Applied Soft Computing Journal 92 (2020) 106278

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

Online RBM: Growing Restricted Boltzmann Machine on the fly for unsupervised representation

Ramasamy Savitha *, ArulMurugan Ambikapathi, Kanagasabai Rajaraman

*Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore*

ARTICLE INFO

Article history:
Received 18 October 2019
Received in revised form 30 March 2020
Accepted 31 March 2020
Available online 24 April 2020

Keywords:
Restricted Boltzmann Machine
Online learning
Unsupervised representation

ABSTRACT

In this work, we endeavor to investigate and propose a novel unsupervised online learning algorithm, namely the Online Restricted Boltzmann Machine (O-RBM). The O-RBM is able to construct and adapt the architecture of a Restricted Boltzmann Machine (RBM) artificial neural network, according to the statistics of the streaming input data. Specifically, for a training data that is not fully available at the onset of training, the proposed O-RBM begins with a single neuron in the hidden layer of the RBM, progressively adds and suitably adapts the network to account for the variations in streaming data distributions. Such an unsupervised learning helps to effectively model the probability distribution of the entire data stream, and generates robust features. We will demonstrate that such unsupervised representations can be used for discriminative classifications on a set of multi-category and binary classification problems for unstructured image and structured signal data sets, having varying degrees of class-imbalance. We first demonstrate the O-RBM algorithm and characterize the network evolution using the simple and conventional multi-class MNIST image dataset, aimed at recognizing hand-written digit. We then benchmark O-RBM performance to other machine learning, neural network and Class RBM techniques using a number of public non-stationary datasets. Finally, we study the performance of the O-RBM on a real-world problem involving predictive maintenance of an aircraft component using time series data. In all these studies, it is observed that the O-RBM converges to a stable, concise network architecture, wherein individual neurons are inherently discriminative to the class labels despite unsupervised training. It can be observed from the performance results that on an average O-RBM improves accuracy by 2.5%–3% over conventional offline batch learning techniques while requiring at least 24%–70% fewer neurons.

1. Introduction

In the recent years, deep learning has revolutionized various fields of computer vision. Deep learning algorithms have superlative capabilities for jointly performing feature mapping, exclusively for tasks such as classification, segmentation, regression etc. Thus, they outperform other machine learning approaches in applications ranging from image classification [1] and medical diagnostics [2] to credit fraud analytics [3]. However, it is challenging to adaptively re-train neural networks to track changes in input data distribution, especially in non-stationary streaming data applications, where the data is not completely available before hand. Further, training multiple layer neural networks requires *a priori* specification of a suitable network architecture, and it is difficult to decide the required architecture for a dynamically varying input data distribution.

Online learning approaches for deep neural networks have the potential to address both these challenges. Several studies have put forth online learning algorithms for training single layer perceptron networks [4–6]. Single layer feedforward neural networks can be trained in an online fashion using Stochastic Gradient Descent [7] or Extended Kalman Filters [8,9] for the parameter update. However, it remains challenging to extend these successes to the task of training deep neural networks in a fully online manner. For example, online algorithms for denoising autoencoders (DAE) [10] have been used for incremental feature learning with streaming data, but need *a priori* training with a DAE architecture as the building block to learn a base set of features first. Further, incremental learning has been applied within a boosting convolutional neural network framework for feature augmentation, loss function updation and fine-tuned back propagation with information accumulating in successive mini-batches [11]. Finally, it has also been shown that updating a greedily pre-trained layer-wise restricted Boltzmann machines (RBMs) in an online fashion automatically learns discriminative features for classification [12,13].

* Corresponding author.
E-mail addresses: ramasamysa@i2r.a-star.edu.sg (R. Savitha), Arulmurugan_Ambikapathi@i2r.a-star.edu.sg (A. Ambikapathi), kanagasa@i2r.a-star.edu.sg (K. Rajaraman).

https://doi.org/10.1016/j.asoc.2020.106278
1568-4946/

Recently, there is an increased interest with lifelong continual learning approaches [14–16], aimed at adapting weights of a pre-trained network to learn new tasks without catastrophic forgetting. Progressive networks [17] incorporate prior knowledge at initialization, retains a pool of pre-trained models throughout training, and fine-tunes the model with new data to learn lateral connections for extracting useful features. Inspired by the neurogenesis of the human hippocampus that enables human beings to learn continuously, and adapting themselves to changing environments, neurogenesis network [18] have been developed. A neurogenesis network adds neurons to a deep neural network, while preserving previously trained data representations. The life long learning ability of a growing self-organizing neural architecture equipped with recurrent neurons, for processing *time-varying patterns* is studied in [19,20]. In order to avoid catastrophic forgetting in a continual learning framework, the representations of a deep neural network are divided into long term and short term memory units in [21], and this division is optimized with a Kalman optimizer.

While all the above-mentioned approaches enable lifelong continual learning, these approaches require an a priori trained network and/or a fixed base network architecture as a precursor for incremental online updates with streaming data. Hence, methods that evolve and train a network architecture from scratch in an online manner, as the data streams in, would offer novel capabilities and thus provide a complete online learning solution. They also enable to understand knowledge distribution across classes, thus, providing an ability to understand the data through the network. Further, it has been proved in [22] that the addition of a hidden neuron to a RBM helps to improve its modeling power, unless it has already perfected the data. Hence, it is important to find the right number of neurons in the hidden layer of the network to perfectly model any data. This can be achieved by letting the model grow its network, depending on the data.

In this work, we present a fully online, unsupervised learning algorithm named the Online Restricted Boltzmann Machine (O-RBM), for an effective and efficient feature representation. At the beginning of training, there are no neurons in the hidden layer of a RBM neural network. As training data samples stream in, the ability of the network to represent the current sample is assessed using the reconstruction error for the sample, from the current architecture. Based on this reconstruction error, the online adaptive learning algorithm either deletes the samples that are well represented, or adds a neuron to the hidden layer to represent the sample and updates the weights for the entire set of existing neurons in the network. As the network updates are tailored to represent the distributions of the distinctive input sample features, the network is compact and inherently discriminative [23]. The unsupervised features that are thus learnt, can be mapped to a set of specific classes via any conventional discriminative learning.

We first propose the methodology of O-RBM with details pertaining to the training and evolution of the network for effective unsupervised feature representations. We then demonstrate the unique abilities of the O-RBM to represent the distinctive class distributions of the feature space and to learn in a manner that is invariant to the training data sequence, through a study on the standard and well-explored MNIST data set,[1] aimed at recognizing handwritten digits. The sequential invariance is much like the invariance to permutations in the training set seen with batch learning algorithms [24]. We then analyze the performance of O-RBM feature representation based binary classification tasks with a variety of data sets having a wide spectrum of imbalances in the data. It is critical to learn the distribution of a minority class from a highly imbalanced data set, and O-RBM provides a premise to efficiently learn the under-represented minority class, owing to its ability to detect novelty in data. Our results show that the O-RBM can perform better than several state-of-the-art networks with lesser network resources than batch methods. The main contributions of the paper are as follows:

- A fully online learning methodology that includes evolving architecture for unsupervised feature representation, for streaming data.
- The adaptive learning algorithm for training sequential data based on contrastive divergence approach.
- Experimental analysis to demonstrate that the neurons trained through the unsupervised O-RBM are inherently discriminative, using MNIST image (hand written numerical characters) data set.
- Rigorous empirical analysis on a variety of data sets (for varying applications) to show that the classification accuracy and the neuron-to-class label associations of O-RBM are independent of the sequence in which the training samples are presented.

The rest of the paper is organized as follows. In Section 2, we propose and discuss the O-RBM architecture and the associated learning algorithm. Next, in Section 3, we analyze the behavior of O-RBM using the MNIST dataset for unsupervised feature representation and the subsequent classification. In Section 4.1 we demonstrate the performance of O-RBM in relation to other algorithms applied to different applications, that involves class imbalance scenarios. Finally, some interesting conclusions and potential future directions are outlined in Section 5.

2. Online Learning Restricted Boltzmann Machine

We describe the Online Restricted Boltzmann Machine (O-RBM) architecture evolution and the corresponding learning algorithm. Fig. 1 shows the bipartite representation of the proposed evolving RBM architecture. Specifically, we denote the training data set as $\left\{\left(\mathbf{x}^1, c^1\right), \ldots, \left(\mathbf{x}^t, c^t\right), \ldots \left(\mathbf{x}^N, c^N\right)\right\}$. Here, $\mathbf{x}^t \in \Re^m = [x_1^t, \ldots, x_j^t, \ldots, x_m^t]^T$ is an m-dimensional input of the tth sample; $c^t \in \{1, 2, \ldots, s\}$ denotes the corresponding class labels or targets among s classes; and N is the total number of samples. Note that for unsupervised feature representations, the class labels are redundant. However, they can be used subsequently to evaluate the discriminative representation capability of O-RBM. The objective of the O-RBM is to best represent the input distribution, by following a fully adaptive and evolving online learning procedure.

In general, a Restricted Boltzmann Machine (RBM) [25] has visible and hidden layers, connected through symmetric weights. The number of neurons in the input layer is fixed, and it is based on the input dimension. In the current scenario, the number of neurons in the input layer is m (as $\mathbf{x} = [x_1 \ldots x_m]^T$). The hidden layer response dictates the feature representation for the input samples, and for a hidden layer with k neurons ($\mathbf{h} = [h_1 \ldots h_k]^T$), the output feature representation belongs to $\Re^k$. The feature representation is derived by learning the symmetrical connecting weights between the visible and the hidden layers i.e., w_{ij}^1; $i = 1, \ldots, m$, $j = 1, \ldots, k$, as can be inferred from Fig. 2. The hidden layer output for a tth input sample can be expressed as:

$$h_j^t = \sigma\left(\sum_{i=1}^{m} x_i^t w_{ij} + b_h^j\right), \ \forall j = 1, \ldots, k, \tag{1}$$

where σ is the standard sigmoid activation function, and b_h^j is the bias associated with the j of the k hidden neurons. It should be noted that the neurons in the same layer of the RBM are not inter-connected.

[1] http://yann.lecun.com/exdb/mnist/

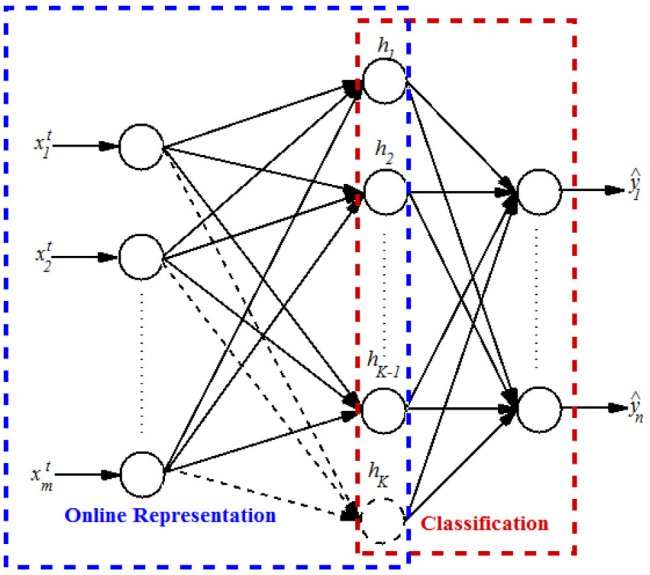


Fig. 1. The network architecture and the training phases. Online representation learning is performed in the first phase, wherein the network begins with zero neurons in the hidden layer, and adds and/or adapts the network learning progresses to derive a feature representation of the data in an unsupervised manner. The next phase performs supervised discriminative learning to associate the feature representation with the class labels.

2.1. Evolving architecture for O-RBM

We now describe the online learning process for feature representation at the hidden layer. Initially, the hidden layer has no neurons. As the data streams in, the online learning algorithm adds a neuron and/or updates the representations of the existing neurons depending on the novelty of the sample. The first neuron in the hidden layer is added based on the first sample $\mathbf{x}^1$ in the training data set.

At a given point during the training process, let the network comprises $k-1$ neurons in the hidden layer, corresponding to $k-1$ novel samples out of a history of $t-1$ samples that has been so far presented to the RBM (with $t \gg k$). Then, for the subsequent tth sample, the reconstruction error of the network is defined as:

$$E^t_{recon} = \frac{1}{m}\sum_{i=1}^{m}(x^t_i - \widehat{x}^t_i)^2, \tag{2}$$

where x^t_i and $\widehat{x}^t_i$ are the ith element of the input $\mathbf{x}^t$ and the corresponding reconstructed (backward path) input element, respectively. Similar to the hidden layer representations (Eq. (1)), the reconstructed input can be expressed as:

$$\widehat{x}^t_i = \sigma\Big(\sum_{j=1}^{k} h^t_j w_{ij} + b^i_x\Big), \ \forall i = 1, \ldots, m, \tag{3}$$

where b^i_x is the bias associated with the ith input neuron.

The reconstruction error (E^t_{recon}) is then compared to two predefined thresholds, namely the novelty threshold (E_n) and the marginal representation threshold (E_m). Based on this comparison, the algorithm chooses one of the following steps for the tth sample:

- **Add a Representative Neuron:** If ($E^t_{recon} > E_n$), the sample is deemed novel and a kth neuron is added to the hidden layer of the network. The initial input weights connecting the kth hidden neuron and the neurons in the input layer are obtained as a function of the inputs $g\left(\mathbf{x}^t\right)$. Here, $g(.)$ can be any function such that $\mathbf{w}^k = [w_1 \cdots w_m]^T \in \Re^m$ and each element of $\mathbf{w}^k$ belongs to [0, 1), so as to confine within the operating regions of the network. In this work, as the inputs are normalized in [0,1), we initialize $\mathbf{w}^k = 0.01{*}\mathbf{x}^t$ to ensure that the weights are initialized in [0,1). The network weights of all the neurons, including the new neuron, are then collectively updated according to the algorithm discussed next in Section 2.2.
- **Adapt Existing Network:** If $E_n > E^t_{recon} > E_m$, the network weight matrix is adapted (as detailed in Section 2.2) such that the probability distribution approximated by the hidden neurons includes the representation of the current training sample.
- **Ignore Sample:** If $E^t_{recon} < E_m$, then the current tth sample is sufficiently represented by the existing network and does not warrant a network or weight matrix update.

Overall, the O-RBM architecture ensures that the neurons in the hidden layer of the network are adaptively added and updated to obtain a compact network structure that is sufficient to yield a strong feature representation for the given set of data. Having discussed the evolving strategy followed by O-RBM, we will now proceed to discuss the online learning algorithm for O-RBM.

2.2. Online contrastive divergence algorithm for O-RBM

The training algorithm for O-RBM is a subtle variation of the contrastive divergence approach proposed for conventional RBM [26]. For sake of completeness and clarity, the inherent details of the algorithm for O-RBM are discussed here. Assuming k is the number of hidden neurons at the current instance of training, the energy function $E(\mathbf{x}, \mathbf{h})$ can be defined as

$$E(\mathbf{x}, \mathbf{h}) = -\mathbf{x}^T\mathbf{W}\mathbf{h} - \mathbf{x}^T\mathbf{b}_\mathbf{x} - \mathbf{h}^T\mathbf{b}_\mathbf{h}, \tag{4}$$

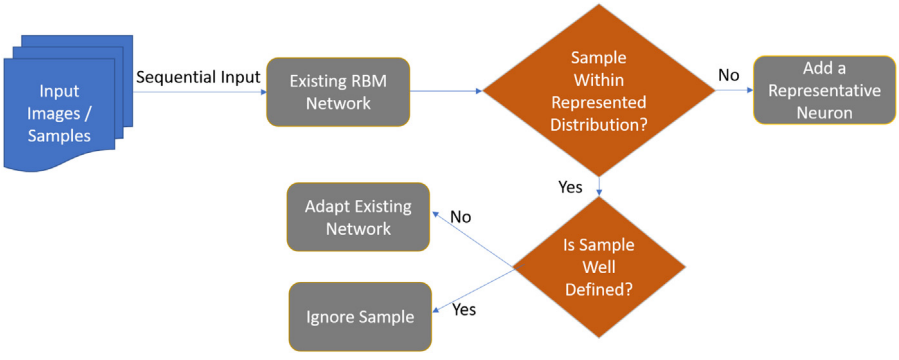


Fig. 2. The overall block diagram for the proposed online restricted Boltzmann machine (O-RBM).

where $\mathbf{x} \in \Re^m$ is the current training sample (with superscript removed for ease of representation), $\mathbf{W} \in \Re^{m\times k}$ is the weight matrix between the m inputs and k neurons, $\mathbf{h} \in \Re^k$ is the response of the k hidden neurons, $\mathbf{b_x} \in \Re^m$ and $\mathbf{b_h} \in \Re^k$, are the biases for the m input neurons and the k hidden neurons, respectively. The probability distribution corresponding to the energy function in Eq. (4) can be expressed as

$$P(\mathbf{x}, \mathbf{h}) = \frac{\exp(-E(\mathbf{x}, \mathbf{h}))}{Z}, \tag{5}$$

where Z is the intractable partition function that is obtained by averaging $\exp(-E(\mathbf{x}, \mathbf{h}))$ over all possible values of $\mathbf{x}$ and $\mathbf{h}$, so that the probability distribution $P(\mathbf{x}, \mathbf{h})$ sums to one. Subsequently, the marginal probability of inputs $\mathbf{x}$ vector can be obtained from $P(\mathbf{x}, \mathbf{h})$ by summing over all possible hidden layer configurations, i.e.,

$$P(\mathbf{x}) = \frac{1}{Z}\sum_{\mathbf{h}} \exp(-E(\mathbf{x}, \mathbf{h})). \tag{6}$$

The purpose of a training algorithm is then to maximize the expectation of $\log P(\mathbf{x})$, for any given $\mathbf{x}$. However, maximization over the probability function involves the computation of expectation over the entire distribution. Hence, the contrastive divergence (CD) approach that approximates the original distribution with a sample point estimation is being employed. However, unlike conventional backpropagation, CD employs back projection (reconstruction) and Gibbs sampling procedures to estimate the weights. Mathematically, CD considers maximizing the following optimization problem:

$$\arg \max_{W} E[\log P(\mathbf{x})]. \tag{7}$$

Then, for a training sample $P(\mathbf{x})$, the CD algorithm aims to compute the conditional probabilities of the hidden units as follows (as the hidden units are not inter connected and hence independent of each other):

$$P(\mathbf{h}|\mathbf{x}) = \prod_{i=1}^{k} P(h_i|\mathbf{x}), \tag{8}$$

where the individual conditional probabilities $P(h_i|\mathbf{x})$ are computed using sigmoidal activations (Eq. (1)). We then sample a hidden feature vector $\mathbf{h}$ based on the obtained distribution. Using the current hidden feature vector $\mathbf{h}$, a reconstruction of $\mathbf{x}$, namely $\widehat{\mathbf{x}}$ (Eq. (3))is obtained by sampling the following distribution (as the input nodes are not inter connected and hence independent of each other):

$$P(\widehat{\mathbf{x}}|\mathbf{h}) = \prod_{i=1}^{m} P(\hat{x}_i|\mathbf{h}). \tag{9}$$

Again, we compute the hidden nodes' conditional probabilities $P(\mathbf{h}|\widehat{\mathbf{x}})$ based on the reconstructed $\widehat{\mathbf{x}}$, by following Eq. (8), and perform Gibbs sampling to sample an hidden feature vector $\widehat{\mathbf{h}}$. Then, the element-wise weights between the input and hidden layers of the network are updated according to:

$$w_{ji} = w'_{ji} + \alpha * (x_j * h_i - \widehat{x}_j * \widehat{h}_i), \tag{10}$$

wherein α denotes a pre-specified learning rate. Also, the respective biases of the input and hidden nodes can be updated as:

$$\mathbf{b_x} = \alpha * (\mathbf{x} - \widehat{\mathbf{x}}) \tag{11}$$

$$\mathbf{b_h} = \alpha * (\mathbf{h} - \widehat{\mathbf{h}}). \tag{12}$$

We have opted for a single-step CD approach wherein the sampling (input and hidden feature representations) is done only once (in both the visible layer and hidden layer) per update [27]. It should be noted that while the dimension of $\mathbf{x}$ remains the same throughout the training, the dimension of $\mathbf{h}$ keep varying according to the number of hidden neurons at the current instance of training. The weight and bias update should be done for each training sample instance if the constraints stipulated in Section 2.1 are met. The entire online learning procedure for O-RBM training is summarized in Algorithm 1.

Algorithm 1 Online Learning of RBM

Step 0: Given A streaming train data set, $\{\mathbf{x}^1, \ldots, \mathbf{x}^t, \ldots \mathbf{x}^N\}$, For each sample $\mathbf{x}^i$ in the training set, do the following:
Step 1: Choose an optimal value for the novelty threshold E_n, and the marginal representation threshold E_m.
Step 2: Obtain a sample hidden vector representation by sampling the distribution obtained by Eq. (8).
Step 3: Back project the sampled $\mathbf{h}$ to obtain a sample of the reconstructed input distribution $\widehat{\mathbf{x}}$, as given by Eq. (9).
Step 4: Compute the reconstruction error between $\mathbf{x}$ and $\widehat{\mathbf{x}}$, as given by Eq. (2). If a new hidden neuron need to be added and / or the hidden neurons need to be updated, go to **Step 5**, else go to the next sample.
Step 5: For $\widehat{\mathbf{x}}$, compute the hidden vector distribution by following Eq. (8) and perform Gibb's sampling (once) to get $\widehat{\mathbf{h}}$.
Step 6: The resultant gradient computation and the element-wise weight update can be done by following the gradient ascent given by Eq. (10). The corresponding bias update is given by Eq. (11).
Step 7: Repeat **Step 1** to **Step 6** sequentially until all the training samples are considered.

2.3. Discriminative training

Next, we briefly discuss the discriminative training, where the feature representation learned during the online generative phase is mapped to the conditional class distributions in a supervised fashion.

The responses of the K neurons in the hidden layer are as below:

$$\mathbf{h} = [h_1 \ldots h_k]^T \tag{13}$$

This feature representation is then used in a supervised discriminative training phase to learn the conditional probability distribution $P(c^t|\mathbf{x}^t)$. The class labels c^t are encoded in $\mathbf{y}^t = [y_1^t, \ldots, y_s^t]$, as below:

$$y_i^t = \begin{cases} 1 & \text{if } c^t = i, \\ 0 & \text{otherwise.} \end{cases} \quad i = 1, \ldots, s; \tag{14}$$

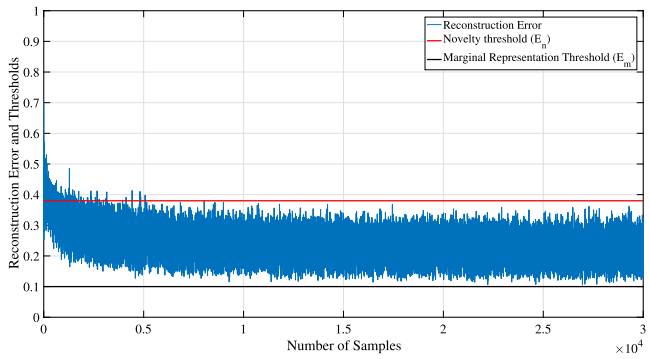
The objective of discriminative training is to minimize the log probability

$$\min_{\mathbf{w}_{ki}^2} \frac{1}{N}\sum_{n\in N} \mathcal{L}_{disc}\left(\mathbf{y}^n|\mathbf{x}^n\right), \tag{15}$$

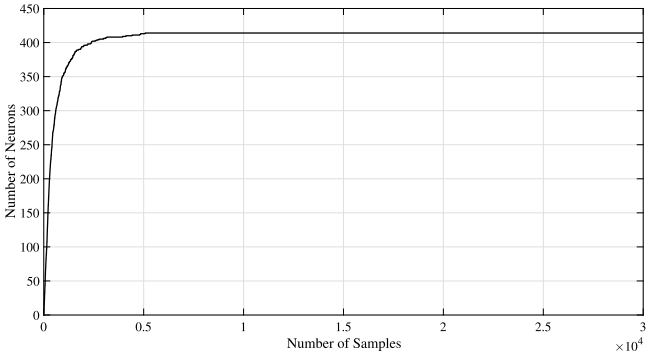
where $\mathcal{L}_{disc}(\mathbf{y}^n|\mathbf{x}^n)$ is a measure of error between $\mathbf{y}^n$ and $\widehat{\mathbf{y}}^{\mathbf{n}}$, and $\mathbf{w}_{ki}^2$ are the weights connecting the kth output neuron and the ith hidden neuron. Here, we perform discriminative training through 10 epochs of supervised training using a Multi-Layer Perceptron (MLP) with sigmoidal activation function.

3. Demonstration of O-RBM learning

We now demonstrate the progression of learning within the proposed O-RBM approach, and make some observations about



(a) MNIST: Reconstruction Error of Samples 1 through 30,000.



(b) MNIST: Growth of the Network with Samples 1 through 30,000.

Fig. 3. MNIST: Reconstruction error and addition of neurons to the generative RBM network through the online learning.

the algorithm. We characterize the algorithm on the MNIST data set [28], as it is a large well-explored multi-category dataset (60,000 training samples, 10 categories). The network is trained in an online fashion, using the training data set. The validity of the trained network is established independently on the test set (10,000 samples, 10 categories) in an offline fashion.

Figs. 3(a) and 3(b) show the evolution of reconstruction error and network architecture, as samples stream in for training. Fig. 3(a) shows that the reconstruction error is high for the initial samples. This is because the model is at infancy and is beginning to learn. Hence, most samples are novel to the network, resulting in neurons being added (see Fig. 3(b)). However, as training progresses, the network learns a sufficient representation of the data and the reconstruction error reduces progressively, resulting in fewer neurons being added to the network. It is evident from Figs. 3(a) and 3(b) that the online generative phase converges to a stable, concise network architecture, and the generative training is complete in about 26 min. It is also evident from Fig. 3(b) that 90% of the neurons in the stable network are added for the first 10% of the training samples (i.e., the first 5000 samples). The remaining 90% of the training samples (i.e., the latter 55000 samples) contribute only about 10% of the neurons in the stable network. Next, we conduct a study on the choice of the novelty threshold (E_n). We do this by training the network using the training data set, varying the novelty threshold from 0.1 to 0.5. We hold a validation data set of 1000 samples, by sampling 100 samples from each class in the testing data set. With the addition of each neuron, the reconstruction error of a subset of the testing data set is computed. It can be observed from Fig. 4 that the choice of E_n affects the convergence, especially, because the network learns in a single pass. Setting a high threshold for E_n results in too few neurons, and vice-versa, with poor learning in either case. On the other hand, setting E_n in the range of $0.25-0.3$ results in an optimal architecture, and optimal learning.

We next validate the effect of the sequence in which the training data is presented, on the performance of the algorithm. We train the O-RBM independently for 50 randomly constructed sequences of the MNIST training samples. In each case, we present different sequences of the training data set to train the network. Across the 50 training trials, the classification accuracy on the testing data set is $97 \pm 2\%$, and the final number of neurons was 403 ± 26. Thus, changing the sequence of presentation of training samples does not change the accuracy or the network architecture significantly, showing that the network is able to generalize well with a concise network architecture.

To evaluate the discriminative potential of the feature representation learned during the online generative training phase, we relate the number of 'novel' samples (where $E^t_{recon} > E_n$) to their corresponding class labels c^t for each of the 50 trials. Fig. 5 shows

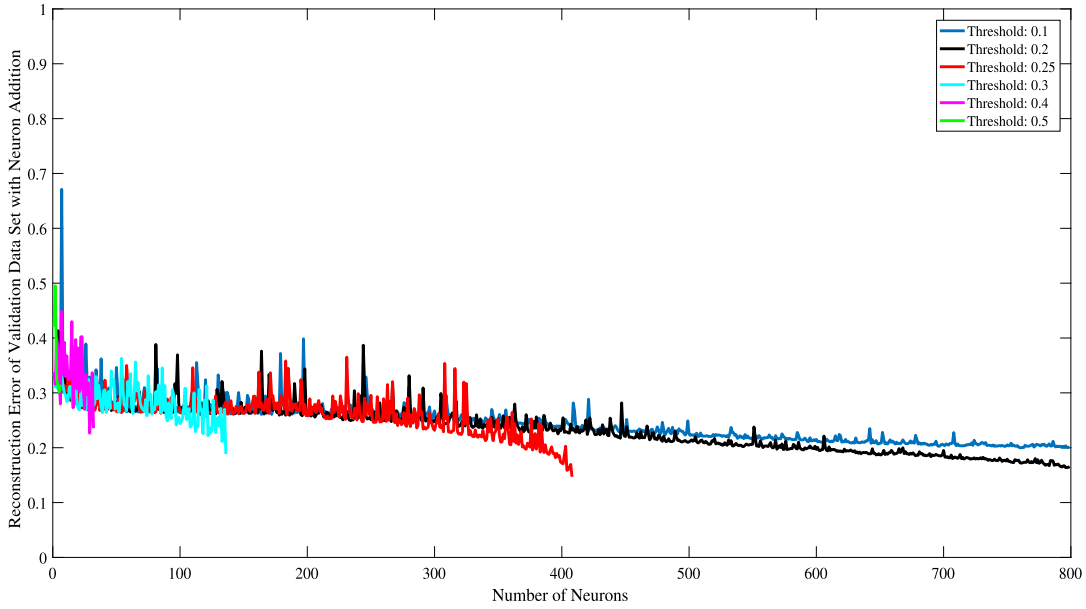


Fig. 4. Choice of threshold E_n of O-RBM for MNIST data set.

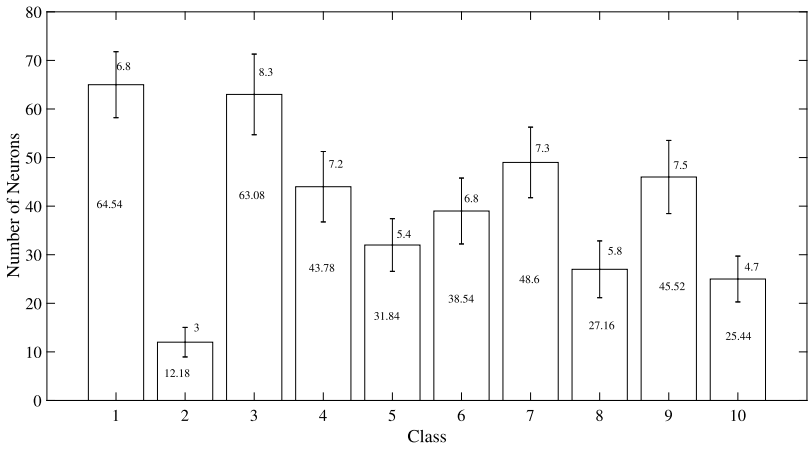


Fig. 5. MNIST Classification: Average number of hidden layer neurons associated with each class of the MNIST dataset, with the standard deviation across 50 trials.

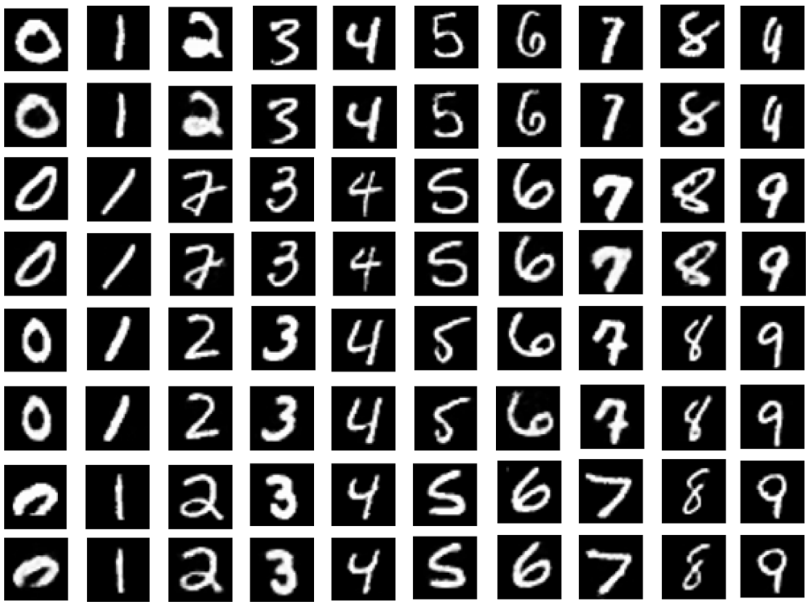


Fig. 6. MNIST Classification: Reconstruction of the handwritten digits using the online generative network. The odd numbered rows represent the original image, while the even numbered rows are the images reconstructed by the RBM network, trained with the O-RBM.

the average number of hidden layer neurons associated with each class of the MNIST dataset, with the standard deviation across the 50 trials. These results show that the individual neurons in the trained network are inherently and collectively discriminative to the class labels, despite the unsupervised nature of the training. Further, we observe that the variability across trials is a small proportion of the average number of neurons in each class, suggesting that the neuron-to-class associations are largely independent of the sequence of training data samples.

To demonstrate the accuracy of reconstructions, we present a subset of the reconstructed image, in comparison to the original image of the MNIST data sets in Fig. 6. It can be observed from the Figure that despite single pass learning, the O-RBM is capable of accurate reconstructions.

4. Experimental performance analysis

As online learning algorithms are particularly suitable for streaming data applications with evolving stream of data, we analyze the performance of the Online Restricted Boltzmann Machine with those of batch learning and online learning techniques for single hidden layer neural networks. Specifically, we evaluate the performances of O-RBM on two problems, namely, the problem of credit scoring of borrowers and for a predictive maintenance of an aircraft component using time series sensor data. We compare the various classifiers on the two problems based on the network size and the performance measures such as the overall efficiency (η_O), the average efficiency (η_A), True Positive Rate (TPR), True Negative Rate (TNR), and Geometric mean accuracy (Gmean) defined as:

$$\eta_O = \frac{\sum_{i=1}^{s} q_{ii}}{N} \text{X}100\% \tag{16}$$

$$\eta_A = \frac{1}{s}\sum_{i=1}^{s} \frac{q_{ii}}{N_i} \text{X}100\% \tag{17}$$

$$\text{TPR} = \frac{\text{Number of TP}}{\text{Number of TP} + \text{Number of FN}} \tag{18}$$

$$\text{TNR} = \frac{\text{Number of TN}}{\text{Number of TN} + \text{Number of FP}} \tag{19}$$

$$\text{Gmean} = \sqrt{\text{TPR} \times \text{TNR}} \tag{20}$$

Table 1
Description of the credit scoring data sets.

Data set	Input features	Size of Data set	*I.F.*
UCI German	24	1000	0.4
UCI AUS	14	690	0.1101
KAGGLE GMSC	10	150000	0.86632

Here, q_{ii} is the number of correctly classified samples in class i and denotes the diagonal elements of the Confusion matrix $Q \in \Re^{s\times s}$ of s classes. The number of samples in class i is denoted by N_i.

4.1. Performance analysis on benchmark data sets: Credit fraud analytics

Credit scoring is the problem of estimating the probability that borrower might default and/or exhibit undesirable behavior in the future. Credit patterns of individual borrowers are (a) bound to evolve over time, with huge inter-personal variability across borrowers, and (b) often characterized by severe imbalances in the data, as the defaulters are few and between. Online Restricted Boltzmann Machine provides a better premise to represent such evolving time series data with severe class imbalances, owing to its ability to detect novel samples better.

Several studies have employed batch machine learning techniques for credit scoring [3,29–31]. We perform analogous evaluations to benchmark our online learning algorithm in relation to these batch learning techniques. Specifically, we perform credit fraud prediction using three publicly available data sets, namely, the UCI German credit data set (UCI German), the UCI Australian credit data set (UCI AUS), and the KAGGLE 'Give me some credit' data set (KAGGLE GMSC). We evaluate the O-RBM classifier, in comparison with the Support Vector Machine classifier (SVM), the Multi-layer Perceptron Neural Network (NN) classifier, the Classification Restricted Boltzmann Machine classifier (ClassRBM) [32] and the Scoring Table (ST) method on the three credit data sets (listed in Table 1) in Table 2.

Table 1 details the public credit scoring data sets, along with the number of classes, the number of training and testing samples, and their imbalance factors (*I.F.*):

$$I.F. = 1 - \frac{s}{N}\min_{i=1\cdots s} N_i, \tag{21}$$

where s is the total number of classes and N_i is the number of samples in class i. It is evident that the 3 public datasets have varying degrees of class-imbalance. While the UCI AUS is mildly imbalanced, the UCI German is partially imbalanced and the KAGGLE GMSC has very high imbalance across classes. This varying degree of class-imbalance provides a unique opportunity to characterize the neuron distribution across classes in the online learning framework. We filled in the missing values in the Kaggle 'Give me some credit' data set by averaging across similar participants in the population, grouped according to ages in intervals of 10.

We now present the results of O-RBM in relation to the batch learning and the state-of-the-art online learning techniques for a single hidden layer network in Table 2. For batch learning techniques, we compare the performances of O-RBM with those of Support Vector Machines, Neural Networks and Classification Restricted Boltzmann Machines [3]. On the other hand, the online learning performance of O-RBM is benchmarked against the Projection Based Learning algorithm of a Radial Basis Function Network (PBL-RBFN). We reproduce previously obtained batch learning results using the SVM, NN, ClassRBM and ST classifiers from [3]. Although the ClassRBM results in [3] are reported with fixed architecture of 100 neurons with a batch size of 100 and learning rate of 0.0001, the architecture of the other classifiers is not specified. Further, the training accuracies of the classifiers have also not been reported. Hence, we perform

Table 2
Performance analysis on benchmark data sets: Credit scoring.

Data set	Classifier	K	Training		Testing				
			η_O	η_A	η_O	η_A	TPR	TNR	Gmean
UCI German	SVM	534	76.429	66.679	74.667	61.378	0.3255	0.8878	0.54
	SVM[a]	–	–	–	–	–	0.484	0.867	0.65
	NN	60	98.571	97.573	72.333	65.105	0.4574	0.8446	0.62
	NN[a]	–	–	–	–	–	0.517	0.814	0.65
	ClassRBM	80	77.428	63.346	74.000	56.738	0.4418	0.8271	0.60
	ClassRBM[a]	100	–	–	–	–	0.479	0.872	0.65
	ST[a]	–	–	–	–	–	0.67	0.68	0.68
	PBL-McRBFN	100	80.14	81.78	67.33	69.11	0.65	0.73	0.69
	O-RBM	48 (32:16)	79	74.2	76.5	71.69	0.60	0.83	0.71
UCI AUS	SVM	192	85.507	86.263	85.507	86.048	0.7946	0.9263	0.8579
	SVM[a]	–	–	–	–	–	0.913	0.71	0.850
	NN	60	94.824	94.767	84.058	83.727	0.7917	0.8828	0.836
	NN[a]	–	–	–	–	–	0.850	0.857	0.854
	ClassRBM	50	86.128	86.391	85.507	86.021	0.8953	0.8264	0.8602
	ClassRBM[a]	100	–	–	–	–	0.880	0.847	0.863
	ST[a]	–	–	–	–	–	0.828	0.805	0.816
	PBL-McRBFN	34	85.71	86.26	81.40	80.60	0.851	0.761	0.80
	O-RBM	38 (20:18)	86.68	86.8	88.49	89	0.92	0.86	0.89
KAGGLE GMSC	SVM	6340	69.970	59.430	72.240	60.018	0.5771	0.8982	0.72
	SVM[a]	–	–	–	–	–	0.114	0.994	0.336
	NN	60	63.896	62.287	74.200	63.017	0.6165	0.8792	0.7363
	NN[a]	–	–	–	–	–	0.229	0.986	0.475
	ClassRBM	100	75.687	74.048	86.160	74.789	0.6	0.8975	0.73
	ClassRBM[a]	100	–	–	–	–	0.182	0.991	0.424
	ST[a]	–	–	–	–	–	0.515	0.622	0.566
	PBL-McRBFN	11	57.04	53.31	80.52	53.18	0.22	0.85	0.43
	O-RBM	13 (3:10)	76.08	74.49	86.25	75.22	0.63	0.88	0.74

[a]Reproduced from [3].

Table 3
Statistics of flights data used.

Failure event	Event 1	Event 2	Event 3	Event 4	Event 5
No. of flights	1	58	215	28	8
No. of samples	84	2936	11580	1920	584
I.R. (%)	0.90	0.95	0.90	0.78	0.79

independent evaluations using SVM, NN, and ClassRBM classifiers, to report an additional performance validation beyond the previously reported results.

From the table, we can observe that the online learning algorithms require fewer neurons, in general, compared to the batch learning techniques. This can be attributed to the ability of online learning algorithms to evolve their network architecture depending on the novelty of samples in the data. Further, the online learning algorithms are capable of generalizing well with better performance on the testing data, compared with the other algorithms. Thus, although the training accuracies of the O-RBM is comparable to those of the batch learning techniques, they have better ability to generalize on a held-out test set. The ability of the O-RBM to represent the minority class is exemplary, in comparison to the batch learning techniques, especially, on the very highly imbalanced KAGGLE GMSC data set. It is encouraging to observe that in general, the O-RBM is better than the state-of-the-art single hidden layer neural network with batch learning algorithm by at least 2%.

4.2. Performance analysis on a real-world application: Predictive maintenance of an aircraft component

Although predictive maintenance solutions for aircraft engine is quite well-established [33,34], there are ongoing efforts in predicting failures of other components that are critical to the operation of the aircraft. In this work, we leverage on data acquired from sensors on the aircraft and recorded in the Quick Access Recorder, to develop predictive maintenance solution for an aircraft Component. In this analysis, we leverage on the operational QAR data from an aircraft fleet, over a period of 2 months pivoted around 5 recorded failure events. In doing so, we dismiss data during manual shutdown of the component within the 2 month period.

The problem of predicting the health of the aircraft component is cast as a predictive maintenance problem, where the O-RBM is employed to represent the distinctions between signal patterns of the component in healthy and failure mode. Table 3 presents the number of flights considered in each event, the number of samples from each event, and the imbalance ratio of the samples in each event. The following observations can be made about the data from the table:

- Event 3 has the highest number of flights and samples. This event alone contributes to more than twice the sum of flights (and samples) of all the other events.
- Although Event 1 has only one flight, it contributes to 84 samples.
- The imbalance ratio of the data in each of these events is very high, with event 2 having the highest imbalance. It must be noted that the valve was manually shutdown when a failure was reported, thus, limiting the volume of usable data in failure mode.

We perform 5 fold Cross Validation (CV), with each of the failure events reported in Table 3 in the test set, and the data from the other failure events in the training data set. We present the results of the O-RBM in comparison with state-of-the-art batch learning single hidden layer networks on this 5-fold cross validation study in Table 4. From the table, it can be observed that although the training accuracies of all the classifiers are comparable, the O-RBM has better generalization ability than the other classifiers used in comparison. Further, the O-RBM requires fewer neurons than the other single hidden layer classifiers, to represent and classify the same imbalanced data set. On the highly imbalanced data set obtained from event 2, the O-RBM has substantially higher testing accuracies than the remaining classifiers. This high testing accuracies can be attributed to the ability of the O-RBM to detect novel samples and represent them accurately. Thus, we observe that on the two sets of problems, namely the credit scoring and the predictive maintenance of aircraft components, O-RBM performs substantially better than state-of-the-art batch learning algorithms, and requires only fewer network resources to represent the data.

Generally, we could make the following inferences from the performance study results on the various problems:

- **Network Size:** Overall, the O-RBM network uses fewer neurons than the other single hidden layer classifiers used in comparison. This is because the O-RBM uses the most novel samples to add neurons to the network, and the neurons are well-representative of the data set.
- **Performance Measures:** Despite having a compact architecture, the proposed O-RBM performs better than all the classifiers used in comparison. This could be attributed to the fact that the learnt distributions represent the data very well. Moreover, while the other algorithms learn the data in batches, and updates gradients in batches, the O-RBM updates gradients based on every sample in the data set.
- **Neuron Distribution Per Class:** Unlike the batch learning algorithms that need *a priori* assumption of the architecture, the O-RBM builds the network as learning progresses. This helps us to infer the number of neurons per class that may help to characterize the distribution of the samples in each class.
- **Effect of Class-Imbalance:** Classes with fewer samples require more neurons for sufficient feature representation. As the class imbalance increases, a greater proportion of the hidden layer neurons is associated with less prevalent classes. This adaptation is a natural consequence of the online learning process, and differentiates our approach from the batch learning algorithms.

5. Conclusion

We introduced a novel Online Restricted Boltzmann Machines (O-RBM) framework that evolves a network architecture in a fully bottom-up online manner as data streams in. We demonstrated that the algorithm converges to a stable compact network architecture wherein (a) hidden layer neurons are implicitly associated with class labels (despite unsupervised training), and (b) classification performance are invariant to the sequence in which the training data samples are presented. Further, O-RBM performed better than batch techniques in credit score classification with streaming data — specifically online learning achieved better accuracy with fewer neurons and showed the unique ability to adapt to class imbalance. Designing online learning frameworks for multilayered deep belief network is currently under investigation. We believe that this work will foster online learning researches in areas that include Convolutional Neural Network (CNN) based supervised and unsupervised learning, generative and interpretable models.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Table 4
Performance study: Predictive maintenance of an aircraft component.

Data set	Classifier	K	Training		Testing				
			η_O	η_A	η_O	η_A	TNR	TPR	Gmean
Event 1	SVM	7879	90.63	90.49	98.81	99.34	100	98.68	0.99
	NN	200	96.63	87.05	100	100	100	100	1
	ClassRBM	80	97.00	89.02	100	100	100	100	1
	PBL-McRBFN	8	89.46	78.76	100	100	100	100	1
	O-RBM	15	96.95	88.86	100	100	100	100	1
Event 2	SVM	6740	91.62	91.52	91.55	78.64	93.12	64.15	0.77
	NN	200	97.16	91.70	95.40	76.52	97.69	55.35	0.74
	ClassRBM	180	97.26	91.00	95.33	75.89	97.70	54.09	0.73
	PBL-McRBFN	6	91.76	85.18	93.21	84.23	95.48	72.98	0.83
	O-RBM	15	97.18	91.8	94.55	82.59	96.00	69.18	0.82
Event 3	SVM	1906	96.66	96.70	86.80	79.56	88.62	70.50	0.79
	NN	120	98.19	95.38	88.78	77.01	91.75	62.26	0.76
	ClassRBM	70	98.46	97.27	83.35	79.05	84.43	73.67	0.79
	PBL-McRBFN	250	90.13	88.96	93.64	82.76	96.39	69.13	0.82
	O-RBM	11	98.38	96.98	85.96	79.81	87.50	72.13	0.79
Event 4	SVM	7109	89.52	89.03	94.27	87.96	99.27	76.65	0.87
	NN	120	96.91	88.43	95.31	89.56	99.87	82.35	0.80
	ClassRBM	70	96.75	88.19	95.37	89.84	99.73	79.25	0.89
	PBL-McRBFN	69	94.75	77.40	99.12	98.76	99.4	98.11	0.98
	O-RBM	41	96.71	87.31	95.68	96.74	96.75	96.72	0.97
Event 4	SVM	7741	90.33	90.13	96.75	94.89	93.07	96.72	0.95
	NN	120	97.06	88.08	96.92	96.24	97.40	95.08	0.96
	ClassRBM	80	96.36	83.95	96.91	96.24	97.4	95.08	0.96
	PBL-McRBFN	7	94.19	80.28	96.75	96.74	96.75	96.72	0.96
	O-RBM	35	96.98	88.28	97.26	97.06	97.4	96.72	0.97

CRediT authorship contribution statement

Ramasamy Savitha: Conceptualization, Formal analysis, Methodoloy, Software, Validation, Writing - original draft. **ArulMurugan Ambikapathi:** Conceptualization, Formal Analysis, Investigation, Writing - original draft, Writing - review & editing. **Kanagasabai Rajaraman:** Conceptualization, Supervision, Writing - review & editing.

Acknowledgments

The first author would like to thank the 'A*STAR Science and Engineering Research Council, Singapore Call for Data Analytics Proposals grant (Grant Number: 1727600058)' for funding this study. We would like to thank Singapore Airlines for sharing the Quick Access Recorder data to complete the study. We thank Dr Vijay Chandrasekhar, Dr Pavitra Krishnaswamy and Dr Senthilnath Jayavelu of Institute for Infocomm Research, A*STAR, Singapore, for the discussions during the work.

References

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Adv. Neural Inf. Process. Syst. 25 (2012).
[2] J.T. Turner, A. Page, T. Mohsenin, T. Oates, Deep Belief Networks used on high resolution multichannel Electroencephalography data for seizure detection, in: AAAI Spring Symposium, 2014.
[3] M.J. Tomczak, M.B. Zie, Classification restricted Boltzmann machine for comprehensible credit scoring model, Expert Syst. Appl. 42 (4) (2015) 1789–1796.
[4] A. Robins, Sequential learning in neural networks: A review and a discussion of pseudorehearsal based methods, J. Intell. Data Anal. 8 (3) (2004) 301–322.
[5] Z.C. Lipton, J. Berkowitz, C. Elkon, A critical review of recurrent neural networks for sequence learning, 2015, arXiv preprint arXiv:1506.00019.
[6] T.G. Dietterich, Structure, Synctactic and Statistical Pattern Recognition, Springer-Verlag, 2002, pp. 15–30, Ch. Machine learning for sequential data: A review.
[7] G. Bouchard, T. Trouillon, J. Perez, A. Galdon, Online learning to sample, 2016, arXiv preprint arXiv:1506.09016v2.
[8] K. Subramanian, S. Suresh, R. Savitha, A metacognitive complex-valued interval type-2 fuzzy inference system, IEEE Trans. Neural Netw. Learn. Syst. 25 (9) (2014) 1659–1672.
[9] S. Suresh, R. Savitha, N. Sundararajan, A sequential learning algorithm for complex-valued resource allocation network-CSRAN, IEEE Trans. Neural Netw. 22 (7) (2011) 1061–1072.
[10] G. Zhou, K. Sohn, H. Lee, Online incremental feature learning with denoising autoencoders, in: Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics, Vol. 22, AISTATS-12, 2012, pp. 1453–1461.
[11] S. Han, Z. Meng, A.S. Khan, Y. Tong, Incremental boosting convolutional neural network for facial action unit recognition, Neural Inf. Process. Syst. (2016).
[12] G. Chen, R. xu, S. Srihari, Sequential labeling with online deep learning, 2015, arXiv preprint arXiv:1412:3397v3.
[13] M.A. Cote, H. Larochelle1, An infinite restricted Boltzmann machine, 2016, arXiv:1502.02476v4.
[14] T. Lesort, H. Caselles-Dupré, M. Garcia-Ortiz, A. Stoian, D. Filliat, Generative models from the perspective of continual learning, in: Advances in Neural Information Processing Systems, 2017.
[15] J. Schwarz, J. Luketina, W.M. Czarnecki, A. Grabska-Barwinska, Y.W. Teh, R. Pascanu, R. Hadsell, Progress and compress: A scalable framework for continual learning, 2018, arXiv preprint arXiv:1805.06370v2.
[16] G.I. Parisi, R. Kemker, J.L. Part, C. Kanan, S. Wermter, Continual lifelong learning with neural networks: A review, Neural Netw. 113 (2019) 54–71.
[17] A.A. Rusu, N.C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, R. Hadsell, Progressive neural networks, 2016, ArXiv Preprint arXiv:1606.04671.
[18] T.J. Draelos, N.E. Miner, C.C. Lamb, J.A. Cox, C.M. Vineyard, K.D. Carlson, W.M. Severa, C.D. James, J.B. Aimone, Neurogenesis deep learning, 2016, ArXiv Preprint arXiv:1612.03770.
[19] G.I. Parisi, J. Tani, C. Weber, S. Wermter, Lifelong learning of humans actions with deep neural network self-organization, Neural Netw. 96 (2017) 137–149.
[20] G.I. Parisi, J. Tani, C. Weber, S. Wermter, Lifelong learning of spatiotemporal representations with dual-memory recurrent self-organization, 2018, ArXiv Preprint arXiv:1805.10966.
[21] H. Li, S. Enshaeifar, F. Ganz, P. Barnaghi, Continual learning in deep neural network by using a kalman optimiser, 2019, ArXiv Preprint arXiv:1905.08119v3.
[22] N.L. Roux, Y. Bengio, Representational power of restricted Boltzmann machines and deep belief networks, Neural Comput. 20 (6) (2008) 1631–1649.
[23] P.A. Szerlip, G. Morse, J.K. Pugh, K.O. Stanley, Unsupervised feature learning through divergent discriminative feature accumulation, in: Proc. of the Twenty-Ninth AAAI Conference on Artificial Intelligence, AAAI-2015, AAAI Press, Menlo Park, CA, 2015.
[24] T. Poggio, S. Voinea, L. Rosasco, Online learning, stability, and stochastic gradient descent, 2011, arXiv preprint arXiv:1105.4701v3.

[25] G.E. Hinton, Training products of experts by minimizing contrastive divergence, Neural Comput. 14 (8) (2002) 1771–1800.
[26] G.E. Hinton, Training products of experts by minimizing contrastive divergence, Neural Comput. 14 (2002) 1771–1800.
[27] Y. Bengio, O. Delalleau, Justifying and generalizing contrastive divergence, Neural Comput. 21 (6) (2009) 1601–1621.
[28] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[29] B. Baesens, T.V. Gestel, S. Viaene, M. Stepanova, J. Suykens, J. Vanthienen, Benchmarking state-of-the-art classification algorithms for credit scoring, J. Oper. Res. Soc. 54 (6) (2003) 627–635.
[30] S. Lessmann, B. Baesens, H.-V. Seow, L.C. Thomas, Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research, European J. Oper. Res. 247 (1) (2015) 124–136.
[31] D.J. Hand, W.E. Henley, Statistical classification models in consumer credit scoring: A review, J. R. Stat. Soc. A 160 (1997) 523–541.
[32] H. Larochelle, M. Mandel, R. Pascanu, Y. Bengio, Learning algorithms for the classification restricted Boltzmann machine, J. Mach. Learn. Res. 13 (2012) 643–669.
[33] H. Powrie, C. Fisher, Engine health monitoring: Towards total prognostics, in: Proc. of the IEEE Aerospace Conference, Vol. 3, 1999, pp. 11–20.
[34] J. Yu, Aircraft engine health prognostics based on logistic regression with penalization regularization and state-space-based degradation framework, Aerosp. Sci. Technol. 68 (2017) 345–361.