Neural Networks 103 (2018) 29–43

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Multilayer bootstrap networks

Xiao-Lei Zhang

Center for Intelligent Acoustics and Immersive Communications, School of Marine Science and Technology, Northwestern Polytechnical University, China

ARTICLE INFO

Article history:
Received 26 June 2017
Received in revised form 4 January 2018
Accepted 6 March 2018
Available online 20 March 2018

Keywords:
Resampling
Ensemble
Nearest neighbor
Tree

ABSTRACT

Multilayer bootstrap network builds a gradually narrowed multilayer nonlinear network from bottom up for unsupervised nonlinear dimensionality reduction. Each layer of the network is a nonparametric density estimator. It consists of a group of k-centroids clusterings. Each clustering randomly selects data points with randomly selected features as its centroids, and learns a one-hot encoder by one-nearest-neighbor optimization. Geometrically, the nonparametric density estimator at each layer projects the input data space to a uniformly-distributed discrete feature space, where the similarity of two data points in the discrete feature space is measured by the number of the nearest centroids they share in common. The multilayer network gradually reduces the nonlinear variations of data from bottom up by building a vast number of hierarchical trees implicitly on the original data space. Theoretically, the estimation error caused by the nonparametric density estimator is proportional to the correlation between the clusterings, both of which are reduced by the randomization steps.

1. Introduction

Principal component analysis (PCA) (Pearson, 1901) is a simple and widely used unsupervised dimensionality reduction method, which finds a coordinate system in the original Euclidean space that the linearly uncorrelated coordinate axes (called principal components) describe the most variances of data. Because PCA is insufficient to capture highly-nonlinear data distributions, many dimensionality reduction methods are explored.

Dimensionality reduction has two core steps. The first step finds a suitable feature space where the density of data with the new feature representation can be well discovered, i.e. a density estimation problem. The second step discards the noise components or small variations of the data with the new feature representation, i.e. a principal component reduction problem in the new feature space.

Dimensionality reduction methods are either linear (He & Niyogi, 2004) or nonlinear based on the connection between the data space and the feature space. This paper focuses on nonlinear methods, which can be categorized to three classes. The first class is kernel methods. It first projects data to a kernel-induced feature space, and then conducts PCA or its variants in the new space. Examples include kernel PCA (Schölkopf, Smola, & Müller, 1998), Isomap (Tenenbaum, De Silva, & Langford, 2000), locally linear embedding (LLE) (Roweis & Saul, 2000), Laplacian eigenmaps (Belkin & Niyogi, 2003; Ng, Jordan, & Weiss, 2002; Shi & Malik, 2000), t-distributed stochastic neighbor embedding (t-SNE) (Van der Maaten & Hinton, 2008), and their generalizations (Nie, Zeng, Tsang, Xu, & Zhang, 2011; Yan et al., 2007). The second class is probabilistic models. It assumes that data are generated from an underlying probability function, and takes the posterior parameters as the feature representation. Examples include Gaussian mixture model and latent Dirichlet allocation (Blei, Ng, & Jordan, 2003). The third class is autoassociative neural networks (Hinton & Salakhutdinov, 2006). It learns a piecewise-linear coordinate system explicitly by backpropagation, and uses the output of the bottleneck layer as the new representation.

However, the feature representations produced by the aforementioned methods are defined in continuous spaces. A fundamental weakness of using a continuous space is that it is hard to find a simple mathematical form that transforms the data space to an ideal continuous feature space, since a real-world data distribution may be non-uniform and irregular. To overcome this difficulty, a large number of machine learning methods have been proposed, such as distance metric learning (Xing, Jordan, Russell, & Ng, 2002) and kernel learning (Lanckriet, Cristianini, Bartlett, Ghaoui, & Jordan, 2004) for kernel methods, and Dirichlet process prior for Bayesian probabilistic models (Teh, Jordan, Beal, & Blei, 2005), in which advanced optimization methods have to be applied. Recently, learning multiple layers of nonlinear transforms, named deep learning, is a trend (Hinton & Salakhutdinov, 2006). A deep network contains more than one nonlinear layers. Each layer consists of a group of nonlinear computational units in parallel. Due to the hierarchical structure and distributed representation at each layer, the representation learning ability of a deep network is exponentially more powerful than that of a shallow network when

E-mail addresses: xiaolei.zhang@nwpu.edu.cn, xiaolei.zhang9@gmail.com.
URL: http://www.xiaolei-zhang.net.

https://doi.org/10.1016/j.neunet.2018.03.005

given the same number of nonlinear units. However, the development of deep learning was mostly supervised, e.g. He, Zhang, Ren, and Sun (2016), Hinton et al. (2012), Schmidhuber (2015), Wang and Chen (2017), Wang, Qin, Nie, and Yuan (2017) and Zhou and Feng (2017). To our knowledge, deep learning for unsupervised dimensionality reduction seems far from explored (Hinton & Salakhutdinov, 2006).

To overcome the aforementioned weakness in a simple way, we revisit the definition of frequentist probability for the density estimation subproblem of dimensionality reduction. *Frequentist probability defines an event's probability as the limit of its relative frequency in a large number of trials* (Wikipedia, 2017). In other words, the density of a local region of a probability distribution can be approximated by counting the events that fall into the local region. This paper focuses on exploring this idea. To generate the events, we resort to *random resampling* in statistics (Efron, 1979; Efron & Tibshirani, 1993). To count the events, we resort to one-nearest-neighbor optimization and binarize the feature space to a discrete space.

To further reduce the small variations and noise components of data, i.e. the second step of dimensionality reduction, we extend the density estimator to a gradually narrowed deep architecture, which essentially builds a vast number of hierarchical trees on the discrete feature space. The overall simple algorithm is named *multilayer bootstrap networks* (MBN).

To our knowledge, although ensemble learning (Breiman, 2001; Dietterich, 2000; Freund & Schapire, 1995; Friedman, Hastie, Tibshirani, et al., 2000; Tao, Tang, Li, & Wu, 2006), which was triggered by random resampling, is a large family of machine learning, it is not very prevalent in unsupervised dimensionality reduction. Furthermore, we did not find methods that estimate the density of data in discrete spaces by random resampling, nor their extensions to deep learning.

This paper is organized as follows. In Section 2, we describe MBN. In Section 3, we give a geometric interpretation of MBN. In Section 4, we justify MBN theoretically. In Section 5, we study MBN empirically. In Section 6, we introduce some related work. In Section 7, we summarize our contributions.

2. Multilayer bootstrap networks

2.1. Network structure

MBN contains multiple hidden layers and an output layer (Fig. 1). Each hidden layer consists of a group of mutually independent k-centroids clusterings; each k-centroids clustering has k output units, each of which indicates one cluster; the output units of all k-centroids clusterings are concatenated as the input of their upper layer. The output layer is PCA.

The network is gradually narrowed from bottom up, which is implemented by setting parameter k as large as possible at the bottom layer and be smaller and smaller along with the increase of the number of layers until a predefined smallest k is reached.

2.2. Training method

MBN is trained layer-by-layer from bottom up.

For training each layer given a d-dimensional input data set $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ either from the lower layer or from the original data space, we simply need to focus on training each k-centroids clustering, which consists of the following steps:

- **Random sampling of features.** The first step randomly selects $\hat{d}$ dimensions of $\mathcal{X}$ ($\hat{d} \leq d$) to form a subset of $\mathcal{X}$, denoted as $\hat{\mathcal{X}} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n\}$.

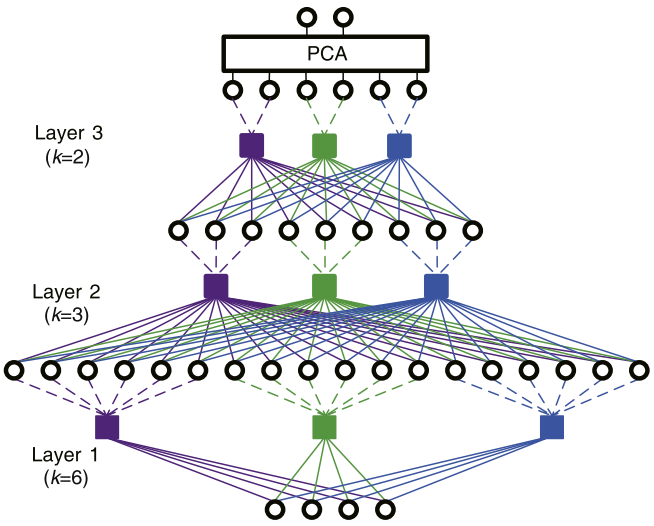

Fig. 1. Network structure. The dimension of the input data for this demo network is 4. Each colored square represents a k-centroids clustering. Each layer contains 3 clusterings. Parameters k at layers 1, 2, and 3 are set to 6, 3, and 2 respectively. The outputs of all clusterings in a layer are concatenated as the input of their upper layer. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

- **Random sampling of data.** The second step randomly selects k data points from $\hat{\mathcal{X}}$ as the k centroids of the clustering, denoted as $\{\mathbf{w}_1, \ldots, \mathbf{w}_k\}$.
- **One-nearest-neighbor learning.** The new representation of an input $\hat{\mathbf{x}}$ produced by the current clustering is an indicator vector $\mathbf{h}$ which indicates the nearest centroid of $\hat{\mathbf{x}}$. For example, if the second centroid is the nearest one to $\hat{\mathbf{x}}$, then $\mathbf{h} = [0, 1, 0, \ldots, 0]^T$. The similarity metric between the centroids and $\hat{\mathbf{x}}$ at the bottom layer is customized, e.g. the squared Euclidean distance $\arg\min_{i=1}^{k} \|\mathbf{w}_i - \hat{\mathbf{x}}\|^2$, and set to $\arg\max_{i=1}^{k} \mathbf{w}_i^T \hat{\mathbf{x}}$ at all other hidden layers.

2.3. Novelty and advantages

Two novel components of MBN distinguish it from other dimensionality reduction methods.

The first component is that each layer is a nonparametric density estimator based on resampling, which has the following major merits:

- It estimates the density of data correctly without any predefined model assumptions. As a corollary, it is insensitive to outliers.
- The representation ability of a group of k-centroids clusterings is exponentially more powerful than that of a single k-centroids clustering.
- The estimation error introduced by binarizing the feature space can be controlled to a small value by simply increasing the number of the clusterings.

The second component is that MBN reduces the small variations and noise components of data by an unsupervised deep ensemble architecture, which has the following main merits:

- It reduces larger and larger local variations of data gradually from bottom up by building as many as $O(k_L 2^V)$ hierarchical trees on the data space (instead of on data points) implicitly, where L is the total number of layers, k_L is parameter k at the Lth layer, V is the number of the clusterings at the layer, and function $O(\cdot)$ is the *order* in mathematics.

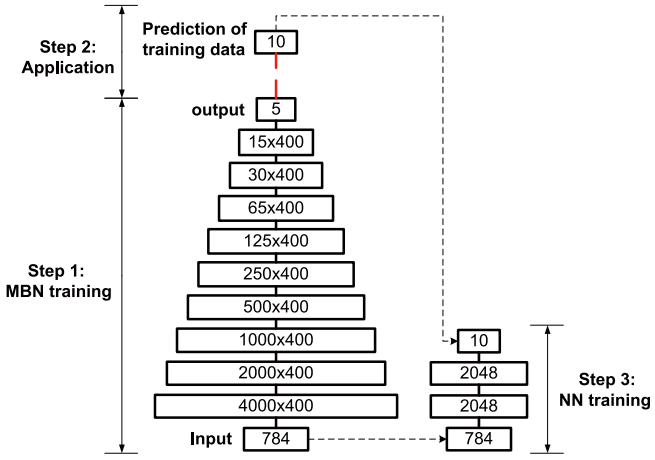

Fig. 2. Principle of compressive MBN. The numbers are the dimensions of representations.

- It does not inherit the problems of deep neural networks such as local minima, overfitting to small-scale data, and gradient vanishing, since it is trained simply by random resampling and stacking. As a result, it can be trained with as many hidden layers as needed and with both small-scale and large-scale data.

See Sections 3 and 4 for the analysis of the above properties.

2.4. Weaknesses

The main weakness of MBN is that the size of the network is large. Although MBN supports parallel computing, its prediction process is inefficient, compared to a neural network.

To overcome this weakness, we propose *compressive MBN* (Fig. 2). Compressive MBN uses a neural network to learn a mapping function from the training data to the output of MBN at the training stage, and uses the neural network for prediction. More generally, if MBN is applied to some specific task, then we can use compressive MBN to learn a mapping function from the input data to the output of the task directly.

3. Geometric interpretation

In this section, we analyze the two components of MBN from the geometric point of view.

3.1. Feature learning by a nonparametric density estimator based on resampling

As presented in Section 1, a core problem of machine learning is to find a suitable similarity metric that maps the original data space to a uniformly-distributed feature space. Here we give an example on its importance. As shown in Fig. 3, the similarity between the data points that are far apart in a distribution with a large variance (e.g., $\mathbf{x}_1$ and $\mathbf{x}_2$ in a distribution $\mathcal{P}_1$) might be the same as the similarity between the data points that are close to each other in a distribution with a small variance (e.g., $\mathbf{x}_3$ and $\mathbf{x}_4$ in a distribution $\mathcal{P}_2$). If we use the Euclidean distance as the similarity metric, then the similarity between $\mathbf{x}_1$ and $\mathbf{x}_2$ is smaller than that between $\mathbf{x}_3$ and $\mathbf{x}_4$, which is not true.

The proposed method provides a simple solution to the above similarity metric learning problem. It outputs a new feature representation of $\mathbf{x}_1$ as follows. Each k-centroids clustering contributes a neighboring centroid of $\mathbf{x}_1$. The centroid partitions the local region of $\mathbf{x}_1$ to two disconnected parts, one containing $\mathbf{x}_1$ and the other not. The data point $\mathbf{x}_1$ owns a local region supported by the centroids of all clusterings that are closest to the data point.[1] These centroids partition the local region to as many as $O(2^V)$ fractions. Each fraction is represented as a unique binary code at the output space of the estimator in a way illustrated in Fig. 4.[2] The new representation of $\mathbf{x}_1$ in the output feature space is a binary code that represents the local fraction of the data space where $\mathbf{x}_1$ is located.

With the new representation, the similarity between two data points is calculated by counting the number of the nearest centroids they share in common—a method in frequentist methodology. In Fig. 3, (i) if the local region in $\mathcal{P}_1$ is partitioned in the same way as the local region in $\mathcal{P}_2$, and if the only difference between them is that the local region in $\mathcal{P}_1$ is an amplification of the local region in $\mathcal{P}_2$, then the surfaces of the two local regions are the same in the discrete feature space. (ii) $\mathbf{x}_1$ and $\mathbf{x}_2$ share 5 common nearest centroids, and $\mathbf{x}_3$ and $\mathbf{x}_4$ also share 5 common nearest centroids, so that the similarity between $\mathbf{x}_1$ and $\mathbf{x}_2$ in $\mathcal{P}_1$ equals the similarity between $\mathbf{x}_3$ and $\mathbf{x}_4$ in $\mathcal{P}_2$.

Note that because V k-centroids clusterings are able to partition the data space to $O(k2^V)$ fractions at the maximum, its representation ability is exponentially more powerful than a single k-centroids clustering.

3.2. Principal component reduction by a deep ensemble architecture

The nonparametric density estimator captures the variances of the input data, however, it is not responsible for reducing the small variances and noise components. To reduce the nonlinear variations, we build a gradually narrowed deep network. The network essentially reduces the nonlinear variations of data by building a vast number of hierarchical trees on the data space (instead of on data points) implicitly. We present its geometric principle as follows.

Suppose MBN has L layers, parameters k at layers 1 to L are $k_1, k_2, \ldots, k_L$ respectively, and $k_1 > k_2 > \cdots > k_L$. From Fig. 4, we know that the lth layer partitions the input data space to $O(k_l 2^V)$ disconnected small fractions. Each fraction is encoded as a single point in the output feature space. The points in the output feature space are the nodes of the trees. Hence, the lth layer has $O(k_l 2^V)$ nodes. The bottom layer has $O(k_1 2^V)$ leaf nodes. The top layer has $O(k_L 2^V)$ root nodes, which is the number of the trees that MBN builds.

For two adjacent layers, because $k_{l-1} < k_l$, it is easy to see that $O(k_{l-1}/k_l)$ neighboring child nodes at the $(l-1)$-th layer will be merged to a single father node at the lth layer on average. To generalize the above property to the entire MBN, a single root node at the Lth layer is a merging of $O(k_1/k_L)$ leaf nodes at the bottom layer, which means that the nonlinear variations among the leaf nodes that are to be merged to the same root node will be reduced completely. Because $k_L \ll k_1$, we can conclude that MBN is highly invariant to the nonlinear variations of data.

One special case is that, if $k_L = 1$, the output feature spaces of all k-centroids clusterings at the top layer are just a single point. However, in practice, we never set $k_L = 1$; instead, we usually set the termination condition of MBN as $k_L \geq 1.5c$ where c is the ground-truth number of classes. The termination condition makes each k-centroids clustering stronger than random guess (Schapire, 1990).

[1] It is important to keep parameter k of the k-centroids clusterings in a layer the same. Otherwise, the density of the centroids between the clusterings is different, that is to say, if we regard the centroids as the coordinates axes of the local region, then the coordinate axes are built in different data spaces.

[2] A small fraction should not have to contain data points as shown in Fig. 4.

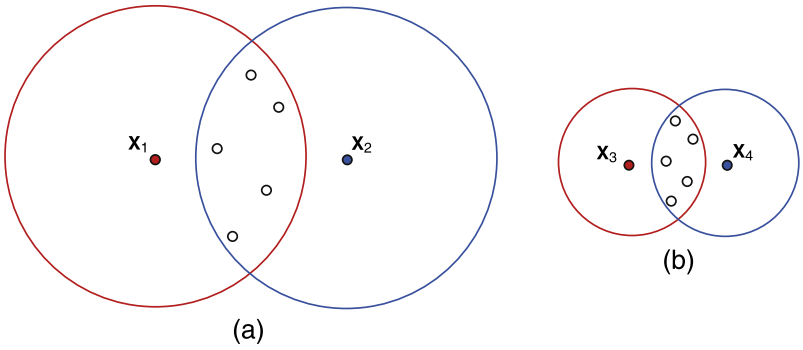

Fig. 3. Illustration of the similarity metric problem at the data space. (a) The similarity problem of two data points $\mathbf{x}_1$ (in red color) and $\mathbf{x}_2$ (in blue color) in a distribution $\mathcal{P}_1$. The local region of $\mathbf{x}_1$ (or $\mathbf{x}_2$) is the area in a colored circle that is centered at $\mathbf{x}_1$ (or $\mathbf{x}_2$). The small hollow points that lie in the cross area of the two local regions are the shared centroids by $\mathbf{x}_1$ and $\mathbf{x}_2$. (b) The similarity problem of two data points $\mathbf{x}_3$ and $\mathbf{x}_4$ in a distribution $\mathcal{P}_2$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

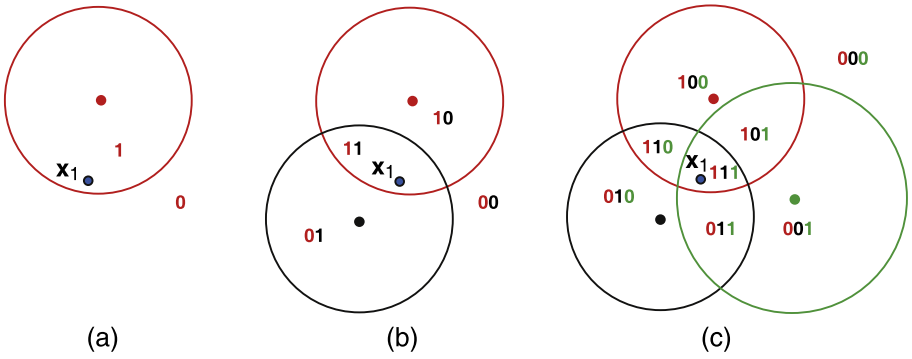

Fig. 4. Encoding the local region of a data point $\mathbf{x}_1$ (in blue color) by (a) one k-centroids clustering, (b) two k-centroids clusterings, and (c) three k-centroids clusterings. The centroids of the three k-centroids clusterings are the points colored in red, black, and green respectively. The local region of a centroid is the area in the circle around the centroid. The centroid itself and the edge of its local region are drawn in the same color. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Note that because $O(k2^V) \gg n$, the data points are distributed sparsely in the output feature space. Hence, MBN seldom merges data points. In other words, MBN learns data representations instead of doing agglomerative data clustering.

4. Theoretical analysis

Although MBN estimates the density of data in the discrete feature space, its estimation error is small and controllable. Specifically, as shown in Fig. 4c, V k-centroids clusterings can partition a local region to $O(2^V)$ disconnected small fractions at the maximum. It is easy to imagine that, for any local region, when V is increasing and the diversity between the centroids is still reserved, an ensemble of k-centroids clusterings approximates the true data distribution. Because the diversity is important, we have used two randomized steps to enlarge it.

In the following, we analyze the estimation error of the proposed method formally:

Theorem 1. *The estimation error of the building block of MBN $\mathcal{E}_{ensemble}$ and the estimation error of a single k-centroids clustering $\mathcal{E}_{single}$ have the following relationship:*

$$\mathcal{E}_{ensemble} = \left(\frac{1}{V} + \left(1 - \frac{1}{V}\right)\rho\right)\mathcal{E}_{single} \tag{1}$$

where ρ is the pairwise positive correlation coefficient between the k-centroids clusterings, $0 \le \rho \le 1$.

Proof. We prove Theorem 1 by first transferring MBN to a supervised regression problem and then using the bias–variance decomposition of the mean squared error to get the bias and variance components of MBN. The detail is as follows.

Suppose the random samples for training the k-centroids clusterings are identically distributed but not necessarily independent, and the *pairwise positive correlation coefficient* between two random samples (i.e., $\{\mathbf{w}_{v_1,i}\}_{i=1}^{k}$ and $\{\mathbf{w}_{v_2,j}\}_{j=1}^{k}, \forall v_1, v_2 = 1, \ldots, V$ and $v_1 \neq v_2$) is ρ, $0 \le \rho \le 1$.

We focus on analyzing a given point $\mathbf{x}$, and assume that the *true* local coordinate of $\mathbf{x}$ is $\mathbf{s}$ which is an invariant point around $\mathbf{x}$ and usually found when the density of the nearest centroids around $\mathbf{x}$ goes to infinity (i.e. $n \to \infty, V \to \infty$, $\{\mathbf{w}_v\}_{v=1}^{V}$ are identically and independently distributed, and parameter k is unchanged). We also suppose that $\mathbf{x}$ is projected to $\hat{\mathbf{s}}$ when given a finite number of nearest centroids $\{\mathbf{w}_v\}_{v=1}^{V}$. The correlation coefficient between the centroids $\{\mathbf{w}_v\}_{v=1}^{V}$ is ρ. Note that $\mathbf{s}$ is used as an invariant reference point for studying $\hat{\mathbf{s}}$.

The effectiveness of MBN can be evaluated by the estimation error of the estimate $\hat{\mathbf{s}}$ to the truth $\mathbf{s}$. The *estimation error* is defined by $E(\|\mathbf{s} - \hat{\mathbf{s}}\|^2)$ where $E(\cdot)$ is the expectation of a random variable.

From the geometric interpretation, we know that each $\mathbf{w}_v$ owns a local space S_v, and moreover, both $\mathbf{s}$ and $\hat{\mathbf{s}}$ are in S_v. When parameter $k \to n$, S_v is small enough to be locally linear. Under the above fact and an assumption that the features of $\mathbf{w}_v$, i.e. $w_{v,1}, \ldots, w_{v,d}, \ldots, w_{v,D}$, are uncorrelated, we are able to assume that $\mathbf{w}_v$ follows a multivariate normal distribution around $\mathbf{s}$: $\mathbf{w}_v \sim \mathcal{MN}(\mathbf{s}, \sigma^2\mathbf{I})$ where $\mathbf{I}$ is the identity matrix, σ describes the

Table 1
Hyperparameters of MBN.

Parameter	Description
δ	A parameter that controls the network structure by $k_{l+1} = \delta k_l$, $\forall l = 1, \ldots, L$
a	Fraction of randomly selected dimensions (i.e., $\hat{d}$) over all dimensions (i.e., d) of input data.
V	Number of k-centroids clusterings per layer.
k_1	A parameter that controls the time and storage complexities of the network. For small-scale problems, $k_1 = 0.5n$. For large-scale problems, k_1 should be tuned smaller to fit MBN to the computing power.

variance of $\mathbf{w}_v$ and $\sigma^2\mathbf{I}$ is the covariance matrix. We can further derive $E(\|\mathbf{s} - \hat{\mathbf{s}}\|^2) = \sum_{d=1}^{D} E((s_d - \hat{s}_d)^2)$ where we denote $\mathbf{s} = [s_1, \ldots, s_D]^T$ and $\hat{\mathbf{s}} = [\hat{s}_1, \ldots, \hat{s}_D]^T$.

In the following, we focus on analyzing the estimation error at a single dimension $E((s_d - \hat{s}_d)^2)$ and will omit the index d for clarity. It is known that the mean squared error of a regression problem can be decomposed to the summation of a squared bias component and a variance component (Hastie, Tibshirani, & Friedman, 2009):

$$\begin{aligned} E((s - \hat{s})^2) &= \left(s - E\left(\hat{s}\right)\right)^2 + E\left(\left(\hat{s} - E\left(\hat{s}\right)\right)^2\right) \\ &= \text{Bias}^2\left(\hat{s}\right) + \text{Var}\left(\hat{s}\right). \end{aligned} \tag{2}$$

Because w_v follows a univariate normal distribution, it is easy to obtain:

$$E(w_v) = s \tag{3}$$

$$E(w_v^2) = \sigma^2 + s^2 \tag{4}$$

$$E(w_{v_1} w_{v_2}) = \rho\sigma^2 + s^2. \tag{5}$$

For a single estimate w_v, we have $\hat{s}_v = w_v$. For a set of estimates $\{w_v\}_{v=1}^{V}$, we have $\hat{s}_\Sigma = \frac{1}{V}\sum_{v=1}^{V} w_v$. Based on Eqs. (3) to (5), we can derive:

$$\text{Bias}^2\left(\hat{s}_v\right) = 0 \tag{6}$$

$$\sigma^2_{\text{single}} = \text{Var}\left(\hat{s}_v\right) = \sigma^2 \tag{7}$$

and

$$\text{Bias}^2\left(\hat{s}_\Sigma\right) = 0 \tag{8}$$

$$\sigma^2_{\text{ensemble}} = \text{Var}\left(\hat{s}_\Sigma\right) = \frac{\sigma^2}{V} + \left(1 - \frac{1}{V}\right)\rho\sigma^2. \tag{9}$$

Substituting Eqs. (6) and (7) to Eq. (2) gets:

$$\mathcal{E}_{\text{single}} = \sigma^2 \tag{10}$$

and substituting Eqs. (8) and (9) to Eq. (2) gets:

$$\mathcal{E}_{\text{ensemble}} = \frac{\sigma^2}{V} + \left(1 - \frac{1}{V}\right)\rho\sigma^2. \tag{11}$$

Theorem 1 is proved. □

From Theorem 1, we can get the following corollaries easily:

Corollary 1. *When ρ is reduced from 1 to 0, $\mathcal{E}_{ensemble}$ is reduced from $\mathcal{E}_{single}$ to $\mathcal{E}_{single}/V$ accordingly.*

Corollary 2. *When $V \to \infty$, $\mathcal{E}_{ensemble}$ reaches a lower bound $\rho\mathcal{E}_{single}$.*

From Corollary 1, we know that increasing parameter V and reducing ρ can help MBN reduce the estimation error.

From Corollary 2, we know that it is important to reduce ρ. We have adopted two randomization steps to reduce ρ. However, although decreasing parameters a and k can help MBN reduce ρ, it will also cause $\mathcal{E}_{\text{single}}$ rise. In other words, reducing $\mathcal{E}_{\text{single}}$ and reducing ρ is a pair of contradictory factors, which needs a balance through a proper parameter setting.

5. Empirical evaluation

In this section, we first introduce a typical parameter setting of MBN, then demonstrate the density estimation ability of MBN on synthetic data sets, and finally apply the low dimensional output of MBN to the tasks of visualization, clustering, and document retrieval.

5.1. Parameter setting

When a data set is small-scale, we use the linear-kernel-based kernel PCA (Canu, Grandvalet, Guigue, & Rakotomamonjy, 2005; Schölkopf et al., 1998) as the PCA toolbox of MBN. When a data set is middle- or large-scale, we use the expectation–maximization PCA (EM-PCA) (Roweis, 1998).[3]

MBN is insensitive to parameters V and a as if $V > 100$ and $a \in [0.5, 1]$. If not specified, we used $V = 400$ and $a = 0.5$ as the default values.

We denote parameter k at layer l as k_l. To control k_l, we introduce a parameter δ defined as $k_{l+1} = \delta k_l$, $\delta \in (0, 1)$. MBN is relatively sensitive to parameter δ: if data are highly-nonlinear, then set δ to a large value, otherwise, set δ to a small value; if the nonlinearity of data is unknown, set $\delta = 0.5$ which is our default.

Finally, if $k_{L+1} < 1.5c$, then we stop MBN training and use the output of the Lth nonlinear layer for PCA. This terminating condition guarantees the validness of data resampling which requires each random sample of data to be stronger than random guess (Schapire, 1990). The hyperparameters of MBN are summarized in Table 1.

In the following experiments, we adopted the above default parameter setting of MBN, unless otherwise specified.

5.2. Density estimation

5.2.1. Density estimation for nonlinear data distributions

Four synthetic data sets with non-uniform densities and nonlinear variations are used for evaluation. They are Gaussian data, Jain data (Jain & Law, 2005), pathbased data (Chang & Yeung, 2008), and compound data (Zahn, 1971), respectively. Parameter a was set to 1.

The visualization result in Fig. 5 shows that the synthetic data in the new feature spaces produced by MBN not only are distributed uniformly but also do not contain many nonlinear variations.

5.2.2. Density estimation in the presence of outliers

A data set of two-class Gaussian data with a randomly generated outlier is used for evaluation. The results of PCA and MBN are shown in Fig. 6. From the figure, we observe that MBN is robust to the presence of the outlier.

[3] The word "large-scale" means that the data cannot be handled by traditional kernel methods on a common personal computer.

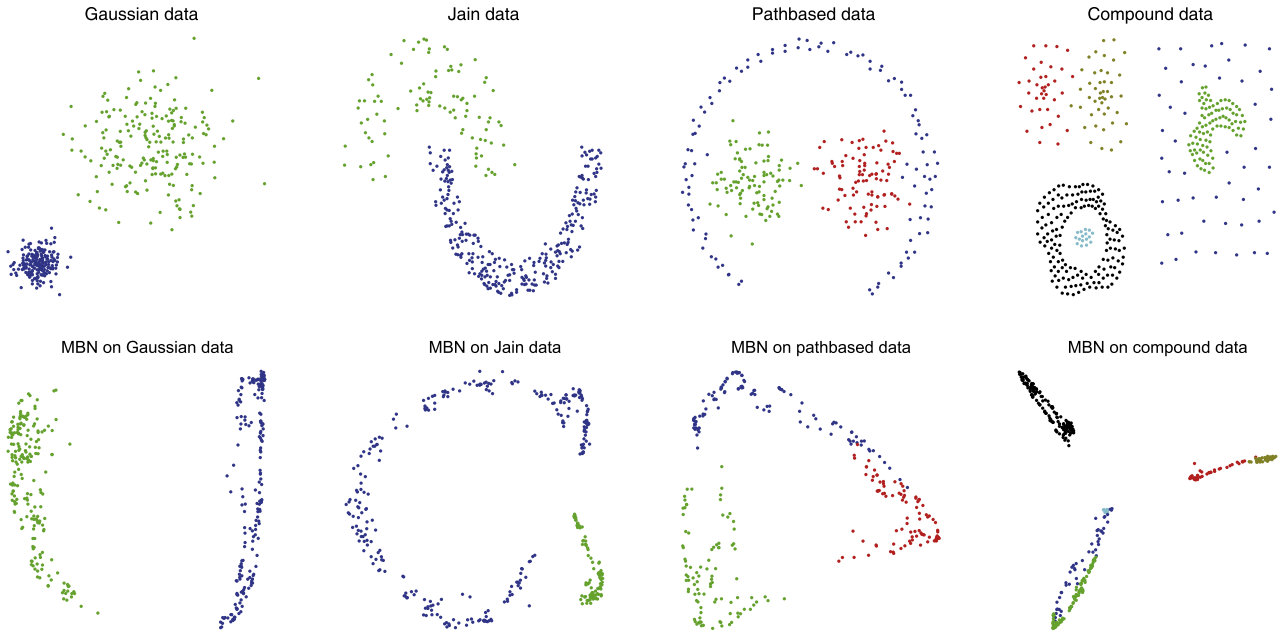

Fig. 5. Density estimation by MBN on synthetic data sets. The data points in different classes are drawn in different colors.

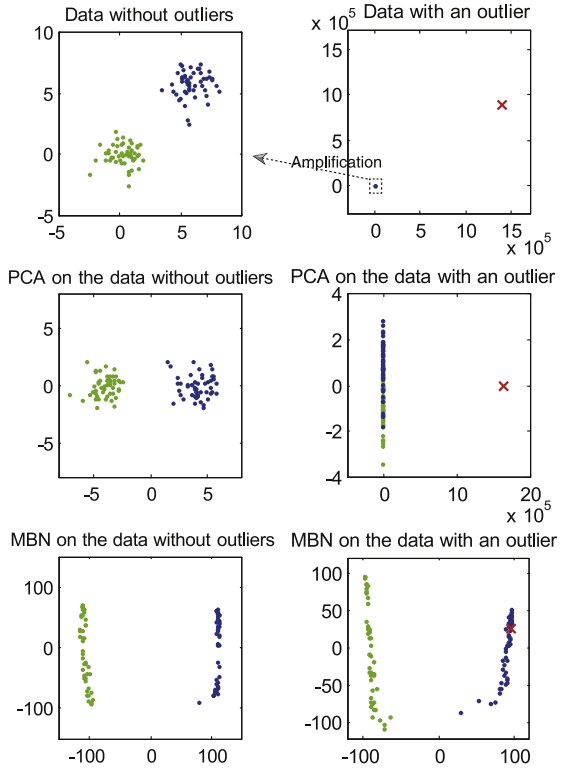

Fig. 6. Density estimation by MBN on a Gaussian data with the presence of an outlier. The red cross denotes the outlier. Because the outlier is far from the Gaussian data, the resolution of the sub-figure in the top-right corner is not high enough to differentiate the two classes of the Gaussian data.

5.3. Data visualization

5.3.1. Visualizing AML–ALL biomedical data

The acute myeloid leukemia and acute lymphoblastic leukemia (AML–ALL) biomedical data set (Golub et al., 1999) is a two-class problem that consists of 38 training examples (27 ALL, 11 AML) and 34 test examples (20 ALL, 14 AML). Each example has 7,129 dimensions produced from 6,817 human genes.

We compared MBN with PCA and 2 nonlinear dimensionality reduction methods which are Isomap (Tenenbaum et al., 2000) and LLE (Roweis & Saul, 2000). We tuned the hyperparameters of Isomap and LLE for their best performance.[4] The visualization result is shown in Fig. 7.

5.3.2. Visualizing MNIST digits

The data set of the MNIST digits (Lecun, Cortes, & Burges, 2004) contains 10 handwritten integer digits ranging from 0 to 9. It consists of 60,000 training images and 10,000 test images. Each image has 784 dimensions.

We compared MBN with PCA, Isomap (Tenenbaum et al., 2000), LLE (Roweis & Saul, 2000), Spectral (Ng et al., 2002), deep belief networks (Hinton & Salakhutdinov, 2006) (DBN), and t-SNE (Van der Maaten & Hinton, 2008). We tuned the hyperparameters of the 5 nonlinear comparison methods for their best performance.

Because the comparison nonlinear methods are too costly to run with the full MNIST data set except DBN, we randomly sampled 5000 images with 500 images per digit for evaluation. The visualization result in Fig. 8 shows that the low-dimensional feature produced by MBN has a small within-class variance and a large between-class distance.

To demonstrate the scalability of MBN on larger data sets and its generalization ability on unseen data, we further trained MBN on all 60,000 training images, and evaluated its effectiveness on the 10,000 test images, where k_1 was set to 8000 to reduce the training cost. The visualization result in Fig. 9 shows that MBN on the full MNIST provides a clearer visualization than that on the small subset of MNIST.

5.4. Clustering

Ten benchmark data sets were used for evaluation. They cover topics in speech processing, chemistry, biomedicine, image processing, and text processing. The details of the data sets are given in

[4] It is difficult and sometimes unable to tune the hyperparameters in practice.

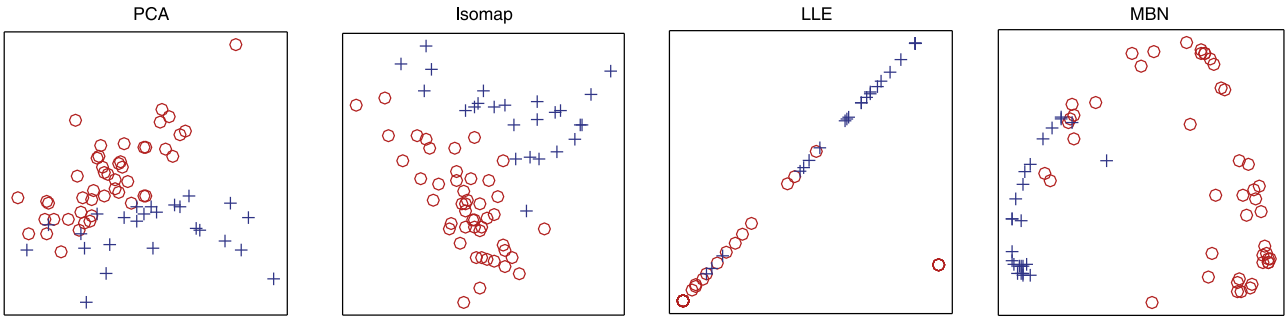

Fig. 7. Visualizations of AML–ALL produced by MBN and 3 comparison methods.

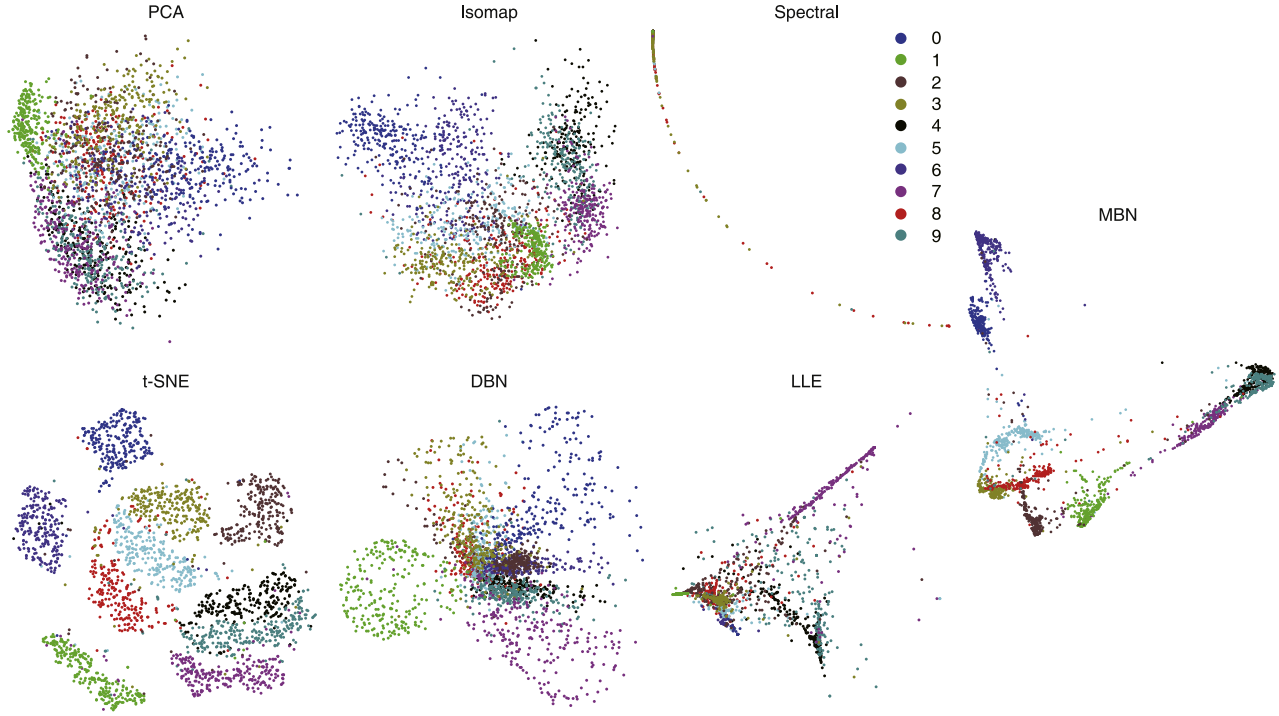

Fig. 8. Visualizations of a subset (5000 images) of MNIST produced by MBN and 6 comparison methods. For clarity, only 250 images per digit are drawn.

Table 2. The data set "20-Newsgroups", which originally has 20,000 documents, was post-processed to a corpus of 18,846 documents, each belonging to a single topic.

We compared MBN with PCA and Spectral (Ng et al., 2002). We applied k-means clustering to the low-dimensional outputs of all comparison methods as well as the original high-dimensional features. To prevent the local minima problem of k-means clustering, we ran k-means clustering 50 times and picked the clustering result that corresponded to the optimal objective value of the k-means clustering among the 50 candidate objective values as the final result. We ran each comparison method 10 times and reported the average performance.

The parameter settings on the data sets with IDs from 1 to 9 are as follows. For PCA, we preserved the top 98% largest eigenvalues and their corresponding eigenvectors. For Spectral, we set the output dimension to the ground-truth number of classes and adopted the RBF kernel. We reported the results with a fixed kernel width $2^{-4}A$ which behaves averagely the best over all data sets, as well as the best result by searching the kernel width from $\{2^{-4}A, 2^{-3}A, \ldots, 2^{4}A\}$ on each data set, where A is the average pairwise Euclidean distance between data. The two parameter selection methods of Spectral are denoted as Spectral^{no_tuning} and Spectral^{optimal} respectively. For MBN, we set the output dimension

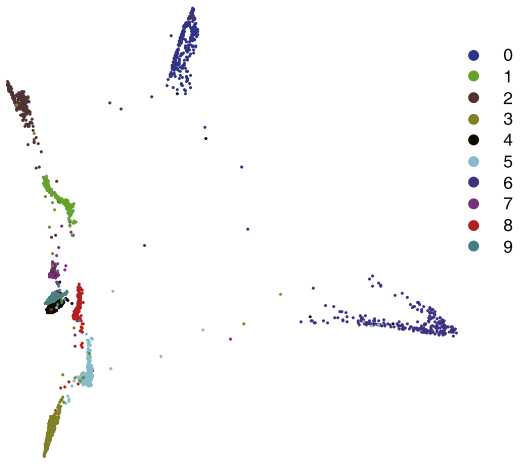

Fig. 9. Visualization of the 10,000 test images of MNIST produced by the MBN at layer 8. For clarity, only 250 images per digit are drawn. See Fig. D.16 in Appendix D for the visualizations produced by other layers.

Table 2
Description of data sets.

ID	Name	# data points	# dimensions	# classes	Attribute
1	Isolet1	1 560	617	26	Speech data
2	Wine	178	13	3	Chemical data
3	New-Thyroid	215	5	3	Biomedical data
4	Dermatology	366	34	6	Biomedical data
5	COIL100	7 200	1 024	100	Images
6	MNIST(small)	5 000	768	10	Images (handwritten digits)
7	MNIST(full)	70 000	768	10	Images (handwritten digits)
8	UMIST	575	1 024	20	Images (faces)
9	ORL	400	1 024	40	Images (faces)
10	20-Newsgroups	18 846	26 214	20	Text corpus

Table 3
NMI on 10 data sets. The methods named raw feature, PCA, Spectral^{no_tuning}, and MBN^{no_tuning} methods are compared with each other, while Spectral^{optimal} and MBN^{optimal} are compared with each other. The numbers after ± are the standard deviations.

		Raw feature	PCA	Spectral^{no_tuning}	MBN^{no_tuning}	Spectral^{optimal}	MBN^{optimal}
1	Isolet1	**77.21% ± 0.92%**	56.74% ± 0.75%	75.51% ± 0.58%	**77.89% ± 0.81%**	**79.66% ± 0.58%**	77.97% ± 0.64%
2	Wine	42.88% ± 0.00%	40.92% ± 0.00%	41.58% ± 0.00%	**55.49% ± 4.07%**	43.02% ± 0.00%	**76.96% ± 3.24%**
3	New-Thyroid	49.46% ± 0.00%	49.46% ± 0.00%	39.63% ± 0.00%	**68.80% ± 5.03%**	43.20% ± 0.00%	**75.78% ± 2.75%**
4	Dermatology	9.11% ± 0.11%	59.50% ± 0.10%	51.04% ± 0.15%	**82.40% ± 2.24%**	51.04% ± 0.15%	**92.79% ± 1.09%**
5	COIL100	76.98% ± 0.27%	69.64% ± 0.45%	**89.28% ± 0.59%**	81.66% ± 0.59%	89.28% ± 0.59%	**90.37% ± 0.65%**
6	MNIST(small)	49.69% ± 0.14%	27.86% ± 0.08%	62.06% ± 0.04%	**77.12% ± 0.35%**	62.06% ± 0.04%	**78.50% ± 0.84%**
7	MNIST(full)	50.32% ± 0.12%	28.05% ± 0.07%	Timeout	**91.36% ± 0.07%**	Timeout	Timeout
8	UMIST	65.36% ± 1.21%	66.25% ± 1.10%	**81.83% ± 0.58%**	74.48% ± 1.91%	**84.66% ± 2.38%**	**84.84% ± 1.98%**
9	ORL	75.55% ± 1.36%	75.81% ± 1.17%	**80.21% ± 1.13%**	**79.22% ± 0.84%**	**83.62% ± 1.13%**	81.73% ± 1.19%
10	20-Newsgroups	Timeout	22.49% ± 1.41%	27.03% ± 0.05%	**41.61% ± 0.05%**	27.03% ± 0.05%	**42.23% ± 0.46%**

Table 4
Clustering accuracy on 10 data sets.

		Raw feature	PCA	Spectral^{no_tuning}	MBN^{no_tuning}	Spectral^{optimal}	MBN^{optimal}
1	Isolet1	**61.47% ± 1.93%**	38.62% ± 0.99%	49.63% ± 1.88	**61.13% ± 2.52%**	**72.29% ± 1.92%**	62.50% ± 1.14%
2	Wine	70.22% ± 0.00%	78.09% ± 0.00%	61.24% ± 0.00	**81.91% ± 2.61%**	70.79% ± 0.00%	**93.20% ± 1.77%**
3	New-Thyroid	86.05% ± 0.00%	86.05% ± 0.00%	79.49% ± 0.94	**93.02% ± 1.60%**	82.79% ± 0.00%	**94.51% ± 0.93%**
4	Dermatology	26.17% ± 0.28%	61.67% ± 0.26%	50.38% ± 0.35	**82.81% ± 7.67%**	50.38% ± 0.35%	**96.07% ± 0.79%**
5	COIL100	49.75% ± 1.31%	43.42% ± 1.21%	**61.83% ± 2.27**	57.38% ± 1.85%	61.83% ± 2.27%	**68.29% ± 1.58%**
6	MNIST(small)	52.64% ± 0.14%	34.49% ± 0.10%	53.30% ± 0.01	**82.36% ± 0.46%**	56.55% ± 0.20%	**87.06% ± 1.75%**
7	MNIST(full)	53.48% ± 0.11%	35.14% ± 0.11%	Timeout	**96.64% ± 0.04%**	Timeout	Timeout
8	UMIST	43.20% ± 1.66%	43.44% ± 1.92%	**70.99% ± 2.19**	56.96% ± 3.73%	70.99% ± 2.19%	**73.22% ± 4.02%**
9	ORL	54.37% ± 2.41%	54.55% ± 2.81%	57.33% ± 2.37	**59.68% ± 1.58%**	**68.85% ± 2.48%**	64.25% ± 3.50%
10	20-Newsgroups	Timeout	22.61% ± 1.29%	28.52% ± 0.03	**46.57% ± 1.10%**	28.52% ± 0.03%	**47.69% ± 0.92%**

to the ground-truth number of classes. We reported the results with $\delta = 0.5$, as well as the best results by searching δ from $\{0.1, 0.2, \ldots, 0.9\}$ on each data set. The two parameter selection methods of MBN are denoted as MBN^{no_tuning} and MBN^{optimal} respectively.

The parameter settings on the 20-Newsgroups are as follows. For PCA, we set the output dimension to 100. For all comparison methods, we used the cosine similarity as the similarity metric.

We evaluated the clustering result in terms of normalized mutual information (NMI) and clustering accuracy. The clustering results in Tables 3 and 4 show that (i) MBN^{no_tuning} achieves better performance than Spectral^{no_tuning} and PCA, and (ii) MBN^{optimal} achieves better performance than Spectral^{optimal}.

Besides the data sets in Table 2, we have also conducted experiments on the following data sets: Lung-Cancer biomedical data, COIL20 images, USPS images, Extended-YaleB images, Reuters-21578 text corpus, and TDT2 text corpus. The experimental conclusions are consistent with the results in Tables 3 and 4.

5.4.1. Effect of hyperparameter δ

We showed the effect of parameter δ on each data set in Fig. 10. From the figure, we do not observe a stable interval of δ where MBN is supposed to achieve the optimal performance across the data sets; if the data are highly variant, then setting δ to a large value yields good performance, and vise versa. Generally, if the nonlinearity of data is unknown, then setting $\delta = 0.5$ is safe.

5.4.2. Effects of hyperparameters a and V

Theorem 1 has guaranteed that the estimation error can be reduced by enlarging parameter V, and may also be reduced by decreasing parameter a. In this subsection, we focus on investigating the stable working intervals of V and a.

Because the experimental phenomena on all data sets are similar, we reported the phenomena on the small subset of MNIST as a representative in Fig. 11. From the figure, we know that (i) we should prevent setting a to a very small value. Empirically, we set $a = 0.5$. (ii) We should set $V \geq 100$. Empirically, we set $V = 400$.

5.5. Document retrieval

We applied MBN to document retrieval and compared it with latent semantic analysis (LSA) (Deerwester, Dumais, Landauer, Furnas, & Harshman, 1990), a document retrieval method based on PCA, on a larger data set—Reuters newswire stories (Lewis, Yang, Rose, & Li, 2004) which consist of 804,414 documents. The data set of the Reuters newswire stories is divided into 103 topics which are grouped into a tree structure. We only preserved the 82 leaf topics. As a result, there were 107,132 unlabeled documents. We preprocessed each document as a vector of 2000 commonest word stems by the *rainbow* software (McCallum, 1998) where each entry of a vector was set to the word count.

We randomly selected half of the data set for training and the other half for test. We recorded the average accuracy over all 402,207 queries in the test set at the document retrieval setting,

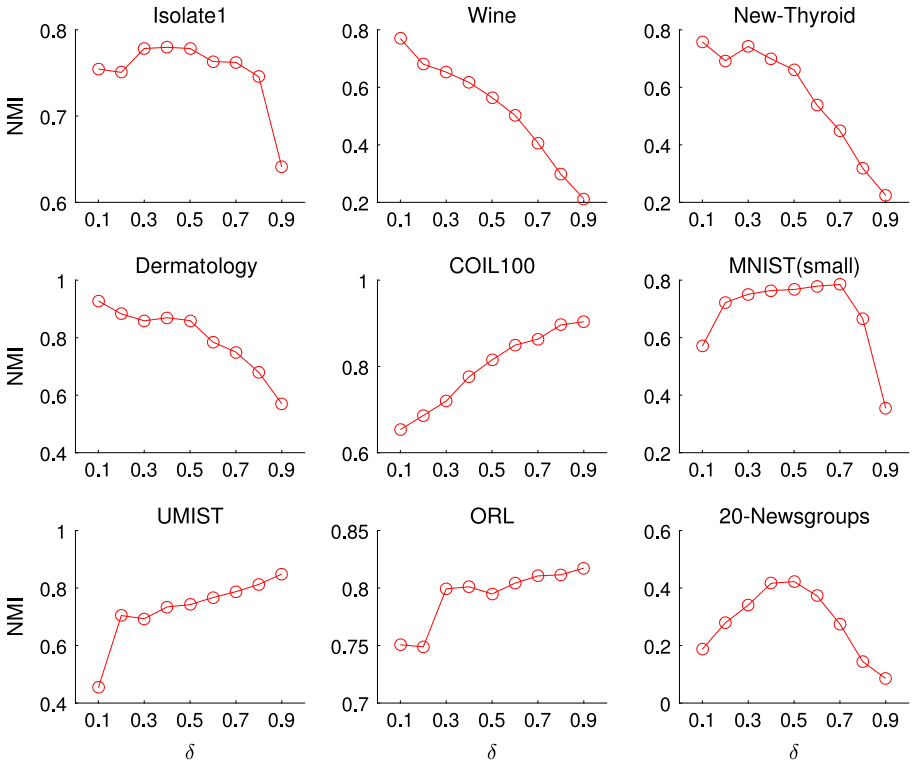

Fig. 10. Effect of decay factor δ on 9 benchmark data sets.

where a query and its retrieved documents were different documents in the test set. If an unlabeled document was retrieved, it was considered as a mistake. If an unlabeled document was used as a query, no relevant documents would be retrieved, which means the precisions of the unlabeled query at all levels were zero.

Because the data set is relatively large, we did not adopt the default setting of MBN where $k_1 = 201103$. Instead, we set k_1 manually to 500 and 2000 respectively, $V = 200$, and kept other parameters unchanged, i.e. $a = 0.5$ and $\delta = 0.5$.

Experimental results in Fig. 12 show that the MBN with the small network reaches an accuracy curve of over 8% higher than LSA; the MBN with the large network reaches an accuracy curve of over 13% higher than LSA. The results indicate that enlarging the network size of MBN improves its generalization ability.

5.6. Empirical study on compressive MBN

This subsection studies the effects of the compressive MBN on accelerating the prediction process of MBN.

We first studied the generalization ability of the compressive MBN on the 10,000 test images of MNIST, where the models of MBN and the compressive MBN are trained with the 60,000 training images of MNIST. The neural network in the compressive MBN contains 2 hidden layers with 2048 rectified linear units per layer. It projects the data space to the 2-dimensional feature space produced by MBN. The visualization results in Fig. 13 show that the compressive MBN produces an identical 2-dimensional feature with MBN on the training set and generalizes well on the test set. The prediction time of MBN and the compressive MBN is 4857.45 and 1.10 s, respectively.

We then studied the generalization ability of the compressive MBN in retrieving the Reuters newswire stories. The neural network here has the same structure with that on MNIST. It projects the data space to the 5-dimensional representation produced by MBN. The result in Fig. 14 shows that the compressive MBN produces an almost identical accuracy curve with MBN. The prediction time of MBN and the compressive MBN is 190,574.74 and 88.80 s, respectively.

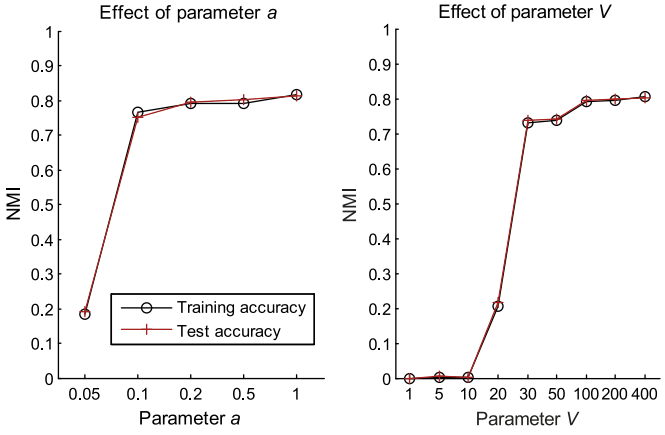

Fig. 11. Effect of parameters a and V on the subset of MNIST.

6. Discussions

MBN is related to many methods in statistics and machine learning. Here we introduce its connection to histogram-based density estimators, bootstrap methods, clustering ensemble, vector quantization, product of experts, sparse coding, and unsupervised deep learning.

6.1. Histogram-based density estimators

Histogram-based density estimation is a fundamental density estimation method, which estimates a probability function by

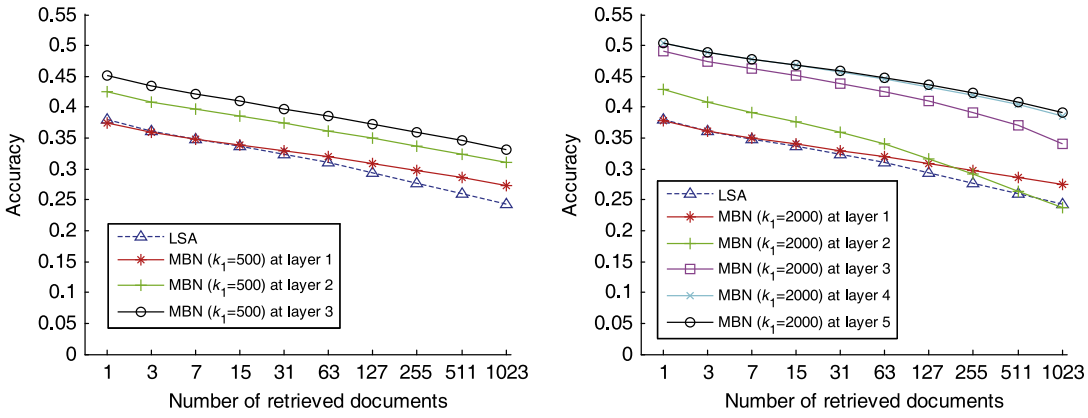

Fig. 12. Average accuracy curves of retrieved documents on the test set (82 topics) of the Reuters newswire stories.

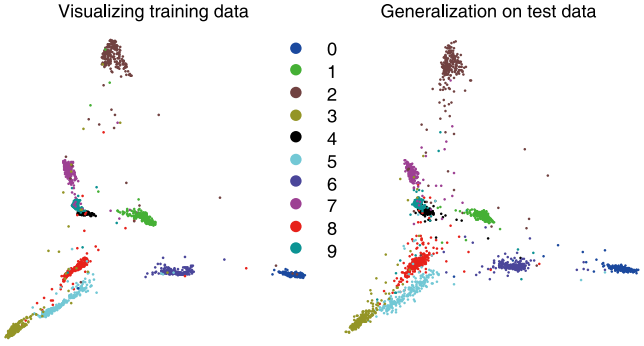

Fig. 13. Visualizations of the training data (left) and test data (right) of MNIST produced by the compressive MBN.

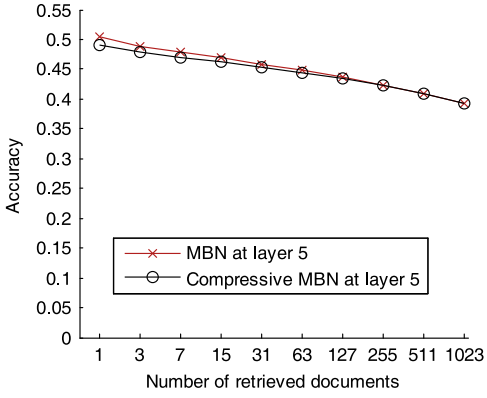

Fig. 14. Comparison of the generalization ability of MBN and the compressive MBN on retrieving the Reuters newswire stories.

accumulating the events that fall into the intervals of the function (Freedman, Pisani, & Purves, 2007) where the intervals may have an equivalent length or not. Our proposed density estimator is essentially such an approach. An interval in our approach is defined as the data space between two data points. The accumulated events in the interval are the shared centroids by the two data points.

6.2. Bootstrap methods

Bootstrap resampling (Efron, 1979; Efron & Tibshirani, 1993) has been applied successfully to machine learning. It resamples data *with replacement*. MBN does not adopt the standard bootstrap resampling. In fact, MBN uses random subsampling *without replacement*, also known as *delete- d jackknife resampling* in statistics (Efron & Tibshirani, 1993). The reason why we do not adopt the standard bootstrap resampling is that the resampling in MBN is used to build local coordinate systems, hence, if a data point is sampled multiple times, the duplicated data points are still viewed as a single coordinate axis. Moreover, it will cause the k-centroids clusterings built in different data spaces. However, MBN was motivated from and shares many common properties with the bootstrap methods, such as building each base clustering from a random sample of the input and de-correlating the base clusterings by the random sampling of features (Breiman, 2001) at each base clustering, hence, we adopted the phrase "bootstrap" in MBN and clarify its usage here for preventing confusion.

6.3. Clustering ensemble

Clustering ensemble (Dudoit & Fridlyand, 2003; Fern & Brodley, 2003; Fred & Jain, 2005; Strehl & Ghosh, 2003; Vega-Pons & Ruiz-Shulcloper, 2011; Zheng, Li, & Ding, 2010) is a clustering technique that uses a *consensus function* to aggregate the clustering results of a set of mutually-independent base clusterings. Each base clustering is usually used to classify data to the ground-truth number of classes. An exceptional clustering ensemble method is Fred and Jain (2005), in which each base k-means clustering produces a subclass partition by assigning the parameter k to a random value that is slightly larger than the ground-truth number of classes.

Each layer of MBN can be regarded as a clustering ensemble. However, its purpose is to estimate the density of data instead of producing an aggregated clustering result. Moreover, MBN has clear theoretical and geometric explanations. We have also found that setting k to a random value does not work for MBN, particularly when k is also small.

6.4. Vector quantization

Each layer of MBN can be regarded as a vector quantizer to the input data space. The codebook produced by MBN is exponentially smaller than that produced by a traditional k-means clustering, when they have the same level of quantization errors. A similar idea, named *product quantization*, has been explored in Jegou, Douze, and Schmid (2011). If product quantization uses random sampling of features instead of a fixed non-overlapping partition of features in Jegou et al. (2011), and uses random sampling of data points to train each k-means clustering instead of the expectation–maximization optimization, then product quantization equals a single layer of MBN. We are also aware of the hierarchical product quantizer (Wichert, 2012), which builds multiple sets of sub-quantizers (where the name "hierarchy" comes) on the original

feature. Our MBN builds each layer of sub-quantizers on the output sparse feature of its lower layer which is fundamentally different from the hierarchical product quantization method.

An important difference between existing vector quantization methods (e.g. binarized neural networks and binary hashing) and MBN is that the former are designed for reducing the computational and storage complexities, while MBN is not developed for this purpose. MBN aims at providing a simple way to overcome the difficulty of density estimation in a continuous space. Hence, it generates larger codebooks than common vector quantization methods and does not transform the sparse codes to compact binary codes.

6.5. Product of experts

PoE aims to combine multiple individual models by multiplying them, where the individual models have to be a bit more complicated and each contains one or more hidden variables (Hinton, 2002). Its general probability framework is:

$$p(\mathbf{x}) = \frac{\prod_{v=1}^{V} g_v(\mathbf{x})}{\sum_{\mathbf{x}'} \prod_{v=1}^{V} g_v(\mathbf{x}')} \tag{12}$$

where $\mathbf{x}'$ indexes all possible vectors in the data space, and g_v is called an *expert*. A major merit of PoE is that, a function that can be fully expressed by a *mixture of experts* with N experts (i.e. mixtures), such as Gaussian mixture model or k-means clustering, can be expressed compactly by a PoE with only $\log_2 N$ experts at the minimum, with the expense of the optimization difficulty of the partition function $\sum_{\mathbf{x}'} \prod_{v=1}^{V} g_v(\mathbf{x}')$ which consists of an exponentially large number of components.

The connection between MBN and PoE is as follows:

Theorem 2. *Each layer of MBN is a PoE that does not need to optimize the partition function.*

Proof. See Appendix B for the proof. □

6.6. Sparse coding

Given a learned dictionary $\mathbf{W}$, sparse coding typically aims to solve $\min_{\mathbf{h}_i} \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{W}\mathbf{h}_i\|_2^2 + \lambda \|\mathbf{h}_i\|_1^1$, where $\|\cdot\|_q$ represents ℓ_q-norm, $\mathbf{h}_i$ is the sparse code of the data point $\mathbf{x}_i$, and λ is a hyperparameter controlling the sparsity of $\mathbf{h}_i$. Each column of $\mathbf{W}$ is called a basis vector.

To understand the connection between MBN and sparse coding, we may view λ as a hyperparameter that controls the number of clusterings. Specifically, if we set $\lambda = 0$, it is likely that $\mathbf{h}_i$ contains only one nonzero element. Intuitively, we can understand it as that we use only one clustering to learn a sparse code. A good value of λ can make a small part of the elements of $\mathbf{h}_i$ nonzero. This choice approximates to the method of partitioning the dictionary to several (probably overlapped) subsets and then grouping the basis vectors in each subset to a base clustering. Motivated by the above intuitive analysis, we introduce the connection between sparse coding and MBN formally as follows:

Theorem 3. *The ℓ_1-norm sparse coding is a convex relaxation of the building block of MBN when given the same dictionary.*

Proof. See Appendix C for the proof. □

6.7. Unsupervised deep learning

Learning abstract representations by deep networks is a recent trend. From the geometric point of view, the abstract representations are produced by reducing larger and larger local variations of data from bottom up in the framework of trees that are built on data spaces.[5] For example, convolutional neural network merges child nodes by pooling. Hierarchical Dirichlet process (Teh et al., 2005) builds trees whose father nodes generate child nodes according to a prior distribution. DBN (Hinton & Salakhutdinov, 2006) merges child nodes by reducing the number of the nonlinear units gradually from bottom up. Subspace tree (Wichert & Moreira, 2015) merges nodes by reducing the dimensions of the subspaces gradually. PCANet (Chan et al., 2015) merges nodes by reducing the output dimensions of the local PCA associated with its patches gradually. Our MBN merges nodes by reducing the number of the randomly sampled centroids gradually.

A fundamental difference between the methods is how to build effective local coordinate systems in each layer. To our knowledge, MBN is a simple method and needs little assumption and prior knowledge. It is only more complicated than random projection which is, to our knowledge, not an effective method for unsupervised deep learning to date. Moreover, MBN is the only method working in discrete feature spaces, and it works well.

7. Conclusions

In this paper, we have proposed multilayer bootstrap network for nonlinear dimensionality reduction. MBN has a novel network structure that each expert is a k-centroids clustering whose centroids are randomly sampled data points with randomly sampled features; the network is gradually narrowed from bottom up.

MBN is composed of two novel components: (i) each layer of MBN is a nonparametric density estimator by random resampling. It estimates the density of data correctly without any model assumption. It is exponentially more powerful than a single k-centroids clustering. Its estimation error is proven to be small and controllable. (ii) The network is a deep ensemble model. It essentially reduces the nonlinear variations of data by building a vast number of hierarchical trees on the data space. It can be trained as many layers as needed with both large-scale and small-scale data.

MBN performs robustly with a wide range of parameter settings. Its time and storage complexities scale linearly with the size of training data. It supports parallel computing naturally. Empirical results demonstrate its efficiency at the training stage and its effectiveness in density estimation, data visualization, clustering, and document retrieval. We have also demonstrated that the high computational complexity of MBN at the test stage can be eliminated by the compressive MBN—a framework of unsupervised model compression based on neural networks.

A problem left is on the selection of parameter δ which controls the network structure. Although the performance of MBN with $\delta = 0.5$ is good, there is still a large performance gap between $\delta = 0.5$ and the best δ. Hence, how to select δ automatically is an important problem.

Acknowledgments

The author thanks Prof. DeLiang Wang for providing his computing resources. The author also thanks the action editor and reviewers for their comments which improved the quality of the paper. This work was supported by the National Natural Science Foundation of China under Grant 61671381.

[5] Some recent unsupervised deep models for data augmentation are out of the discussion, hence we omit them here.

Appendix A. Complexity analysis

Theorem 4. *The computational and storage complexities of MBN are:*

$$O_{time} = O(dskVn) \tag{A.1}$$

$$O_{storage} = O((ds + V)n + kV + kds) \tag{A.2}$$

respectively at the bottom layer, and are:

$$O_{time} = O\left(kV^2n\right) \tag{A.3}$$

$$O_{storage} = O(2V(n + k)) \tag{A.4}$$

respectively at other layers, where d is the dimension of the original feature, s is the sparsity of the data (i.e., the ratio of the non-zero elements over all elements).

If MBN is not used for prediction, then MBN needs not to be saved, which further reduces the storage complexity to $O((ds + V)n)$ at the bottom layer and $O(2Vn)$ at other layers.

Proof. Due to the length limitation of the paper, we omit the simple proof. □

Fortunately, as shown in Fig. A.15, the empirical time complexity grows with $O(V)$ instead of $O(V^2)$. The only explanation is that the input data is sparse. Specifically, the multiplication of two sparse matrices only considers the element-wise multiplication of two elements that are both nonzero, as a result, when the input data is sparse, one factor V is offset by the sparsity factor s. The empirical time and storage complexities with other parameters are consistent with our theoretical analysis. We omit the results here.

Note that, our default parameter $k_1 = 0.5n$ makes the time complexity scale squarely with the size of the data set n. To reduce the computational cost, we usually set k_1 manually to a small value irrelevant to n as what we have done for visualizing the full MNSIT and retrieving the RCV1 documents in Section 5.

Appendix B. Proof of Theorem 2

The optimization objective of k-means clustering can be written as:

$$\begin{aligned} &\max_{\mathbf{W},\mathbf{h}} g(\mathbf{x}|\mathbf{W},\mathbf{h}) \\ &= \max_{\mathbf{W},\mathbf{h}} \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x} - \mathbf{W}^T\mathbf{h}\|_2^2\right) \\ &\text{subject to} \quad \mathbf{h} \text{ is a one-hot code} \end{aligned} \tag{B.1}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_k]$ is the weight matrix whose columns are centroids, $\mathbf{h}$ is a vector of hidden variables, and the covariance matrix of the clustering is $\sigma^2\mathbf{I}$ with $\mathbf{I}$ being the identity matrix and $\sigma \to 0$ (Bishop et al., 2006).

Substituting $g(\mathbf{x}|\mathbf{W},\mathbf{h}) = \exp\left(-\|\mathbf{x} - \mathbf{W}^T\mathbf{h}\|_2^2/2\sigma^2\right)$ to Eq. (12) gets:

$$\begin{aligned} &p(\mathbf{x}|\{W_v,\mathbf{h}_v\}_{v=1}^V) \\ &= \frac{\prod_{v=1}^V \exp\left(-\|\mathbf{x} - \mathbf{W}_v^T\mathbf{h}_v\|_2^2/2\sigma^2\right)}{\sum_{\mathbf{x}'}\prod_{v=1}^V \exp\left(-\|\mathbf{x}' - \mathbf{W}_v^T\mathbf{h}_v\|_2^2/2\sigma^2\right)} \\ &= \frac{\exp\left(-\sum_{v=1}^V \|\mathbf{x} - \mathbf{W}_v^T\mathbf{h}_v\|_2^2/2\sigma^2\right)}{\sum_{\mathbf{x}'}\exp\left(-\sum_{v=1}^V \|\mathbf{x}' - \mathbf{W}_v^T\mathbf{h}_v\|_2^2/2\sigma^2\right)} \end{aligned} \tag{B.2}$$

where we assume that all experts have the same covariance matrix $\sigma^2\mathbf{I}$. Maximizing the likelihood of Eq. (B.2) is equivalent to the following problem:

$$\begin{aligned} \mathcal{J}_{\text{PoE}} &= \min_{\{W_v,\mathbf{h}_v\}_{v=1}^V} -\log p(\mathbf{x}|\{W_v,\mathbf{h}_v\}_{v=1}^V) \\ &= \min_{\{W_v,\mathbf{h}_v\}_{v=1}^V} \frac{1}{2\sigma^2}\sum_{v=1}^V \|\mathbf{x} - \mathbf{W}_v^T\mathbf{h}_v\|_2^2 \\ &\quad + \log\sum_{\mathbf{x}'}\exp\left(-\frac{1}{2\sigma^2}\sum_{v=1}^V \|\mathbf{x}' - \mathbf{W}_v^T\mathbf{h}_v\|_2^2\right) \\ &\propto \min_{\{W_v,\mathbf{h}_v\}_{v=1}^V} \sum_{v=1}^V \|\mathbf{x} - \mathbf{W}_v^T\mathbf{h}_v\|_2^2 \\ &\quad + \log\left(\sum_{\mathbf{x}'}\exp\left(-\frac{1}{2\sigma^2}\sum_{v=1}^V \|\mathbf{x}' - \mathbf{W}_v^T\mathbf{h}_v\|_2^2\right)\right)^{2\sigma^2} \\ &\quad \text{subject to} \quad \mathbf{h}_v \text{ is a one-hot code.} \end{aligned} \tag{B.3}$$

Because $\sigma \to 0$, problem (B.3) can be rewritten as:

$$\begin{aligned} \mathcal{J}_{\text{PoE}} &\propto \min_{\{W_v,\mathbf{h}_v\}_{v=1}^V} \sum_{v=1}^V \|\mathbf{x} - \mathbf{W}_v^T\mathbf{h}_v\|_2^2 \\ &\quad + \log\left(\max_{\mathbf{x}'}\left(\exp\left(-\frac{1}{2\sigma^2}\sum_{v=1}^V \|\mathbf{x}' - \mathbf{W}_v^T\mathbf{h}_v\|_2^2\right)\right)\right)^{2\sigma^2} \\ &= \min_{\{W_v,\mathbf{h}_v\}_{v=1}^V} \sum_{v=1}^V \|\mathbf{x} - \mathbf{W}_v^T\mathbf{h}_v\|_2^2 - \min_{\mathbf{x}'}\sum_{v=1}^V \|\mathbf{x}' \\ &\quad - \mathbf{W}_v^T\mathbf{h}_v\|_2^2 \\ &\quad \text{subject to} \quad \mathbf{h}_v \text{ is a one-hot code.} \end{aligned} \tag{B.4}$$

Because $\mathbf{x}'$ can be any possible vector in the data space, we have $\min_{\mathbf{x}'}\sum_{v=1}^V \|\mathbf{x}' - \mathbf{W}_v^T\mathbf{h}_v\|_2^2 = 0$. Eventually, the optimization objective of PoE is:

$$\begin{aligned} &\mathcal{J}_{\text{PoE}} \propto \min_{\{W_v,\mathbf{h}_v\}_{v=1}^V} \sum_{v=1}^V \|\mathbf{x} - \mathbf{W}_v^T\mathbf{h}_v\|_2^2 \\ &\text{subject to} \quad \mathbf{h}_v \text{ is a one-hot code} \end{aligned} \tag{B.5}$$

which is irrelevant to the partition function.

It is clear that problem (B.5) is the optimization objective of an ensemble of k-means clusterings. When we assign W_v by random sampling, then problem (B.5) becomes:

$$\begin{aligned} &\mathcal{J}_{\text{PoE}} \propto \min_{\{\mathbf{h}_v\}_{v=1}^V} \sum_{v=1}^V \|\mathbf{x} - \mathbf{W}_v^T\mathbf{h}_v\|_2^2 \\ &\text{subject to} \quad \mathbf{h}_v \text{ is a one-hot code} \end{aligned} \tag{B.6}$$

which is the building block of MBN. Theorem 2 is proved.

Appendix C. Proof of Theorem 3

Each layer of MBN maximizes the likelihood of the following equation:

$$p(\mathbf{x}) = \prod_{v=1}^V g_v(\mathbf{x}) \tag{C.1}$$

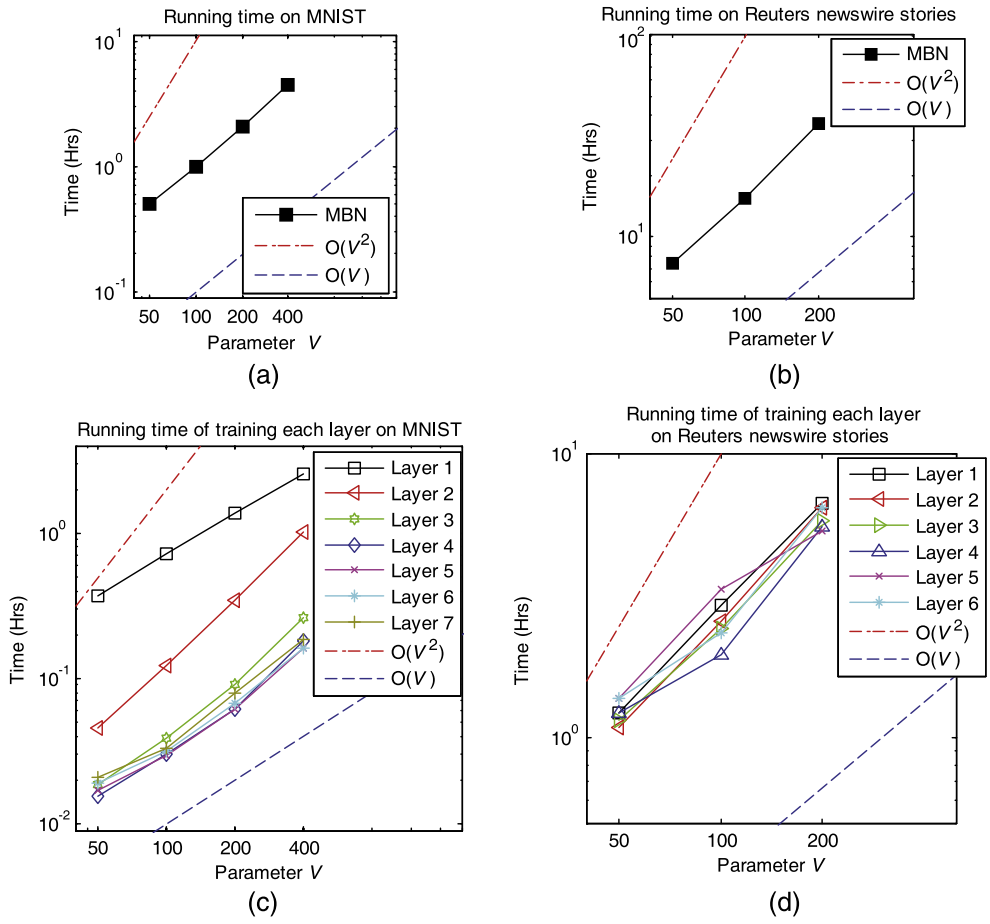

Fig. A.15. Time complexity of MBN with respect to parameter V.

where $g_v(\mathbf{x})$ is a k-means clustering with the squared error as the similarity metric:

$$g_v(\mathbf{x}) = \mathcal{MN}\left(\mathbf{x}; \mathbf{W}_v\mathbf{h}_v, \sigma^2\mathbf{I}\right) \tag{C.2}$$

subject to $\mathbf{h}_v$ is a one-hot code

where $\mathcal{MN}$ denotes the multivariate normal distribution, $\mathbf{W}_v = [\mathbf{w}_{v,1}, \ldots, \mathbf{w}_{v,k}]$ is the weight matrix whose columns are centroids, $\mathbf{I}$ is the identity matrix, and $\sigma \to 0$.

Given a data set $\{\mathbf{x}_i\}_{i=1}^n$. We assume that $\{\mathbf{W}_v\}_{v=1}^V$ are fixed, and take the negative logarithm of Eq. (C.1):

$$\min_{\{\{\mathbf{h}_{v,i}\}_{i=1}^n\}_{v=1}^V} \sum_{v=1}^{V}\sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{W}_v\mathbf{h}_{v,i}\|_2^2, \tag{C.3}$$

subject to $\mathbf{h}_{v,i}$ is a one-hot code.

If we denote $\mathbf{W} = [\mathbf{W}_1, \ldots, \mathbf{W}_V]$ and further complement the head and tail of $\mathbf{h}_{v,i}$ with multiple zeros, denoted as $\mathbf{h}'_{v,i}$, such that $\mathbf{W}\mathbf{h}'_{v,i} = \mathbf{W}_v\mathbf{h}_{v,i}$, we can rewrite Eq. (C.3) to the following equivalent problem:

$$\min_{\{\{\mathbf{h}_{v,i}\}_{i=1}^n\}_{v=1}^V} \sum_{v=1}^{V}\sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{W}\mathbf{h}'_{v,i}\|_2^2, \tag{C.4}$$

subject to $\mathbf{h}_{v,i}$ is a one-hot code.

It is an integer optimization problem that has an integer matrix variable $\mathbf{H}'_v = [\mathbf{h}'_{v,1}, \ldots, \mathbf{h}'_{v,n}]$. Suppose there are totally $|\mathcal{H}'_v|$ possible solutions of $\mathbf{H}'_v$, denoted as $\mathbf{H}'_{v,1}, \ldots, \mathbf{H}'_{v,|\mathcal{H}'_v|}$, we first relax Eq. (C.4) to a convex optimization problem by constructing a *convex hull* (Boyd & Vandenberghe, 2004) on $\mathbf{H}'_v$:

$$\min_{\left\{\{\mu_{v,k}\}_{k=1}^{|\mathcal{H}'_v|}\right\}_{v=1}^V} \sum_{v=1}^{V}\sum_{i=1}^{n} \left\|\mathbf{x}_i - \mathbf{W}\left(\sum_{k=1}^{|\mathcal{H}'_v|}\mu_{v,k}\mathbf{h}'_{v,k,i}\right)\right\|_2^2 \tag{C.5}$$

$$\text{subject to } 0 \le \mu_{v,k} \le 1, \sum_{k=1}^{|\mathcal{H}'_v|}\mu_{v,k} = 1, \quad \forall v = 1, \ldots, V.$$

Because Eq. (C.5) is a convex optimization problem, according to Jensen's inequality, the following problem learns a lower bound of Eq. (C.5):

$$\min_{\left\{\{\mu_{v,k}\}_{k=1}^{|\mathcal{H}'_v|}\right\}_{v=1}^V} V\sum_{i=1}^{n} \left\|\mathbf{x}_i - \mathbf{W}\mathbf{h}''_i\right\|_2^2 \tag{C.6}$$

$$\text{subject to } 0 \le \mu_{v,k} \le 1, \sum_{k=1}^{|\mathcal{H}'_v|}\mu_{v,k} = 1, \quad \forall v = 1, \ldots, V$$

where $\mathbf{h}''_i = \frac{1}{V}\sum_{v=1}^{V}\sum_{k=1}^{|\mathcal{H}'_v|}\mu_{v,k}\mathbf{h}'_{v,k,i}$ with $\mu_{v,k}$ as a variable.

Recalling the definition of sparse coding given a fixed dictionary $\mathbf{W}$, we observe that Eq. (C.6) is a special form of sparse coding with more strict constraints on the format of sparsity.

Therefore, given the same dictionary $\mathbf{W}$, each layer of MBN is a distributed sparse coding that is lower bounded by the common ℓ_1-norm sparse coding. When we discard the expectation–maximization optimization of each k-means clustering (i.e., dictionary learning) but only preserve the default initialization

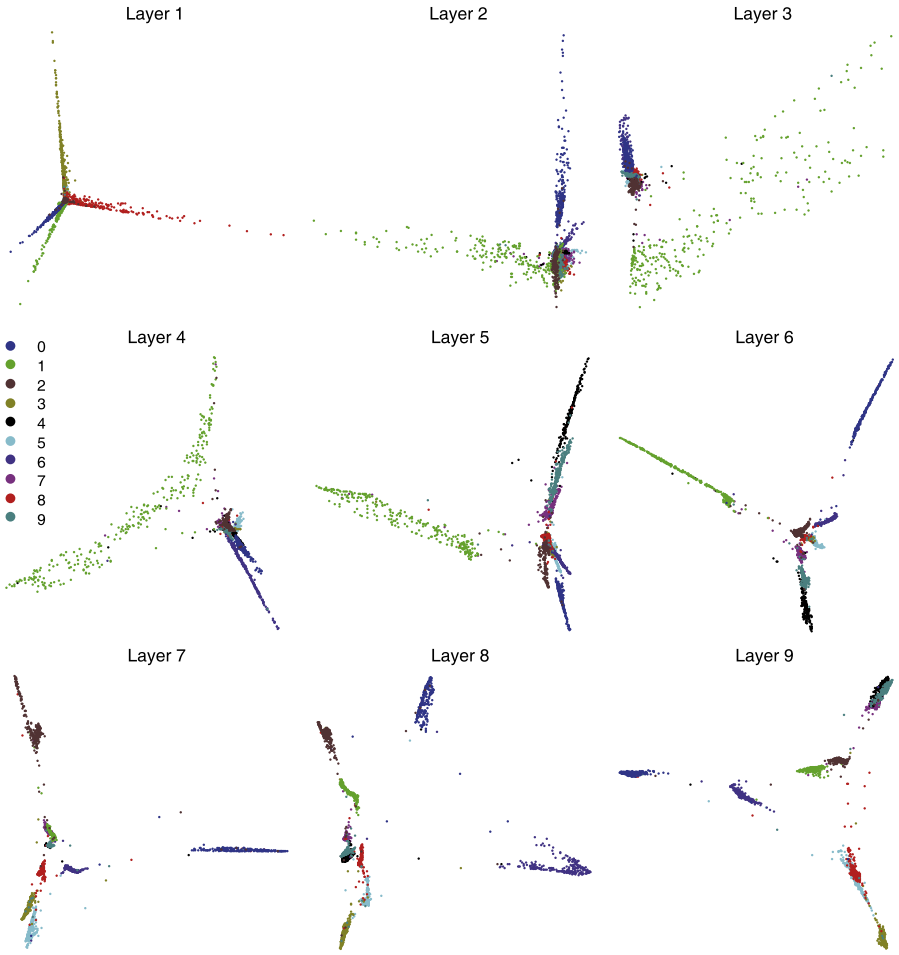

Fig. D.16. Visualizations of MNIST produced by MBN at different layers. This figure is a supplement to Fig. 9.

method — random sampling, Eq. (C.1) becomes the building block of MBN. Given the same dictionary, the ℓ_1-norm-regularized sparse coding is a convex relaxation of the building block of MBN. Theorem 3 is proved.

Appendix D. Visualizations produced by intermediate layers

See Fig. D.16.

References

Belkin, M., & Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, *15*(6), 1373–1396.

Bishop, C. M., et al. (2006). *Pattern recognition and machine learning*. Springer New York:.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*, 993–1022.

Boyd, S. P., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Canu, S., Grandvalet, Y., Guigue, V., & Rakotomamonjy, A. (2005). SVM and kernel methods Matlab toolbox. http://asi.insa-rouen.fr/enseignants/~arakoto/toolbox/index.html.

Chan, T.-H., Jia, K., Gao, S., Lu, J., Zeng, Z., & Ma, Y. (2015). Pcanet: A simple deep learning baseline for image classification? *IEEE Transactions on Image Processing*, *24*(12), 5017–5032.

Chang, H., & Yeung, D.-Y. (2008). Robust path-based spectral clustering. *Pattern Recognition*, *41*(1), 191–203.

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., & Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391–407.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Multiple classifier system* (pp. 1–15). Cagliari, Italy: Springer.

Dudoit, S., & Fridlyand, J. (2003). Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*, *19*(9), 1090–1099.

Efron, B. (1979). Bootstrap methods: another look at the jackknife. *The Annals of Statistics*, *7*(1), 1–26.

Efron, B., & Tibshirani, R. (1993). *An introduction to the bootstrap*. CRC press.

Fern, X. Z., & Brodley, C. E. (2003). Random projection for high dimensional data clustering: A cluster ensemble approach. In *Proceedings of the 20th international conference on machine learning, Vol. 3* (pp. 186–193).

Fred, A. L., & Jain, A. K. (2005). Combining multiple clusterings using evidence accumulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(6), 835–850.

Freedman, D., Pisani, R., & Purves, R. (2007). *Statistics (4th edition)*. Norton, New York.

Freund, Y., & Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the 2nd european conference on computational learning theory* (pp. 23–37), Barcelona, Spain.

Friedman, J., Hastie, T., Tibshirani, R., et al. (2000). Additive logistic regression: A statistical view of boosting (with discussion and a rejoinder by the authors). *The Annals of Statistics*, *28*(2), 337–407.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., et al. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, *286*(5439), 531–537.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning*. Springer.

He, X., & Niyogi, X. (2004). Locality preserving projections. In *Advances in neural information processing systems 17, Vol. 16* (pp. 153–160), Vancouver, British Columbia, Canada.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of 2016 IEEE international conference on computer vision and pattern recognition* (pp. 770–778), Las Vegas, NV.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, *14*(8), 1771–1800.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-. R, Jaitly, N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, *29*(3), 82–97.

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504–507.

Jain, A. K., & Law, M. H. (2005). Data clustering: A user's dilemma. *Lecture Notes in Computer Science*, *3776*, 1–10.

Jegou, H., Douze, M., & Schmid, C. (2011). Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33*(1), 117–128.

Lanckriet, G. R. G., Cristianini, N., Bartlett, P., Ghaoui, L. E., & Jordan, M. I. (2004). Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, *5*, 27–72.

Lecun, Y., Cortes, C., & Burges, C. J. C. (2004). The mnist database of handwritten digits. http://yann.lecun.com/exdb/mnist/index.html.

Lewis, D. D., Yang, Y., Rose, T. G., & Li, F. (2004). RCV1: a new benchmark collection for text categorization research. *Journal of Machine Learning Research*, *5*, 361–397.

McCallum, A. (1998). Rainbow. http://www.cs.cmu.edu/~mccallum/bow/rainbow.

Ng, A. Y., Jordan, M. I., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems 14* (pp. 849–856), Vancouver, British Columbia, Canada.

Nie, F., Zeng, Z., Tsang, I. W., Xu, D., & Zhang, C. (2011). Spectral embedded clustering: A framework for in-sample and out-of-sample spectral clustering. *IEEE Transactions on Neural Networks*, *22*(11), 1796–1808.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, *2*(11), 559–572.

Roweis, S. T. (1998). EM algorithms for PCA and SPCA. In *Advances in neural information processing systems 10* (pp. 626–632), Denver, CO.

Roweis, S. T., & Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*(5500), 2323–2326.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, *5*(2), 197–227.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85–117.

Schölkopf, B., Smola, A., & Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, *10*(5), 1299–1319.

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 888–905.

Strehl, A., & Ghosh, J. (2003). Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, *3*, 583–617.

Tao, D., Tang, X., Li, X., & Wu, X. (2006). Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(7), 1088–1099.

Teh, Y. W., Jordan, M. I., Beal, M. J., & Blei, D. M. (2005). Sharing clusters among related groups: Hierarchical Dirichlet processes. In *Advances in neural information processing systems 17* (pp. 1385–1392).

Tenenbaum, J. B., De Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*(5500), 2319–2323.

Van der Maaten, L., & Hinton, G. E. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, *9*(85), 2579–2605.

Vega-Pons, S., & Ruiz-Shulcloper, J. (2011). A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, *25*(03), 337–372.

Wang, D. L., & Chen, J. (2017). Supervised speech separation based on deep learning: an overview. arXiv preprint arXiv:1708.07524.

Wang, Q., Qin, Z., Nie, F., & Yuan, Y. (2017). Convolutional 2d lda for nonlinear dimensionality reduction. In *Proceedings of the 26th international joint conference on artificial intelligence* (pp. 2929–2935), Melbourne, Australia.

Wichert, A. (2012). Product quantization for vector retrieval with no error. In *ICEIS (1)* (pp. 87–92).

Wichert, A., & Moreira, C. (2015). On projection based operators in lp space for exact similarity search. *Fundamenta Informaticae*, *136*(4), 461–474.

Wikipedia (2017). Frequentist probability. https://en.wikipedia.org/wiki/Frequentist_probability.

Xing, E. P., Jordan, M. I., Russell, S., & Ng, A. (2002) Distance metric learning with application to clustering with side-information. In *Advances in nueral information processing systems 15* (pp. 505–512), Vancouver, British Columbia, Canada.

Yan, S., Xu, D., Zhang, B., Zhang, H.-J., Yang, Q., & Lin, S. (2007). Graph embedding and extensions: a general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(1), 40–51.

Zahn, C. T. (1971). Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, *100*(1), 68–86.

Zheng, L., Li, T., & Ding, C. (2010). Hierarchical ensemble clustering. In *Proceedings of 2010 IEEE international conference on data mining* (pp. 1199–1204), Sydney, Australia.

Zhou, Z.-H., & Feng, J. (2017). Deep Forest: Towards an alternative to deep neural networks. In *Proceedings of the 26th international joint conference on artificial intelligence* (pp. 3553–3559), Melbourne, Australia.