ISSN 1060-992X, Optical Memory and Neural Networks, 2016, Vol. 25, No. 3, pp. 127–141. © Allerton Press, Inc., 2016.

# The Nature of Unsupervised Learning in Deep Neural Networks: A New Understanding and Novel Approach

**V. Golovko[a, b, *], A. Kroshchanka[a], and D. Treadwell[c]**

[a] *Brest State Technical University, Belarus*

[b] *National Research Nuclear University (MEPHI), Moscow, Russia*

[c] *5339 Iron Horse Pkwy, Dublin CA 94568, USA*

**e-mail: gva@bstu.by*

Received July 12, 2016

**Abstract**—Over the last decade, the deep neural networks are a hot topic in machine learning. It is breakthrough technology in processing images, video, speech, text and audio. Deep neural network permits us to overcome some limitations of a shallow neural network due to its deep architecture. In this paper we investigate the nature of unsupervised learning in restricted Boltzmann machine. We have proved that maximization of the log-likelihood input data distribution of restricted Boltzmann machine is equivalent to minimizing the cross-entropy and to special case of minimizing the mean squared error. Thus the nature of unsupervised learning is invariant to different training criteria. As a result we propose a new technique called "REBA" for the unsupervised training of deep neural networks. In contrast to Hinton's conventional approach to the learning of restricted Boltzmann machine, which is based on linear nature of training rule, the proposed technique is founded on nonlinear training rule. We have shown that the classical equations for RBM learning are a special case of the proposed technique. As a result the proposed approach is more universal in contrast to the traditional energy-based model. We demonstrate the performance of the REBA technique using well-known benchmark problem. The main contribution of this paper is a novel view and new understanding of an unsupervised learning in deep neural networks.

*Keywords*: deep neural networks, deep learning, restricted Boltzmann machine, data visualization, machine learning

**DOI:** 10.3103/S1060992X16030073

## INTRODUCTION

Deep learning is a revolutionary technique in the domain of machine learning and has been successfully applied to many problems in artificial intelligence, namely speech recognition, computer vision, natural language processing, data visualization, etc. [1–14]. Deep learning allows computational model, which consist of multiple processing layers to learn representations of data with multiple levels of abstraction. Deep neural network (DNN) can perform a deep hierarchical transformation of the input data, and as a result have been found to have better performance and more representational power than shallow neural networks. This kind of neural network has been investigated in many studies [1–14].

This paper deals with an unsupervised learning technique for restricted Boltzmann machine (RBM), which can be applied for the training of deep neural networks. The conventional approach to unsupervised training the RBM uses an energy-based model and is based on maximization of the log-likelihood input data distribution using gradient descent approach. In this paper we consider the unsupervised deep learning from another point of view, which provides a deeper understanding of the nature of unsupervised learning in deep neural networks. First of all we use two training criteria, namely square error and cross-entropy, instead of energy-based technique. Next, we present the RBM as PCA or auto-encoder neural network, which consist of three layers: visible, hidden and visible. Finally, the Gibbs sampling in order to define mean square error and cross-entropy loss function is used. As a result we have proved that maximization of the log-likelihood input data distribution of restricted Boltzmann machine is equivalent to minimizing the cross-entropy and to special case of minimizing the mean squared error. It follows that the nature of unsupervised learning is invariant to the different learning criteria. We propose also a new technique called "REBA" for RBM learning, and this approach in contrast to an energy-based model is based

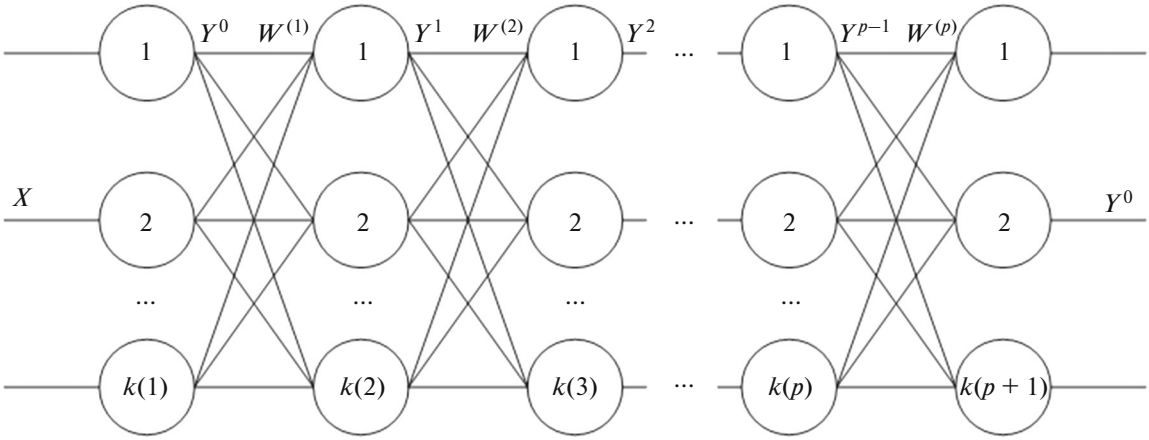


**Fig. 1.** Deep perceptron.

on the minimization of the reconstruction mean square error in the hidden and last visible layers of the RBM. We will show that classical equations for RBM training are a special case of the proposed technique. Therefore the proposed approach is more universal in contrast to the traditional energy-based model.

The rest of the paper is organized as follows. Section 1 introduces the conventional approach for restricted Boltzmann machine training based on an energy model. In Section 2 we propose the novel techniques for inference of RBM training rules. Section 3 demonstrates the results of experiments, and finally we give our conclusion.

## 1. RELATED WORKS

Let us consider the related works in this domain [1–14]. As mentioned earlier, the deep neural network has been the hottest topic in the domain of machine learning in recent years. Deep neural networks can accomplish a deep hierarchical representation of their input data. The first layer can extract low-level features; the second layer can extract higher level features, and so on. There are different kinds of deep neural networks: deep belief neural networks, deep perceptron, deep convolutional neural networks, deep recurrent neural networks, deep auto-encoder, deep R-CNN and so on. If we take, for instance, the perceptron, it will be deep if more than two hidden layer is used. The analogical situation is for deep belief and deep recurrent neural networks. It should be noted that the training rules are identical for different kind of deep neural networks. Therefore we will take the many-layered perceptron as a deep neural network in order to investigate deep learning rules. As already mentioned the DNN consists of many hidden layers and can perform a deep hierarchical representation of the input data as shown in Fig. 1.

Let's $p$ is the number of deep perceptron layers, except of first layer. Then the common number of synaptic weights (weights and thresholds) is defined by

$$V = \sum_{i=1}^{p} k(i)k(i+1). \tag{1}$$

The $j$-th output unit for $k$-th layer is given by

$$y_j^k = F(S_j^k), \tag{2}$$

$$S_j^k = \sum_{i=1} \omega_{ij}^k y_i^{k-1} + T_j^k, \tag{3}$$

where $F$ is the activation function, $S_j^k$ is the weighted sum of the $j$-th unit, $\omega_{ij}^k$ is the weight from the $i$-th unit of the $(k-1)$-th layer to the $j$-th unit of the $k$-th layer, and $T_j^k$ is the threshold of the $j$-th unit.

For the first layer

$$y_i^0 = x_i. \tag{4}$$

In the common case we can write that

$$Y^k = F(S^k) = F(W^k Y^{k-1} + T^k), \tag{5}$$

where $W$ is a weight matrix, $Y^{k-1}$ is the output vector for $(k-1)$-th layer and $T^k$ is the threshold vector.

It should also be noted that the output of a DNN is often defined using the softmax function:

$$y_j^F = \text{softmax}(S_j) = \frac{e^{S_j}}{\sum_l e^{S_l}}. \tag{6}$$

There exist the two main techniques for learning of deep neural networks: learning with pre-training using a greedy layer-wise approach and stochastic gradient descent approach (SGD) with rectified linear unit (ReLU) transfer function [6].

The learning with pre-training consists of two stages [1–4]. The first stage is the pre-training of neural network using greedy layer-wise approach. This procedure is started from the first layer and performed in unsupervised manner. The second one is fine-tuning all of parameters of neural network using back-propagation algorithm.

The training with stochastic gradient descent approach is the online or mini-batch learning using conventional backpropagation algorithm [14]. The use of ReLU activation function can help to avoid of vanishing gradient problem, poor local minima and unstable gradient problem due to the greater linearity of such kind of activation function [6].

At present the following paradigm for DNN learning is used. If training data set is large then SGD with ReLU is used for deep neural network learning. Otherwise pre-training and fine-tuning is applied. So, for instance, for smaller data sets, unsupervised pre-training helps to prevent overfitting [6]. As stated in paper [6]:"Although at present the supervised training with ReLU is used mainly for deep neural networks learning, we expect unsupervised learning to become far more important in the longer term. Human and animal learning is largely unsupervised: we discover the structure of the world by observing it, not by being told the name of every object". Thus the unsupervised learning has great perspectives. Therefore let's consider this problem in more detail.

The most important stage of deep neural network training is the pre-training of each layer of the DNN in unsupervised manner. There exist two main techniques for DNN pre-training. As a rule the DNN pre-training is based on either the restricted Boltzmann machine (RBM) or auto-encoder approach [1, 9, 12]. In accordance with the greedy layer-wise training procedure, in the beginning the first layer of the DNN is trained using RBM or auto-encoder training rule and its parameters are fixed. After this the next layer is trained, and so on. As a result a good initialization of the neural network is achieved and we can then use back-propagation or the wake-sleep algorithm for fine tuning the parameters of the whole neural network.

At present the approach based on RBM is more popular for DNN pre-training. Therefore we will consider the DNN pre-training technique based on the restricted Boltzmann machine. In this case the deep neural network can be represented as a set of restricted Boltzmann machines and the RBM is the main building block of deep neural network. The traditional approach to RBM training was proposed by G. Hinton and is based on an energy model and training rules which take into account only a linear nature of neural units, as will be shown in Section 3.

Let's consider the conventional restricted Boltzmann machine, which consists of two layers of units: visible and hidden (Fig. 2).

The restricted Boltzmann machine can represent any discrete distribution if enough hidden units are used [9]. The layers of neural units are connected by bidirectional weights $W$. Often the binary units are used [1–4]. The RBM is a stochastic neural network and the states of visible and hidden units are defined using a probabilistic version of the sigmoid activation function:

$$p(y_j|x) = \frac{1}{1+e^{-S_j}}, \quad S_j = \sum_{i=1}^{n} w_{ij} x_i + T_j, \tag{7}$$

$$p(x_i|x) = \frac{1}{1+e^{-S_i}}, \quad S_i = \sum_{j=1}^{m} w_{ij} y_j + T_i. \tag{8}$$

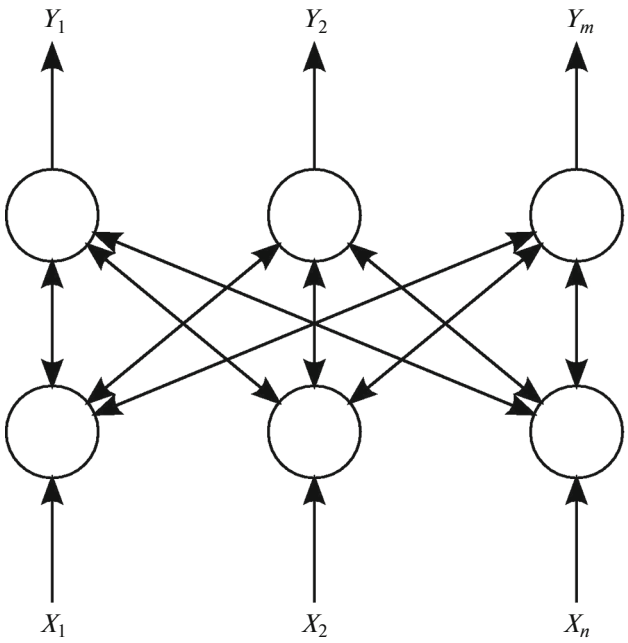


**Fig. 2.** Restricted Boltzmann machine.

It should be noted that the visible and hidden units are conditionally independent:

$$P(x|y) = \prod_{i=1}^{n} P(x_i|y),$$
$$P(y|x) = \prod_{j=1}^{m} P(y_j|x). \tag{9}$$

As can be seen, the states of all the units are obtained through a probability distribution. The hidden units of the RBM are feature detectors which capture the regularities of the input data. The key idea of RBM training is to reproduce as closely as possible the distribution of the input data using the states of the hidden units. This is equivalent to maximizing the likelihood of the input data distribution $P(x)$ by the modification of synaptic weights using the gradient of the log probability of the input data. Then the modification of synaptic weights is defined by

$$\omega_{ij}(t+1) = \omega_{ij}(t) + \alpha \frac{\partial \operatorname{Ln} P(x)}{\partial \omega_{ij}(t)}. \tag{10}$$

Using this approach Hinton proposed to use contrastive divergence (CD) technique for RBM learning [1]. It is based on Gibbs sampling. As a result we can obtain the RBM training rules. In the case of CD-1 the training rule is defined as

$$\begin{aligned} \omega_{ij}(t+1) &= \omega_{ij}(t) + \alpha(x_i(0)y_j(0) - x_i(1)y_j(1)), \\ T_i(t+1) &= T_i(t) + \alpha(x_i(0) - x_i(1)), \\ T_j(t+1) &= T_j(t) + \alpha(y_j(0) - y_j(1)). \end{aligned} \tag{11}$$

In case of CD-k

$$\begin{aligned} \omega_{ij}(t+1) &= \omega_{ij}(t) + \alpha(x_i(0)y_j(0) - x_i(k)y_j(k)), \\ T_i(t+1) &= T_i(t) + \alpha(x_i(0) - x_i(k)), \\ T_j(t+1) &= T_j(t) + \alpha(y_j(0) - y_j(k)). \end{aligned} \tag{12}$$

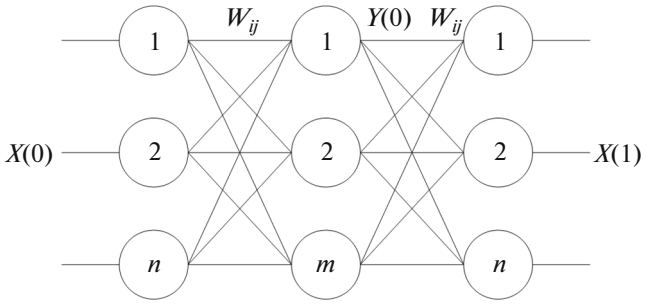


**Fig. 3.** Unfolded representation of RBM.

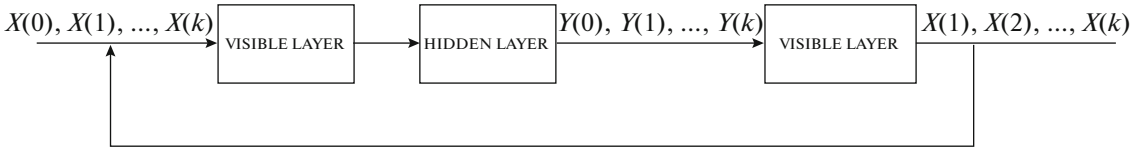


**Fig. 4.** Gibbs sampling.

Here $\alpha$ is the learning rate. In this equations the first term in the training rule denotes the data distribution at the time $t = 0$ and the second term is the model distribution of reconstructed states at the step $t = k$. Therefore the CD-k procedure can be represented as follows:

$$x(0) \to y(0) \to x(1) \to y(1) \to \ldots. \to x(k) \to y(k). \tag{13}$$

As can be seen from these equations, the training rules for RBMs are essentially minimizing the difference between the original data and the synthesized samples from model. The synthesized data can be obtained using a Gibbs sampling algorithm.

Training an RBM is based on presenting a training sample to the visible units, then using the CD-$k$ procedure to compute the binary states of the hidden units $p(y|x)$, sampling the visible units (reconstructed states) $p(x|y)$, and so on. After performing these iterations the weights and biases of the restricted Boltzmann machine are updated. Then we stack on another hidden layer to train a new RBM. This approach is applied to all layers of the deep neural network (greedy layer-wise training). As a result of this unsupervised pre-training we can obtain a good initialization of the neural network. Finally, supervised fine-tuning of the whole neural network is performed. As mentioned before the energy-based model leads to the linear nature of neural units as will be shown in the next section.

## 2. A NEW UNDERSTANDING OF UNSUPERVISED LEARNING IN RBM

In this section we will consider the restricted Boltzmann machine from another point of view, namely as auto-encoder or the PCA neural network. We will use two training criteria in order to obtain RBM learning rule. As a result we have proposed a new unsupervised learning rule and the novel techniques to infer the RBM training rules. It is based on minimization of the reconstruction mean square error and cross-entropy error function, which we can obtain using simple iterations of Gibbs sampling. In contrast to the traditional energy-based method, which is based on a linear representation of neural units, the proposed approach permits us to take into account the nonlinear nature of neural units.

Let's examine the restricted Boltzmann machine. We will represent the RBM using three layers (visible, hidden and visible) [15] as shown in Fig. 3. As can be seen such a representation of RBM is equivalent to PCA neural network, where the hidden and last visible layer is respectively compression and reconstruction (inverse) layer.

Let's consider the Gibbs sampling using unfolded representation of RBM. In this case we can represent Gibbs sampling as shown in Fig. 4.

Then Gibbs sampling will consist of the following procedure. Let $x(0)$ be the input data, which arrives at the visible layer at time 0. Then the output of the hidden layer is defined as follows:

$$y_j(0) = F(S_j(0)), \tag{14}$$

$$S_j(0) = \sum_i \omega_{ij} x_i(0) + T_j. \tag{15}$$

The inverse layer reconstructs the data from the hidden layer. As a result we can obtain $x(1)$ at time 1:

$$x_i(1) = F(S_i(1)), \tag{16}$$

$$S_i(1) = \sum_j \omega_{ij} y_j(0) + T_i. \tag{17}$$

After this, $x(1)$ enters the visible layer and we can obtain the output of the hidden layer the following way:

$$y_j(1) = F(S_j(1)), \tag{18}$$

$$S_j(1) = \sum_i \omega_{ij} x_i(1) + T_j. \tag{19}$$

Continuing the given process we can obtain on a step $k$, that

$$\begin{aligned} x_i(k) &= F(S_i(k)), \\ S_i(k) &= \sum_j \omega_{ij} y_j(k-1) + T_i. \end{aligned} \tag{20}$$

$$\begin{aligned} y_j(k) &= F(S_j(k)), \\ S_j(k) &= \sum_i \omega_{ij} x_i(k) + T_j. \end{aligned} \tag{21}$$

There exist the different ways for RBM training. It is based on the use of the different learning criteria. As mentioned before G. Hinton proposed an energy-based model, which is based on maximization of the log-likelihood input data distribution $P(x)$. We suggest using the two loss functions for RBM learning [15–17]. The first training criterion is based on minimization of mean square error (MSE). The second one involves the minimization of cross entropy error function. Both training criteria have the attractive properties and have been studied in many papers [18, 19]. Our main goal here is to show, that the use of different training criteria leads to the same learning rules. In the next subsections we will study these criteria in more detail.

### *2.1. MSE Training Criterion*

Let's consider the use of mean square error function for RBM learning. Then the primary goal of training RBM is to minimize the reconstruction mean squared error (MSE) in the hidden and visible layers. The MSE in the hidden layer is proportional to the difference between the states of the hidden units at the various time steps. Then in case of CD-k

$$E_h(k) = \frac{1}{2} \sum_{l=1}^{L} \sum_{j=1}^{m} \sum_{p=1}^{k} (y_j^l(p) - y_j^l(p-1))^2. \tag{22}$$

Similarly, the MSE in the inverse layer is proportional to the difference between the states of the inverse units at the various time steps:

$$E_v(k) = \frac{1}{2} \sum_{l=1}^{L} \sum_{i=1}^{n} \sum_{p=1}^{k} (x_i^l(p) - x_i^l(p-1))^2, \tag{23}$$

where $L$ is the number of training patterns.

In case of CD-$k$ the common reconstruction mean squared error is defined as the sum of errors:

$$E_s(k) = E_h(k) + E_v(k). \tag{24}$$

Then

$$E_s(k) = \frac{1}{2}\sum_{l=1}^{L}\sum_{j=1}^{m}\sum_{p=1}^{k}(y_j^l(p) - y_j^l(p-1))^2 + \frac{1}{2}\sum_{l=1}^{L}\sum_{i=1}^{n}\sum_{p=1}^{k}(x_i^l(p) - x_i^l(p-1))^2. \quad (25)$$

Using the same approach for CD-1 we can write the reconstruction mean square error by the following way [16, 17]:

$$E_s(1) = \frac{1}{2}\sum_{l=1}^{L}\sum_{j=1}^{m}(y_j^l(1) - y_j^l(0))^2 + \frac{1}{2}\sum_{l=1}^{L}\sum_{i=1}^{n}(x_i^l(1) - x_i^l(0))^2. \quad (26)$$

**Theorem 1.** *Maximization of the log-likelihood input data distribution P(x) in the space of synaptic weights of the restricted Boltzmann machine is equivalent to special case of minimizing the reconstruction mean squared error in the same space.*

**Proof.** Let's consider online training of an RBM. In this case the weights and thresholds are updated iteratively in accordance with the gradient descent technique:

$$\omega_{ij}(t+1) = \omega_{ij}(t) - \alpha\frac{\partial E(k)}{\partial \omega_{ij}(t)}, \quad (27)$$

$$T_i(t+1) = T_i(t) - \alpha\frac{\partial E(k)}{\partial T_i(t)}, \quad (28)$$

$$T_j(t+1) = T_j(t) - \alpha\frac{\partial E(k)}{\partial T_j(t)}. \quad (29)$$

If we use CD-$k$, the cost function $E(k)$ for one sample is defined by the following expression:

$$E(k) = \frac{1}{2}\sum_{j=1}^{m}\sum_{p=1}^{k}(y_j(p) - y_j(p-1))^2 + \frac{1}{2}\sum_{i=1}^{n}\sum_{p=1}^{k}(x_i(p) - x_i(p-1))^2. \quad (30)$$

Then differentiating (30) with respect to $\omega_{ij}$ we get the following result:

$$\frac{\partial E(k)}{\partial \omega_{ij}} = \frac{\partial E(k)}{\partial y_j(p)}\frac{\partial y_j(p)}{\partial S_j(p)}\frac{\partial S_j(p)}{\partial \omega_{ij}} + \frac{\partial E(k)}{\partial x_i(p)}\frac{\partial x_i(p)}{\partial S_i(p)}\frac{\partial S_i(p)}{\partial \omega_{ij}}$$

$$= \sum_{p=1}^{k}(y_j(p) - y_j(p-1))x_i(p)F'(S_j(p)) + \sum_{p=1}^{k}(x_i(p) - x_i(p-1))y_j(p-1)F'(S_i(p)).$$

Let us suppose a linear activation function is used. This is equivalent to

$$F'(S_j(p)) = \frac{\partial S_j(p)}{\partial \omega_{ij}} = F'(S_i(p) = \frac{\partial S_i(p)}{\partial \omega_{ij}} = 1.$$

Then

$$\frac{\partial E(k)}{\partial \omega_{ij}} = \sum_{p=1}^{k} y_j(p)x_i(p) - y_i(p-1)x_i(p-1) = y_j(k)x_i(k) - y_j(0)x_i(0).$$

As a result we can obtain the CD-$k$ training rule for RBMs:

$$\omega_{ij}(t+1) = \omega_{ij}(t) + \alpha(x_i(0)y_j(0) - x_i(k)y_j(k)). \quad (31)$$

Similarly, for thresholds we have

$$T_j(t+1) = T_j(t) + \alpha(y_j(0) - y_j(k)), \quad (32)$$

$$T_i(t+1) = T_i(t) + \alpha(x_i(0) - x_i(k)). \quad (33)$$

Analogically, for CD-1 we can obtain the following RBM training rule:

$$\begin{aligned} \omega_{ij}(t+1) &= \omega_{ij}(t) + \alpha(x_i(0)y_j(0) - x_i(1)y_j(1)), \\ T_i(t+1) &= T_i(t) + \alpha(x_i(0) - x_i(1)), \\ T_j(t+1) &= T_j(t) + \alpha(y_j(0) - y_j(1)). \end{aligned} \tag{34}$$

As can be seen the last equations are identical to the conventional RBM training rule. Therefore the proof of Theorem 1 is completed.

Thus the nature of the conventional RBM training rule is linear in terms of minimizing the MSE. Therefore we will call such a machine a linear RBM.

**Corollary 1.** *A linear restricted Boltzmann machine from the training point of view is equivalent to the linear PCA (auto associative) neural network if we use Gibbs sampling during learning.*

**Corollary 2.** *The training rule for a nonlinear restricted Boltzmann machine in the case of CD-k is defined as*

$$\begin{aligned} \omega_{ij}(t+1) &= \omega_{ij}(t) - \alpha\left(\sum_{p=1}^{k}(y_j(p) - y_j(p-1))x_i(p)F'(S_j(p) + (x_i(p) - x_i(p-1))y_j(p-1)F'(S_i(p))\right), \\ T_j(t+1) &= T_j(t) - \alpha\left(\sum_{p=1}^{k}(y_j(p) - y_j(p-1))F'(S_j(p))\right), \\ T_i(t+1) &= T_i(t) - \alpha\left(\sum_{p=1}^{k}(x_i(p) - x_i(p-1))F'(S_i(p))\right). \end{aligned} \tag{35}$$

**Corollary 3.** *The training rule for a nonlinear restricted Boltzmann machine in the case of CD-1 is the following:*

$$\begin{aligned} \omega_{ij}(t+1) &= \omega_{ij}(t) - \alpha\left(\left(y_j(1) - y_j(0)\right)F'(S_j(1))x_i(1) + \left(x_i(1) - x_i(0)\right)F'(S_i(1))y_j(0)\right), \\ T_i(t+1) &= T_i(t) - \alpha\left(x_i(1) - x_i(0)\right)F'(S_i(1)), \\ T_j(t+1) &= T_j(t) - \alpha\left(y_j(1) - y_j(0)\right)F'(S_j(1)). \end{aligned} \tag{36}$$

Thus as can be seen the classical equations for RBM training are a particular case of the proposed technique.

If batch learning is used we can write:

$$\begin{aligned} \omega_{ij}(t+1) &= \omega_{ij}(t) - \alpha\frac{\partial E_s(k)}{\partial \omega_{ij}(t)}, \quad T_i(t+1) = T_i(t) - \alpha\frac{\partial E_s(k)}{\partial T_i(t)}, \\ T_j(t+1) &= T_j(t) - \alpha\frac{\partial E_s(k)}{\partial T_j(t)}. \end{aligned} \tag{37}$$

**Theorem 2.** *If we use CD-k for a nonlinear RBM and use batch learning the training rule is defined by the following equations:*

$$\begin{aligned} \omega_{ij}(t+1) &= \omega_{ij}(t) - \alpha\sum_{l=1}^{L}\sum_{p=1}^{k}\left((y_j^l(p) - y_j^l(p-1))x_i^l(p)F'(S_j^l(p)) + (x_i^l(p) - x_i^l(p-1))y_j^l(p-1)F'(S_i^l(p))\right), \\ T_j(t+1) &= T_j(t) - \alpha\sum_{l=1}^{L}\sum_{p=1}^{k}(y_j^l(p) - y_j^l(p-1))F'(S_j^l(p)), \\ T_i(t+1) &= T_i(t) - \alpha\sum_{l=1}^{L}\sum_{p=1}^{k}(x_i^l(p) - x_i^l(p-1))F'(S_i^l(p)). \end{aligned} \tag{38}$$

The proof of this theorem is similar to the proof of Theorem 1.

**Corollary 4.** *If we use CD-1 for a nonlinear RBM, with batch learning, the training rule is defined by the following equations:*

$$\omega_{ij}(t+1) = \omega_{ij}(t) - \alpha\sum_{l=1}^{L}\left((y_j^l(1) - y_j^l(0))x_i^l(1)F'(S_j^l(1)) + (x_i^l(1) - x_i^l(0))y_j^l(0)F'(S_i^l(1))\right),$$
$$T_j(t+1) = T_j(t) - \alpha\sum_{l=1}^{L}(y_j^l(1) - y_j^l(0))F'(S_j^l(1)), \tag{39}$$
$$T_i(t+1) = T_i(t) - \alpha\sum_{l=1}^{L}(x_i^l(1) - x_i^l(0))F'(S_i^l(1))).$$

**Corollary 5.** *If we use CD-k for a linear RBM, with batch learning, the training rule is defined by the following equations:*

$$\omega_{ij}(t+1) = \omega_{ij}(t) + \alpha\sum_{l=1}^{L}(x_i^l(0)y_j^l(0) - x_i^l(k)y_j^l(k)),$$
$$T_j(t+1) = T_j(t) + \alpha\sum_{l=1}^{L}(y_j^l(0) - y_j^l(k)), \tag{40}$$
$$T_i(t+1) = T_i(t) + \alpha\sum_{l=1}^{L}(x_i^l(0) - x_i^l(k)).$$

**Corollary 6.** *If we use CD-1 for a linear RBM, with batch learning, the training rule is defined by the following equations:*

$$\omega_{ij}(t+1) = \omega_{ij}(t) + \alpha\sum_{l=1}^{L}(x_i^l(0)y_j^l(0) - x_i^l(1)y_j^l(1)),$$
$$T_j(t+1) = T_j(t) + \alpha\sum_{l=1}^{L}(y_j^l(0) - y_j^l(1)), \tag{41}$$
$$T_i(t+1) = T_i(t) + \alpha\sum_{l=1}^{L}(x_i^l(0) - x_i^l(1)).$$

In this section we have obtained the novel unsupervised learning rules for restricted Boltzmann machines, using MSE training criterion. The traditional energy-based method is based on maximization of the log-likelihood input data distribution and leads to the linear representation of neural units in terms of minimizing the MSE. The proposed approach, which we can be obtained using simple iterations of Gibbs sampling is based on minimization of reconstruction mean square error and leads to nonlinear and linear representation of neurons. In contrast to energy-based approach, the MSE training criterion permits us to take into account the derivatives of a nonlinear activation function for neural network units. As a result the proposed approach is more universal in comparison with traditional energy-based model. We will call the proposed approach the reconstruction error-based approach (REBA). We have shown that the classical equations for RBM training are a special case of the proposed technique. Such a technique, based on minimization MSE training criterion can be applied for deep neural networks learning. For the first time, the approach described above has been proposed in [15] for the CD-1 and in [16, 17] for CD-*k*.

### *2.2. Cross-Entropy Training Criterion*

The cross-entropy measure (CE) can be used as an alternative to mean squared error. Let's consider a sigmoid neural network and the cross entropy error function instead of mean square error. The goal of training

RBM is to minimize the cross-entropy in the hidden and visible layers. In the case of CD-$k$ the cross-entropy error function in the inverse layer is defined as

$$CE_v(k) = -\sum_{l=1}^{L}\left[\sum_{p=1}^{k}\sum_{i=1}^{n}\left(x_i^l(p-1)\log(x_i^l(p)) + (1-x_i^l(p-1))\log(1-x_i^l(p))\right)\right]. \quad (42)$$

Similarly, the cross-entropy error function in the hidden layer

$$CE_h(k) = -\sum_{l=1}^{L}\left[\sum_{p=1}^{k}\sum_{j=1}^{m}\left(y_j^l(p-1)\log(y_j^l(p)) + (1-y_j^l(p-1))\log(1-y_j^l(p))\right)\right]. \quad (43)$$

The common cross entropy error function in case of CD-$k$ is defined as the sum of errors:

$$CE_s(k) = CE_h(k) + CE_v(k). \quad (44)$$

**Theorem 3.** *Maximization of the log-likelihood input data distribution P(x) in the space of synaptic weights restricted Boltzmann machine is equivalent to minimizing the cross-entropy error function $CE_s(k)$.*

**Proof.** In order to simplify the proof of the theorem let's consider the cross entropy for CD-1. In this case the cross entropy error function for a single example is

$$CE(1) = -\sum_{i=1}^{n}\left(x_i(0)\log(x_i(1)) + (1-x_i(0))\log(1-x_i(1))\right) - \sum_{j=1}^{m}\left(y_j(0)\log(y_j(1)) + (1-y_j(0))\log(1-y_j(1))\right).$$

Then

$$\frac{\partial CE(1)}{\partial \omega_{ij}} = -\frac{x_i(0)}{x_i(1)}x_1(1)(1-x_i(1))y_j(0) + \frac{1-x_i(0)}{1-x_i(1)}x_i(1)(1-x_i(1))y_j(0) - y_j(0)(1-y_j(1))x_i(1)$$
$$+ (1-y_j(0))y_j(1)x_i(1) = -x_i(0)(1-x_i(1))y_j(0) + (1-x_i(0))x_i(1)y_j(0) - y_j(0)(1-y_j(1))x_i(1)$$
$$+ (1-y_j(0))y_j(1)x_i(1) = -x_i(0)y_j(0) + x_i(0)x_i(1)y_j(0) + x_i(1)y_j(0) - x_i(0)x_i(1)y_j(0)$$
$$- y_j(0)x_i(1) + y_j(0)y_j(1)x_i(1) + y_j(1)x_i(1) - y_j(0)y_j(1)x_i(1) = x_i(1)y_j(1) - x_i(0)y_j(0).$$

Accordingly, for the thresholds

$$\frac{\partial CE(1)}{\partial T_i} = x_i(1) - x_i(0),$$
$$\frac{\partial CE(1)}{\partial T_j} = y_j(1) - y_j(0).$$

The theorem is proved. As follows from theorem the RBM learning rules can be obtained in a simpler way compared to the conventional energy-based approach. Thus using minimization of the cross-entropy error function and simple iterations of Gibbs sampling we have received the conventional linear RBM learning rules.

The obtained results can be summarized in the following general theorem.

**Theorem 4.** *Maximization of the log-likelihood input data distribution P(x) in the space of synaptic weights restricted Boltzmann machine is equivalent to minimizing the cross-entropy and to special case of minimizing the mean squared error:*

$$\max(\ln P(x)) = \min(CE_s) = \min(E_s). \quad (45)$$

Theorem 4 represents a generalization of the previous results in this paper. It follows from the theorem that the use of various training criteria leads to the same learning rules. Therefore the nature of unsupervised learning of RBM is the same, even if we use different objective function. The maximization of the log-likelihood input data distribution and minimization cross-entropy error function leads to the linear representation of neural units in terms of minimizing the MSE. It should be noted, that applying of training criterion, which is based on minimization of MSE, we can take into account also nonlinear representation of neurons. Thus an obvious advantage of the use MSE as error function compared to applying log-

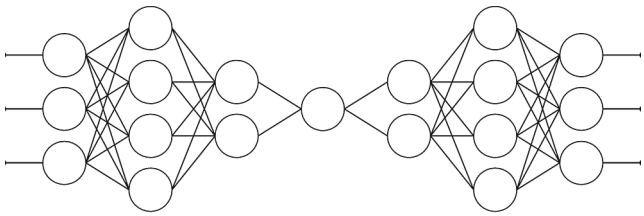

**Fig. 5.** Deep auto-encoder.

likelihood and cross-entropy loss function is that using MSE training criterion can be received both linear and nonlinear learning rules, but not vice versa. Therefore such an approach is more universal, fundamental and flexible compared to the energy-based and cross-entropy techniques and can open us new possibilities to unsupervised learning in deep neural networks.

In conclusion of this section let's consider the analogy with a single layer perceptron (SLP) learning rule. The delta rule for training of a single layer perceptron proposed by Rosenblatt and Widrow–Hoff has linear nature, but could be applied also for SLP learning with threshold activation function [20, 21]. This delta rule can be obtained from MSE minimization using linear activation function of neural unit or using nonlinear transfer function, if we take the derivative of the activation function equal to unity. We can apply the linear or nonlinear training delta rule depending on real-life application and it gives us learning's diversity and additional possibilities for SLP learning.

## 3. EXPERIMENTAL INVESTIGATIONS

In this section we present some numerical results to show effectiveness of the proposed approach. To check the performance of the proposed technique, we used a well-known benchmark problem. In order to train neural network, we applied the previously described rules without any additional techniques, such as dropout, etc.

### *3.1. Data Compression*

To assess the performance of the proposed learning technique experiments were conducted on an artificial data set. The artificial data $x$ lie on a one-dimensional manifold (a helical loop) embedded in three dimensions [22] and were generated from a uniformly distributed factor $t$ in the range $[-1, 1]$:

$$\begin{cases} x_1 = \sin(\pi t) + \mu, \\ x_2 = \cos(\pi t) + \mu, \\ x_3 = t + \mu, \end{cases} \tag{46}$$

where $\mu$ – Gaussian noise with mean 0 and standard deviation 0.05.

In order to verify the proposed approach experimentally, we trained seven-layer deep auto-encoder using data subsets of 1000 samples. The deep auto-encoder is shown in Fig. 5. We used the sigmoid activation function for all layers of the neural network except for the bottleneck layer. The linear activation function is used in the bottleneck layer (fourth layer).

The average results are provided in Table 1. Here MSE is the mean square error on the training data set; MS is the mean square error on the test data set in order to check generalization ability. The size of the test patterns is 1000. The learning rate $\alpha$ is 0.1 for conventional RBM and 0.5 for REBA in all experiments.

The number of epochs for the pre-training of each layer is 10. The number of epochs for fine-tuning is 1000. It is evident from the simulation results that the use of the REBA technique can improve the generalization capability of a deep auto-encoder in the case of CD-1 and CD-10. Figures 6 and 7 depict the original training data and the reconstructed data from one nonlinear component, using test data. As can be seen, the auto-encoder reconstructs the data from one nonlinear component with well accuracy.

**Table 1.** Comparison of RBM and REBA techniques

| Training procedure | CD-$k$ | MSE | MS |
|---|---|---|---|
| RBM | 1 | 0.699 | 0.886 |
| | 5 | **0.710** | **0.932** |
| | 10 | 0.689 | 0.916 |
| | 15 | **0.688** | **0.873** |
| REBA | 1 | **0.673** | **0.851** |
| | 5 | 0.719 | 0.966 |
| | 10 | **0.677** | **0.907** |
| | 15 | 0.700 | 0.895 |

**Table 2.** Comparative analysis

| Training procedure | MSE | MS |
|---|---|---|
| RBM | 3.7801 | 4.0115 |
| REBA | 3.6490 | 3.8726 |

### *3.2. Data Visualization*

In order to illustrate the performance of REBA technique we present simulation results for visualization of handwritten digits using MNIST dataset [23]. The MNIST dataset contains 28 × 28 handwritten digits in gray-scale and has a training set of 60000 samples, and a test set of 10000 samples. For mapping 784D digits data to a 2D feature space, the deep auto-encoder with topology 784-1000-500-250-2 is used. The identity function for neural units is applied in bottleneck layer. In other layers the sigmoid activation function is used. We have realized pre-training of deep auto-encoder using greedy layer-wise approach with RBM and REBA techniques. This procedure is started from first layer and performed in unsupervised manner. Finally fine-tuning all of parameters of neural network using simple back-propagation algorithm is implemented. We have compared two techniques: REBA and conventional RBM. We have used the following training parameters: the learning rate for pre-training is 0.2 for REBA and 0.05 for conventional RBM in all layers except of bottleneck layer. The training rate for bottleneck layer is 0.001. The comparative analysis of two techniques is shown in the Table 2.

Visualization of MNIST dataset on the basis of REBA is shown in Fig. 8 for the 500 test images for each class of digits.

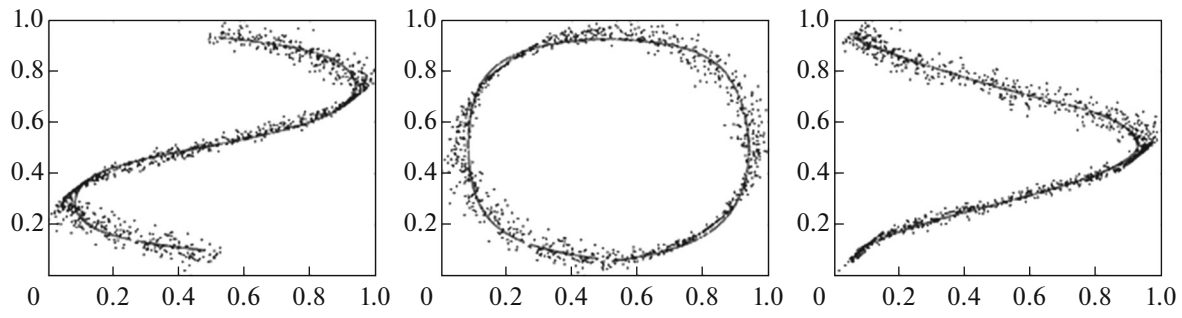


**Fig. 6.** 2D views of original and reconstructed datasets.

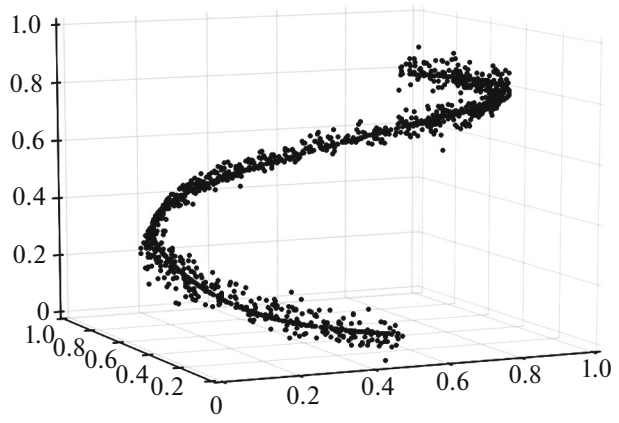


**Fig. 7.** 3D views of original and reconstructed datasets.

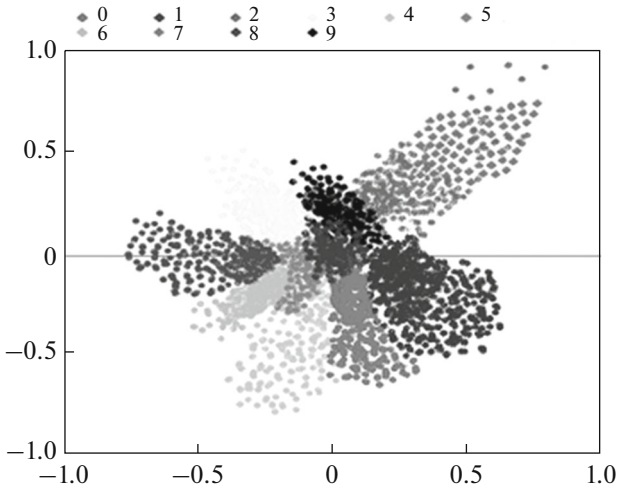


**Fig. 8.** Visualization of handwritten digits.

As can be seen from Fig. 8 and the Table 2 the use of REBA technique permits to obtain the better performance compared conventional approach.

### *3.3. Data Classification*

In order to check the performance of proposed technique we present simulation results also for handwritten digits classification using again MNIST dataset.

The deep perceptron with topology 784-500-500-2000-10 and sigmoid activation function in all layers of neural network is used. The classification of input image is performed by the following way. At the beginning the $k$-th output neuron, which has a maximum output value is defined:

$$y_k = \arg\max y_j. \tag{47}$$

The output value of the $k$-th neuron is assigned a unit value, and the output values of the other neurons are equal to zero:

$$y_j = \begin{cases} 1, & j = k; \\ 0, & \text{otherwise}. \end{cases} \tag{48}$$

**Table 3.** Comparison of RBM and Hybrid REBA techniques

| Training procedure | MSE | MS | Test error rate (%) | NIT |
|---|---|---|---|---|
| Classic RBM | 6.178e-6 | 0.0235 | 1.23 | 10 |
| Hybrid REBA (9 + 1) | 5.962e-6 | 0.0224 | **1.09** | 9 + 1 |

Also we can use probabilistic coding the output values of the deep perceptron. In this case the output value of the $j$-th neuron is determined as follows:

$$y_j = \frac{y_j}{\sum_j y_j}. \tag{49}$$

We applied here the following training parameters: learning rate is 0.1 for REBA and 0.2 for conventional RBM; size of mini batch is 100 samples; the number of pre-training epoch is 10, the number of fine-tuning epoch is 100. Also we use here weight decay technique with decay parameter 0.00001. It should be noted that the best performance has been obtained using the hybrid approach to pre-training of deep neural network, when we use both conventional and proposed approach. The results of experiments are shown in the Table 3. The NIT is the number of pre-training epochs. In this case we used for pre-training 9 epoch of classical RBM approach and 1 epoch of REBA. As can be seen from the Table 3 we can reach the test error rate 1.09% using hybrid approach.

## CONCLUSIONS

In this paper we have addressed the key aspects of unsupervised learning in deep neural networks. We described both the traditional energy-based method, which is based on a linear representation of neural units, and the proposed approach, which is based on nonlinear representation of neurons. We have proved that maximization of the log-likelihood input data distribution of restricted Boltzmann machine is equivalent to minimizing the cross-entropy and to special case of minimizing the mean squared error. It follows that the nature of unsupervised learning is invariant to the different learning criteria. We have proposed also a new technique called "REBA" for RBM learning, and this approach in contrast to an energy-based model is based on the minimization of the reconstruction mean square error in the hidden and last visible layers of the RBM. Thus using MSE training criterion we can get both conventional and novel learning rules. We have shown that classical equations for RBM training are a special case of the proposed technique. Therefore the proposed approach is more universal in contrast to the traditional energy-based model. The simulation results demonstrate the effectiveness of the proposed technique.

## REFERENCES

1. Hinton, G., Osindero, S., and Teh, Y., A fast learning algorithm for deep belief nets, *Neural Computation,* 2006, vol. 18, pp. 1527–1554.
2. Hinton, G., Training products of experts by minimizing contrastive divergence, *Neural Computation,* 2002, vol. 14, pp. 1771–1800.
3. Hinton, G. and Salakhutdinov, R., Reducing the dimensionality of data with neural networks, *Science,* 2006, vol. 313, no. 5786, pp. 504–507.
4. Hinton, G.E., A practical guide to training restricted Boltzmann machines, *Tech. Rep. 2010-000,* Toronto: Machine Learning Group, University of Toronto, 2010.
5. Krizhevsky, A., Sutskever, L., and Hinton, G., *ImageNet classification with deep convolutional neural networs*, *Proc. Advances in Neural information Processing Systems,* 2012, vol. 25, pp. 1090–1098.
6. LeCun, Y., Bengio, Y., and Hinton, G., *Deep Learning Nature,* 2015, vol. 521, no. 7553, pp. 436–444.
7. Mikolov, T., Deoras, A., Povey, D., Burget, L., and Cernocky, J., Strategies for training large scale neural network language models, in *Automatic Speech Recognition and Understanding*, 2011, pp. 195–201.
8. Hinton, G., et al., Deep neural network for acoustic modeling in speech recognition, *Proc. IEEE Signal Processing Magazine,* 2012, vol. 29, pp. 82–97.
9. Bengio, Y., Learning deep architectures for AI, *Foundations and Trends in Machine Learning*, 2009, vol. 2, no. 1, pp. 1–127.
10. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., et al., Greedy layer-wise training of deep networks, *Advances in Neural Information Processing Systems,* 2007, vol. 19, p. 153.

11. Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., and Bengio, S., Why does unsupervised pretraining help deep learning?, *Journal of Machine Learning Research,* 2010, vol. 11, pp. 625–660.
12. Larochelle, H., Bengio, Y., Louradour, J., and Lamblin, P., Exploring strategies for training deep neural networks, *Journal of Machine Learning Research,* 2009, vol. 1, pp. 1–40.
13. Bengio, Y., Courville, A., and Vincent, P., Representation learning a review and new percpectives. *Proc. IEEE Trans. Pattern Anal. Machine Intell.,* 2013, vol. 35, pp. 1798–1828.
14. Glorot, X., Bordes, A., and Bengio, Y., Deep sparse rectifier networks, *Proc. 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP,* 2011, vol. 15, pp. 315–323.
15. Golovko, V., A learning technique for deep belief neural networks, in *Neural Networks and Artificial Intelligence,* Golovko, V., Kroshchanka, A., Rubanau, U., and Jankowski, S., Eds., Springer, *Communication in Computer and Information Science,* 2014, vol. 440, pp. 136–146.
16. Golovko, V., A new technique for restricted Boltzmann machine learning, Kroshchanka, A., Turchenko, V., Jankowski, S., and Treadwell, D., Eds., *Proc. 8th IEEE International Conference IDAACS-2015*, Warsaw 24–26 September 2015, Warsaw, 2015, pp. 182–186.
17. Golovko, V., From multilayers perceptrons to deep belief neural networks: training paradigms and application, *Lections on Neuroinformatics,* Golovko, V.A., Ed., Moscow: NRNU MEPhI, 2015, pp. 47–84 [in Russian].
18. Golik, P., *Cross-entropy vs. squared error training: A theoretical and experimental comparison*, Golik, P., Doetsch, P., and Ney, H., Eds., in *Interspeech* Lyon, France, 2013, pp. 1756–1760.
19. Glorot, X. and Bengio, Y., Understanding the difficulty of training deep feed-forward neural networks, *Proc. of Int. Conf. on Artificial Intelligence and Statistics*, vol. 9, Chia Laguna Resort, Italy, 2010, pp. 249–256.
20. Rosenblatt, F., *Principles of Neurodynamics; Perceptrons and the Theory of Brain Mechanisms,* Washington: Spartan Books, 1962.
21. Widrow, B. and Hoff, M., Adaptive switching circuits, *Proc. 1960 IRE WESCON Convention Record.-DUNNO,* 1960, pp. 96–104.
22. Scholz, M., Fraunholz, M., and Selbig, J., Nonlinear principal component analysis: neural network models and applications, in *Principal Manifolds for Data Visualization and Dimension Reduction,* Springer Berlin Heidelberg, 2008, pp. 44–67.
23. Qiao, Yu., THE MNIST DATABASE of handwritten digits: http://www.gavo.t.u-tokyo.ac.jp/~qiao/database.html.