Winter 2018

Deep Recurrent Learning for Efficient Image Recognition Using Small Data

Mahbubul Alam
Old Dominion University

Recommended Citation

Alam, Mahbubul. "Deep Recurrent Learning for Efficient Image Recognition Using Small Data" (2018). Doctor of Philosophy (PhD), dissertation, Electrical/Computer Engineering, Old Dominion University, DOI: 10.25777/1527-2348
https://digitalcommons.odu.edu/ece_etds/43

DEEP RECURRENT LEARNING FOR EFFICIENT IMAGE RECOGNITION USING SMALL DATA

by

Mahbubul Alam
B.S. November 2008, Jahangirnagar University
M.S. December 2011, Jahangirnagar University

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

ELECTRICAL AND COMPUTER ENGINEERING

OLD DOMINION UNIVERSITY
December 2018

ABSTRACT

DEEP RECURRENT LEARNING FOR EFFICIENT IMAGE RECOGNITION USING SMALL DATA

Mahbubul Alam
Old Dominion University, 2018
Director: Dr. Khan M. Iftekharuddin

Recognition is fundamental yet open and challenging problem in computer vision. Recognition involves the detection and interpretation of complex shapes of objects or persons from previous encounters or knowledge. Biological systems are considered as the most powerful, robust and generalized recognition models. The recent success of learning based mathematical models known as artificial neural networks, especially deep neural networks, have propelled researchers to utilize such architectures for developing bio-inspired computational recognition models. However, the computational complexity of these models increases proportionally to the challenges posed by the recognition problem, and more importantly, these models require a large amount of data for successful learning. Additionally, the feedforward-based hierarchical models do not exploit another important biological learning paradigm, known as recurrency, which ubiquitously exists in the biological visual system and has been shown to be quite crucial for recognition.

Consequently, this work aims to develop novel biologically relevant deep recurrent learning models for robust recognition using limited training data. First, we design an efficient deep simultaneous recurrent network (DSRN) architecture for solving several challenging image recognition tasks. The use of simultaneous recurrency in the proposed model improves the recognition performance and offers reduced computational complexity compared to the existing hierarchical deep learning models. Moreover, the DSRN architecture inherently learns meaningful representations of data during the training process which is essential to achieve superior recognition performance. However, probabilistic models such as deep generative models are particularly adept at learning representations directly from unlabeled input data. Accordingly, we show the generalization of the proposed deep simultaneous recurrency concept by developing a probabilistic deep simultaneous recurrent belief network (DSRBN) architecture which is more efficient in learning the underlying representation of the data compared to the state-of-the-art generative models. Finally, we propose a deep recurrent learning framework for solving the image recognition task using small data. We incorporate Bayesian statistics to the DSRBN generative model to propose a deep recurrent generative Bayesian model that addresses the challenge of learning from a small amount of data. Our findings suggest that the proposed deep recurrent Bayesian framework demonstrates better image recognition performance compared to the state-of-the-art models in a small data learning scenario. In conclusion, this dissertation proposes novel deep recurrent learning pipelines, which utilize not only limited training data to achieve improved image recognition performance but also require significantly reduced training parameters.

ACKNOWLEDGMENTS

I want to express my sincere gratitude to my advisor, Dr. Khan M. Iftekharuddin, for giving me the opportunity to work in his research group and providing guidance and advice with enthusiasm and patience. I feel honored that I got the opportunity to learn from such a knowledgeable and visionary person like him. I thank the members of my examining committee including Dr. Chunsheng Xin, Dr. Norou Diawara, and Dr. Hongyi Wu for their time and willingness to advise me and review the dissertation. I respectfully acknowledge my parents for their constant inspiration and encouragement. With all of their favor and sacrifices, I have reached where I am today. Moreover, I would like to humbly acknowledge my uncle, Dr. A. A. Mamun, a world famous dusty plasma physicist, who inspired me to become a researcher from the very beginning of my life. I would also like to show my love and affection to my younger sister Farhana Sharmin for constantly motivating me to finish my Ph.D. My special thanks to Lasitha Vidyaratne and Alexandar Glandon for their suggestions from time to time and becoming true friends of mine while working together in multiple different projects. I would like to appreciate all my colleagues in the Vision Lab for providing a great deal of knowledge and instruction. My Bangladeshi, American and International friends from different countries have been wonderful company and cooperation beyond my studies. Especially, I would like to thank my dearest friend Muhammad Hasib for being on my side in all my difficult situations. I also respectfully thank my elder cousin brother Raquibul Alam Shamim for supporting me like a friend and giving me valuable advice from my childhood. Finally and most importantly, I would like to thank my wife, Zinat Afrose, for her enormous support and patience during my Ph.D. study. At last, I would like to dedicate my dissertation to my beloved son Zaaib Alam who is the source of all my energy, happiness and inspiration to move forward with my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION

The twenty first century has seen rapid growth in computing power and a massive accumulation of human-centric data to an unprecedented scale. These advancements have rejuvenated computational intelligence (CI) that has now become an indispensable part of everyday life. In the past, the scope for CI had been limited to the application of industrial control and robotics. However, recent advancements in CI expand the application towards more complex domains such as computer vision, bio-medical image processing, natural language processing, cyber security and many more. Computer vision has become ubiquitous in our society, with applications in search, image understanding, medical imaging, and autonomous vehicles. The crucial parts of these applications are solving visual recognition tasks such as image recognition. Robust image recognition is a fundamental yet open and challenging problem in computer vision. Research has gradually been evolved for decades after starting with traditional machine learning at the core of intelligent systems to solve complex pattern recognition problems such as image recognition. However, machine learning techniques have limitations in their ability to process natural data or images in raw formats. Different pre-processing steps are used to extract representative features from raw data or images, which are more amenable to machine learning models. This intermediate representation of raw data, also known as the "hand-engineered" feature, requires domain expertise and human interpretation of physical patterns such as texture, shape, and geometry among others. There are two major problems with "hand-engineered" features that impede any major progress in intelligent systems. First, the choice of "hand-engineered" features is application dependent and requires independent evaluation. Second, "hand-engineered" features are extracted from each training sample in a stand-alone manner without the knowledge of inevitable noise and variations in data. Finally, “hand-engineering” of features may perform excellently with some input instances but may completely fail to extract quality features in other instances, which can lead to high variability in visual recognition performance.

A solution to the limitations of “hand-engineered” features has emerged through mimicking functions of biological neurons in artificial neural networks (ANN). The potential of ANNs is recently being exploited with access to large trainable datasets, efficient learning algorithms, and powerful computational resources. These new techniques in machine learning over the last decade are referred to as deep learning [1, 2] which is largely impacting the computer vision domain, especially visual image recognition. The rapid success of deep learning over traditional machine learning may be attributed to three aspects. First, it offers end-to-end trainable architectures that integrate feature extraction, dimensionality reduction, and final classification. Second, useful and intermediate features can be optimally learned from both input examples and classification targets without using one generic feature extractor for all applications. Third, deep learning methods are flexible enough to capture underlying nonlinear relationships between inputs and output targets at a level far beyond the capacity of ‘hand-engineered’ features. However, current state-of-the-art deep learning models have two major limitations. First, the depth, complexity, and training parameters of these models increase proportionally to the challenges posed by the image recognition task. Second, these models require thousands or millions of labeled training examples to achieve a good generalization for solving the image recognition task. The existing research efforts in deep learning for image or visual recognition are designing efficient architectures following the information processing in the primate visual systems. Hence, it is essential to study the working principle of the recognition pipeline in the primate visual system for designing a more efficient deep learning recognition model.

1.1 RECOGNITION IN PRIMATE VISUAL SYSTEMS

An understanding of the neuronal circuits architecture involved in the primate visual system is very important towards building a more efficient computational deep learning recognition model. Recognition in primate visual systems starts from the eyes. A high-level schematic representation of the visual object recognition areas in the primate visual system is shown in Fig. 1.

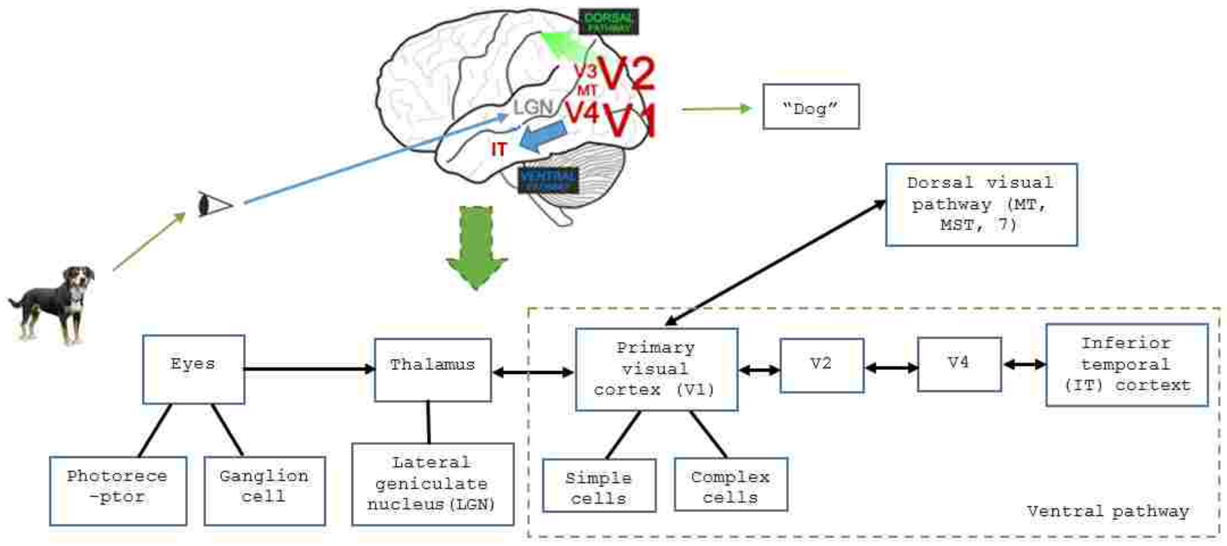


Fig.1. Schematic representation of the visual object recognition areas in the primate visual system. This is a high-level representation of information processing stages along the ventral visual stream.

The photoreceptors in the retina receive a light signal which is considered as input to the visual system. The retina itself includes a complex distributed circuitry to process the incoming visual information into a signal called the retinal ganglion cells [3-5]. This signal is then conveyed to the lateral geniculate nucleus (LGN) in the thalamus which, after processing, sends the signal to the primary visual cortex (V1) [6-8]. V1 is composed of both simple and complex cells [9]. These cells are responsible for extracting low level information from the signal received from the thalamus [10]. Several studies [11-13] have shown that, in addition to receiving feed-forward input from the thalamus, V1 receives feedback signals from higher cortical regions. The information from V1 emerges into two main pathways: *ventral* and *dorsal*. The *ventral* pathway is particularly involved in recognizing objects, whereas, the *dorsal* pathway is responsible for the localization of objects and action towards those objects [14]. In the *ventral* pathway, the information flows from V1 to V2 and then to V4. In addition, V1 also back-projects the signal to the thalamus [11]. In fact, all the visual areas mentioned so far back project the signal to the previous cortical regions [15]. Finally, the processed signal from V4 is sent to the inferior temporal (IT) cortex region of the ventral stream [13]. IT represents the last exclusively visual area where the actual recognition happens through a decision making process. On the other hand, the *dorsal* pathway is thought to perform object localization, and motion detection tasks primarily through medial temporal (MT) and medial superior temporal (MST) regions [16].

1.2 BIOLOGICALLY INSPIRED COMPUTATIONAL VISION MODELS: CURRENT TECHNIQUES AND LIMITATIONS

The early computational recognition models inspired by the biological visual system are developed based on handcrafted techniques [17-20]. These techniques, however, are applicable for solving simple object recognition tasks. Additionally, the features extracted using handcrafted techniques are not general enough to represent different types of objects. To address this limitation, learning-based hierarchical models are developed to extract features directly from the input samples. The fundamental units of these models are artificial neural networks (ANNs) that closely mimic the biological neural architectures found in the primate brain. Moreover, the hierarchical working principle of the learning based recognition models is inspired by the hierarchical information processing observed in the ventral pathway (see Fig. 1) of the primate visual system [21]. Several successful hierarchical ANN (also known as deep neural network (DNNs)) models proposed in the literature are convolutional neural networks (CNNs) [22, 23], deep belief networks (DBNs) [1] and stacked autoencoders (SAEs) [24]. These models are capable of extracting both simple and complex features similar to the ones witnessed in the V1 to V4 regions of the ventral stream. Consequently, they have shown excellent performance in solving several computer vision tasks, especially complex object recognition [25]. However, these hierarchical models utilize thousands of neurons connected via millions of synaptic weights to perform the complex recognition task. This significantly increases the computational complexity of the models and requires a huge amount of computing resources. Also, current state-of-the-art CNN, DBN, and SAE models require thousands or millions of labeled training examples to achieve good generalization for solving the intricate image recognition task. In comparison, human learners usually require just one or a few examples to perform the learning task of a new image category and make meaningful generalizations to novel instances.

The above mentioned biologically inspired hierarchical models share several properties with the biological visual system. However, a prominent limitation is that these models are based on generic feed-forward architectures while in the visual system both local (ventral pathway), and global recurrent connections are abundant.

1.3 CONTRIBUTION OF LOCAL BACK-PROJECTIONS IN VISUAL PATHWAY FOR RECOGNITION

Several neurobiological studies [26-28] have shown the importance of local recurrent back-projections in the visual areas of the animal brain for recognizing objects.
Fig. 1 shows that the visual areas (V1 ↔ V2 ↔ V4 ↔ IT) in the ventral pathway back project information to the earlier areas enabling local recurrent information processing for the object recognition task. Local recurrent processing can be thought of as a top-down process, except that the signal originates from the ventral stream itself [29]. In a recent study, Koivisto et al. [30] reveal strong evidence of recurrent feedback circuits engaged during visual processing. The authors use external stimulation (transcranial magnetic stimulation (TMS)) to temporarily prevent a targeted brain area from responding for a specific time period. The time period is intentionally set to disrupt the recurrent response of the targeted areas. The experiment is conducted using functional magnetic resonance imaging (fMRI)-localized TMS to selectively inactivate V1/V2 for a couple of milliseconds while subjects categorize images. Experimental findings show that applying TMS to V1/V2 greatly impairs subjects' categorization performance. Earlier work from Corthout et al. [31] demonstrates that applying TMS over V1 impairs letter recognition performance. Collectively, these experiments show that the disruption of information processing in early visual areas impairs visual recognition. The above-mentioned biological evidence suggests that local recurrent processing in early visual areas plays an important role in visual recognition.

As mentioned in the previous subsection hierarchical feed-forward based computational deep learning models do not exhibit the local back-projections observed in the ventral stream of the visual system. Therefore, these models can be viewed as crude approximations of the biological recognition system. We believe that a hierarchical/deep model with local recurrent connections may lead to a more biologically plausible and efficient computational recognition model.

1.4 LEARNING FROM SMALL DATA

In typical computer vision applications, such as image recognition, machine learning, especially deep learning techniques, requires thousands, or even millions of training examples to achieve good generalization. Conversely, human learners are capable of effectively performing complex image recognition tasks when the training data are very sparse. In fact, in many cases, only one example is often sufficient for humans to comprehend a novel category and make meaningful generalizations of new instances even when it is not possible to classify precisely [32]. Several studies [33-35] have shown that humans are able to perform accurate classification after observing just three or four examples. Consequently, learning from limited training data is an essential characteristic for computational recognition models such as deep learning. However, current state-of-the-art deep learning techniques such as CNNs, DBNs, and SAEs perform poorly for solving any computer vision problems, especially image recognition in scenarios where labeled training data is scarce. Therefore, in recent years, the challenge of learning from a limited amount of data has drawn increasing attention in the machine learning and deep learning research community. Nevertheless, there are very few known techniques for handling such an intricate task. These techniques are still in the early stage and do not generalize well for solving complex computer vision problems. Therefore, the problem of learning with small data is still an open challenge in the machine learning domain and requires extensive research for designing an efficient deep learning framework.

1.5 PROPOSED WORK AND CONTRIBUTIONS

This dissertation addresses two major limitations of the current state-of-the-art computational deep learning models: 1) the need for a large number of training parameters, and 2) the need a large amount of training data. Consequently, we propose biologically inspired novel deep recurrent learning frameworks for solving complex computer vision tasks such as image recognition using a significantly fewer training parameters and limited training data. The dissertation has contributed to three journal manuscripts and three conference proceedings as follows.

A novel biologically inspired deep simultaneous recurrent network (DSRN) is proposed in order to improve image recognition. The DSRN model utilizes extensive weight sharing in the hidden recurrent layers that significantly reduces the number of trainable parameters. The simultaneous recurrency offers further depth within each layer in addition to the overall deep structure of the network that in turn enables more robust image recognition. Moreover, we show the generalization of the deep simultaneous recurrency concept in a generative model by proposing a deep recurrent generative model known as the deep simultaneous recurrent belief network (D-SRBN). Deep generative models are quite adept at learning meaningful representation from unlabeled data. We design the joint and conditional probability distribution functions required for the proposed D-SRBN model followed by the inference and learning procedure of the D-SRBN. The proposed D-SRBN model achieves superior representation learning performance while utilizing less trainable parameters compared to the state-of-the art generative models. Finally, this dissertation incorporates Bayesian statistics to the proposed DSRBN deep recurrent probabilistic generative model to solve the problem of learning using small data. More specifically, we address the intricate one-shot image recognition task which is a well-known learning problem with small data, using the proposed deep simultaneous recurrent generative Bayesian model. The proposed model achieves better or comparable one-shot image recognition performance compared to state-of-the-art deep learning frameworks while utilizing a significantly reduced fewer training parameters. These findings have been published in IEEE Transactions on Neural Networks and Learning Systems (TNNLS), Neural Networks, Elsevier Journal. Moreover, part of the findings is also published in the proceedings of the International Joint Conference on Neural Networks (IJCNN) and the Society of Photo-Optical Instrumentation Engineers (SPIE). For the first time in the literature, we propose a novel deep learning framework which learns from limited training data while utilizing several orders of reduced training parameters.

1.6 ORGANIZATION OF THE DISSERTATION

The dissertation is organized as follows. Chapter 2 describes the necessary background to understand the proposed deep recurrent learning framework. This chapter covers the brief understanding of artificial neural networks (ANNs), different architectures and the learning procedure, different type of deep learning architectures and their learning scheme, and a machine learning based classification technique such as metric learning, and hierarchical Bayesian models. Chapter 3 discusses our proposed deep simultaneous recurrent learning technique, DSRN, for efficient image recognition. This chapter provides a detailed explanation of the architecture, and learning technique followed by experimental results for the proposed DSRN model. Chapter 4 illustrates our proposed deep simultaneous recurrent generative model, D-SRBN for efficient representation learning to demonstrate the generalization of our proposed DSRN concept. This chapter demonstrates the joint and conditional distribution functions, the learning and inference procedure developed for the D-SRBN model, and experimental results.

Chapter 5 demonstrates the extension of the deep recurrent generative model, D-SRBN with a hierarchical Bayesian technique for solving difficult learning challenges using small data. This chapter provides detailed formulation of our proposed deep recurrent generative Bayesian model, DSRBN-HB. The learning and inference procedure of the proposed DSRBN-HB framework are explained in detail followed by the experimental results. Finally, the dissertation concludes with a summary and future work in Chapter 6.

CHAPTER 2

BACKGROUND OF THE STUDY

This chapter provides a brief discussion of the techniques required to understand this dissertation. We illustrate the fundamental working principle of artificial neural networks (ANNs), different types of ANN architectures and their learning mechanism, a brief discussion on deep neural networks and different architectures along with training machine learning classification techniques such as metric learning, and hierarchical Bayesian models. In the next few subsections, we briefly go over each of these techniques.

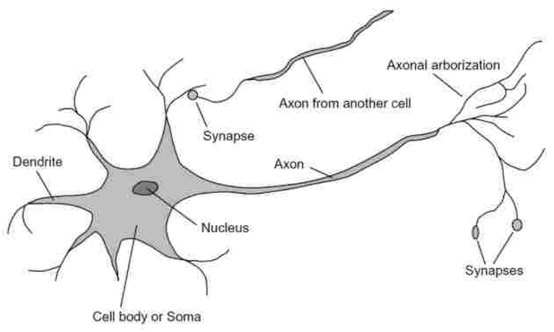
2.1 ARTIFICIAL NEURAL NETWORKS

Artificial neural networks (ANNs) are designed to closely mimic the information processing of biological nervous systems such as the brain. A human or animal brain contains billions of biological neurons that are densely interconnected with each other. Fig. 2 (a) shows a typical example of the biological neuron. It is a simple processing unit (*soma*) that receives and combines signals from other neurons through input paths called *dendrites* which contain *synaptic connections*. An artificial neuron is a computational model inspired by these biological neurons [36]. The artificial neurons are the basic building blocks to design ANNs.

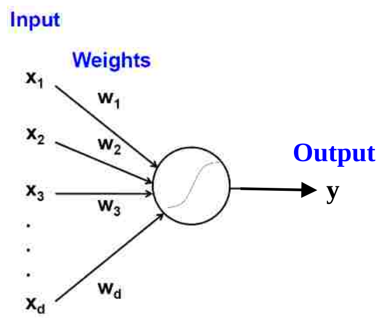
The first computational model of an artificial neuron also known as *perceptron* is proposed by Rosenblatt [38]. Fig. 2 (b) shows an example perceptron where the inputs are denoted by $x_i (i = 1, 2, \dots, d)$ and d denotes the number of inputs. The output y is computed by transforming the weighted sum of the inputs via a non-linear activation function. The mathematical model of a perceptron can be written as,

$$\mathrm{y} = \sigma\left(\sum_{i=1}^{d} w_i \cdot x_i + b\right) \tag{1}$$

where, $w_i's$ represent weight values and $\sum_i w_i = c$ where c represents a small value, b denotes the constant bias value usually set to 1 and σ denotes the non-linear activation function (e.g. *sigmoid*, *hyperbolic tangent*, *rectified linear*).



(a)



(b)

Fig. 2. (a) Biological neuron [37], (b) Artificial neuron or perceptron.

2.2 ARCHITECTURES OF BASIC ANNS

ANNs are modeled using the artificial neurons interconnected with each other and arranged in a layered fashion. Based on the connection between neurons and signal flow, ANNs can be broadly divided into two groups: *feed-forward* and *recurrent*. Using these two basic structures, more complex network architectures such as convolutional neural networks, deep neural networks, and gated recurrent networks are designed.

2.2.1 FEED-FORWARD NEURAL NETWORKS

In a feed-forward neural network (FFNNs), the neurons are arranged in multiple layers. The neurons in one layer are usually fully connected with neurons in the consecutive layer.

Because of this layered architecture, FFNNs are also known as multi-layered perceptrons (MLPs) [39]. Typically, MLPs have three layers: *input*, *hidden* and *output*. An example 3 layer MLP is shown in Fig. 3. MLPs are considered as one of the most popular types of ANNs because of their ability to approximate any function that is sufficiently smooth; hence, they are called universal approximators. The hidden layer of the MLPs can also be expanded into multiple layers for solving more challenging approximation functions.

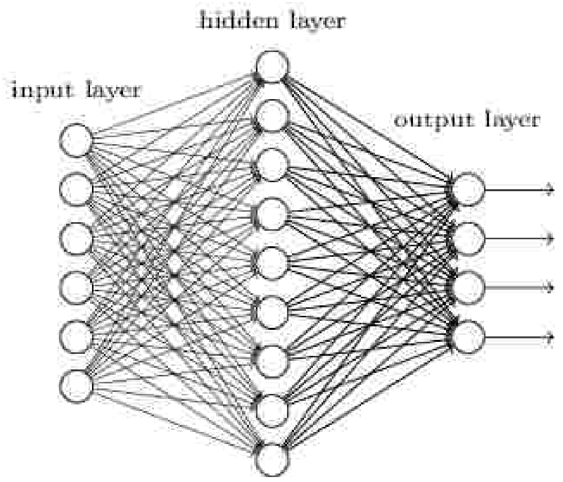


Fig. 3. A simple 3 layer feed-forward network.

2.2.2 RECURRENT NEURAL NETWORKS

FFNNs or MLPs are shown to be efficient for approximating non-linear functions that are used in different applications. However, these networks are unsuitable for resembling the dynamic behavior of a system. In addition, the one directional signal flow of FFNNs ignore the bidirectional recurrent behavior of biological neurons in the brain [40, 41]. Recurrent neural networks (RNNs) achieve this important property by incorporating feedback connections in the

NN architecture. Depending on the nature of recurrent connections, RNNs can be further divided into two categories: *time-lagged recurrent networks* and *simultaneous recurrent networks.*

2.2.2.1 TIME-LAGGED RECURRENT NETWORKS

The structure of time-lagged recurrent neural networks (TLRNs) is similar to that of standard FFNNs, with the distinction that they (RNNs) allow feedback connections in the hidden layer or output layer neurons. The purpose of TLRN is to predict or classify time-varying systems using recurrency as a way to provide memory of the past. Fig. 4 shows the basic topology of a TLRN.

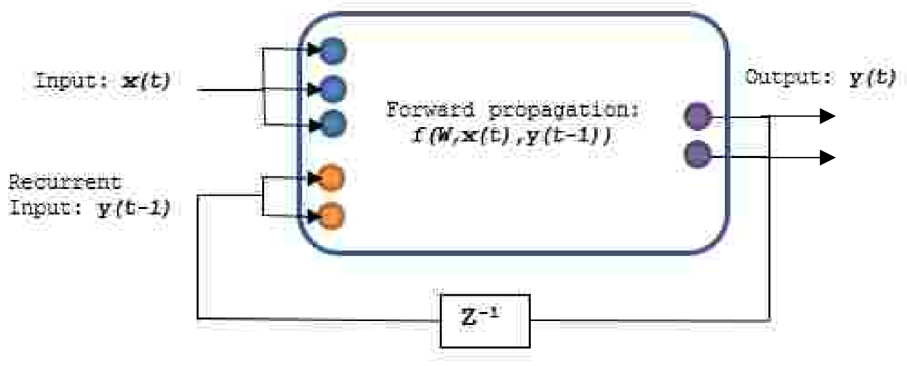


Fig. 4. Basic topology of time-lagged recurrent neural network (TLRN). The current output of the network, *y(t)* is obtained by using the current input *x(t)*, the output from the previous time step *y(t-1)*, and z^{-1} represents unit delay. The forward propagation function of TLRN is represented by *f( )* and the connection weights are denoted by *W*.

As shown in the Fig. 4, TLRNs apply a unit time delay to the output feedback. Commonly used TLRNs are Elman recurrent network [42], Jordan recurrent network [43], long-short term memory (LSTM) network [44], and gated recurrent unit (GRU) [45].

2.2.2.2 SIMULTANEOUS RECURRENT NETWORKS

Observe the graphical representation of a simultaneous recurrent network (SRN) shown in Fig. 5. Though one can draw similarities between SRNs and previously mentioned TLRNs, they are designed to perform fundamentally different tasks. Unlike in TLRN, an input in SRN is applied throughout several iterations (or time steps) and the corresponding output is obtained only after the disappearance of the initial transition and the network stabilizes in an equilibrium state [46, 47]. In other terms, an SRN is based on a FFNN with simultaneous feedback from outputs of the network to its inputs. This simultaneous recurrency is obtained without utilizing any unit time delay in the feedback connections.

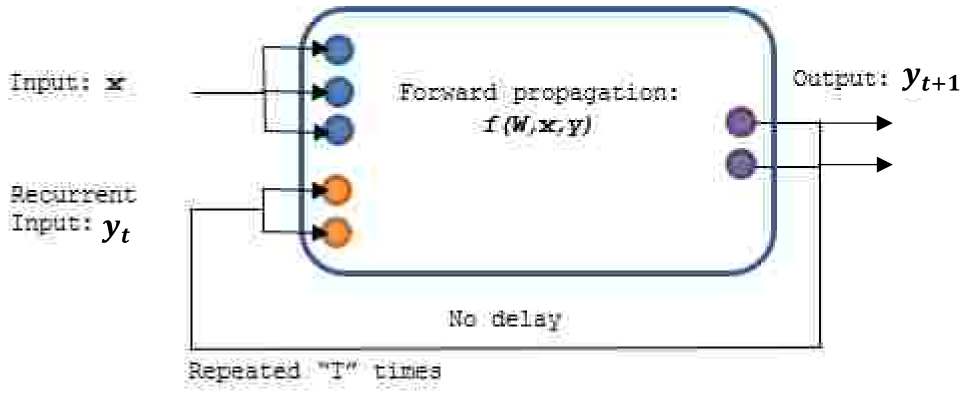


Fig. 5. Graphical representation of simultaneous recurrent network (SRN). f denotes a feed-forward mapping function. Note the absence of unit time delay in the architecture.

The SRN is defined by the following mapping function,

$$\tilde{y} = f(W, x); \tag{2}$$

where, x and W denote input vector and network weights respectively. $\tilde{y}$ in (2) is computed by iterating over the following equation [48],

$$y_{t+1} = f(W, x, y_t); \tag{3}$$

where, f is some sort of feed-forward mapping function and $\tilde{y}$ is obtained by,

$$\tilde{y} = \lim_{T \to \infty} y_T; \tag{4}$$

In practical applications T is assigned a finite value rather than ∞ (usually $T \leq 20$). Note that SRN behaves like a feed-forward network when $T = 1$.

From the operational point of view, SRNs are capable of efficiently approximating functions that can be approximated by MLPs, but the opposite is not true [49]. Moreover, the recurrent behavior of SRN also makes them appropriate for dynamic systems with feedback. Additionally, from the biological viewpoint, the simultaneous recurrrency of SRN closely mimics the activity in the human brain [50].

2.3 TRAINING ANN

The artificial neural networks mentioned in the previous section require training to perform a specific task. Training ANN means adjusting the free parameters, i.e. weights (*W's*) connecting the neurons with an appropriate algorithm. In the beginning, it was hard to develop a suitable training algorithm because of the difficulty in computing the derivatives of the error function. However, for the first time, Werbos [51] and other researchers [52] have successfully trained an ANN, more specifically MLP with a suitable training algorithm known as back-propagation (BP). The BP algorithm computes the derivative of the error function and back-propagates the error derivative through the network. The weights of the network that need to be adjusted are initially set to some random values. Once the derivatives are obtained at each layer, the weights are updated by applying the commonly known delta rule.

The above mentioned BP algorithm is the most popular neural network training algorithm for MLPs. However, this generic BP algorithm is not appropriate for training more complex recurrent networks. The main challenge of training RNNs using BP is back-propagating the error through the recurrent layers. In order to address this problem, Werbos et al. [53] have proposed a modified BP algorithm known as back-propagation through time (BPTT) to effectively train RNNs. The BPTT first “unfolds” the recurrent neural network to a certain depth prior to training. Specifically, this “unfolding” process creates a pseudo feed-forward network consisting of replications of the original network with the recurrent link being fed forward into the successive copy. If the network stabilizes, the output may not change after running through a finite number of replications; in this case the replication process is stopped. The multi-layered feed-forward network resulting from the above process can be considered as equivalent to the recurrent network and the resulting network can be trained using the regular back-propagation algorithm. However, the weights in each replication must be equal, and therefore, cannot be updated individually. Weight updating in BPTT is done by updating the weights simultaneously by using the sum of all the derivatives.

2.4 DEEP NEURAL NETWORKS (DNNs)

The ANN architectures that we discussed so far are shallow in nature (only three layers with a small number of neurons) and can be utilized for solving simple approximation tasks. One naive way to address complex problems with shallow architectures is using millions of neurons in the hidden layer. However, training an ANN with millions of hidden neurons becomes significantly complex and unrealistic for practical applications. The introduction of deep learning alleviates this problem by providing the flexibility of increasing the number of hidden layers in the neural network architecture. This technique of increasing the number of hidden layers in turn increases the number of neurons and, hence, improves the capability of ANN for solving more complex and challenging problems. In the next few sections, we briefly discuss the most popular state-of-the-art deep neural network architectures such as convolutional neural networks, deep auto-encoders, and different types of deep generative models, which are utilized in this study for comparing our proposed deep recurrent learning methods.

2.4.1 CONVOLUTIONAL NEURAL NETWORKS

One of the first hierarchical models, known as convolutional neural networks (CNNs/ConvNets) [22, 54], learns hierarchical image patterns at multiple layers with a 2D convolutional operation. CNNs are designed to process multidimensional data structured in the form of multiple arrays or tensors. For example, a color image has three color channels represented by three 2D arrays. Typically, CNNs process input data using three basic ideas: local connectivity, shared weights, and pooling, arranged in a series of connected layers. A CNN architecture is shown in Fig. 6. The first few layers are convolutional and pooling layers. The convolutional operation processes parts of the input data in small localities to take advantage of local data dependency within a signal. The convolutional layers gradually yield more highly abstract representations of the data in deeper layers of the network. Another aspect of the convolution operation is that filtering is repeated over the data, which makes use of redundant patterns in the data.

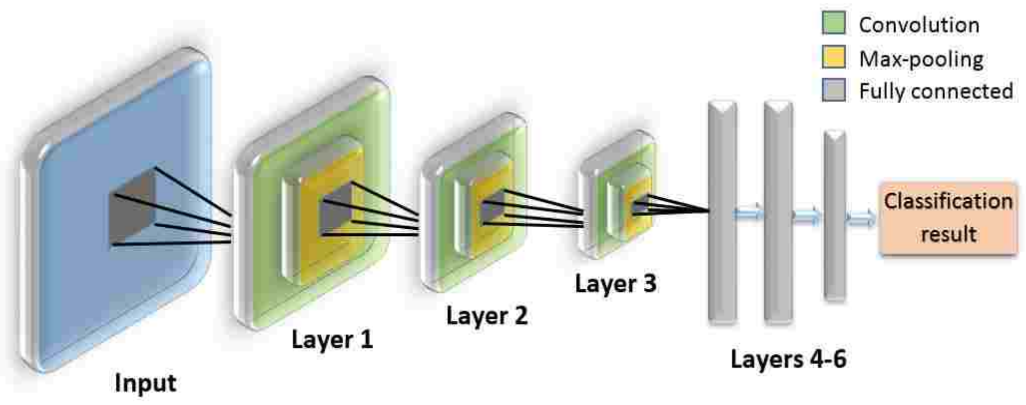


Fig. 6. Generic architecture of CNN.

While the convolutional layers detect local conjunctions of features from the previous layer, the role of the pooling layer is to merge local features into a more global representation and higher level of abstraction. This helps a network become robust to small shifts and distortions in data. The final layers of CNN architecture are typically fully-connected neural networks followed by a “softmax” regression layer that performs classification using highly abstracted features from the previous layers. The training of all the weights in the CNN architecture is performed by applying a regular backpropagation algorithm commonly known as gradient descent optimization algorithm [51, 52].

2.4.2 DEEP GENERATIVE MODELS AND AUTO-ENCODERS

The hierarchical model of CNN is designed to efficiently handle images and videos by learning meaningful features from raw data during training. However, the major breakthrough of hierarchical models is the introduction of the “greedy layer-wise” training algorithm for deep belief networks (DBNs) proposed by Hinton *et al.* [1]. A DBN is built in a layer-by-layer fashion by training each learning module known as the restricted Boltzmann machine (RBM) [55].

RBMs are composed of a visible and a hidden layer. The visible layer represents raw data in a less abstract form, and the hidden layer is trained to represent more abstract features by capturing correlations in the visible layer data [11]. Fig. 7 (a) shows a standard architecture of a DBN. DBNs are considered hybrid networks that do not support direct end-to-end learning. Consequently, a more efficient architecture known as deep Boltzmann machines (DBMs) [56] has been introduced. Similar to DBNs, DBMs are structured by stacking layers of RBMs. However, unlike DBNs, the inference procedure of DBMs is bidirectional, allowing them to learn in the presence of more ambiguous and challenging datasets.

Undirected models such as RBM and its extensions suffer from a major limitation in training due to the associated computationally intractable partition function . Consequently, *directed* generative models such as sigmoid belief networks (SBNs) [57-59] have drawn increasing attention. SBNs accompany a much simpler partition function [60], making the computation of full-likelihood trivial. As such, RBMs are increasingly being replaced by SBNs as the basic learning module in the DBNs [60]. Other generative models such as variational auto-encoder (VAE) [61-63] and generative adversarial network (GAN) [64-67] have shown to be quite adept at the representation learning task. GAN, unlike the probabilistic generative models, takes a deterministic approach by employing an adversarial scheme for representation learning. As such, GAN simultaneously trains a generator network and a discriminator network based on the accuracy of the generated outcome [64]. Specifically, the samples generated by the generator network are challenged by the discriminator network in terms of the difference between the generated sample and the actual data.

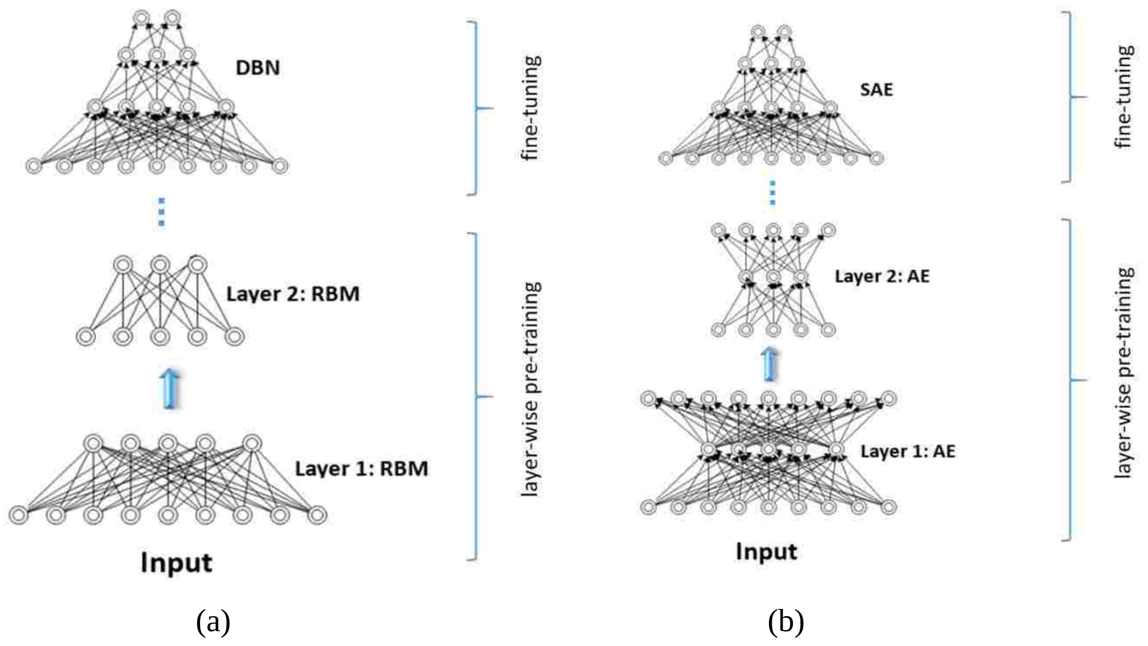


Fig. 7. A typical architecture including layer-wise pre-training and fine-tuning procedure of (a) deep belief network (DBN); (b) Stacked auto-encoder (SAE).

The introduction of generative models has led to the development of the stacked auto-encoder (SAE) [24, 68], which is also formed by stacking multiple layers. Unlike generative models, SAEs utilize auto-encoders (AE) [69] as the basic learning module. An AE is trained to learn a copy of the input at its output. In doing so, the hidden layer learns an abstract representation of inputs in a compressed form. Fig. 7 (b) shows the architecture of an SAE. A greedy layer-wise training algorithm is used to train SAE networks, where the parameters of each layer are trained individually by keeping parameters in other layers fixed. After greedy layer-wise training of all layers, called pre-training, the layers are stacked together and the entire network is simultaneously fine-tuned by adding a “softmax” regression layer at the end, to adjust all the parameters as illustrated in Fig. 7 (b).

2.4.3 TRAINING DEEP NEURAL NETWORKS

2.4.3.1 TRAINING DETERMINISTIC DEEP NEURAL NETWORKS

Deterministic DNN architectures such as CNNs, and SAEs utilize huge number of training parameters; hence, training such DNN models is generally difficult. The most popular training method for DNNs is the gradient descent (GD) learning method. However, conventional GD techniques are not adequate to achieve good convergence for the DNNs. Therefore, stochastic gradient descent (SGD) [70] and mini-batch gradient descent [71] techniques are introduced to handle DNN training more efficiently. Nevertheless, GD based techniques require adjustments of hyper-parameters such as learning rate, step size, weight decay, momentum, etc. for successful training of DNNs. Consequently, more sophisticated GD techniques such as AdaGrad [72], AdaDelta [73], and Adam [74] are introduced to handle the hyper-parameter adjustment efficiently using adaptive techniques. Though recently introduced GD based learning techniques generally work well for DNN training, nevertheless, DNNs with thousands of adjustable parameters easily succumb to the overfitting problem. Consequently, several regularization techniques such as L1, L2 weight regularization, and dropout learning have been developed for tackling this overfitting problem. Several studies [75, 76] mathematically and experimentally show that dropout learning is the most effective overfitting prevention technique for DNNs. Dropout learning was first introduced by Hinton et al. [77] for DNNs. The idea of dropout learning is to randomly drop some of the hidden units with a predefined probability during the forward pass of training. The connection weights associated with the dropped units are not updated during the backward pass of training. The process is repeated for each input sample and each training epoch. By doing this dropout learning discourages any unit relying on the output of any other units, and it forces the units to rely on the population behavior of the inputs [76]. This, in turn, prevents the overfitting problem associated with DNNs and also improves the generalization capability of the network. Furthermore, dropout inherently regularizes the network weights that favor small hidden activations [75]. During the testing phase of dropout learning, all units are present and the network weights are scaled appropriately based on the dropout probability.

2.4.3.2 TRAINING DEEP GENERATIVE NEURAL NETWORKS

Unlike deterministic DNNs, probabilistic generative models such as DBM, DSBN, and VAE models require special learning techniques. The training of undirected generative models such as the DBM model is performed by applying a variational approach [78] where mean-field inference is used to estimate data-dependent expectations. In order to better initialize the model parameters of a DBM, the stack of RBMs are pretrained by applying a modified greedy layer-wise pretraining technique [78]. A desirable property of the RBM is that the calculation of the gradient estimates on the model parameters is straightforward and the stochastic gradient descent (SGD) provides relatively efficient inference. However, evaluating the probability of a data point under an RBM is non-trivial due to the computationally intractable partition function [60]. The estimation of this partition function is usually performed by a sampling algorithm known as annealed importance sampling (AIS) [79]. Directed generative models such as DSBNs mitigate this problem by modifying the energy function to obtain a simple partition function [60]. Therefore, the full-likelihood under a DSBN is trivial to compute. However, training such directed generative models may be difficult [1]. Simple sampling based gradient estimation methods are proposed in [57, 80] to train the SBN model. Nonetheless, these methods are not scalable and practical for learning large models. This problem is tackled by utilizing recently developed variational inference methods in the Bayesian statistics literature. One such method is known as neural variational inference and learning (NVIL) [59] algorithm which is shown to be very efficient in training any deep generative models.

2.4.3.2.1 NEURAL VARIATIONAL INFERNCE AND LEARNING TECHNIQUE FOR DEEP GENERATIVE MODELS

The key idea behind the NVIL algorithm is the use of an inference model to implement an efficient exact sampling from the variational posterior for the given observation [12, 59]. The parameters of the inference model are jointly trained with the true model by maximizing a variational lower bound on the log-likelihood. The variational objective function is obtained by following the standard variational inference approach [81].

Suppose $P_\theta(v, h)$ with input space v and latent space h denotes a generative model with parameters θ where the exact inference of the model is intractable. The model is trained by maximizing a variational lower bound on the marginal log-likelihood. Since the exact posterior $P_\theta(h|v)$ of the model is difficult to obtain, a new parametric distribution $Q_\varphi(h|v)$ with parameters φ is introduced which serves as an approximation to the exact posterior. The variational posterior Q is chosen to have a simpler form than the exact posterior; hence, easier to work with. For a given observation v the variational lower bound objective function is written in terms of the Kullback-Leibler (KL) divergence as follows [59],

$$£(v, \theta, \varphi) = \log P_\theta(v, h) - KL\left(Q_\varphi(h|v) \,||\, P_\theta(h|v)\right). \tag{5}$$

The KL divergence in (5) determines the tightness between the variational objective function and the exact posterior. The tightness is obtained by maximizing (5) with respect to the parameters φ of the variational posterior Q which makes this distribution a better approximation to the exact posterior $P_\theta(h|v)$. Now, given a training set D, the model P is trained by maximizing $£(D, \theta, \varphi)$ using an appropriate gradient ascent technique with respect to the model and inference parameters.

The effectiveness of the NVIL algorithm is obtained by defining the variational posterior distribution $Q_\varphi(h|v)$ with an efficient inference model. This inference model is represented by a flexible neural network architecture to compute the variational distribution from the given observation. The naïve gradient estimation of the inference network parameters in NVIL exhibit high variance. Hence, several straightforward and general variance reduction techniques are applied to make the NVIL algorithm practical [59].

2.5 CLASSIFICATION USING DISTANCE METRIC LEARNING

In addition to the logistic regression based "softmax" classification technique which is widely used as an integral part of DNN models, distance metric learning (DML) is another popular machine learning technique to perform the classification task. DMLs aim to learn a metric to determine the distance or similarity between elements in a vector space. For a set of points $U = \{x_i\}_{i=1}^{l}$, the distance metric M can be learnt using the following pairwise real valued metric function,

$$d_M(x_i, x_j) = \|x_i - x_j\|_M = \sqrt{(x_i - x_j)^T M (x_i - x_j)} \quad \text{where } i, j = 1, 2, 3, \dots, l \tag{6}$$

Most metric learning methods learn the metric $M_{l \times l}$ using information obtained from training examples. The information is usually available in the form of pairwise constraints: pairs of data points known to be *similar* (S) and pairs of data points that are *dissimilar* (D). Using these, a convex problem is formulated to minimize the distance between the similar pairs and maximize the distance between the dissimilar pairs. The convex problem is then optimized using a semi supervised learning technique to approximate the underlying distance metric, M. The learned metric can be utilized in any distance based classification/clustering algorithm such as k-nearest-neighbor (k-NN), k-means clustering, and SVMs.

In [82] the authors show an iterative algorithm to solve the metric learning problem. The iterative method is less computationally expensive and therefore results in slow convergence. To alleviate this problem several other methods such as large margin nearest neighbor [83], and information theoretic based metric learning [84] have been proposed. While these algorithms show efficient techniques for solving the metric learning problem, they involve a full eigen-decomposition step that is computationally expensive for large-scale problems. Consequently, Ying et al. [85] propose an alternative method for solving the distance metric learning problem using eigenvalue optimization (DML-eig). Unlike other learning algorithms, DML-eig only computes the largest eigenvector at each iteration of the learning process making the technique suitable for large scale classification problems [85]. Due to the fast convergence property and state-of-the-art performance, this dissertation utilizes DML-eig as a classification step in of the proposed models.

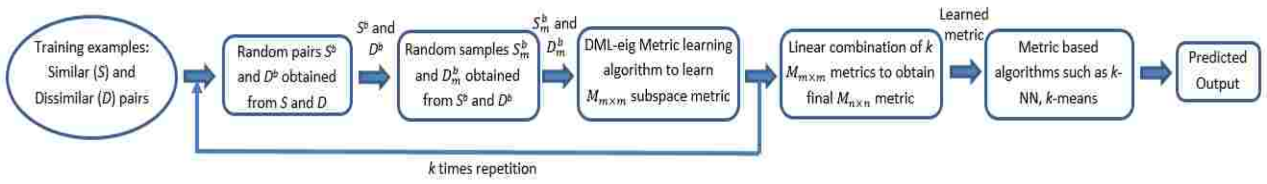

Training examples: Similar (S) and Dissimilar (D) pairs
Random pairs S^b and D^b obtained from S and D
S^b and D^b
Random samples S^b_m and D^b_m obtained from S^b and D^b
S^b_m and D^b_m
DML-eig Metric learning algorithm to learn $M_{m\times m}$ subspace metric
Linear combination of k $M_{m\times m}$ metrics to obtain final $M_{n\times n}$ metric
Learned metric
Metric based algorithms such as k-NN, k-means
Predicted Output
k times repetition

Fig. 8. Flow diagram of the randomized DML-eig metric learning algorithm ($\mathcal{6}_r$).

Langmead [86] introduces a randomized approach for metric learning to alleviate two common problems including computational complexity and overfitting. Given similar (S) and dissimilar (D) pairs, the randomized algorithm first randomly selects a set of pairs to obtain S^b and D^b, where b is the number of randomly chosen pairs from z number of pairs ($b < z$). Next, the algorithm randomly samples the input features from S^b and D^b to generate S_m^b and D_m^b. This reduces the dimensionality of the data from n space to m subspace ($m < n$). The subspace training sets S_m^b and D_m^b are then used as input to the metric learning algorithm. The procedure is then repeated k times with different random subspaces. During each iteration of k, the metric learning algorithm returns a trained $m \times m$ dimensional metric, $M_{m\times m}$ over the random subspace. The final $n \times n$ metric, $M_{n\times n}$, is then obtained by linearly combining all the k trained $M_{m\times m}$ metrics. The authors in [86] show that the k times repetition over different random subspaces can be performed in parallel. Therefore, the randomized algorithm may achieve a constant factor speedup when running on multi-core processors. The flow diagram of $\mathcal{G}_r$ is shown in Fig. 8.

2.6 HIERARCHICAL BAYESIAN MODEL

Bayesian models are powerful models due to its principled ability to combine prior information with data. Bayesian methods provide a complete representation of parameter uncertainty that can be directly interpreted [87]. Hierarchical Bayesian (HB) models are essentially hierarchical stacking of statistical sub-models that work in conjunction to estimate the posterior distribution of input data using Bayes principles [88]. HB models are defined to reflect the dependencies of the model parameters on each other. The chain of dependencies among parameters exemplifies HB model. Let us consider that the observed data denoted by X, are described by a model with parameter β and γ. The probability of the data is represented by the conditional probability distribution function $p(X|\beta,\gamma)$ which is called the likelihood function of the parameters. The prior probability of the parameters is defined using $p(\beta,\gamma)$. Generally, the probability of the data weighted by the probability of the parameters is defined using the product form, $p(X|\beta,\gamma)\ p(\beta,\gamma)$. However, the HB model factors the product term as a chain of dependencies among parameters as follows,

$$p(X|\beta,\gamma)\ p(\beta,\gamma) = p(X|\beta)p(\beta|\gamma)p(\gamma) \tag{7}$$

where, X denotes the observed data, and β and γ denote the model parameters. HB models are commonly used in natural language processing (NLP) applications such as topic modeling for document classification. More recently, HB models are utilized in image categorization applications to obtain better understanding of the input data which is essential for developing a more efficient categorization technique. HB models have shown to be particularly adept at learning categorization, or similarity metrics from a very few examples. Consequently, HB models have been adopted by many studies [34, 89, 90] to introduce an efficient learning mechanism where a few examples are typically sufficient for accurate categorization of a new object. In particular, the hierarchical Bayes model developed by Salakhutdinov et al. [90] accomplished “one-shot learning” in which the knowledge in categorization obtained previously is used to establish a novel category based on the similarity characteristics using just one example.

CHAPTER 3

DEEP SIMULTANEOUS RECURRENT LEARNING FOR EFFICIENT IMAGE RECOGNITION

3.1 CHAPTER OVERVIEW

This chapter proposes a novel deep simultaneous recurrent learning architecture for efficient image recognition. Recent advances in feed-forward deep neural networks (DNNs) have offered improved image recognition performance. Sparse feature learning in feed-forward DNN models offers further improvement in performance when compared to the earlier handcrafted techniques. However, the depth of the feed-forward DNNs and the computational complexity of the models increase proportionally to the challenges posed by the facial expression recognition problem. Moreover, the feed-forward DNN architectures do not exploit another important learning paradigm, known as recurrency, which is ubiquitous in the human visual system. Consequently, this chapter proposes a novel biologically relevant deep simultaneous recurrent network (DSRN) for robust image recognition. The feature sparsity is obtained by adopting dropout learning in the proposed DSRN as opposed to the usual handcrafting of additional penalty terms for sparse representation of data. This chapter provides a theoretical analysis to study the sparsity and over-fitting property of the proposed DSRN model. Finally, this chapter shows the superiority of the proposed DSRN model by solving several challenging image recognition tasks. Experimental results suggest that the proposed method yields better performance accuracy, requires fewer parameters and offers reduced computational complexity than that of the previously reported state-of-the-art feed forward DNNs.

3.2 LITERATURE REVIEW

This section provides a brief literature review of the recently introduced DNN models for image recognition. Image recognition is a fundamental yet open and challenging problem in computer vision. Computer vision models have seen great advancements in the recent years. Though these models may vary in terms of methodology and functionality, most feature-based methods generally involve an overall two-step process: 1) Feature extraction and selection, and 2) Feature classification. Numerous classification methods with varying capabilities have been introduced. However, in feature-based classification, the overall recognition performance heavily depends on the ability of extracting and selecting appropriate features.

3.2.1 FEATURE EXTRACTION TECHNIQUES IN COMPUTER VISION SYSTEMS

Early computer vision systems relied on hand crafted feature extraction and selection algorithms and simple classifier designs to perform object recognition tasks [91-95]. Though these techniques show limited success in certain simple object (e.g. rigid objects viewed at certain angles) recognition tasks, they did not extend well into recognizing more complex object categories such as human faces. The idea of learning the solution to a specific recognition task, instead of hand crafting feature extractors and classifiers, led to the recent progress in vision applications. Instead of a pre-determined set of features, an appropriate model is learnt or selected from a set of possibilities, using a set of known examples [96]. This type of “learning from examples” provided more efficient vision models with higher accuracy even with simple classification methods. More recently, the idea led to more complex deep hierarchical feature extraction and selection techniques commonly known as deep neural networks (DNNs). These multi-layered architectures such as deep belief networks (DBNs) [1, 97], stacked auto encoders (SAEs) [24, 98] and convolutional neural networks (CNN) [22, 23] are capable of extracting and refining features of incremental complexity. Hence, the automated and efficient feature extraction technique offered by DNNs are essentially used for solving complex image recognition tasks.

3.2.2 DEEP NEURAL NETWORKS FOR IMAGE RECOGNITION

Deep neural networks (DNNs) have shown excellent success in addressing a few of the challenges in image processing and computer vision. DNNs can successfully handle large scale complex object recognition tasks by utilizing thousands of neurons arranged in multiple layers. The hierarchical architecture of DNN mimics the information processing of the human vision system. Convolutional neural networks (CNNs) [22, 23] are one of the first hierarchical deep models. Later, researchers introduce other hierarchical deep architectures such as deep belief networks (DBNs) [1] and stacked auto-encoders (SAEs) [24]. All these deep networks and their variants are designed to mimic the hierarchical processing architecture of the human vision system using large scale feed-forward layers. However, recent biological evidence suggests that recurrent connections exist in the human vision system and in some regions, the recurrent connections even outnumber the feed-forward connections [99].

Different types of deep feed-forward based networks have been utilized to solve the object recognition tasks. CNNs show excellent performance for solving various recognition tasks such as document recognition [22], multi-class object recognition [25], and face recognition [100, 101]. The main benefit of CNN is the ability to learn features from raw images and videos during the training process. The introduction of greedy layerwise learning algorithm [1] revolutionized the training of DNNs and popularized other deep networks such as DBNs and SAEs. Both DBNs and SAEs utilize unsupervised feature learning techniques to efficiently handle recognition problems such as object recognition [25], face recognition [102, 103] and handwritten digit recognition [1, 104]. However, these feed-forward DNN architectures utilize thousands of neurons connected using millions of weights for solving the intricate image recognition task. Optimizing an increasing number of connection weights between these neurons quickly becomes cumbersome and computationally intensive. Moreover, these deep models lack the recurrent connections found in the human brain.

3.3 DEEP SIMULTANEOUS RECURRENT LEARNING FOR IMAGE RECOGNITION

This section illustrates the details of the proposed deep simultaneous recurrent network (DSRN) for image recognition. The architecture, mathematical explanation and experimental results of the proposed technique are discussed below.

3.3.1 DSRN ARCHITECTURE FOR UNSUPERVISED FEATURE LEARNING AND CLASSIFICATION

The internal architecture, a time unfolded version and an example deep version of the proposed DSRN are shown in Fig. 9 (a)-(c). In Fig. 9 (a), the hidden recurrent layer outputs of the deep SRN can be obtained as follows,

$$y_{t+1}^{h} = \sigma\big(dropout((W_e^h \,.\, y_T^{h-1} \,+ R^h \,.\, y_t^h), p^h) + b_e^h\big); \qquad (8)$$

$$for\ t = 1, 2, \dots, T, h = 1,2, \dots, F\ and\ y_T^0 = x;$$

where, superscript h indicates layer number, y_t^h indicates the n dimensional recurrent output from the hidden units at time step t and layer h with $y_{t=0}^h = 0$, T denotes the maximum number of recurrent steps, F denotes the number of layers, b_e^h denotes the bias vector, p^h indicates dropout probability, σ is the non-linear activation function (e.g., sigmoid, tanh or ReLU), y_T^{h-1} denotes an m dimensional feature vector with $y_T^0 = x$, W_e^hdenotes $n \times m$ feed-forward weight matrix (at layer h) which is projected to all the hidden recurrent layers, R^h denotes $n \times n$ recurrent weight matrix (at layer h) which is shared in the hidden recurrent layers.

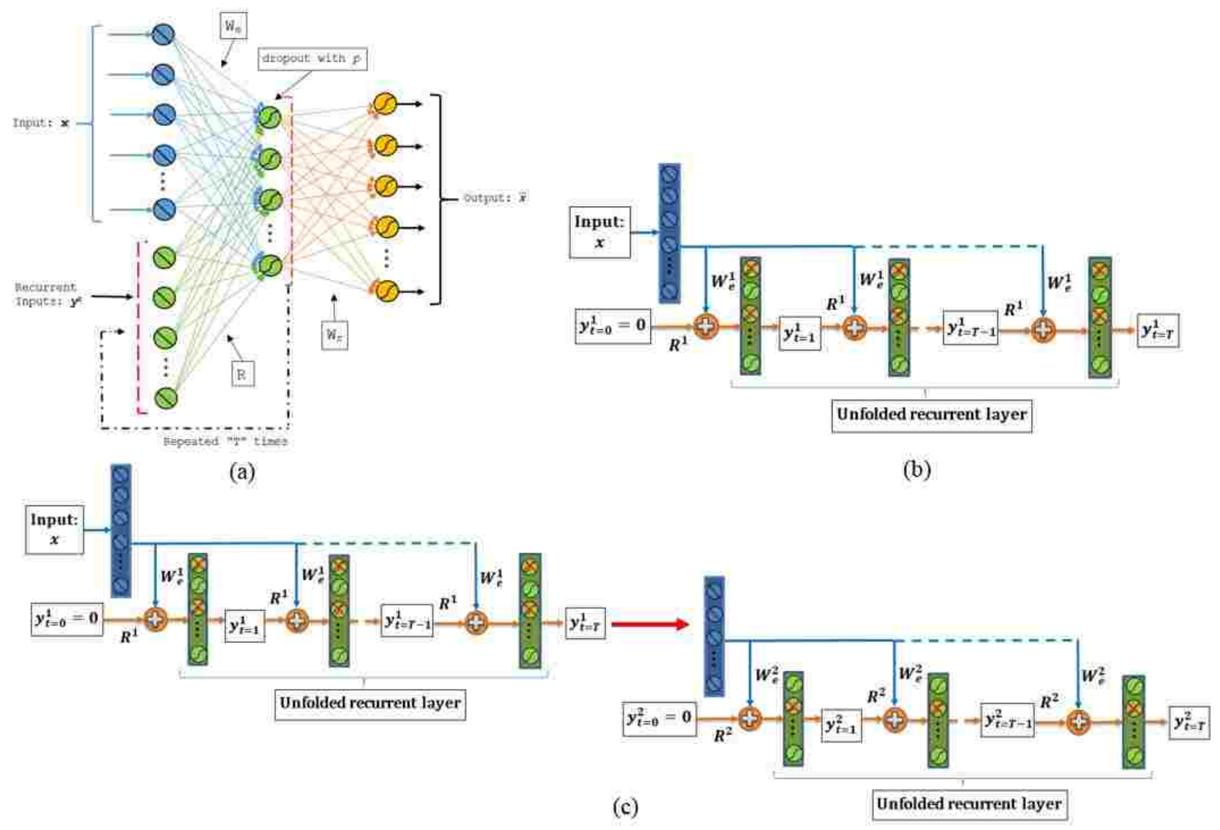


Fig. 9. (a) Internal architecture of an SRN in sparse deep simultaneous recurrent network DSRN). This figure also shows the SRN setup for unsupervised feature learning. Circle with straight line indicates linear units and circle with curved line denotes units activated with a non-linear function; (b) A single layer time unfolded version of DSRN ($h = 1$) that demonstrates weight sharing in the hidden recurrent layers. The red-cross in this figure indicates that the unit is dropped; (c) An example two layer time unfolded DSRN ($h = 2$). The network is expandable to $h = F$ layers similarly.

Though dropout does not always guarantee sparse activations, a recent study [76] shows that applying dropout regularization in a deep feed-forward network with sigmoid non-linear activation function in all the hidden layers ensures feature sparsity. Therefore, in this study we use σ as sigmoid non-linear activation function in all the hidden layers of our proposed DSRN architecture to achieve sparsity using dropout. The dropout probability p^h in **(**8), specified by the user, provides control over the sparseness of the hidden units at different layers h. Consequently, unlike existing sparse auto-encoder methods in the literature, this technique does not require any auxiliary regularization to enforce sparsity in the deep model.

The output of the DSRN is obtained by using $m \times n$ reconstruction weight matrix, W_r^h given as,

$$\ddot{x}^h = \sigma\left(W_r^h \, . \, y_T^h + b_r^h\right); \;\; for\ h = 1,2, \ldots, F; \tag{9}$$

where, $\ddot{x}^h$ indicates m dimensional output vector that is also considered as the reconstructed input, and b_r^h denotes the bias vector. The unsupervised loss function of DSRN is written as,

$$E^h = \frac{1}{2} \left\| y_T^{h-1} - \dddot{x}^h \right\|_2^2; \tag{10}$$

$$for\ h = 1,2, \ldots, F \quad and \quad with\ y_T^0 = x;$$

This unsupervised loss function is optimized using the back-propagation through time (BPTT) with stochastic gradient descent (SGD). The learning procedure of DSRN is shown in Algorithm 1. The proposed DSRN utilizes a novel SRN as the core learning module rather than generic feed-forward networks.

A “softmax” layer [1] is added to classify the feature vectors obtained from the last layer (layer F) of DSRN to c categories whose output is given by,

$$z_k = \frac{\exp(W_\varsigma y_T^F)}{\sum_{l=1}^{k} \exp(W_{\varsigma_l} y_T^F)}; \quad \text{where, } k = 1, 2, \ldots, c; \tag{11}$$

where, W_ς indicates $c \times n$ classification weight matrix, z_k denotes the predicted probability of kth class, and y_T^F indicates the features extracted from the final layer of the DSRN. In summary, the

S-DSRN is first pretrained for unsupervised feature learning via the layer-wise learning procedure as shown in Algorithm 1. Subsequently, the DSRN with classification layer is optimized using (11) to perform the classification task.

Algorithm 1: Learning procedure of DSRN using BPTT with SGD

```
Initialization:
  - Set network weights $\{W_e^h, R^h, W_r^h\}$ with random values
  - Set learning rate, $\propto$ with a small value

Training:
for each epoch do
    for each mini-batch do
        a. Perform DSRN forward propagation:
           for each simultaneous recurrent replication
               Compute hidden layer outputs using Eq. (8)
           end
           - Compute final layer output using Eq. (9)
        b. Compute overall loss function using Eq. (10)
        c. Perform DSRN back propagation (BPTT):
           - Compute partial derivative of the loss function ($E^h$) in terms of
             $W_r^h$ and $b_r^h$: $\nabla_{W_r^h} E^h$ and $\nabla_{b_r^h} E^h$
           - Set: $\Delta W_r^h = \Delta W_r^h + \nabla_{W_r^h} E^h$ and
             $\Delta b_r^h = \Delta b_r^h + \nabla_{b_r^h} E^h$
           for each simultaneous recurrent replications (backwards)
               - Compute partial derivative of recurrent weights: $\nabla_{R^h} E^h$
               - Compute partial derivative of projected feed forward weights and bias: $\nabla_{W_e^h} E^h$
                 and $\nabla_{b_e^h} E^h$
           end
           - Set:
               • $\Delta R^h = \Delta R^h + \nabla_{R^h} E^h$;
               • $\Delta W_e^h = \Delta W_e^h + \nabla_{W_e^h} E^h$ and
               • $\Delta b_e^h = \Delta b_e^h + \nabla_{b_e^h} E^h$
    end
     d.Update the weight parameters:
        - $W_e^h(new) = W_e^h(old) - \propto * \Delta W_e^h$;
        - $R^h(new) = R^h(old) - \propto * \Delta R^h$;
        - $W_r^h(new) = W_r^h(old) - \propto * \Delta W_r^h$;
        - $b_e^h(new) = b_e^h(old) - \propto * \Delta b_e^h$ and
        - $b_r^h(new) = b_r^h(old) - \propto * \Delta b_r^h$.
end
```

3.3.2 COMPLEXITY ANALYSIS OF THE PROPOSED DSRN ARCHITECTURE

This section analyzes the complexity of the proposed DSRN architecture compared to the state-of-the-art deep feedforward neural network (FDNN) architectures in terms of *Big O* notation. One advantage of DSRN is the use of extensive weight sharing in the hidden recurrent layers as shown in Fig. 9 (b). This enables efficient control over the depth of the architecture while utilizing fewer trainable parameters than feed-forward deep networks. An example two-layer ($h = 2$) DSRN is shown in Fig. 9 (c) which can be expanded to $h = F$ layers in a similar manner. The number of parameters (N_{DSRN}) of a h layer DSRN with T hidden recurrent layers is equal to the number of independent weights and biases over all the layers (including pretrained and classification weights) and is given by,

$$N_{DSRN} = \underbrace{h \times (n \times m)}_{\text{(feed-forward weights)}} + \underbrace{h \times (n \times n)}_{\text{(recurrent weights)}} + \underbrace{h \times n}_{\text{(biases)}} + \underbrace{c \times n}_{\text{(classifier)}}. \quad (12)$$

In contrast, the required number of parameters (N_{FDNN}) to achieve similar depth by using a feedforward DNN with classification layer is given by,

$$N_{FDNN} = \underbrace{h \times (T + 1) \times (n \times m)}_{\text{(feed-forward weights)}} + \underbrace{h \times n}_{\text{(biases)}} + \underbrace{c \times n}_{\text{(classifier)}}. \quad (13)$$

Equation (12) suggests that the upper bound of the required number of parameters for DSRN is $O(h \times (n \times m))$ when $m > n$. In comparison, the upper bound of N_{FDNN} from (13) is $O(h \times (T + 1) \times (n \times m))$. This clearly shows that generic FDNN requires $O(T \times (n \times m))$ more parameters than DSRN. This parameter reduction is due to the weight sharing property utilized in the hidden recurrent layers of DSRN. Therefore, a h layer DSRN with T hidden recurrent layers may be considered as equivalent (in terms of depth) to a $h \times (T + 1)$ layer FDNN.

3.3.3 IMAGE RECOGNITION PIPELINE WITH DSRN

The proposed pipeline for the image recognition algorithm using DSRN is shown in Fig. 10. The images are preprocessed and converted to vectors before feeding as input to the pipeline. At each layer h, feed-forward weights (W_e^h) and the shared recurrent weights (R^h) play major roles in meaningful feature extraction. The addition of sparseness further enhances the feature quality by reducing redundant and unstructured features. The features are then classified by the "softmax" classification layer of DSRN. From here onwards we refer to this technique as S-DSRN+softmax. We also train the proposed deep SRN architecture without using dropout learning for comparison and this technique is denoted as DSRN+softmax in the rest of this work.

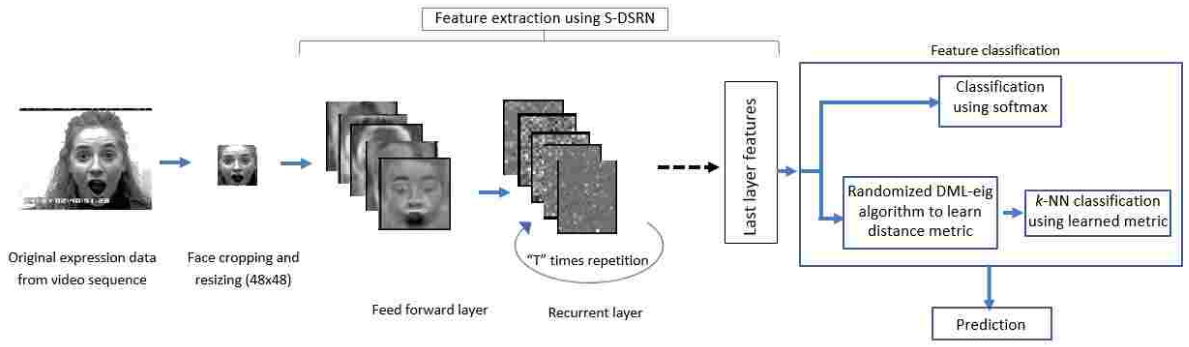

Feature extraction using S-DSRN
Feature classification
Classification using softmax
Last layer features
Randomized DML-eig algorithm to learn distance metric
k-NN classification using learned metric
Original expression data from video sequence
Face cropping and resizing (48x48)
"T" times repetition
Feed forward layer
Recurrent layer
Prediction

Fig. 10. Deep SRN (DSRN) based image expression recognition pipeline with (a) standard classification (S-DSRN+softmax); and (b) extended classification (S-DSRN with $б_r$) stages.

Fig. 10. shows that the classification stage is further enhanced by the introduction of metric learning. In addition, this study utilizes a randomized DML-eig metric learning technique [105] as discussed in Chapter 2, section 2.5 to further enhance the effectiveness of the feature classification task. Once features are extracted by the DSRN, the class labels of the dataset are used to obtain the similar (S) and dissimilar (D) pairs. These pairs of training samples are used by the randomized DML-eig algorithm to train the metric. A k-NN classification technique then utilizes the learned metric to perform the expression classification task. We use $𝒷_r$ to denote the randomized DML-eig as mentioned in Chapter 2 section 2.5; hence, the extended expression recognition pipeline is denoted by S-DSRN with $𝒷_r$ in the rest of the paper.

3.3.4 RESULTS AND DISCUSSION

The efficacy of the proposed deep recurrent network based image recognition framework is investigated by performing three complex recognition tasks: facial expression recognition, face recognition and character recognition. The following few sections illustrate our findings.

3.3.4.1 FACIAL EXPRESSION RECOGNITION USING DSRN

This section discusses the performance of the proposed DSRN framework for solving the facial expression recognition task.

3.3.4.1.1 DATASET PREPARATION

We perform the extremely difficult human facial expression recognition task by conducting extensive experiments on the well-known Extended Cohn-Kanade (CK+) expression dataset [106, 107]. CK+ dataset has 593 video sequences from 123 subjects and 309 out of 593 sequences are labeled as one of the seven basic expressions: anger (An), contempt (Co), disgust (Di), fear (Fe), happiness (Ha), sadness (Sa) and surprise (Su). Some example facial expression images are shown in Fig. 11. In order to obtain a fair comparison with other state-of-the-art methods, we drop the “contempt” expression and set up a 7-class classification task (including neutral expression). Each image sequence starts from the neutral pose and gradually transitions to the peak expression with each frame. The corresponding label is provided by the last frame in the sequence. The training and testing data are formed by randomly selecting the last five frames of each sequence. Note the last five frames adequately capture the peak expression while providing a sufficient number of samples for training and testing. Additionally, we select some of the first few frames from 309 labeled sequences for the neutral expression. The Viola-Jones face detection technique [108] is used as a pre-processor for each image (640 × 490) in CK+ dataset to extract the frontal face. The images are resized (48 × 48) using bicubic interpolation and then the intensity is normalized to ensure brightness invariance in the feature extraction process. Therefore, the experimental dataset formed with the sampling procedure described above contains a total of 2067 images, which is then divided into 10 subsets to obtain a 10 fold cross validation.

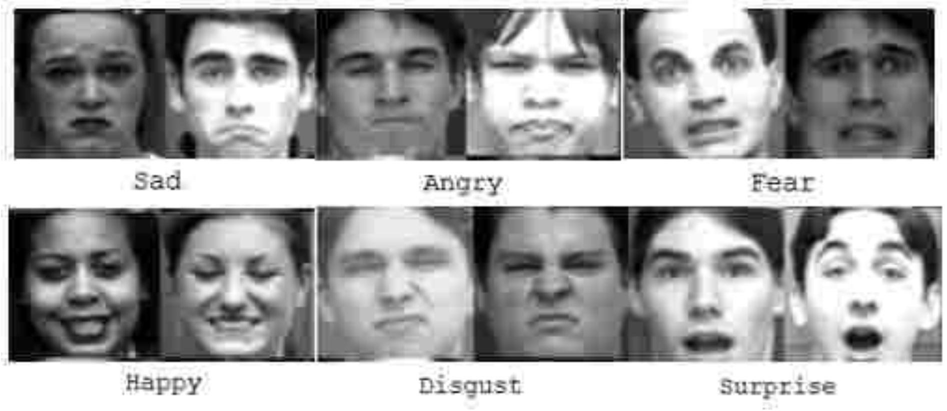


Fig. 11. Example facial expression images from CK+ dataset. Neutral expression is not shown in this figure.

3.3.4.1.2 DSRN ARCHITECTURE FOR FACIAL EXPRESSION RECOGNITION

The input to the proposed DSRN is the raw intensity values from the expression images of size 2304 (48 × 48). The architecture of the DSRN is given as: $2304 \times 1500^T \times 1000^T \times 500^T \times 7$, where superscript T denotes the number of repetitions in the hidden recurrent layer. In this study, we consider $T = 11$ and sigmoid as the non-linear activation function. The S-DSRN+softmax pipeline is trained using Algorithm 1. For weight initialization of S-DSRN, a simple normalized random initialization method [109] is used. The neuronal connection weights, W_e^h, R^h and W_r^h at each layer h is initialized using the following function,

$$\{W_e^h, R^h, W_r^h\} \sim U\left[-\frac{\sqrt{6}}{\sqrt{f^h+f^{h-1}}}, \frac{\sqrt{6}}{\sqrt{f^h+f^{h-1}}}\right] \tag{14}$$

where, $U[-a, a]$ is the uniform distribution in the interval $(-a, a)$ and f^h is the feature dimension at layer h.

3.3.4.1.3 PERFORMANCE OF FACIAL EXPRESSION RECOGNITION USING DSRN

We first conduct an experiment to compare our proposed expression recognition techniques: S-DSRN with $\mathscr{b}_r$, S-DSRN+softmax and DSRN+softmax. In the training process of S-DSRN, each hidden unit is dropped with a probability of 0.5 whereas each input unit is dropped with a probability of 0.25 when a batch of training samples are presented. In the testing phase, all weights are scaled down with appropriate dropout rates to compute the network outputs. The value of T (i.e. the number of hidden recurrent steps in our proposed network) is chosen experimentally by varying T from 5 to 20. Fig. 12 shows the effect of different values of T on the average 10-fold test classification accuracy. Fig. 12 illustrates that $T = 11$ offers an optimum result.

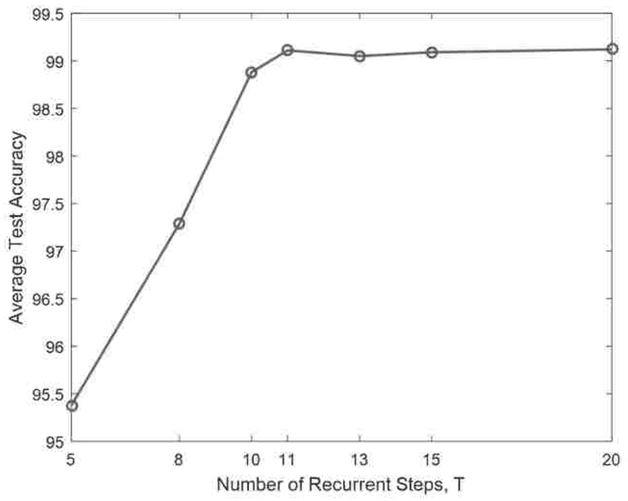


Fig. 12. The effect of different values of T (number of hidden recurrent steps of DSRN) on the average test classification accuracy. The classification accuracies are obtained from 10-fold cross validation experiment performed on CK+ dataset.

Table 1 summarizes performance of all three DSRN techniques for facial expression recognition. The table shows average test classification accuracies along with the standard deviation over 10-fold cross validation with the CK+ dataset.

TABLE 1

PERFORMANCE COMPARISON FOR THREE DSRN BASED EXPRESSION RECOGNITION TECHNIQUES USING CK+ DATASET

Methods	Proposed recognition techniques		
	S-DSRN with $\mathscr{b}_r$	S-DSRN+softmax	DSRN+softmax
Recognition performance	**99.11 ± 0.87%**	**97.68 ± 1.39%**	**92.69 ± 2.83%**

Fig. 13 shows comparison among the three recognition techniques using true positive rate (TPR), false positive rate (FPR) and F1 score. F1 score indicates the harmonic average of the precision and recall rate. The values are obtained by averaging 10-fold cross validation results for the CK+ dataset. Fig. 13 shows that S-DSRN with $\mathscr{6}_r$ offers the best performance with high TPR, high F1 scores and very low FPR for all the expression classes. On the other hand, S-DSRN+softmax shows a slightly lower TPR and F1 score, and higher FPR (except for “Di”) than that of S-DSRN with $\mathscr{6}_r$. Moreover, DSRN+softmax achieves the lowest performance across all the criteria with low TPR and F1, and high FPR.

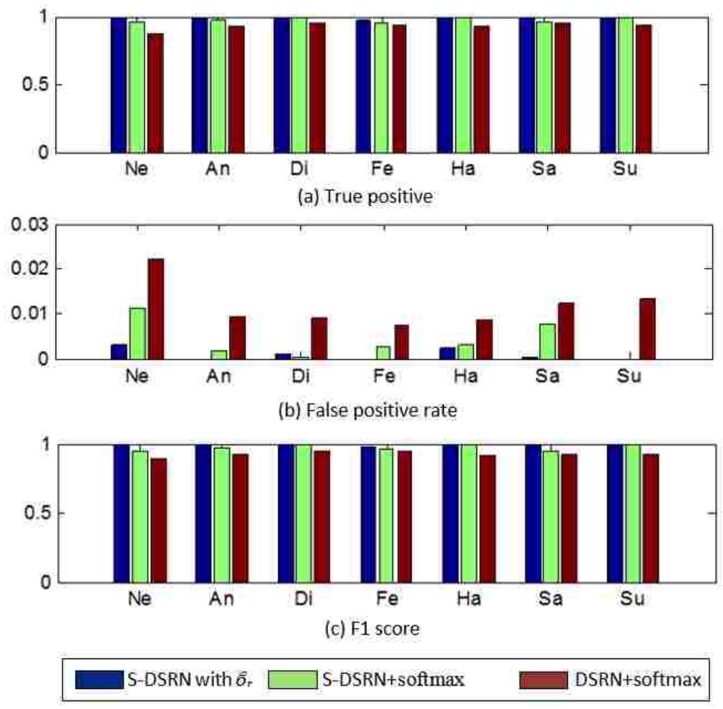


Fig. 13. From top to bottom, performance comparison on the CK+ dataset in terms of a) True positive rate, b) False positive rate and c) F1 score for 7 basic expressions.

A GPU based acceleration of the proposed S-DSRN is obtained for real-time implementation. Table 2 shows a comparison between the training time required by the CPU and GPU based implementation of the proposed S-DSRN based recognition pipeline. Note that the CPU based experiment is conducted on a single core 2.30GHz PC.

TABLE 2

TRAINING TIME COMPARISON OF CPU VS GPU IMPLEMENTATION OF S-DSRN

Implementation	S-DSRN (CPU: single core 2.3 GHz)	S-DSRN (GPU: TESLA M2090)
Training Time (10 fold cross validation experiment)	**90.15 hours** (3.75 days)	**18.23 hours**

Table 2 shows that distributed processing capability of GPU offers several order of magnitude improvements in the training time when compared to a single core CPU based implementation.

Finally, the performance of the proposed S-DSRN with δ_r based expression recognition technique is compared with a few state-of-the-art deep neural network and metric learning based expression recognition techniques using the CK+ dataset. The methods for comparison include several facial expression recognition methods including a boosted deep belief network (BDBN) [110]; a 3D Convolutional Neural Network (3D CNN) with deformable action parts (3DCNN-DAP) [111] and a K-means based unsupervised feature extraction technique combined with two layer RBMs known as an action unit aware deep network (AUDN) [112]. For a fair comparison, only the methods that use the CK+ expression dataset with a similar experimental setting are selected. Table 3 summarizes the performance comparison.

TABLE 3

PERFORMANCE COMPARISON OF OUR PROPOSED S-DSRN WITH THE STATE OF THE ART ON THE CK+ DATASET

	Our proposed	**State of the art**		
Methods	S-DSRN with $\mathcal{B}_r$	BDBN [110]	3DCNN-DAP [111]	AUDN [112]
Validation settings	10-fold	8-fold	15-fold	10-fold
Recognition performance	**99.11%**	**96.70%**	**92.40%**	**92.05%**
Number of Layers	4	6	7	8
Total number of trainable parameters (approx. in millions (M))	9M	164M	70M	30M

Table 3 shows that the proposed S-DSRN based expression recognition framework outperforms all other methods. The number of trainable parameters required by each method is also shown in Table 3. This number of trainable weights can be considered as a direct estimation of computational resource requirements for each model. The number of parameters is obtained by calculating the total number of weights and biases over all layers from the description of the deep neural network models [110-112]. Note the table shows that the proposed DSRN based expression recognition architecture requires fewer trainable parameters than that of the other networks which implies usage of a much reduced usage of computing resources for DSRN.

3.3.4.1.4 SPARSITY AND OVERFITTING ANALYSIS OF DSRN

This section provides mathematical and experimental analysis of sparsity followed by overfitting analysis for DSRN. We first obtain a detailed mathematical analysis of DSRN with dropout learning. For the ease of analysis we rewrite the forward propagation term of DSRN in a simplified and generalized form [76]. We simplify **(**8**)** by removing the bias terms (b_e^h). This is justified because dropout in **(**8**)** is only applied to the hidden unit outputs obtained using the feed-forward (W^h) and recurrent (R^h) weights while the bias term is unaffected. Next, we write **(**8**)** to obtain a generalized form for "h" number of layers which yields,

$$y_{t+1}^h = \sigma\left(S_{t+1}^h\right) = \sigma\left(\sum_{l<h}\left[W_e^h . y_T^l + R^h . y_t^h\right]\delta^l\right), \tag{15}$$

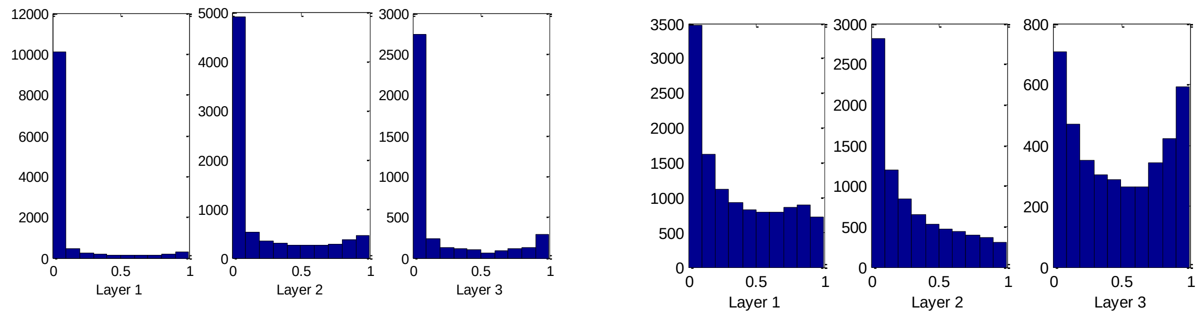
where, W_e^h is the feed-forward weights, R^h denotes the hidden recurrent weights at layer h, σ indicates the sigmoid non-linear activation function, $S_{t+1}^h = \sum_{l<h}\left[W_e^h . y_T^l + R^h . y_t^h\right]$ denotes the hidden layer output of the DSRN before applying the non-linear activation function σ, y_{t+1}^h indicates the same after applying the non-linear activation function σ to S_{t+1}^h, $y_T^{l=1} = x$, $t = 1, 2, \dots, T$ and $h = 1, 2, \dots, F$, and δ^l is a Bernoulli selector random variable with probability $P(\delta^l = 1) = p^l$. Since the recurrent weights of the DSRN are shared in the hidden layers, we apply dropout at $t = 1$ and maintain the probability as a constant for that layer until $t = T$.

As a consequence of SGD based training, the variance of the DSRN units is considered approximately constant and relatively small when the learning converges. Therefore, by analyzing the variance of the units it is possible to understand the effect of dropout on the weights and activities of DSRN. The variance of the units of DSRN is computed as follows,

$$Var\left(S_{t+1}^{h}\right) = \sum_{l<h}\left[W_e^{h}.y_T^{l} + R^{h}.y_t^{h}\right]^2 p^{l}(1-p^{l}). \tag{16}$$

For simplicity, we assume that dropout is applied only in layer h and the random variables δ^{l} are independent of each other. From (16) it can be observed that $Var\left(S_{t+1}^{h}\right)$ is reduced when the term $\left[W_e^{h}.y_T^{l} + R^{h}.y_t^{h}\right]^2$ is minimized. In other words, dropout forces SGD to converge with small weights for both W^{h} and $R^{h}$and minimal activations (S_{t+1}^{h}) with lower variance in the units. The term ($p^{l}(1-p^{l})$) introduced by the random dropout variable δ^{l} provides further sparseness in the weights and the activations. Consequently, this analysis shows that dropout favors small weights and unit activations which leads to sparsity for the proposed DSRN architecture.

Next, experimental analysis investigates the effects of dropout learning on sparsity for the proposed DSRN architecture. The sparsity on DSRN is observed by studying the histograms of the hidden layer unit activations (y_T^{h}) at the final recurrent step (at t = T). Fig. 14 (a) shows the histograms of three hidden recurrent layer activations obtained from the trained S-DSRN, averaged over a random test batch of 50 images from the CK+ dataset. For comparison, Fig. 14 (b) obtains histograms of the same hidden layer activations for a regular DSRN model trained without dropout.



(a) DSRN (dropout with $p = 0.5$) (b) DSRN (no dropout)

Fig.14. Sparsity of deep SRN using dropout: Histogram of the activated units for three recurrent hidden layers (a) DSRN with dropout; and (b) DSRN without dropout. The activations are obtained from the last recurrent iteration i.e. at $t = T$ for all three layers.

Fig. 14 shows that the hidden unit activations of DSRN trained with dropout is far sparser than that of the DSRN without dropout. This demonstrates that the use of dropout regularization in our proposed DSRN network with sigmoid non-linearity induces sparsity in the hidden recurrent unit activations with no additional requirement of user defined sparsity regularization terms.

Finally, we study the capability of dropout learning to prevent the overfitting problem otherwise associated with deep models such as DSRN. Prevention of overfitting is necessary for smaller datasets such as the ones used in this study. An expression recognition experiment is conducted with proposed DSRN with and without dropout for comparison. Fig. 15 shows the plot of DSRN without dropout, suggesting very high variation in the test accuracies for the 10-fold experiment (87% - 96%). Note the training accuracy for each fold in this case is always close to 100%, suggesting possible overfitting scenario. On the other hand, DSRN with dropout shows very consistent test accuracies for all folds (95% – 99%). The training accuracies in this case are consistently close to the test accuracies. The plot also shows a distinct improvement on the average test accuracy when dropout learning is applied for training deep SRN architecture.

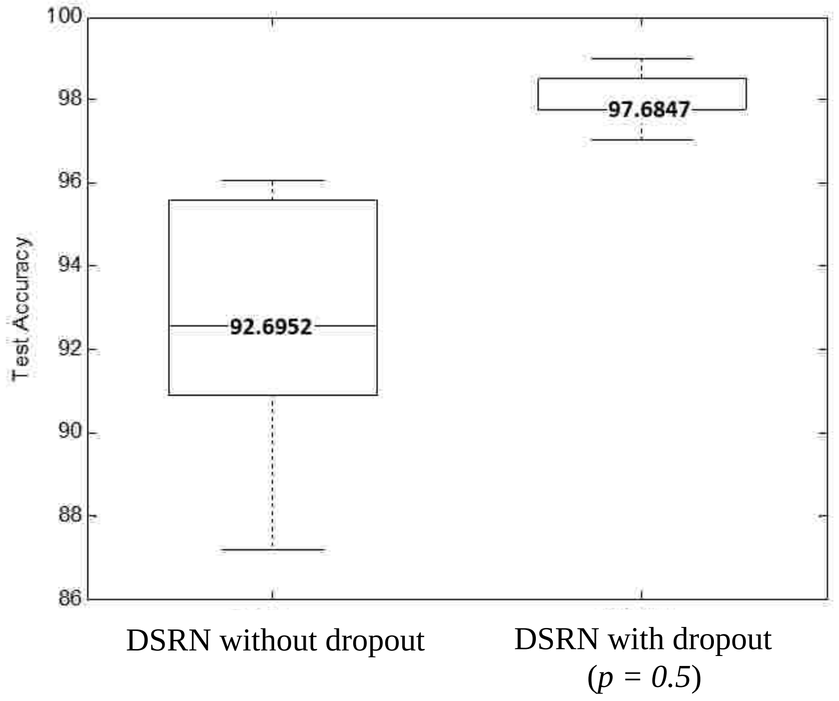


Fig. 15. Significance of dropout learning for training S-DSRN. Test accuracy (%) of facial expression recognition experiment (10-fold) obtained using S-DSRN and DSRN model. Values inside the box represent average test classification accuracy.

Furthermore, the detailed mathematical analysis of the model averaging property and convergence of S-DSRN are provided in Appendix A.

3.3.4.2 FACE RECOGNITION USING DSRN

This section illustrates the performance of the proposed DSRN technique for solving the face recognition task.

3.3.4.2.1 DATASET PREPARATION

The face dataset utilized in this experiment are obtained from low-resolution videos carefully captured by our research group in the Vision Lab. The videos are taken from 10 people at different settings with various lighting conditions, different facial expressions, pose variations and changes in details (glasses, no glasses, beard, no beard, etc). The face dataset is then formed by detecting, cropping and resizing (48x48) the faces from the video frames. By doing this we obtain 4609 facial images in total (the number of images of each person varies from 350 to 550). The dataset is randomized and divided into 10 subsets to obtain a 10 fold cross validation configuration.

3.3.4.2.2 DSRN FOR FACE RECOGNITION: ARCHITECTURE AND PERFORMANCE ANALYSIS

The architecture of the DSRN for the face recognition experiment is similar to the previously mentioned facial expression recognition model except in this case we use a single layer DSRN architecture given as: $2304 \times 500^T \times 10$. We perform a comparison with a state-of-the-art feed forward network based five layer stacked auto encoder (SAE). In this case also we have utilized 500 neurons in all the hidden layers. The architecture of the SAE used in this paper can be written as: $2304 \times 500 \times 500 \times 500 \times 500 \times 500 \times 10$ with a “softmax” classification layer. The SAE is pre-trained and fine-tuned using backpropogation with SGD. Moreover, in this experiment, we use an $L2$ weight regularization technique to prevent both networks from overfitting. The face recognition results are shown in Table 4. The results demonstrate that

DSRN achieves better face classification accuracy than SAE for the 10-fold cross validation experiment.

TABLE 4

COMPARISON BETWEEN PROPOSED DSRN AND FIVE LAYER SAE ON FACE RECOGNITION EXPERIMENT

Network	**Classification Accuracy (10-fold)**	**No. of layers**	**Total # of trainable parameters**
DSRN	**98.97%**	1	1407K
SAE	93.14%	5	2157K

Table 4 also shows that the five-layer SAE requires 750K more trainable parameters than the DSRN. The recurrent connections of DSRN increase the depth of the network while keeping the number of trainable parameters constant by weight sharing. Even with substantial reduction in trainable parameters, the above results show that our proposed network provides more representational power than conventional SAEs.

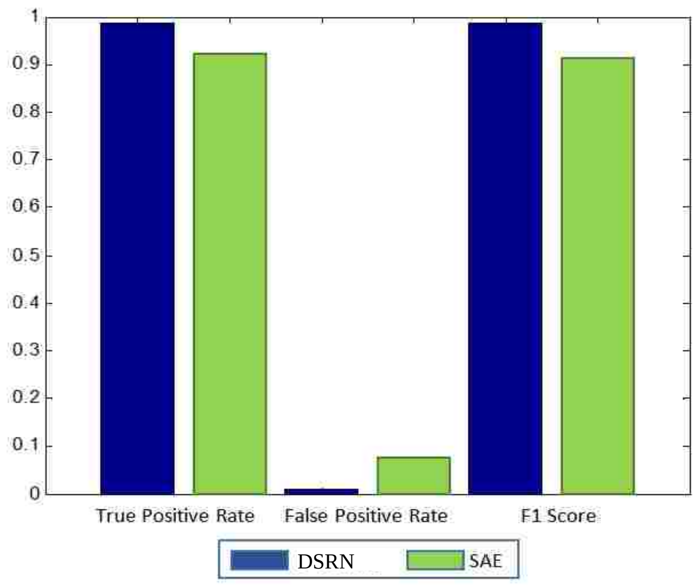


Fig. 16. Performance comparison between DSRN and SAE on the face dataset in terms of true positive rate, false positive rate and F1 score. The values are obtained by averaging over all classes and all folds.

Furthermore, Fig. 16 shows the comparison between the two deep models using TPR, FPR and F1 score measures. Note that DSRN achieves a higher TPR and F1 score than SAE. On the other hand, the FPR of DSRN is much lower than that of SAE. This demonstrates the superior face classification performance of DSRN with low false positives.

3.3.4.3 CHARACTER RECOGNITION USING DSRN

This section discusses the performance of the proposed DSRN architecture for solving the character recognition task.

3.3.4.3.1 DATASET PREPARATION

The character images are obtained from the challenging Chars74K dataset [113]. The dataset is formed by obtaining the characters from natural images (mostly from Google street view images). It also contains hand drawn and synthetic computer-generated character images. In total the dataset has 74K images consisting of 64 classes (0-9, A-Z, a-z). This work considers a subset of the Char74K dataset that contains 36 classes (0-9, A-Z) with 42,371 images. The subset dataset is then divided into five non-overlapping sets to form a five-fold cross validation setup. Similar to the expression dataset the character images are converted to gray scale and resized to 48×48 . Some example character images are shown in Fig. 17.

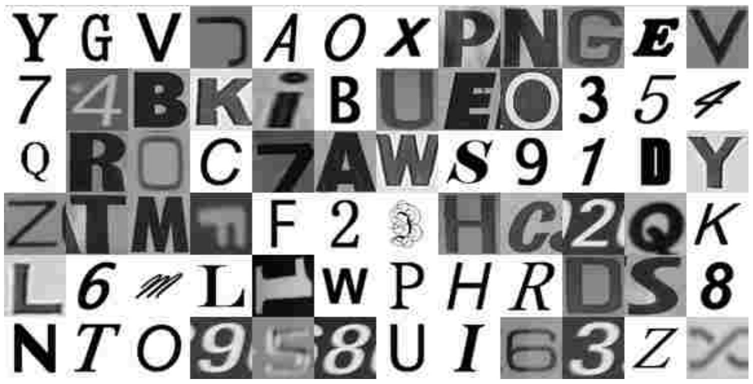


Fig. 17. Example character images from Char74K dataset.

3.3.4.3.2 DSRN FOR CHARACTER RECOGNITION: ARCHITECTURE AND PERFORMANCE ANALYSIS

The character recognition experiment is performed using the same deep architectures for both DSRN and SAE explained in the previous section. However, in this case the classification task is performed by the randomized DML-eig metric learning technique, β_r rather than "softmax". Hence, we refer to the pipelines as DSRN with $\mathscr{6}_r$ and SAE with $\mathscr{6}_r$, respectively. The average 5-fold cross-validation accuracy obtained on the character dataset is shown in Table 5.

TABLE 5

COMPARISON BETWEEN PROPOSED DSRN with $\mathscr{6}_r$ AND FIVE LAYER SAE with $\mathscr{6}_r$ ON CHARACTER RECOGNITION EXPERIMENT

Network	**Classification Accuracy (5-fold)**	**No. of layers**	**Total # of trainable parameters**
DSRN with $\mathscr{6}_r$	**92.62%**	1	1402K
SAE with $\mathscr{6}_r$	88.79%	5	2152K

Table 5 shows that as DSRN with $\mathscr{6}_r$ achieves better classification accuracy than SAE with $\mathscr{6}_r$, for the character recognition task. We once again point out that DSRN requires far fewer trainable parameters compared to SAE (750K less than SAE) as shown in Table 5.

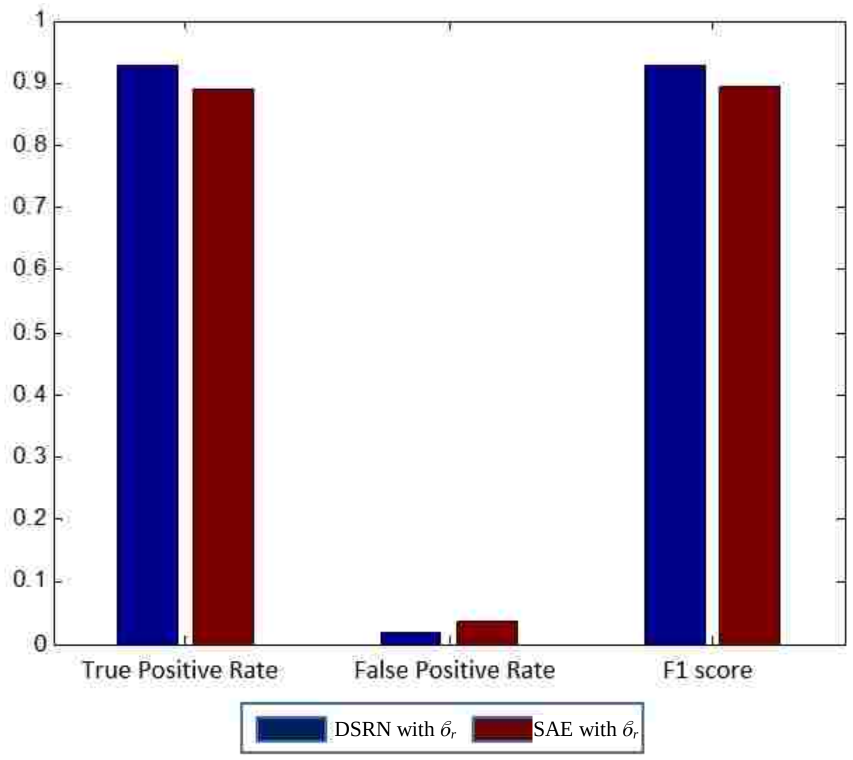


Fig. 18. Performance comparison between DSRN with $б_r$ and SAE with $б_r$ on Chars74K dataset in terms of true positive rate, false positive rate and F1 score. The values are obtained by averaging over all classes and all folds.

Furthermore, we show a performance comparison between the classification pipelines in terms of TPR, FPR and F1 score in Fig. 18 which clearly shows that DSRN with $б_r$ achieves higher TPR and F1 scores than SAE with $б_r$. On the other hand, DSRN with $б_r$ offers much lower FPR than that of SAE with $б_r$.

3.4 SUMMARY

This chapter proposes a novel biologically inspired deep recurrent model, DSRN, for effective image recognition. The DSRN model enables recurrent information processing in addition to the feed-forward information processing within each layer, which allows the model to learn more complex features from the input data. Moreover, the use of simultaneous recurrency in the DSRN model provides efficient control over the depth and the number of training parameters. Our findings suggest that the proposed DSRN model shows improved image recognition performance compared to the state-of-the-art models while utilizing a significantly fewer training parameters. These findings are published in [114-116].

The proposed DSRN architecture shows improved image recognition performance compared to the state-of-the-art models. However, the DSRN model is discriminative and designed to solve task specific classification problems which require huge labeled examples for training. Such a huge number of labeled examples is not available in many practical applications and deep learning models are required to learn from a large collection of unlabeled data to achieve an understanding of the underlying distribution of the data also known as the representation learning problem. Task specific discriminative models perform poorly in this scenario; hence, a special type of deep learning models may be required to perform the representation learning task. Consequently, probabilistic generative models are introduced to perform the representation learning task from unlabeled data. Moreover, generative models are used as the basic building blocks of many deep learning models. Accordingly, in the next chapter, we extend the concept of deep simultaneous recurrency in a novel deep recurrent generative model to perform the representation learning task effectively.

CHAPTER 4

DEEP GENERATIVE SIMULTANEOUS RECURRENT MODEL FOR EFFICIENT REPRESENTATION LEARNING

4.1 CHAPTER OVERVIEW

This chapter proposes a novel deep simultaneous recurrent probabilistic generative model to effectively perform the representation learning task from unlabeled data. Representation learning plays an important role for building effective deep neural network models. Deep generative probabilistic models have shown to be efficient in the data representation learning task which is usually carried out in an unsupervised fashion. Throughout the past decade, there has been almost exclusive focus on the learning algorithms to improve representation capability of the generative models. However, effective data representation requires improvement in both the learning algorithm and architecture of the generative models. Therefore, improvement to the neural architecture is critical for improved data representation capability of deep generative models. Furthermore, the prevailing class of deep generative models such as deep belief network (DBN), deep Boltzman machine (DBM), deep sigmoid belief network (DSBN), and variational autoencoder (VAE) are inherently unidirectional and lack recurrent connections ubiquitous in the biological neuronal structures. Introduction of recurrent connections may offer further improvement in data representation learning performance to the deep generative models. Consequently, this chapter proposes a deep recurrent generative model known as deep simultaneous recurrent belief network (D-SRBN) to efficiently learn representations from unlabeled data. The proposed D-SRBN model is a logical extension of the DSRN based discriminative model proposed in the previous chapter to perform more generalized representation learning task.

4.2 LITERATURE REVIEW

Representation learning is considered as one of the critical steps for building robust deep learning models. The task of representation learning involves learning different explanatory factors of variation embedded in the data without explicitly knowing the labels of the data in an unsupervised fashion. Once the representation is learnt, the model may be applied in other machine learning applications such as recognition, classification, segmentation, reasoning, decision-making and many more. Deep generative models are particularly adept at learning representations directly from the unlabeled data. Similar to the majority of neural models, the performance and efficacy of such models generally depend on the architecture and the associated learning algorithm. Throughout the past decade, there has been extensive research [57, 59, 80, 117-121] to introduce faster and efficient learning algorithms for probabilistic generative models. However, in addition to learning algorithms, improvements in the underlying network architectures are critical for effective representation learning. The rapid increase in the complexity of the large-scale datasets necessitates more sophisticated yet efficient architectures for the generative models to capture complex patterns from the data, and improve representation learning performance.

4.2.1 GENERATIVE MODELS: DEFINITION AND CLASSICAL MODELS

Generative models learn to represent a dataset as a joint probability distribution over its features. Therefore, the statistical samples drawn from the model represents similar types of observations found in the input dataset. Due to this ability of generating samples from the learned distribution, these models are referred to as generative models. Probabilistic generative models are formally expressed as $p(v, h)$, a probabilistic model over the joint space of the latent variable h and the observed data or visible variables v. Feature values are obtained as the result of an inference technique to determine the probability distribution of the latent variables given the data, $p(h|v)$. Learning is perceived in terms of estimating a set of model parameters that maximizes the regularized likelihood of the training data. Though several classic dimensionality reduction methods such as principal component analysis (PCA), linear discriminant analysis (LDA) and manifold learning have been proposed for representation learning applications [122, 123], neural networks remain as one of the widely used form of generative models. During the neural network learning process, the trainable parameters are adjusted such that the probability distribution represented by the neural network model that fits the input data as best as possible.

4.2.2 GENERATIVE MODELS: RECENT DEVELOPMENT

Restricted Boltzman machine (RBM) was the earliest neural network based parametric generative model and was later utilized as the basic building block of the more expressive deep generative models such as deep belief network (DBN) and deep Boltzman machine (DBM). Deep belief network (DBN) [1] and deep Boltzman machine (DBM) [78] are two widely used deep probabilistic generative models that contain many layers of non-linear hidden units. These models utilize restricted Boltzman machine (RBM) [117-119] as the basic building block. The RBM is an *undirected* graphical model which consists of an input layer (visible layer) and a hidden layer of stochastic binary units. The visible and hidden layer units are connected by trainable weights with no connections between units in the same layer. The information propagation between visible and hidden units occurs in two ways: recognition, where visible unit activations propagate to the hidden units and reconstruction, in which the information propagates from hidden to visible units [117].

The following sections present brief discussions on existing state-of-the-art deep generative models such as DBN, DBM, deep sigmoid belief network (DSBN), variational auto-encoder (VAE) and generative adversarial network (GAN).

4.2.2.1 DEEP BELIEF NETWORK (DBN)

The DBN model has been successfully applied in many different applications such as image recognition [124], natural language processing [125] and acoustic modeling [126], etc. The DBNs are constructed by stacking layers of RBM on top of each other. Arranging RBMs in this fashion allows the DBN to progressively capture more complex patterns of the input data at each non-linear layer [120]. DBNs are considered a hybrid deep model since the training of the DBN model is performed in two stages [1]: unsupervised pretraining, and supervised finetuning. The pretraining of the DBN requires greedy layerwise training [121] by optimizing an unsupervised loss function. Consequently, a generative model with “pretrained weights” is obtained that captures the features of the raw input. In the second stage, first the pretrained weights from the RBMs are copied to a regular deep feed-forward neural network model to replace its hidden layer weights. Then an additional layer such as a classification layer is incorporated on top of the newly formed deep network to perform finetuning of the weights by optimizing a supervised loss function.

4.2.2.2 DEEP BOLTZMAN MACHINE (DBM)

Inference in DBNs is problematic, so more efficient DBM generative models are introduced [78]. DBMs are successfully applied in various applications such as object and speech recognition [127], multimodal learning [128], etc. Similar to DBNs, DBMs are formed by stacking layers of RBMs in which each layer captures complicated, and higher-order correlations between the activities of hidden features in the layer below [78]. This enables the DBM to learn internal representations directly from the raw input data that become increasingly complex at each layer. However, unlike DBNs, the approximate inference procedure in DBMs is bidirectional: bottom-up and top-down, allowing the DBMs to better propagate the uncertainty of ambiguous inputs. This makes the DBM a more robust generative graphical model than the DBN. The training of the DBM model is performed by applying a variational approach [78] where mean-field inference is used to estimate data-dependent expectations. In order to better initialize the model parameters of a DBM, the stack of RBMs are pretrained by applying a modified greedy layerwise pretraining technique [78]. Though there are differences between DBN and DBM models, they both utilize the RBM as the basic learning module. A desirable property of the RBM is that the calculation of the gradient estimates on the model parameters is straightforward and the stochastic gradient descent (SGD) provides relatively efficient inference. However, evaluating the probability of a data point under an RBM is non-trivial due to the computationally intractable partition function [60]. The estimation of this partition function is usually performed by a sampling algorithm known as annealed importance sampling (AIS) [79].

4.2.2.3 DEEP SIGMOID BELIEF NETWORK (DSBN)

In recent years *directed* generative models such as sigmoid belief networks (SBNs) [57-59] have drawn increasing attention. The SBN models are closely related to their undirected counterparts, RBMs. As mentioned above, one major limitation associated with RBMs is the use of an intractable partition function in the energy function. SBNs mitigate this problem by modifying the energy function to obtain a simple partition function [60]. Therefore, the full-likelihood under an SBN is trivial to compute. As such, SBNs are now utilized as the basic learning module for the DBNs [60]. Moreover, deep directed generative models known as deep sigmoid belief network (DSBN) are also introduced using SBNs and successfully applied in applications such as image representation [60, 129] and polyphonic music and motion capture [130]. However, training such directed generative models may be difficult [1]. Simple sampling based gradient estimation methods are proposed in [57, 80] to train the SBN model. Nonetheless, these methods are not scalable and practical for learning large models. This problem is tackled by utilizing recently developed variational inference methods in the Bayesian statistics literature. One such method is known as neural variational inference and learning (NVIL) [59] algorithm which is shown to be very efficient in training DSBN models.

4.2.2.4 GENERATIVE ADVERSARIAL NETWORK (GAN) AND VARIATIONAL AUTOENCODER (VAE)

Recent studies introduce more robust generative models such as variational auto-encoder (VAE) [61-63] and generative adversarial network (GAN) [64-67] for the representation learning task. VAE is a probabilistic graphical model whose explicit goal is to perform non-linear latent variable modeling by marginalizing out certain variables as part of the modeling process [61]. During the learning process, the latent variables capture meaningful representation from the observed data, which is not immediately visible from the raw observations. Subsequently, the learned latent variable model is utilized to generate the input sample from some latent or unobserved space. VAE utilizes a gradient-based learning procedure inspired from variational inference principle. More specifically, VAE is trained by maximizing the evidence lower bound (ELBO) cost function by applying the gradient descent technique over the model parameters [61]. VAE and its variants such as conditional VAE (CVAE) [131] have been successfully applied in applications such as diverse colorization [132], attribute to image [133] and forecasting motion [131]. GAN is another generative model which attempts to train a generator network by simultaneously training a discriminator network [64]. Unlike existing probabilistic generative models such as DBN, DBM, DSBN and VAE, GAN utilizes an adversarial learning scheme where samples produced by the generator network are challenged by the discriminator network which determines the difference between the generated sample and the real sample. The training is carried out until the discriminator network is maximally confused i.e. the discriminator network cannot differentiate the generated sample from real sample [64]. This learning scheme of GAN falls into the category of semi-supervised learning. GAN has been successfully applied in many image processing applications such as image super resolution [134], text to image synthesis [135] and image inpainting [136]. Though GAN is commonly referred as a class of generative model, it utilizes a deterministic approach to build the generator and the discriminator network. More specifically, the generator network is designed using an inverse convolution mechanism while the discriminator network is a standard convolutional binary classifier network. Hence, a direct model comparison from a probabilistic standpoint between GAN and other probabilistic generative models is not feasible.

4.2.3 LIMITATIONS OF CURRENT GENERATIVE MODELS

All the deep generative models discussed above such as DBN, DBM, DSBN, VAE and GAN utilize hierarchical feed-forward information processing in the architecture to learn meaningful representations from input data. This hierarchical information processing is generally inspired by biological neural information processing systems. However, several studies [99, 137] also suggest the presence of recurrent information processing in biology for learning efficient representations from the input stimuli. Hence, recurrency in generative models may assist in learning more meaningful and efficient representations of data.

Consequently, this work proposes a novel directed deep recurrent probabilistic generative model known as deep simultaneous recurrent belief network (D-SRBN) for efficient representation learning. The proposed model utilizes a unique type of recurrency found in simultaneous recurrent networks (SRNs) [48]. Several studies [46, 114-116] have proven the superiority of SRNs in efficacy and performance compared to regular feed-forward based architectures by solving challenging problems such as topological mapping, decision making and image recognition.

4.3 DEEP SIMULTANEOUS RECURRENT BELIEF NETWORK (D-SRBN) MODEL

This section provides the detailed architecture, mathematic formulation followed by inference and learning procedure of the proposed D-SRBN model.

4.3.1 ARCHITECTURE AND PROBABILITY DISTRIBUTION OF D-SRBN MODEL

The D-SRBN is a directed deep recurrent generative model. The architecture of the proposed deep simultaneous recurrent belief network (D-SRBN) generative model is shown in Fig. 19 (a). Moreover, a time unfolded version of the generative model is shown in Fig. 19 (c) for further clarity. Each "h" in Fig. 19 (a) and Fig. 19 (c) represents a recurrent layer at layer l with J_l hidden units. For the first layer $h^{(1)} = v$ represents the visible unit (input). The simultaneous recurrency in the hidden recurrent layers are applied for $t = 1, \ldots, T$ steps. The input/feature at each layer is simultaneously applied T times at each hidden recurrent step t along with the previous recurrent input at $t - 1$. More specifically, each recurrent layer at each time step t, receives input from the last recurrent layer output of the previous layer $h_T^{(l-1)}$ as well as from the previous time step $h_{(t-1)}^{(l)}$ of the same layer as shown in Fig. 19 (c). The addition of simultaneous recurrency provides t additional non-linear processing capability within each hidden layer, which may enable the model to learn more complex features. This eventually facilitates the D-SRBN model to learn better representation of the input data.

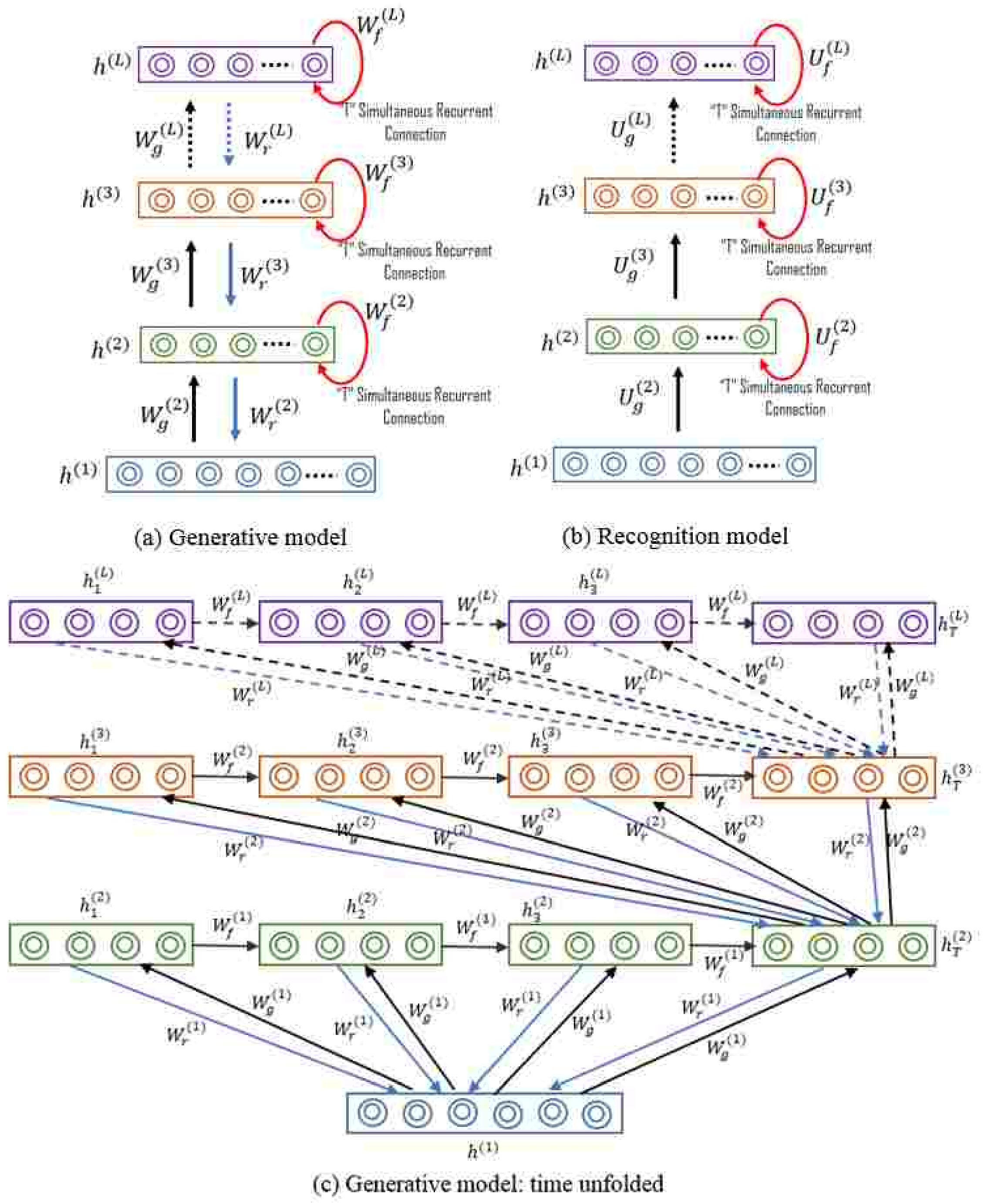

"T" Simultaneous Recurrent Connection
(a) Generative model
(b) Recognition model
(c) Generative model: time unfolded

Fig. 19. Architecture of the proposed Deep Simultaneous Recurrent Belief Network (D-SRBN) model: (a) Generative model, (b) Recognition model and (c) Time unfolded version of the generative model. Each "h" represents a recurrent layer containing J_l hidden units at layer l except the first layer where $h^{(1)} = v$ indicates the visible unit (input).

Accordingly, the joint probability of the D-SRBN is written as,

$$p_\theta\left(h^{(1)} \middle| H^{(L)}\right) =$$

$$\prod_{l=2}^{L}\left[p\left(h_1^{(l)}\right)p\left(h_1^{(l)}\middle|h_1^{(l+1)}\right)\prod_{t=2}^{T}(p\left(h_t^{(l)}\middle|h_{t-1}^{(l)},h_T^{(l-1)},h_t^{(l+1)}\right)p\left(h_T^{(l-1)}\middle|h_t^{(l)}\right)\right], \quad (17)$$

$$for\ l = L, h_t^{(l+1)} = 0;$$

where, $h^{(1)} = v$ indicates the visible layer (input), $H^{(L)} = [h_1^{(L)}, h_2^{(L)}, \ldots, h_T^{(L)}]$, $h_t^{(l)}$ represents hidden recurrent units at layer l and time step t where each $h_t^{(l)} \in \{0,1\}^{J_l}$, L denotes the number of layers and T denotes the last recurrent step. Moreover, $h_0^{(l)}$ needed for the prior model $p\left(h_1^{(l)}\right)$ and $p\left(h_1^{(l)}\middle|h_1^{(l+1)}\right)$ are defined as zero vectors. Each conditional distribution in (17) is expressed as,

$$p\left(h_t^{(l)}\middle|h_{t-1}^{(l)},h_T^{(l-1)},h_t^{(l+1)}\right) = \sigma\left({W_r^{(l)}}^T h_t^{(l+1)} + {W_f^{(l)}}^T h_{t-1}^{(l)} + {W_g^{(l)}}^T h_{\mathrm{T}}^{(l-1)} + b^{(l)}\right);$$

$$for\ l = 2\ to\ L;\ for\ l = L, h_t^{(l+1)} = 0; t = 2\ to\ T \text{ and } h_{\mathrm{T}}^{(1)} = v; \quad (18)$$

and

$$p\left(h_T^{(l-1)}\middle|h_t^{(l)}\right) = \sigma\left({W_r^{(l)}}^T h_t^{(l)} + b^{(l)}\right)$$

$$for\ l = 2\ to\ L; t = 2\ to\ T; \quad (19)$$

where, the model parameters $\theta \in \{W_r^{(l)}, W_f^{(l)},\ W_g^{(l)}, b^{(l)}\}$ are specified as $W_r^{(l)} \in \mathbb{R}^{J_l \times J_{(l-1)}}$, $W_f^{(l)} \in \mathbb{R}^{J_l \times J_l}$, $W_g^{(l)} \in \mathbb{R}^{J_{(l-1)} \times J_l}$ and $b^{(l)} = [b_1^{(l)}, b_2^{(l)}, \ldots, b_{J_l}^{(l)}]^T$ are bias terms. The conditional distribution in (5) shows that at any layer l the hidden units at recurrent step t ($h_t^{(l)}$) are computed from the previous recurrent steps $t-1$ ($h_{t-1}^{(l)}$), the last recurrent step T from the previous layer $l-1$ ($h_{\mathrm{T}}^{(l-1)}$) and the recurrent steps from the layer above $l+1$ ($h_t^{(l+1)}$). This computation can be considered as going bottom up in the graphical model based on the directed connection shown in Fig. 19 (c). Conversely, the conditional distribution in (19) performs the top down computation which can be considered as the reconstruction step, where the recurrent units of the last step T at any layer $l-1$ ($h_T^{(l-1)}$) are computed from the recurrent steps of the layer above l ($h_t^{(l)}$). Further detailed derivations of the proposed D-SRBN generative model are provided in the Appendix B (Equation (B.1)-(B.3)).

4.3.2 INFERENCE AND PARAMETER LEARNING OF D-SRBN

The exact posterior computation of the D-SRBN model shown in (17) is intractable. Therefore, in this section we obtain approximate posterior distribution for the inference model of the proposed D-SRBN model. This inference model is utilized to derive the variational lower bound objective function.

Given an observation $h^{(1)} = v$, the parameters θ of the D-SRBN model, $p_\theta\left(h^{(1)} \middle| H^{(L)}\right)$ shown in (17), are trained by defining the variational lower bound objective function. First, a fixed-form distribution, $q_\varphi\left(H^{(L)} \middle| h^{(1)}\right)$ with parameters φ is introduced which approximates the true posterior distribution, $p\left(H^{(L)} \middle| h^{(1)}\right)$. We utilize the approximate posterior distribution, $q_\varphi\left(H^{(L)} \middle| h^{(1)}\right)$ and follow the variational principle to derive the lower bound on the marginal log-likelihood which is expressed as,

$$£\left(h^{(1)}, \theta, \varphi\right) = \mathbb{E}_{q_\varphi\left(H^{(L)} \middle| h^{(1)}\right)}\left[\log p_\theta\left(h^{(1)}, H^{(L)}\right) - \log q_\varphi\left(H^{(L)} \middle| h^{(1)}\right)\right]; \tag{20}$$

where, $h^{(1)} = v$ denotes the input, $H^{(L)}$ indicates the last hidden recurrent layer, θ and φ denotes the model and the approximate model parameters, respectively. The approximate posterior, $q_\varphi\left(H^{(L)} \middle| h^{(1)}\right)$ is defined as a recognition model [15] and the graphical architecture is shown in Fig. 19 (b). The recognition model is expressed as follows,

$$q_{\varphi}\left(H^{(L)} \middle| h^{(1)}\right) = \prod_{l=2}^{L} \left[q\left(h_1^{(l)} \middle| h_T^{(l-1)}\right) \prod_{t=2}^{T} (q\left(h_t^{(l)} \middle| h_{t-1}^{(l)}, h_T^{(l-1)}\right) \right]; \tag{21}$$

The conditional distribution in (21) is specified as,

$$q\left(h_t^{(l)} \middle| h_{t-1}^{(l)}, h_T^{(l-1)}\right) = \sigma\left(U_g^{(l)^T} h_T^{(l-1)} + U_f^{(l)^T} h_{t-1}^{(l)} + c^{(l)}\right),$$
$$for\ l = 2\ to\ L;\ t = 2\ to\ T \text{ and } h_{\mathrm{T}}^{(1)} = v. \tag{22}$$

where, the model parameters $\varphi \in \{U_f^{(l)},\ U_g^{(l)}, c^{(l)}\}$ are specified as $U_f^{(l)} \in \mathbb{R}^{J_l \times J_l}$, $U_g^{(l)} \in \mathbb{R}^{J_{(l-1)} \times J_l}$ and $c^{(l)} = [c_1^{(l)}, c_2^{(l)}, \dots, c_{J_l}^{(l)}]^T$ are bias terms. The conditional distribution in (22) shows that at any layer l the hidden recurrent units at time step t ($h_t^{(l)}$) of the recognition model is computed from the recurrent units computed at the time steps $t-1$ ($h_{t-1}^{(l)}$) and the last recurrent step T from the previous layer $l-1$ ($h_T^{(l-1)}$). Defining the approximate posterior using such a recognition model enables both fast inference and efficient parameter computation where the variational parameters φ are computed simultaneously for all v rather than per data point [130]. Moreover, the parameters φ of the recognition model are learned simultaneously with the parameters of the generative model θ.

The parameters $\{\theta, \varphi\}$ of the D-SRBN are learned by optimizing (20). We use the NVIL algorithm [59, 130] which utilizes Monte Carlo methods to approximate expectations and stochastic gradient descent (SGD) to optimize the parameters, θ and φ. The gradients in terms of the model parameters θ and φ are expressed as,

$$\nabla_{\theta} £\left(h^{(1)}\right) = \mathbb{E}_{q_{\varphi}\left(H^{(L)} \middle| h^{(1)}\right)} \left[\nabla_{\theta} \log p_{\theta}\left(h^{(1)}, H^{(L)}\right)\right], \tag{23}$$

and

$$\nabla_{\varphi}£\left(h^{(1)}\right) = \mathbb{E}_{q_{\varphi}\left(H^{(L)}\middle|h^{(1)}\right)}[(\log p_{\theta}\left(h^{(1)}, H^{(L)}\right) - \log q_{\varphi}\left(H^{(L)}\middle|h^{(1)}\right)) \times \nabla_{\varphi} \log q_{\varphi}\left(H^{(L)}\middle|h^{(1)}\right)]. \quad (24)$$

where, $\nabla_{\theta}£$ and $\nabla_{\varphi}£$ denote the gradient of £ in terms of θ and φ.

Algorithm 2 shows the learning procedure of D-SRBN using the NVIL algorithm. The detailed equations (as shown in Algorithm 2) necessary to understand the generative model, the recognition model and the lower bound objective function derived for the D-SRBN are provided in Appendix B.

Algorithm 2: Learning procedure of D-SRBN using NVIL

Initialization:

- ■ Set generative model parameters, $\theta \in \{W_r^{(l)}, W_f^{(l)},\ W_g^{(l)}, b^{(l)}\}$ and recognition model parameters, $\varphi \in \{U_f^{(l)},\ \mathrm{U}_\mathrm{g}^{(l)}, c^{(l)}\}$ with random values
- ■ Set $\Delta\theta \leftarrow 0$, $\Delta\varphi\ \leftarrow 0$ and £ $\leftarrow 0$
- ■ Set learning rate, $\propto$ with a small value
- ■ Set $h^{(1)} = v$ and $h_0^{(l)} = 0$

Training:

```
for each epoch
    for each mini-batch
        a. Perform generative step:
            for $t \leftarrow 1\ to\ T$
                Compute $p_\theta\left(h^{(1)} \middle| h_t^{(L)}\right)$ using Eq. (B1)-(B3)
            end
        b. Perform recognition step:
            for $t \leftarrow 1\ to\ T$
                Compute $q_\varphi\left(h_t^{(L)} \middle| h^{(1)}\right)$ using Eq. (B4)
            end
        c. Compute variational lower bound, £
            for $t \leftarrow 1\ to\ T$
                ■ Compute $e_t$using Eq. (B6)-(B10)
                ■ $£ \leftarrow £ + e_t$
            end
        d. Compute gradients:
            for $t \leftarrow 1\ to\ T$
                ■ $\Delta\theta \leftarrow \Delta\theta + \nabla_\theta \log p_\theta\left(h^{(1)} \middle| h_t^{(L)}\right)$
                ■ $\Delta\varphi \leftarrow \Delta\varphi + e_t \nabla_\varphi \log q_\varphi\left(h_t^{(L)} \middle| h^{(1)}\right)$
            end
    end
        e. Update the model parameters:
            ■ $\theta(new) \leftarrow \theta(old) - \propto * \Delta\theta$
            ■ $\varphi(new) \leftarrow \varphi(old) - \propto * \Delta\varphi$
end
```

4.4 RESULTS AND DISCUSSION

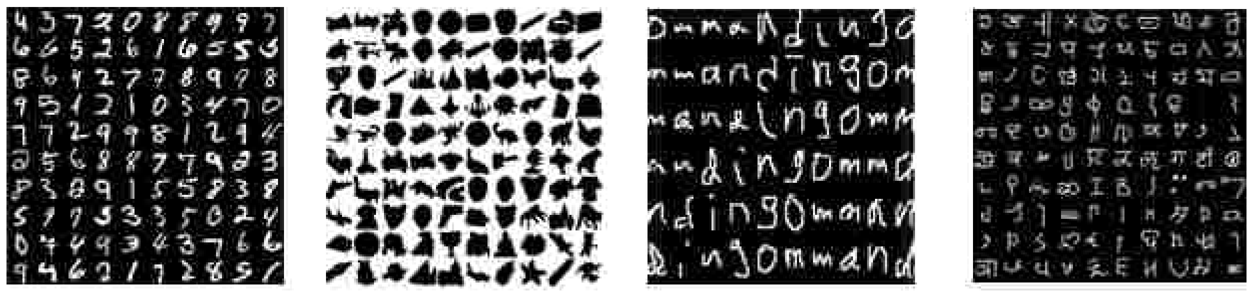
This section discusses the representation learning performance of the proposed D-SRBN model compared to the state-of-the-art models using four widely used standard benchmark datasets such as MNIST [22], Caltech 101 Silhouettes [138], OCR letters[1] and Omniglot [139], respectively. These datasets are the most widely used datasets in the literature for evaluating the performance of the generative models. Some random example images from each of the above datasets are shown in Fig. 20. Further, we compare the proposed model with three state-of-the-art deep generative models such as Deep Belief Network (DBN), Deep Boltzman Machine (DBM), Deep Sigmoid Belief Network (DSBN) and Variational Auto-Encoder (VAE), respectively. Quantitative evaluation of deep generative models is crucial to measure and compare different probabilistic models. The performance of the models are assessed by generating samples from a specific model and obtaining the average log probability metric using a test dataset for that model. As mentioned above, the specific performance metric used in this study is known as negative log-likelihood [59, 60, 78, 127, 140]. This metric calculation varies depending on the variational inference technique used in the probabilistic model. For our proposed D-SRBN model the average test negative log-probability is computed as follows.

$$\text{Average test log-probability} = \frac{1}{n_{test}} \sum_{i \in n_{test}} \sum_{t=1}^{T} E_{q_{\varphi}\left(H^{(L)} \middle| x_{test}^{i}\right)}([e_t]) \tag{25}$$

where, n_{test} denotes the number of testing samples, x_{test}^{i} denotes the i^{th} testing sample, $H^{(L)}$ denotes the latent representation of the test data x_{test}^{i} and e_t denotes the variational lower bound which is shown in (B.6) in the appendix. The average test log-probability computation for the

[1] http://ai.stanford.edu/~btaskar/ocr/

deep generative models in comparison can be found in [60, 61, 141]. The next few sections summarize and discuss the results for each benchmark dataset.



(a) (b) (c) (d)

Fig. 20. Random example images from the dataset; (a) MNIST, (b) Caltech 101 Silhouettes, (c) OCR letters, (d) Omniglot.

4.4.1 MNIST DATASET

The MNIST dataset [22] contains 60,000 training and 10,000 test images of ten handwritten digits (0 to 9) with image size of 28×28. The binarized version of the dataset is used for these experiments [141]. We report the average log-probability of the test data obtained by the proposed D-SRBN model trained with NVIL and SGD. The detailed hyper parameters setup to train the D-SRBN model are as follows: number of layers = 2, number of units at each layer = 200, $T = 15$, learning rate = 0.001, weight decay = 0.0001, momentum = 0.9, step size = 0.001 and mini batch size = 256. The DSBN model is trained using NVIL and the test log-probability is approximated from the variational lower bound for comparison. We follow [60] to obtain the architecture of the DSBN model along with the learning parameters for the NVIL algorithm. On the other hand, the performance of the DBN and DBM models are evaluated by estimating the variational lower bound for the average log-likelihood on the test set using the annealed importance sampling (AIS) method [141]. The architecture and learning parameters of the DBN and the DBM models are obtained from [79] and [78], respectively. Moreover, the architecture and the learning parameters for the VAE model is obtained from [61].

Fig. 21 shows the average log-probability achieved by the proposed D-SRBN, DBN, DBM, DSBN and VAE model on the test dataset at each iteration, respectively. Moreover, Fig.21 shows the average test log probability achieved by a single layer version (SRBN) of the D-SRBN model for comparison. The D-SRBN model achieves a higher average test log-probability than the three comparative models after 50 iterations. This demonstrates the effectiveness of the proposed model for achieving faster model-fit of the data compared to the three state-of-the-art deep generative models. It should be noted here that the VAE model achieves a marginally higher average test log-probability than the proposed D-SRBN model towards the end of the training iterations as shown in Fig. 21. However, the performance of the D-SRBN model is significantly better for the first 300 epochs compared to the VAE model. This shows that the proposed D-SRBN model achieves a considerably faster convergence compared to the VAE model.

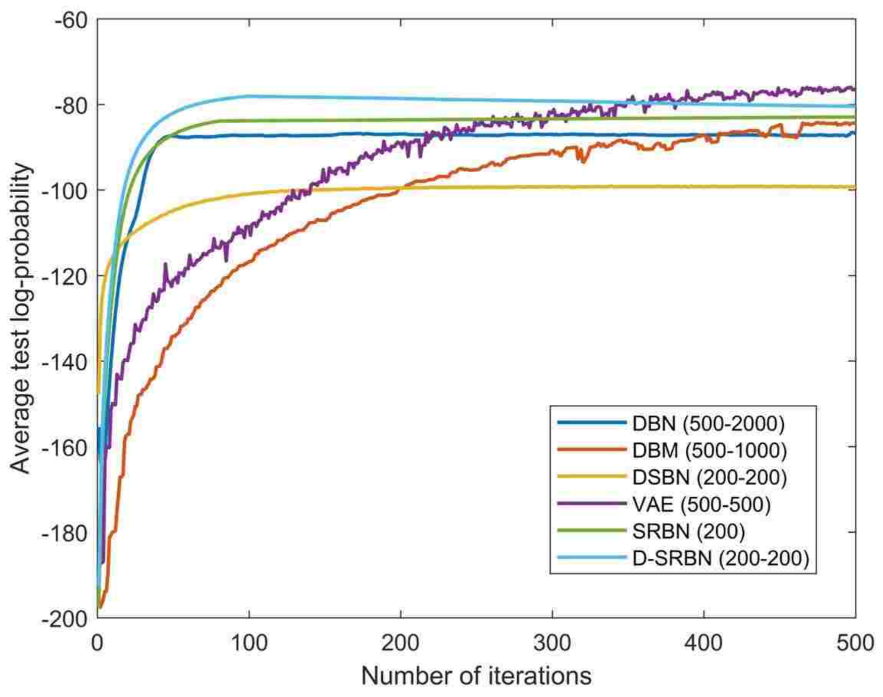


Fig. 21. Comparison of the proposed D-SRBN model with four state-of-the-art generative models: DBN, DBM, DSBN and VAE based on the average test log-probability using the MNIST dataset.

Table 6 shows the best test log-probability estimate with corresponding iteration number achieved by all five deep generative models. The parameters are tuned to obtain the best test log-probability metric for each of the DBN, DBM and DSBN models to reflect the results reported in studies [9], [2] and [8] on MNIST the dataset, respectively. For the VAE model we have achieved better performance than the one reported in [61] for the MNIST dataset. Table 6 demonstrates that the proposed D-SRBN model is faster and achieves better performance than the three state-of-the-art DBN, DBM and DSBN deep generative models. Conversely, the VAE model achieves slightly better performance than the D-SRBN model. However, this slight performance improvement achieved by the VAE model is at the expense of 2.5 times more hidden units (500 units at each layer) than the D-SRBN model (200 units at each layer). Additionally, Table 6 shows that the D-SRBN model achieves the best performance at training epoch 101 whereas the VAE model achieves the best performance at epoch 497. This demonstrates that the D-SRBN model converges with a minimal number of training epochs compared to the VAE and other state-of-the art deep generative models.

TABLE 6

COMPARISON OF THE PROPOSED D-SRBN MODEL WITH THE STATE-OF-THE-ART DEEP GENERATIVE MODELS: DBN, DBM, DSBN AND VAE BASED ON THE LOG PROBABILITY OF TEST DATA ON MNIST DATASET

Model	**Number of units in each hidden layer**	**Best test log-probability**	**Number of iterations taken**
DBN	500-2000	-86.56	498
DBM	500-1000	-84.27	475
DSBN	200-200	-99.11	408
VAE	500-500	-76.07	497
SRBN	200	-82.96	500
D-SRBN	200-200	**-78.12**	101

Further, the D-SRBN model utilizes fewer hidden units when compared to the DBN and DBM models and the same number of hidden units when compared to DSBN.

4.4.2 CALTECH 101 SILHOUETTTES DATASET

We perform the second experiment using the Caltech 101 Silhouettes dataset [138]. The dataset is composed of 6,364 training and 2,307 test images of size 28×28, representing object silhouettes of 101 classes. The training and the testing protocols for the remaining deep generative models discussed in this work are similar to the one used for the MNIST dataset. Additionally, the hyper parameters used to train the D-SRBN model are unchanged from the previous experiment. Fig. 22 shows the average log-probability of the test dataset obtained at each iteration for the five generative models. The proposed D-SRBN model achieves a higher average test log-probability than the three comparative models after 20 iterations. Once again this demonstrates the effectiveness of the proposed model for achieving faster model-fit of the data compared to the four state-of-the-art deep generative models.

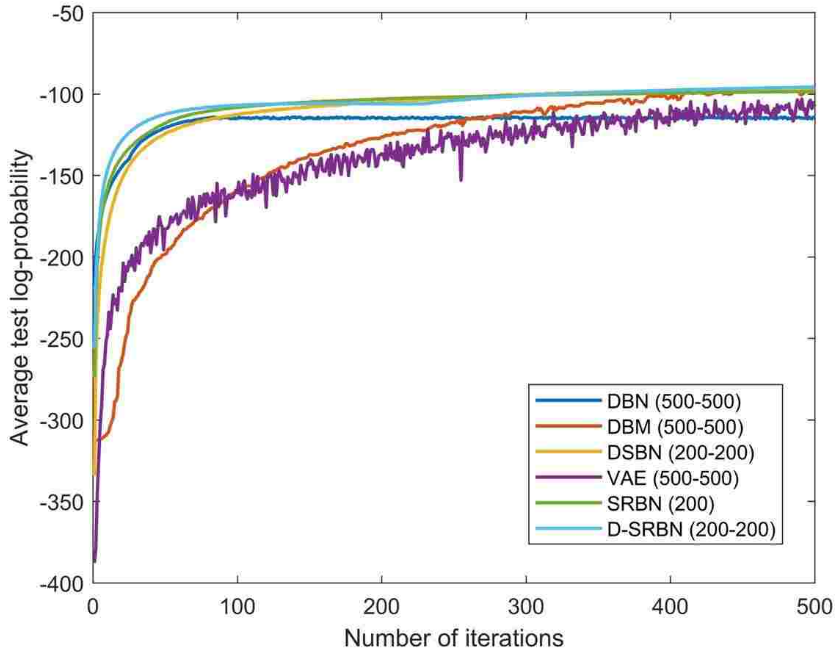


Fig. 22. Comparison of the proposed D-SRBN model with four state-of-the-art generative models: DBN, DBM, DSBN and VAE based on the average test log-probability using the Caltech 101 Silhouettes dataset.

Table 7 shows a comparison between the proposed D-SRBN model and the four state-of-the-art generative models in terms of the best test log-probability metric with associated iteration number. Additionally, we show the average test log likelihood performance obtained by the SRBN model. The parameters of the DBN, DBM, DSBN and VAE models are tuned to obtain the best test log-probability as reported in the literature [60, 142]. Table 7 demonstrates that both the proposed SRBN and D-SRBN model achieves better performance than the state-of-the-art models. Nevertheless, the performance of the D-SRBN model is better than the single layer SRBN model. Though the D-SRBN model achieves the best test log-probability at iteration number 499, the performance improvement compared to the other generative models occurs after just 20 iterations as mentioned above.

TABLE 7

COMPARISON OF THE PROPOSED D-SRBN MODEL WITH THE STATE-OF-THE-ART DEEP GENERATIVE MODELS: DBN, DBM, DSBN AND VAE BASED ON THE LOG PROBABILITY OF TEST DATA ON CALTECH 101 DATASET

Model	**Number of units in each hidden layer**	**Best test log-probability**	**Number of iterations taken**
DBN	500-500	-114.21	477
DBM	500-500	-98.20	473
DSBN	200-200	-97.32	482
VAE	500-500	-103.13	491
SRBN	200	-98.43	497
D-SRBN	200-200	**-95.74**	499

Moreover, here again the D-SRBN model utilizes fewer hidden units when compared to DBN and DBM models and similar number of hidden units when compared to DSBN.

4.4.3 OCR LETTERS DATASET

The third experiment is based on the OCR letters dataset which contains images of 26 letters of the English alphabet. The dataset is composed of 42,152 training and 10,000 test image examples of size 16×8. The training and testing protocols for the deep generative models are similar to the previous experiments. The hyper parameter setup for the D-SRBN model is similar to the previous experiments except T is set to 11 which is experimentally found to provide the best performance for the OCR letters dataset. The architecture and learning parameters of the DSBN and DBM models are obtained from [60] and [127], respectively. Additionally, we perform an in-house experiment to obtain the best architecture for the DBN and VAE model since to the best of our knowledge there are no reported results for the OCR letters dataset using the DBN and VAE models. Our results suggest that a DBN model with two hidden layers containing 1000 hidden units at each layer and the VAE model with 500 hidden units at each layer achieves the best performance. The average log-probability of the test dataset observed at each iteration for all deep generative models is shown in Fig. 23 which shows that the proposed D-SRBN model demonstrates faster and much better performance than the state-of-the-art deep generative models. Moreover, Fig. 23 demonstrates that the single layer SRBN model shows improved performance compared to the state-of-the-art models.

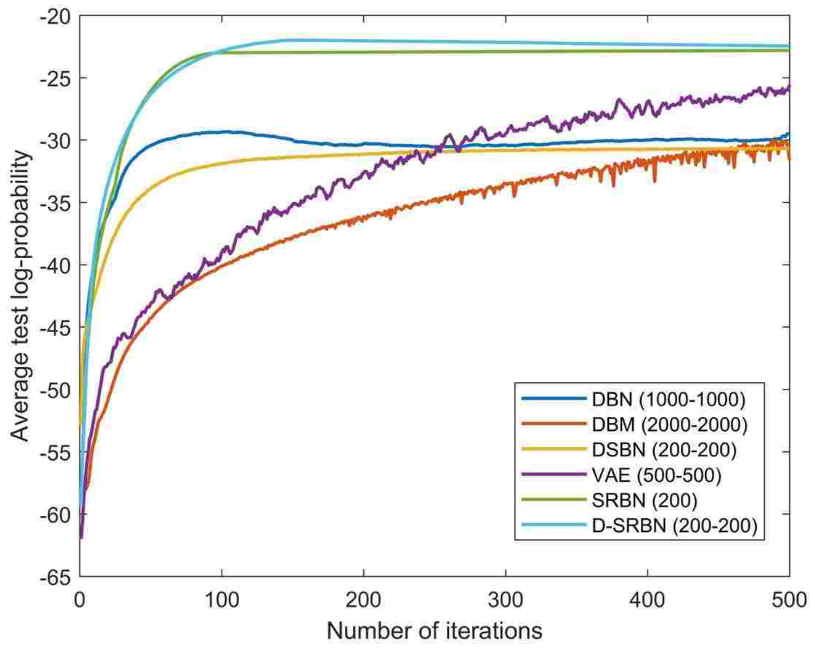


Fig. 23. Comparison of the proposed D-SRBN model with three state-of-the-art generative models: DBN, DBM, DSBN and VAE based on the average test log-probability using the OCR letters dataset.

We also present a comparison between the four deep generative models in terms of the best test log-probability metric for the corresponding iteration number in Table 8. The parameters of the DBN, DBM, DSBN and VAE models are tuned to obtain the best test log-probability. Note that our experiments using the state-of-the-art DBN, DBM, DSBN and VAE models show better results than the best results reported in the literature [60, 127]. However, in this case also the proposed D-SRBN achieves a significantly higher performance than that of the state-of-the-art deep generative models while utilizing fewer or similar numbers of hidden units. Table 8 shows that the performance of the D-SRBN model is slightly better than the SRBN model while the SRBN model achieves improved performance compared to the state-of-the-art models. Though the D-SRBN model takes a few more iterations (150) to achieve the best test log-probability compared to the DBN model (104), the performance improvement begins after just the 15th iteration and the improvement is noticeably better as shown in Fig. 23 and Table 8, respectively.

TABLE 8

COMPARISON OF THE PROPOSED D-SRBN MODEL WITH THE STATE-OF-THE-ART DEEP GENERATIVE MODELS: DBN, DBM, DSBN AND VAE BASED ON THE LOG PROBABILITY OF TEST DATA ON OCR LETTERS DATASET

Model	**Number of units in each hidden layer**	**Best test log-probability**	**Number of iterations taken**
DBN	1000-1000	-29.33	104
DBM	2000-2000	-30.12	489
DSBN	200-200	-30.66	471
VAE	500-500	-25.57	500
SRBN	200	-22.82	500
D-SRBN	200-200	**-21.99**	150

4.4.4 OMNIGLOT DATASET

The final experiment utilizes the Omniglot dataset [139] that contains images of hand-written characters across many world alphabets. We partition and preprocess the dataset following [143] which results in 24,345 training and 8,070 test image examples of size 28×28, representing 50 classes. The architecture and the hyper parameter setup for the D-SRBN model is unchanged from the MNIST experiment. The architectures of the DBN and DBM models are implemented following [144] as they show the best reported results. However, to the best of our knowledge, there are no reported results for the Omniglot dataset using DSBN and VAE; hence, we conduct an in-house experiment to determine the best architecture for the DSBN and VAE model. Our results suggest that a DSBN model with three hidden layers containing 200 hidden units at each layer and the VAE model with two layers containing 500 units achieves the best performance. Fig. 24 shows the iteration-wise average log-probability of the test dataset for the four deep generative models. Fig. 24 demonstrates that for the first few iterations the performance of the DBN, DBM and the DSBN models are better than our proposed D-SRBN model. However, as the learning progresses the D-SRBN model achieves a higher performance than the state-of-the-art DBN, DBM and DSBN models. Moreover, the performance of the D-SRBN model remains superior to the VAE model in all training epochs.

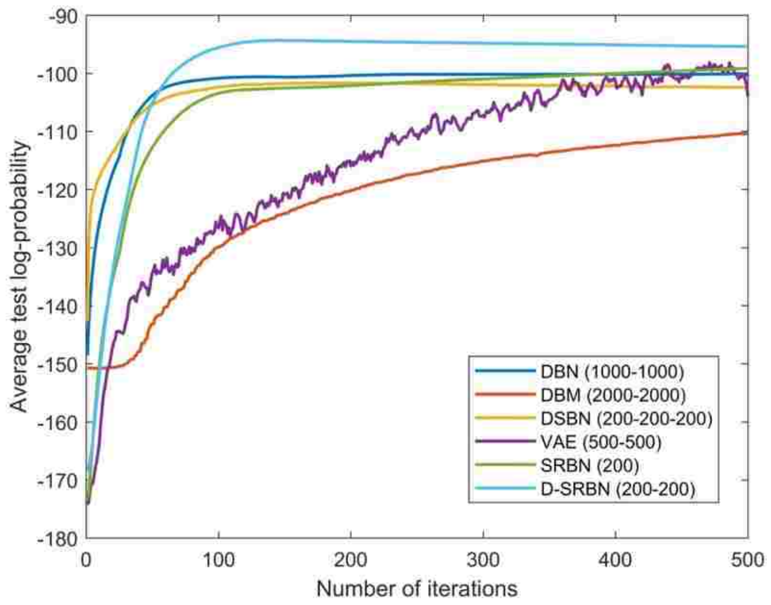


Fig. 24. Comparison of the proposed D-SRBN model with three state-of-the-art generative models: DBN, DBM, DSBN and VAE based on the average test log-probability using the Omniglot dataset.

Table 9 shows a comparison among all the deep generative models considered in this work for the test log-probability metric and corresponding iteration number. In this experiment the parameters of the deep generative models are tuned to obtain the best test log-probability following [144]. Table 9 demonstrates that the proposed D-SRBN model achieves faster convergence and outperforms the state-of-the-art deep generative models while utilizing fewer hidden units. It should be noted here that the performance of the SRBN model is lower than both the D-SRBN and VAE model.

TABLE 9

COMPARISON OF THE PROPOSED D-SRBN MODEL WITH THE STATE-OF-THE-ART DEEP GENERATIVE MODELS: DBN, DBM, DSBN AND VAE BASED ON THE LOG PROBABILITY OF TEST DATA ON OMNIGLOT DATASET

Model	**Number of units in each hidden layer**	**Best test log-probability**	**Number of iterations taken**
DBN	1000-1000	-100.02	499
DBM	2000-2000	-110.32	495
DSBN	200-200-200	-101.64	206
VAE	500-500	-98.16	486
SRBN	200	-99.13	500
D-SRBN	200-200	**-94.30**	145

Note for all four experiments, the proposed D-SRBN offers better or comparable log probability metric in fewer training iterations when compared to all other state-of-the-art models studied in this work. Moreover, our experimental results suggest that the single layer SRBN model achieves better or comparable performance than the state-of the-art deep generative models. However, the performance of the D-SRBN model is consistently better than the SRBN model. This in turn suggests that the proposed D-SRBN generative model offers better model-fit of the data when compared to the state-of-the-art models in this study. Furthermore, Table 6 to Table 9 demonstrate that the D-SRBN model utilizes fewer or the same number of hidden units compared to the above mentioned models. The number of training parameters utilized by the D-SRBN model is far fewer than that of the DBN, DBM and VAE models for all four experiments. The D-SRBN model utilizes slightly more parameters compared to the DSBN model (though the number of hidden units is the same). This is due to the additional recurrent layers incorporated in the hidden layers of the D-SRBN model. However, these recurrent layers offer further depth within each hidden layer of the D-SRBN model which eventually enables the D-SRBN model to achieve significantly better performance than the DSBN model.

4.5 SUMMARY

This chapter proposes a novel deep recurrent probabilistic generative D-SRBN model for efficient representation learning which is a logical extension of our proposed DSRN based discriminative model in the previous chapter. However, unlike, DSRN, the probabilistic D-SRBN model allows representation learning from the input data in an unsupervised fashion. Our experiments use four benchmark datasets, MNIST, OCR Letters, Caltech 101 Silhouettes, and Omniglot, to demonstrate that the proposed D-SRBN model achieves better representation learning performance compared to the state-of-the-art deep generative models such as DBN, DBM, DSBN, and VAE while utilizing fewer training parameters. These findings are published in [145]. This suggests that the D-SRBN model can be considered as an efficient building block for designing more sophisticated deep learning frameworks to handle more challenging tasks such as small data learning.

CHAPTER 5

DEEP RECURRENT GENERATIVE HIERARCHICAL BAYESIAN MODEL FOR LEARNING WITH SMALL DATA

5.1 CHAPTER OVERVIEW

This chapter proposes a novel deep recurrent generative Bayesian learning model for addressing the challenge of learning with small data. D-SRBN is introduced in the previous chapter. The D-SRBN model efficiently handles representation learning task using unlabeled data which is essential for any classification task. The proposed D-SRBN model shows improved representation learning performance while utilizing significantly fewer training parameters compared to the state-of-the-art models. However, the D-SRBN model alone is not sufficient to handle the problem of learning using a small amount of data since the model still requires a considerable number of labeled examples to attain good generalization for solving challenging image classification tasks. Bayesian statistics are historically known for learning from limited training data. However, the Bayesian models usually suffer from lack of good priors resulting in low performance quality in difficult image classification tasks. Consequently, this chapter proposes a deep simultaneous recurrent belief network-hierarchical Bayesian (DSRBN-HB) model for solving several challenging image classification tasks using very limited labeled training examples. Specifically, we address the intricate one-shot image classification problem where a model is required to classify images from a previously unseen category.

5.2 LITERATURE REVIEW

Deep learning models have been quite successful in solving challenging problems in various application domains such as computer vision, pattern recognition, medical image analysis, cyber-security and many more. The performance of the deep learning models depends on several factors such as the type of the architecture, the depth of the model, the learning algorithm, the hyper-parameter settings, and most importantly the number of training examples. Deep learning models are supervised models and require thousands or millions of training examples to achieve a good generalization. In comparison, human learners usually require just one or a few examples of a new category to make meaningful generalizations to novel instances [34, 35, 146]. In many practical applications such as medical image analysis and cybersecurity massive amounts of training data may not be available, so a sophisticated deep learning framework is desirable that can effectively handle the challenge of learning with a small amount of data.

5.2.1 LEARNING TECHNIQUES WITH SMALL DATA AND ITS LIMITATIONS

In recent years, the challenge of learning with small data has drawn increasing attention in the machine learning research community. The challenge of learning with small data is commonly called one-shot learning, few-shot learning and zero-shot learning in the literature. Therefore, in this paper, we use these terms interchangeably. Several studies [77, 139, 147-160] propose a variety of machine learning, deep learning and statistical techniques to approach the challenge of learning with small data. These approaches can be broadly grouped into three categories: meta learning based approach, transfer learning based approach, and Bayesian statistics based approach. In the next few sections we briefly discuss these approaches and their corresponding limitations.

5.2.1.1 META LEARNING BASED APPROACH FOR LEARNING WITH SMALL DATA

Meta-learning is a popular technique for solving the challenge of learning with small data. In meta-learning, a model for a specific task is learned by a specialized, trainable algorithm called the meta-learner. The learning occurs in two levels, gradual learning, which learns the meta-learner, and rapid learning, where the meta-learner trains the task-specific model. Bengio *et al.* [161] and Schmidhuber *et al.* [162] propose meta-learning techniques for training a meta-learner that learns to update the parameters of the learner's model. This approach is applied for optimizing parameters of deep neural networks [163, 164] and for learning dynamically changing recurrent neural networks [165]. Recently, Ravi *et al.* [166] extend these techniques to develop a meta-learning approach that learns both the weight initialization and the optimizer for solving few-shot image recognition task. The proposed method utilizes long short-term memory (LSTM) as the meta-learner to model the parameters of a learner, a convolutional neural network (CNN). However, this technique is prohibitively complex since each parameter of the learner is updated independently in each step. Koch *et al.* [152] and Vinyl *et al.* [148] introduce a metrics based meta-learner such as k-nearest neighbor (k-NN) and nearest neighbor with cosine similarities to train a Siamese network as the learner for solving the one-shot image recognition task. However, a metric does not really train a learner, rather it modifies the pairwise distance between examples. Consequently, Finn *et al.* [167] propose an optimizer, stochastic gradient descent (SGD) based meta-learner known as model agnostic meta-learning for solving the few-shot learning task. The proposed method works well in practice when compared to the state-of-the-art meta-learning techniques. From the above discussion, it is evident that the main challenge for the meta-learning strategy is in designing the appropriate meta-learners to be learned. Moreover, the technique is relatively problem and data specific and, hence, cannot be easily generalizable.

5.2.1.2 TRANSFER LEARNING BASED APPROACH FOR LEARNING WITH SMALL DATA

Transfer learning allows deep neural network models to learn from a pre-trained state rather than learning from scratch. Usually, a deep neural network model is first trained using massive amounts of available labeled or unlabeled training examples. Subsequently, the parameters of the trained model are utilized as the starting point and re-trained using a different dataset where number of labeled training examples is scarce. This simple yet efficient technique has shown success for solving intricate classification tasks. The ability to quickly learn from fewer training examples is the main motivation behind using the transfer learning technique for solving the problem of learning with small data. Moreover, transfer learning techniques are easily adaptable and generalizable for different applications. Anderson *et al.* [158] proposes a transfer learning based technique for learning from limited training data. Their method combines a pre-trained VGG network with an un-trained residual network (ResNet) to learn the shift between data sets. This modular approach adds new features to the network rather than replacing representations via fine-tuning. Blaes *et al.* [153] proposes another transfer learning method for solving the few-shot learning task by utilizing a pre-trained deep network on the ImageNet dataset. The authors introduce a proto-type based learning procedure by adding additional global feature layers at the end of the pre-trained deep network. This global proto-type learning technique enables the proposed model to learn from few training examples of new categories. Another seminal transfer learning based technique in [156] proposes a two-step process: representation learning and few-shot learning, for solving the few-shot visual recognition task. In the representation learning step the authors utilize a CNN trained on ImageNet dataset to learn feature representation from many training instances of base classes. During the few-shot learning step, the pre-trained model is exposed to novel categories to learn from only a few examples per class. The proposed model learns a classifier over the joint label space of the base classes and novel classes. However, the performance of the transfer learning based techniques for learning with small data heavily depends on how well the deep neural network model learns representation from the available labeled or unlabeled training examples. Hence, efficient learning using small data requires sophisticated deep neural networks for representation learning.

5.2.1.3 BAYESIAN TECHNIQUES FOR LEARNING WITH SMALL DATA

Bayesian techniques are powerful tools for understanding the underlying distribution of the data. However, Bayesian techniques require good prior distribution of the data to offer useful classification results. On the other hand, Bayesian techniques are historically popular for their ability to learn from limited training data; hence, they may be suitable for solving classification challenge using small data. Li *et al.* [168] propose a Bayesian approach for solving the one-shot learning task of object categories in an unsupervised fashion. The proposed method utilizes a variational Bayesian framework where object categories are represented by probabilistic models and prior knowledge is represented as a probability density function on the parameters of these models. The posterior model for a novel object category is obtained by updating the prior using one or very few observations. However, this method utilizes hand-crafted techniques for image feature extraction which may not be optimal for learning the Bayesian framework. Maas *et al.* [169] introduces a novel Bayesian network for addressing the one-shot learning problem. The authors mention that conventional Bayesian networks fail to identify and exploit near-deterministic relationships between attributes which is essential for learning a novel category from few examples. The proposed Bayesian network overcomes this limitation by learning a hyperparameter from each distribution in the network that specifies whether it is nondeterministic or near-deterministic. However, this technique is only tested for text data and cannot be readily extended for solving problems encountered in computer vision. Lake *et al.* [160, 170] introduces a Bayesian program learning (BPL) framework which is capable of learning a large class of simple visual concepts from a single example. The authors claim that the proposed BPL technique achieves human level performance for learning handwriting characters. In addition to utilizing pixel intensity information, BPL utilizes the stroke pattern information captured during the handwriting data collection process. Therefore, this technique is hard to generalize for solving a one-shot natural image classification task where such stroke pattern information is not available. Salakhutdinov *et al.* [77] proposes a hierarchical nonparametric Bayesian model for solving the one-shot image classification task. The proposed hierarchical Bayesian (HB) model leverages higher order knowledge abstracted from previously learned categories. This helps the model classify a novel category using very limited training samples. However, the HB model considers raw images as input without applying any feature extraction technique; hence, it may not be suitable for more complex object categorization tasks. To address this limitation, Salakhutdinov *et al.* [159] proposes a hierarchical deep model that combines the DBM probabilistic generative model with a hierarchical Dirichlet process (HDP) based HB model to solve the one-shot classification task. The HDP model works on the feature space generated by the DBM model. The DBM learns meaningful features from the input data, substantially improving the categorization performance of the HDP model. However, as mentioned in Chapter 4, feed-forward based DBM and other similar deep generative models such as DBN, DSBN, and VAE require a huge amount of training parameters to perform the feature learning task which, in turn, affects the training time. Conversely, our proposed deep recurrent generative model, D-SRBN, in Chapter 4 shows improved feature learning performance while utilizing fewer training parameters.

5.3 PROPOSED DEEP RECURRENTE GENERATIVE HIERARCHICAL BAYESIAN MODEL FOR LEARNING WITH SMALL DATA

We propose a combined deep recurrent and Bayesian learning approach to address the intricate learning challenge using small data. More specifically, the deep simultaneous recurrent belief network (DSRBN) model performs efficient representation learning from unlabeled data. The learned representation is considered as input feature space for the HB model. The combined DSRBN-HB model is learned jointly to solve the one-shot learning task. The use of simultaneous recurrency in the DSRBN architecture may enable the model to learn more compact and complex representation from the input data while significantly reducing the number of hidden neurons, which in turn reduces the number of trainable parameters. The compact representation may facilitate the HB model to perform a faster object categorization learning task using very limited training data. Consequently, our proposed DSRBN-HB model efficiently performs the one-shot classification task while ensuring reduction in the number of training parameters..

5.3.1 DSRBN HIERARCHICAL BAYESIAN (DSRBN-HB) FRAMEWORK FOR ONE-SHOT LEARNING

The hierarchical Bayesian (HB) model [77] works on the feature space obtained from a trained DSRBN model. The detailed explanation of the architecture and learning procedure of the DSRBN model is provided in Chapter 4 Section 4.3. Particularly, the features are obtained by passing the inputs to the trained DSRBN recognition model (See Fig. 19 (b)). The HB model operates on the top-level features obtained from the DSRBN model. Let us consider that for input, $X,$ DSRBN top-level features $h^{(L)}$ are obtained from N objects. For simplicity, we first consider a two-level HB model where N objects are partitioned into B basic-level or level-1 categories. Such partition is represented by z^b of length N, where, each entry is $z_n^b \epsilon \{1, \dots, B\}$. Next, we assume that B basic-level categories are partitioned into S super-categories or level-2 categories which is represented by z^s of length B, where, each entry is $z_b^s \epsilon \{1, \dots, S\}$ [77]. The distribution over the DSRBN feature vector for any basic level category is given as follows,

$$P\left({h^{(L)}}^n \middle| z_n^b = b, \theta^1\right) = \prod_{d=1}^{D} \mathcal{N}\left({h^{(L)}}^n{}_d \middle| \mu_d^b, {}^{1}\!/_{\tau_d^b}\right), \tag{26}$$

where, $\mathcal{N}()$ denotes a Gaussian distribution with mean μ and precision τ, d denotes the feature dimension index, D represents the upper limit of the feature dimension, and $\theta^1 = \{\mu^b, \tau^b\}_{b=1}^{B}$ denotes the level-1 category parameters. Next, a conjugate Normal-Gamma prior is placed over $\{\mu^b, \tau^b\}$ to obtain level-2 category written as follows,

$$P\left(\mu_d^b, \mu_d^s \middle| \theta^2\right) = \prod_{d=1}^{D} P\left(\mu_d^b, \mu_d^s \middle| \theta^2, z^s\right), \tag{27}$$

where, $\theta^2 = \{\mu^s, \tau^s, \alpha^s\}_{s=1}^{S}$ denotes the level-2 parameters. Each dimension, d, in (27) is given as,

$$P\left(\mu_d^b, \mu_d^s \middle| \theta^2\right) = \mathcal{N}\left(\mu_d^b \middle| \mu_d^s, {}^{1}\!/_{\tau_d^b}\right) \Gamma\left(\tau_d^b \middle| \alpha_d^s, {}^{\alpha_d^s}\!/_{\tau_d^s}\right), \tag{28}$$

where, $\Gamma()$ denotes Gamma density function. Note for level-2 parameters θ^2, the following conjugate priors are assumed,

$$P(\mu_d^s) = \mathcal{N}\left(\mu_d^s \middle| 0, {}^{1}\!/_{\tau^0}\right), \tag{29}$$

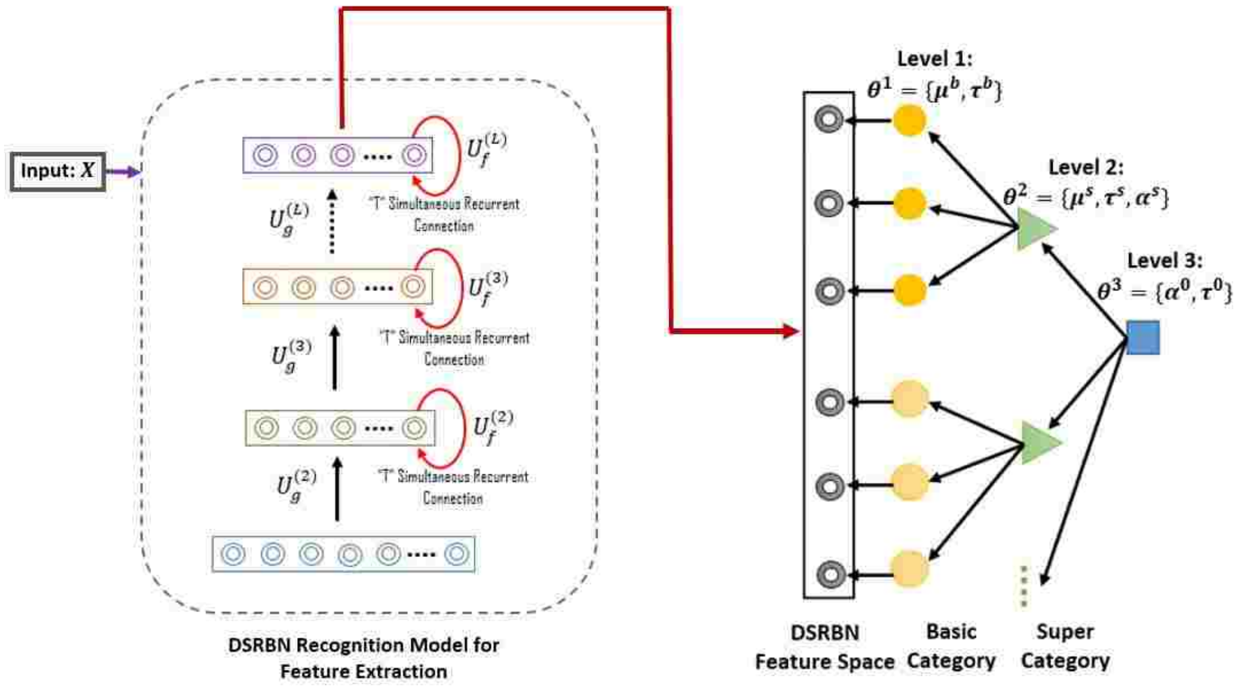
$$P(\alpha_d^s | \alpha^0) = Exp(\alpha_d^s | \alpha^0), and \tag{30}$$

$$P(\tau_d^s | \theta^0) = IG(\tau_d^s | a^0, b^0); \tag{31}$$

where, Exp() denotes an exponential distribution, $IG()$ denotes an inverse-gamma distribution, and $a^0 = b^0 = 1$. A Gamma prior $\Gamma(1,1)$ is placed over the level-3 parameters $\theta^3 = \{\alpha^0, \tau^0\}$. The HB model in (26) - (27) only allows generating fixed two-level categories, which is not generalizable. Blei *et al.* [171] proposes a nonparametric two-level nested Chinese Restaurant Prior (nCRP) over the partition z. This allows the HB model to define prior over tree structures and is generalizable to learn arbitrary hierarchies. The basic building block of the nCRP is the Chinese restaurant process, which defines a distribution on partition of integers. Consider a process where customers enter a restaurant with an unbounded number of tables. According to CRP, the n^{th} customer occupying a table k is drawn from the following distribution,

$$P(z_n|z_1, \dots, z_{n-1}) = \begin{cases} \dfrac{n^k}{n-1+\varrho}; & when\ n^k > 0 \\ \dfrac{\varphi}{n-1+\varrho}; & when\ k\ is\ new \end{cases}, \tag{32}$$

where, n^k is the number of previous customers at table k and ϱ is the concentration parameter. The nCRP(φ) extends CRP to nested sequence of partitions, one for each level of the tree. In the above two-level case, we first assign each observation n to the super-category z_n^s and then recursively assign the basic-level category z_n^b under a super-category z_n^s. This two-level nCRP technique allows us to generate a potentially unbounded number of super-categories and an unbounded number of basic-level categories under each super category.



Input: X
$U_f^{(L)}$
"T" Simultaneous Recurrent Connection
$U_g^{(L)}$
$U_f^{(3)}$
"T" Simultaneous Recurrent Connection
$U_g^{(3)}$
$U_f^{(2)}$
"T" Simultaneous Recurrent Connection
$U_g^{(2)}$
DSRBN Recognition Model for Feature Extraction
Level 1: $\theta^1 = \{\mu^b, \tau^b\}$
Level 2: $\theta^2 = \{\mu^s, \tau^s, \alpha^s\}$
Level 3: $\theta^3 = \{\alpha^0, \tau^0\}$
DSRBN Feature Space
Basic Category
Super Category

Fig. 25. The DSRBN hierarchical Bayesian (DSRBN-HB) model for one-shot learning. "U" represents the trained DSRBN recognition model weights to obtain features from the input X, $h^{(L)}$ denotes the top-level DSRBN features. The HB model operates on the feature space, $h^{(L)}$ and $\theta = \{\theta^1, \theta^2, \theta^3\}$ represents hierarchical Bayesian model parameters for different levels. The blue box represents the root of the HB tree and the green triangles represent the super-category learned by the HB model from the basic categories.

Finally, we perform the one-shot learning task using the DSRBN-HB model as shown in Fig. 25. Let us consider a new test instance x^* which belongs to a novel category b^*. First, the trained DSRBN model, $q_\gamma(h^*|x^*)$ is utilized to obtain the feature vector, h^* from the test instance, x^*. The HB model sets a prior over the feature space using level-1 parameters, θ^1. Next, using the existing tree structure z and current setting of the level-2 parameters θ^2 we infer the super-category to which the novel category belongs. Given the model parameters $\theta = \{\theta^1, \theta^2\}$ the posterior over the assignment z_b^* is computed as follows,

$$p(z_b^* | \theta, \sim z_b^*, h^*) \propto p(h^* | \theta, z_b^*) p(z_b^* | \sim z_b^*) \tag{33}$$

where, $\sim z_b^*$ denotes variables h for all observation other than z_b^*. This inferred assignment z_b^* is used to infer the posterior mean and precision terms $\{\mu^*, \tau^*\}$ for the novel category. Subsequently, the DSRBN-HB model determines the novel category b^* of the test input x^* by computing the following conditional probability,

$$p(b^* | h^*) = \frac{p(h^* | z_b^*) p(z_b^*)}{\sum_z p(h^* | z) p(z)} \tag{34}$$

where, h^* denotes the feature vector obtained using the DSRBN recognition model, $q_\gamma(h^* | x^*)$, and the prior is given by the nCRP(ϱ).

5.4 RESULTS AND DISCUSSION

This section evaluates the performance of the proposed DSRBN-HB model for solving the one-shot learning task using four widely used standard benchmark datasets such as MNIST [22], Omniglot [139], OCR letters[2] and CIFAR-100 [172], respectively. These are a few most widely used datasets in the literature for evaluating the performance of the one-shot learning models. We conduct the one-shot learning experiment using the DSRBN-HB model in two-steps. First, we pre-train the DSRBN model in an unsupervised fashion using a larger dataset which is completely different from any of the above mentioned datasets. This pre-training step ensures good generalization of the DSRBN model for feature extraction. Accordingly, we train the DSRBN-HB model using limited labeled training data to conduct the one-shot classification experiments. Further, we compare the proposed model with two state-of-the-art deep generative

[2] http://ai.stanford.edu/~btaskar/ocr/

models such as Deep Boltzman Machine (DBM), and Variational Auto-Encoder (VAE) combined with the HB model, respectively. From here onwards, we refer to the DBM and VAE model combined with the HB model as DBM-HB and VAE-HB. In the next few sections we discuss our experimental results obtained using the above mentioned datasets.

5.4.1 MNIST DATASET

Our first one-shot classification experiment is conducted using MNIST dataset. The DSRBN generative model is first pre-trained using a large dataset in an unsupervised fashion. Note that this large dataset called EMNIST [173] is different from the MNIST dataset which we use for training and testing of our proposed one-shot learning framework. The EMNIST dataset contains 814,255 characters from 62 unbalanced classes. We use a 2-layer DSRBN architecture to perform the unsupervised learning task. The DSRBN model is trained using NVIL and the AdaGrad gradient descent method [72]. The detailed hyper parameter setup for the DSRBN model is as follows: number of layers = 2, number of units at each layer = 256, $T = 15$, learning rate = 0.001, weight decay = 0.0001, momentum = 0.9, step size = 0.001, rmsdecay = 0.95 and mini batch size = 128. We use this pre-trained DSRBN model to train our DSRBN-HB model using the MNIST dataset to perform the one-shot classification task. We first investigate the ability of the DSRBN-HB model to learn from very limited training data per class. In this experiment, we randomly choose 100 example images from each class, so there are a total of 1000 training images from 10 classes to train the DSRBN-HB model. The DSRBN model extracts features from the training images, which are subsequently used as the input feature space for the HB model as shown in Fig. 25. The HB model is trained on the DSRBN feature space to construct a hierarchical Bayesian tree structure. The parameters of the HB models are obtained following the work in [77]. The tree structure obtained from the DSRBN-HB model is shown in

Fig. 26 which shows that the HB model groups similar digits such as digits 4, 7, and 9 together under the same super-category and maintains a separate super-category for the digits that are sufficiently different from other digits such as 1, 2, and 3.

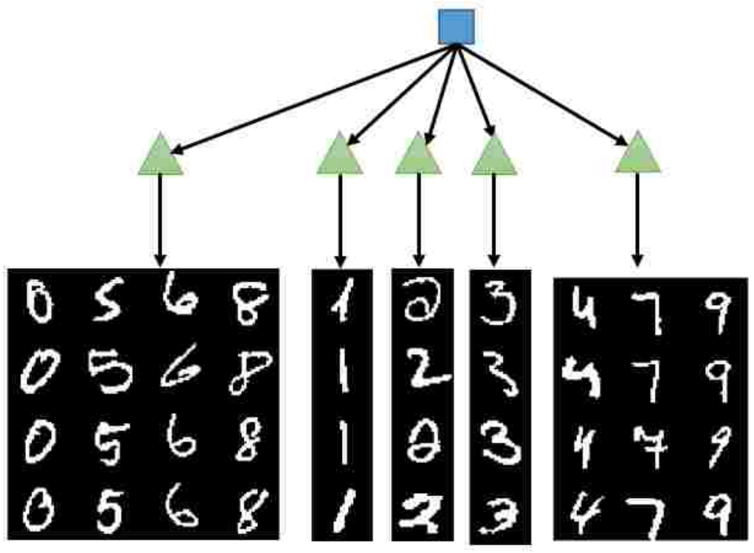


Fig. 26. A partition over 10 MNIST digits discovered by the DSRBN-HB model. The blue box represents the root of the HB tree and the green triangles represent the super-category learned by the HB model from the basic categories.

In order to evaluate one-shot classification performance, we train the DSRBN-HB model using 9 digit classes leaving 1 class as a novel category to test the model. This gives the DSRBN-HB model 900 training samples from 9 classes (100 images per class). We obtain 1000 test samples for the novel class from the test split of the MNIST dataset. We compute the area under the receiver operating characteristic (AUROC) curve for classifying 1000 test images as belonging to the novel versus all the other 9 classes. The results are averaged over 10 classes using a *leave-one-out* testing format. Next, we compare our proposed DSRBN-HB model with state-of-the-art DBM-HB and VAE-HB generative models. Both these models are pre-trained using the EMNIST dataset as mentioned above. The architecture and learning parameters of the DBM and VAE are obtained from the best performing models reported in the literature [61, 78]. In order to make a fair comparison with our proposed DSRBN-HB model, we keep the parameters of the HB model the same for the DBM-HB and VAE-HB models. Table 10 quantifies and compares the performance of the proposed DSRBN-HB model using average AUROC over all 10 classes with leave-one-out testing format.

TABLE 10

ONE-SHOT LEARNING PERFORMANCE COMPARISON OF THE PROPOSED DSRBN-HB MODEL WITH THE STATE-OF-THE-ART DBM-HB AND VAE-HB MODELS USING THE AREA UNDER THE ROC CURVE (AUROC) ON THE MNSIT DATASET. THE RESULTS ARE AVERAGED OVER ALL 10 CLASSES USING LEAVE-ONE-OUT TESTING FORMAT

Model	**Number of units in each hidden layer of the generative model**	**Average AUROC**
DSRBN-HB	256-256	**0.8705**
DBM-HB	500-1000	0.8135
VAE-HB	500-500	0.75

Table 10 shows that the proposed DSRBN-HB model achieves better one-shot classification performance compared to the DBM-HB and VAE-HB models. Learning better representation or features of the data allows the HB model to effectively discover the partition, and, hence, improves one-shot classification performance for the novel category in the test case. Moreover, the DSRBN generative model utilizes considerably fewer hidden neurons compared to the state-of-the-art DBM and VAE generative models.

5.4.2 OCR LETTERS DATASET

The second experiment is conducted using OCR letters dataset. We use the same DSRBN architecture pre-trained with EMNIST as mentioned in the experiment with the MNIST dataset in section 5.4.1. In order to use this pre-trained DSRBN model, we resize the images of the OCR letters dataset from 16×8 to 28×28 to match the image size of the EMNIST dataset. However, the architecture of the DBM and VAE models are modified to match the best performing models reported in the literature for the OCR letters dataset [127]. Therefore, rather than using the pre-trained DBM utilized for the MNIST dataset, the model is trained from scratch using the EMNIST dataset. The best performing architecture of the VAE model for the OCR letters dataset is similar to the one used for MNIST dataset, so we use the same pre-trained model as mentioned above for the MNIST case.

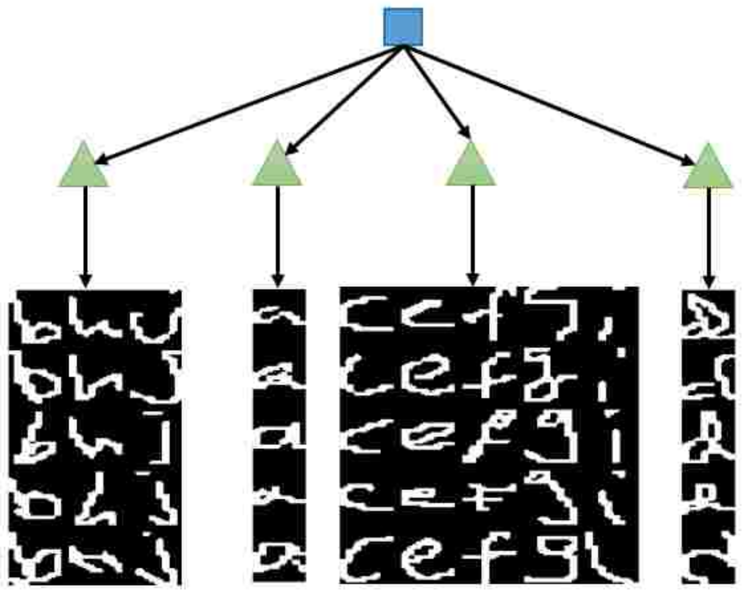


Fig. 27. A partition over some of the example basic level categories of the OCR letters dataset discovered by the DSRBN-HB model. The blue box represents the root of the HB tree and the green triangles represent the super-category learned by the HB model from the basic categories.

Subsequently, we conduct a one-shot classification experiment using the proposed DSRBN-HB model. First, we take a subset of the OCR letters dataset for training the DSRBN-HB model using limited data. We consider 100 images from each class of the OCR letters dataset to construct the training dataset. Accordingly, there are a total of 2600 images for training the DSRBN-HB model. The pre-trained DSRBN model is utilized to obtain feature space from the training images for the HB model, and subsequently, the HB model is trained over the feature space. The HB model constructs a hierarchical tree as shown in Fig. 27 which demonstrates that the model groups similar classes under the same super-category while it keeps separate super-categories for classes that are sufficiently different from other classes. The one-shot classification performance of the DSRBN-HB model is evaluated by training the model using

2500 images from 25 classes leaving 1 class as the novel category for testing. We obtain 250 images of the novel category from the test split of the OCR letters dataset. We report the AUROC for classifying 250 test images as belonging to the novel versus all the other 25 classes. The results are averaged over 26 classes using *leave-one-out* testing format. We perform the same experiment using the DBM-HB and VAE-HB models for comparison. Table 11 quantifies the one-shot classification performance of the DSRBN-HB, DBM-HB, and VAE-HB models. Table11 shows that the DSRBN-HB model shows better performance than the DBM-HB model. However, the performance of the VAE-HB model performs slightly better than the proposed DSRBN-HB model. However, the number of hidden units utilized by the DSRBN-HB model is significantly lower than that of the VAE-HB model.

TABLE 11

ONE-SHOT LEARNING PERFORMANCE COMPARISON OF THE PROPOSED DSRBN-HB MODEL WITH THE STATE-OF-THE-ART DBM-HB AND VAE-HB MODELS USING THE AREA UNDER THE ROC CURVE (AUROC) ON THE OCR LETTERS DATASET. THE RESULTS ARE AVERAGED OVER ALL 26 CLASSES USING LEAVE-ONE-OUT TESTING FORMAT

Model	**Number of units in each hidden layer of the generative model**	**Average AUROC**
DSRBN-HB	256-256	**0.8301**
DBM-HB	2000-2000	0.7926
VAE-HB	500-500	0.8511

5.4.3 OMNIGLOT DATASET

We perform the third experiment using the Omniglot dataset [139] which is one of the most popular datasets for evaluating one-shot classification performance. The architecture of the DSRBN model for the Omniglot dataset is similar to the one used for the MNIST dataset. Moreover, we utilize the same DSRBN model pre-trained using the EMNIST dataset. However, the DBM model is pre-trained using the best reported architecture in the literature for the Omniglot dataset [144]. We use the same pre-trained VAE model as mentioned in the above two experiments since the best reported architecture is unchanged. In order to evaluate the one-shot classification performance of the DSRBN-HB model, we consider 100 training images from each of the 50 classes to constitute our training data. Consequently, there are a total of 5000 training examples for training the DSRBN-HB model. These training images are first processed using the pre-trained DSRBN model for feature extraction. Subsequently, the HB model considers the features input to construct the hierarchical Bayesian tree. However, we observe that the HB model struggles to form the tree using 50 classes and soon becomes computationally impractical. Hence, we evaluate the one-shot classification performance by dividing the problem into smaller sub-problems. We perform this by considering 10 classes at a time i.e. we conduct the experiment for classes 1 to 10, 11 to 20 and so on. For each of the10 classes we consider 9 classes for training the DSRBN-HB model and leave 1 class as the novel category to test the model. Accordingly, the DSRBN-HB model is trained using 900 images from 9 classes (100 images per class). In order to perform the testing, we take 100 images from the test split of the Omniglot dataset for the novel category. We compute the average AUROC for classifying 100 test images as belonging to the novel versus all the other 9 classes. The experiment is repeated 5 times for each of the 10 sub-classes. For each of the 10 sub-classes the DSRBN-HB model is trained and tested from scratch using the above-mentioned *leave-one-out* testing format. The final result is obtained by averaging over the average AUROC obtained from the 5 experiments. The same experiment is repeated for the DBM-HB and VAE-HB model to obtain a fair comparison with our proposed DSRBN-HB model.

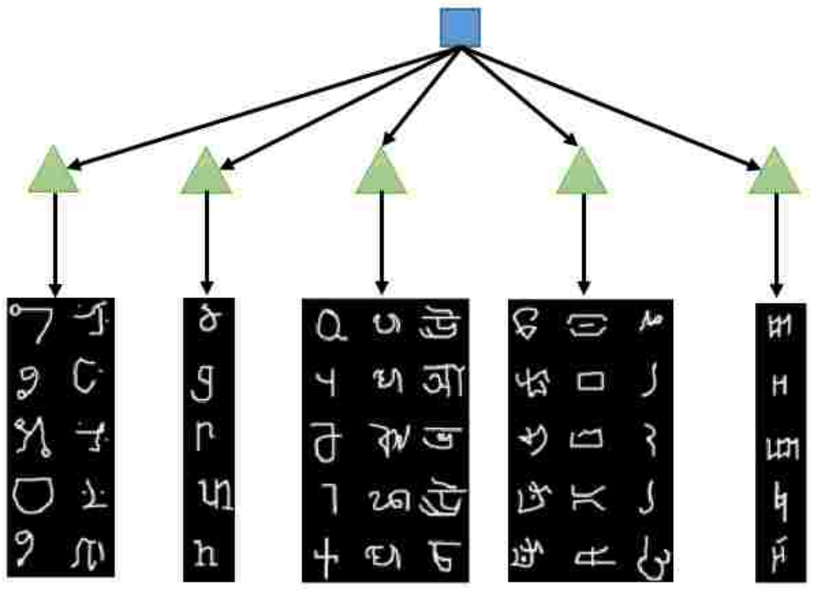


Fig. 28. Some example learned super-categories over the basic-level categories of the Omniglot dataset using the proposed DSRBN-HB model. The blue box represents the root of the HB tree and the green triangles represent the super-category learned by the HB model from the basic categories.

Fig. 28 shows a typical partition of some of the classes from the Omniglot dataset using the DSRBN-HB model. The partition of the similar classes are grouped in the super-categories, which share the same prior distribution over the classes. Table 12 quantifies the one-shot classification performance of the proposed DSRBN-HB model and compares with the DBM-HB and VAE-HB models using the average AUROC obtained from the experiments. The DSRBN-HB model achieves better performance while utilizing fewer hidden neurons compared to the state-of-the-art models.

TABLE 12

ONE-SHOT LEARNING PERFORMANCE COMPARISON OF THE PROPOSED DSRBN-HB MODEL WITH THE STATE-OF-THE-ART DBM-HB AND VAE-HB MODELS USING THE AREA UNDER THE ROC CURVE (AUROC) ON THE OMNIGLOT DATASET. THE RESULTS ARE AVERAGED OVER ALL 50 CLASSES USING LEAVE-ONE-OUT TEST FORMAT

Model	**Number of units in each hidden layer of the generative model**	**Average AUROC**
DSRBN-HB	256-256	**0.8279**
DBM-HB	2000-1000	0.7861
VAE-HB	500-500	0.8104

5.4.4 CIFAR-100 DATASET

The final experiment is conducted using the CIFAR-100 dataset. CIFAR-100 is a challenging image classification dataset which contains color images from 100 different classes of size $32 \times 32 \times 3$. The dataset has 50,000 training and 10,000 test images of a balanced number of examples per class. In this experiment we pre-train the DSRBN model using 4 million tiny color images obtained from an 80 million tiny image dataset [174]. Using the unlabeled tiny images, we perform the unsupervised learning task using a 2-layer DSRBN architecture with

NVIL and AdaGrad. The detailed hyper parameter setup for the DSRBN model is as follows: number of layers = 2, number of units at each layer = 500, $T = 17$, learning rate = 0.001, weight decay = 0.0001, momentum = 0.9, step size = 0.001, rmsdecay = 0.95, and mini batch size = 512. The architectures of the DBM and VAE models are obtained from the best performing models in the literature [159].

Subsequently, we perform the one-shot classification experiment using the CIFAR-100 dataset. Similar to the previous experiments, we consider 100 training images from each class of the CIFAR-100 dataset, which constitutes 10,000 examples for evaluating the performance of the proposed DSRBN-HB model. The pre-trained DSRBN model is used for extracting features from the training images. The HB model considers this feature space as input and constructs a hierarchical Bayesian tree. In order to assess the one-shot classification performance of the DSRBN-HB model, we take images from 99 classes for training the DSRBN-HB model and leave 1 class as the novel category for testing. However, similar to the Omniglot case, the HB model becomes computationally unreasonable to form the hierarchical Bayesian tree from 99 classes. Therefore, we divide the 100 class problem into smaller 10 class sub-problems to perform the one-shot classification task. More specifically, for each sub-problem, we consider classes 1 to 10, 11 to 20 until 91-100, respectively. The one-shot classification task for each of these 10 classes is performed by taking 900 images from 9 classes for training leaving 1 class as the novel category for testing. We take 100 images of the novel category from the test split of the CIFAR-100 dataset. We compute the average AUROC for classifying the 100 test images as belonging to the novel versus all other 9 classes. The experiment is repeated 10 times for each of the 10 sub-classes individually. The DSRBN-HB model is trained and tested from scratch for each of the 10 sub-class experiments. We obtain the final result by averaging the average

AUROC obtained from all 10 experiments. We repeat the same experiment for the DBM-HB and VAE-HB model to compare with our proposed DSRBN-HB model.

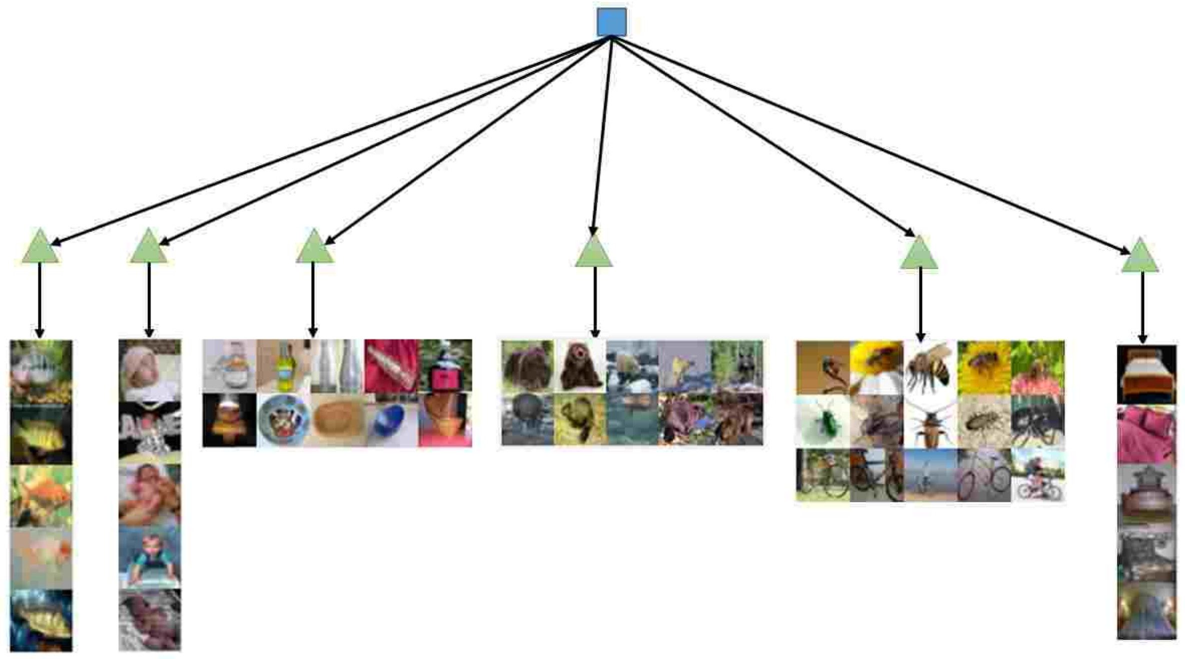


Fig. 29. DSRBN-HB model learns to group similar basic level categories under the same super-category for some of the example CIFAR-100 classes. The blue box represents the root of the HB tree and the green triangles represent the super-category learned by the HB model from the basic categories.

Fig. 29 shows a partition learned by the DSRBN-HB model over some of the example basic-level CIFAR-100 classes. Fig. 29 shows that the classes belonging to the same super-category exhibit some underlying similarity. For example, the model groups bottle and bowl classes under the same super-category. Similarly, bee and insect classes are categorized under the same super-category. However, the model incorrectly groups bicycle with bee and insect classes. During the testing case, the model will classify this as belonging to the wrong super-category. For example, in this case, when we perform one-shot classification experiment using bicycle as the novel category, the model is expected to confuse the bi-cycle class with the insect and bee classes. This is one limitation of the HB model. However, this happens on rare occasions when the HB model fails to differentiate one class as a separate super-category due to lack of sufficient dissimilarity discovered by the model. Table 13 shows a quantitative comparison of the proposed DSRBN-HB model with that of DBM-HB and VAE-HB models using average AUROC. The results demonstrate that the DSRBN-HB model achieves significantly improved one-shot classification performance compared to the DBM-HB and VAE-HB models. Additionally, the DSRBN-HB model utilizes fewer hidden units compared to the state-of-the-art models demonstrating the superior efficiency of the proposed DSRBN-HB model.

TABLE 13

ONE-SHOT LEARNING PERFORMANCE COMPARISON OF THE PROPOSED DSRBN-HB MODEL WITH THE STATE-OF-THE-ART DBM-HB AND VAE-HB MODELS USING THE AREA UNDER THE ROC CURVE (AUROC) ON THE CIFAR-100 DATASET. THE RESULTS ARE AVERAGED OVER ALL 100 CLASSES USING LEAVE-ONE-OUT TESTING FORMAT

Model	**Number of units in each hidden layer of the generative model**	**Average AUROC**
DSRBN-HB	500-500	**0.9123**
DBM-HB	5000-1000	0.7435
VAE-HB	1000-1000	0.8341

5.5 SUMMARY

This chapter proposes a novel deep recurrent Bayesian learning framework, DSRBN-HB for solving challenging classification tasks with a small amount of data. We solve the intricate one-shot classification task, which is a well-known learning challenge with small data, using the DSRBN-HB model. More specifically, the proposed DSRBN model performs efficient representation learning from unlabeled data. The learned representation is considered as input feature space for the HB model. The combined DSRBN-HB model is learned jointly to solve the one-shot learning task. The performance of the proposed DSRBN-HB model is evaluated using four widely used benchmark datasets: MNIST, Omniglot, OCR Letter and CIFAR-100. We compare our proposed method with two state-of-the-art deep generative models based one-shot learning frameworks, namely DBM-HB and VAE-HB. Our results suggest that the proposed DSRBN-HB model achieves better or comparable one-shot classification performance while utilizing significantly fewer training parameters when compared to the state-of-the-art deep learning frameworks.

CHAPTER 6

CONCLUSIONS AND FUTURE WORKS

The overall goal of this dissertation is to propose novel biologically inspired deep recurrent learning models for efficient image recognition using a small amount of data. In order to achieve this, our first goal is to design an efficient deep learning framework with simultaneous recurrency for efficiently handling the image recognition task. The novel deep recurrent learning model is expected to provide efficient control over the depth of the model, extract more complex features from the input data utilizing the recurrency and achieve superior recognition performance while reducing the number of training parameters by several orders of magnitude. Secondly, our goal is to show the generalization of the proposed deep simultaneous recurrency concept in a probabilistic generative model by solving the challenging representation learning task from unlabeled data. The proposed deep simultaneous recurrent generative model is expected to achieve superior representation learning performance while significantly reducing the number of training parameters compared to the state-of-the-art models similar to the case of deep simultaneous recurrent image recognition model. Our final goal is to extend the proposed deep simultaneous recurrent generative model by incorporating Bayesian techniques for solving the intricate problem of learning with small data. The overall outcomes of this dissertation are summarized in Table 14 and further discussed below.

TABLE 14

SUMMARY OF THE RESEARCH FINDINGS RELATED TO THE PROPOSED METHODS

Chapter	Topic	Contributions	Results	Comments
3	Deep simultaneous recurrent network (DSRN) for efficient image recognition	• Introduced unique simultaneous recurrency in a deep learning model • Developed a biologically inspired novel deep neural network model	DSRN model shows significantly improved recognition performance compared to the state-of-the-art deep learning models	• DSRN provides efficient control over the depth • DSRN utilizes less training parameters • DSRN extracts more complex features from the input data using feed-forward and recurrent weights
4	Deep Simultaneous recurrent belief network (D-SRBN) for efficient representation learning from unlabeled data	• Introduced deep simultaneous recurrency in a probabilistic generative model • Developed joint and conditional probability distribution functions for the D-SRBN generative model	D-SRBN achieves improved representation learning performance compared to the state-of-the-art deep generative models	• D-SRBN utilizes both recurrent and feed-forward information processing for learning meaningful representations • D-SRBN utilizes fewer training parameters
5	Deep Simultaneous recurrent hierarchical Bayesian model (DSRBN-HB) for solving the problem of learning with small data	• Introduced Bayesian technique with D-SRBN generative model • Developed efficient one-shot image recognition pipeline using highly expressive D-SRBN model and hierarchical Bayesian model	DSRBN-HB demonstrates improved one-shot image recognition performance compared to the state-of-the-art methods	• DSRBN-HB requires very limited training data to perform one-shot image recognition task • DSRBN-HB requires significantly reduced training parameters to achieve high accuracy

Firstly, we propose a novel deep recurrent learning model called DSRN for solving complex image recognition tasks. The simultaneous recurrency of DSRN provides efficient control over the depth of the model while keeping the number of trainable parameters constant by sharing weights between hidden recurrent layers. We introduce sparsity to the proposed DSRN architecture by utilizing dropout learning that avoids the addition of user defined external sparsity penalty terms to the loss function. Extensive mathematical and experimental analyses of sparsity and overfitting of DSRN show efficacy of DSRN. Moreover, we solve several challenging image recognition tasks such as facial expression recognition, face recognition, and character recognition to show the effectiveness of the proposed DSRN model. Our experimental results using several widely used publicly available datasets show that the proposed DSRN architecture outperforms the state-of-the-art deep neural network models such as DBN, SAE, and CNN for each of the above mentioned image recognition tasks. Moreover, our findings demonstrate that the proposed network requires a lower number of trainable parameters and thus offers enhanced efficiency with reduced computational resources than that of the state-of-the-art feed forward DNNs. We further extend the proposed DSRN recognition framework by incorporating a randomized metric learning technique (DML-eig) for object categorization. The results suggest that integration of DML-eig for the proposed DSRN model offers a considerable performance improvement over generic regression based classification such as softmax. Finally, we show that a parallel GPU based implementation of DSRN attains several orders of magnitude speedup over CPU based implementation wherein the training time is reduced from days to hours. The quick training times achieved through GPU acceleration and general processing time with fewer trainable parameters makes the proposed pipeline appropriate for real time image recognition applications.

Secondly, we show the generalization of the proposed deep simultaneous recurrency concept in a deep probabilistic generative model. Generative models are well known for their ability to learn meaningful representation from unlabeled data. Consequently, we propose a deep recurrent generative model known as, D-SRBN for efficiently solving the representation learning task. The D-SRBN model utilizes both recurrent and feed-forward information processing for learning meaningful representations from unlabeled data. Due to this unique and novel formulation of the D-SRBN architecture, we first design the joint and conditional distribution function required for the model. Subsequently, we show the inference and learning procedure of the proposed model using a well-known NVIL algorithm. Finally, we perform extensive benchmark evaluation using four widely used publicly available datasets such as MNIST, Caltech 101 Silhouettes, OCR letters and Omniglot. The effectiveness of the proposed D-SRBN model for representation learning is demonstrated by performing a comparison with three existing state-of-the-art deep generative models such as DBN, DBM, DSBN and VAE, respectively. The performance of the generative models is evaluated by computing a commonly used metric known as the average negative log-likelihood of the test dataset to determine the representation learning performance of the generative models. The results suggest that the D-SRBN model consistently achieves improved or similar performance compared to the state-of-the-art deep generative models. Moreover, experimental results demonstrate that the single layer SRBN model offers better or comparable performance compared to the benchmark deep generative models. However, the performance of the D-SRBN model is consistently better than the SRBN model. Furthermore, the results demonstrate that the D-SRBN model utilizes significantly fewer (than the DBN, DBM and VAE) or comparable (to the DSBN) number of trainable parameters while achieving a higher performance. The improved performance of the D-

SRBN with fewer training parameters may be due to the additional depth introduced by the simultaneous recurrency within each hidden layer. Hence, the D-SRBN model may be considered as an efficient alterative to the state-of-the-art deep generative models such as DBN, DBM, DSBN and VAE for learning efficient representation from unlabeled static input data.

Finally, we propose a novel DSRBN-HB network for solving difficult learning challenges using small data. Specifically, we address the intricate one-shot image classification task using DSRBN-HB. The DSRBN generative model is pre-trained using a large unlabeled dataset to obtain good feature generalization. This pre-trained model is then utilized to extract features from the dataset with a very limited number of labeled examples. Subsequently, the HB model trains on the feature space to construct the hierarchical Bayesian tree which groups basic categories into meaningful super-categories. Our experiments show that the proposed DSRBN-HB model efficiently infers appropriate super-category with correct underlying novel categories enabling improved one-shot classification using only one or very few examples. Both training and testing of DSRBN-HB model involve the *leave-one-out* method. Average AUROC is used to evaluate the proposed DSRBN-HB model using four widely used benchmark datasets: MNIST, OCR letters, Omniglot and CIFAR-100. A small subset of data is considered from each of these datasets for one-shot classification evaluation. Moreover, the performance of the proposed DSRBN-HB model is compared with two state-of-the-art generative models, DBM-HB and VAE-HB, respectively. Our results show that the proposed DSRBN-HB model achieves either improved or comparable one-shot classification performance using these benchmark datasets when compared to DBM-HB and VAE-HB models, respectively. Our experiments also show that the DSRBN-HB model achieves improved classification accuracy with far fewer training parameters in comparison to the DBM-HB and VAE-HB models.

Our future plan is to introduce a more scalable and versatile deep recurrent learning framework for small data which may be applicable in different application domains and various different data types such as static, time dependent, and multi-dimension. In order to achieve this, we first plan to address some of the limitations of our proposed deep recurrent learning models, DSRN and D-SRBN. Both DSRN and D-SRBN models process multidimensional input as a sequence of vectors. This may be a potential limiting factor for the DSRN and D-SRBN models in applications which involve large scale inputs. Hence, our future plan is to incorporate efficient windowing-based convolution or a cellular information processing technique to tackle large scale inputs. Moreover, convolution and cellular techniques inherently allow parallel information processing. Hence, the use of convolution or cellular techniques may allow our DSRN and D-SRBN models to achieve better scalability and distributed information processing capability through parallel processing. Moreover, our future plan is to propose better adaptive hyper-parameter adjustment techniques during learning of the DSRN and D-SRBN models. More specifically, we plan to adaptively adjust the important hyper-parameter, number of simultaneous recurrent steps, T of the DSRN and D-SRBN models to ensure better convergence. This may require us to make appropriate modifications to the existing gradient descent learning techniques. Additionally, in future, we plan to investigate the proposed deep recurrent hierarchical Bayesian model, DSRBN-HB for introducing a more efficient learning technique for small data. The DSRBN-HB model requires off-line pre-training of the DSRBN model for feature extraction and the HB model requires appropriate prior assignment for building the HB tree. This may limit the application of the DSRBN-HB model in a new domain which consists of un-familiar data types. Therefore, our future plan includes introduction of a better learning scheme such as online learning to avoid pre-training of the DSRBN model using external data sources. Furthermore, study to improve prior distributions for the HB model that facilitates better learning from different data types and distributions may also be pursued. Finally, the above mentioned modifications to our proposed deep recurrent learning framework may contribute a more efficient and versatile learning scheme for small data, applicable in a diverse application domain which involves different data types.

REFERENCES

[1] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527-1554, 2006.

[2] G. E. Hinton, "Learning multiple layers of representation," *Trends in cognitive sciences*, vol. 11, no. 10, pp. 428-434, 2007.

[3] C. Koch, T. Poggio, and V. Torres, "Retinal ganglion cells: a functional interpretation of dendritic morphology," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 298, no. 1090, pp. 227-263, 1982.

[4] M. Meister, L. Lagnado, and D. A. Baylor, "Concerted signaling by retinal ganglion cells," *Science*, vol. 270, no. 5239, pp. 1207-1210, 1995.

[5] M. Meister, "Multineuronal codes in retinal signaling," *Proceedings of the National Academy of sciences*, vol. 93, no. 2, pp. 609-614, 1996.

[6] P. Reinagel, D. Godwin, S. M. Sherman, and C. Koch, "Encoding of visual information by LGN bursts," *Journal of neurophysiology*, vol. 81, no. 5, pp. 2558-2569, 1999.

[7] S. M. Sherman, "Tonic and burst firing: dual modes of thalamocortical relay," *Trends in neurosciences*, vol. 24, no. 2, pp. 122-126, 2001.

[8] N. A. Lesica and G. B. Stanley, "Encoding of natural scene movies by tonic and burst spikes in the lateral geniculate nucleus," *The Journal of Neuroscience*, vol. 24, no. 47, pp. 10731-10740, 2004.

[9] R. J. Douglas and K. A. Martin, "Neuronal circuits of the neocortex," *Annu. Rev. Neurosci.*, vol. 27, pp. 419-451, 2004.

[10] D. J. Felleman and D. C. Van Essen, "Distributed hierarchical processing in the primate cerebral cortex," *Cerebral cortex*, vol. 1, no. 1, pp. 1-47, 1991.

[11] E. M. Callaway, "Feedforward, feedback and inhibitory connections in primate visual cortex," *Neural Networks*, vol. 17, no. 5, pp. 625-632, 2004.

[12] E. M. Callaway, "Local circuits in primary visual cortex of the macaque monkey," *Annual review of neuroscience*, vol. 21, no. 1, pp. 47-74, 1998.

[13] D. H. Hubel and T. N. Wiesel, "Early exploration of the visual cortex," *Neuron*, vol. 20, no. 3, pp. 401-412, 1998.

[14] M. A. Goodale and A. D. Milner, "Separate visual pathways for perception and action," *Trends in neurosciences*, vol. 15, no. 1, pp. 20-25, 1992.

[15] R. J. Douglas, C. Koch, M. Mahowald, K. Martin, and H. H. Suarez, "Recurrent excitation in neocortical circuits," *Science*, vol. 269, no. 5226, pp. 981-985, 1995.

[16] C. J. Perry and M. Fallah, "Feature integration and object representations along the dorsal stream visual hierarchy," *Frontiers in computational neuroscience*, vol. 8, 2014.

[17] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, "Robust object recognition with cortex-like mechanisms," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 3, pp. 411-426, 2007.

[18] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 4, pp. 509-522, 2002.

[19] C. P. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," in *Computer vision, 1998. sixth international conference on*, 1998, pp. 555-562: IEEE.

[20] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio, "Pedestrian detection using wavelet templates," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, 1997, pp. 193-199: IEEE.

[21] N. Kruger *et al.*, "Deep hierarchies in the primate visual cortex: What can we learn for computer vision?," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 35, no. 8, pp. 1847-1871, 2013.

[22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, no. 11, pp. 2278-2324, 1998.

[23] F. J. Huang and Y. LeCun, "Large-scale learning with svm and convolutional for generic object categorization," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, 2006, vol. 1, pp. 284-291: IEEE.

[24] J. Gehring, Y. Miao, F. Metze, and A. Waibel, "Extracting deep bottleneck features using stacked auto-encoders," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 2013, pp. 3377-3381: IEEE.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097-1105.

[26] D. Wyatte, D. J. Jilk, and R. C. O'Reilly, "Early recurrent feedback facilitates visual object recognition under challenging conditions," 2014.

[27] R. C. O'Reilly, D. Wyatte, S. Herd, B. Mingus, and D. J. Jilk, "Recurrent processing during object recognition," *Frontiers in psychology,* vol. 4, no. 124, pp. 1-14, 2013.

[28] S. M. Crouzet and M. Cauchoix, "When does the visual system need to look back?," *The Journal of Neuroscience,* vol. 31, no. 24, pp. 8706-8707, 2011.

[29] P. E. Roland *et al.*, "Cortical feedback depolarization waves: a mechanism of top-down influence on early visual areas," *Proceedings of the National Academy of Sciences,* vol. 103, no. 33, pp. 12586-12591, 2006.

[30] M. Koivisto, H. Railo, A. Revonsuo, S. Vanni, and N. Salminen-Vaparanta, "Recurrent processing in V1/V2 contributes to categorization of natural scenes," *The Journal of Neuroscience,* vol. 31, no. 7, pp. 2488-2492, 2011.

[31] E. Corthout, B. Uttl, V. Walsh, M. Hallett, and A. Cowey, "Timing of activity in early visual cortex as revealed by transcranial magnetic stimulation," *Neuroreport,* vol. 10, no. 12, pp. 2631-2634, 1999.

[32] S. Pinker, "How the mind works," *Annals of the New York Academy of Sciences,* vol. 882, no. 1, pp. 119-127, 1999.

[33] F. Xu and J. B. Tenenbaum, "Word learning as Bayesian inference," *Psychological review,* vol. 114, no. 2, p. 245, 2007.

[34] C. Kemp, A. Perfors, and J. B. Tenenbaum, "Learning overhypotheses with hierarchical Bayesian models," *Developmental science,* vol. 10, no. 3, pp. 307-321, 2007.

[35] A. Perfors and J. Tenenbaum, "Learning to learn categories," 2009: Cognitive Science Society.

[36] S. Haykin, "Neural networks, a Comprehensive Foundation. Preditice Hall, Englewood Cliffs," *New Jersey,* 2001.

[37] A. K. Jain, J. Mao, and K. Mohiuddin, "Artificial neural networks: A tutorial," *Computer,* no. 3, pp. 31-44, 1996.

[38] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain," *Psychological review,* vol. 65, no. 6, p. 386, 1958.

[39] M. Minsky and P. Seymour, "Perceptrons," 1969.

[40] C. Von der Malsburg and W. Schneider, "A neural cocktail-party processor," *Biological cybernetics*, vol. 54, no. 1, pp. 29-40, 1986.

[41] H.-J. Chang and W. J. Freeman, "Parameter optimization in models of the olfactory neural system," *Neural Networks,* vol. 9, no. 1, pp. 1-14, 1996.

[42] P. Rodriguez, J. Wiles, and J. L. Elman, "A recurrent neural network that learns to count," *Connection Science,* vol. 11, no. 1, pp. 5-40, 1999.

[43] J. L. Elman, "Finding structure in time," *Cognitive science,* vol. 14, no. 2, pp. 179-211, 1990.

[44] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014, pp. 338-342.

[45] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555,* 2014.

[46] R. Ilin, R. Kozma, and P. J. Werbos, "Beyond feedforward models trained by backpropagation: A practical training tool for a more efficient universal approximator," *Neural Networks, IEEE Transactions on,* vol. 19, no. 6, pp. 929-937, 2008.

[47] K. Anderson, K. Iftekharuddin, E. White, and P. Kim, "Binary image registration using cellular simultaneous recurrent networks," in *Computational Intelligence for Multimedia Signal and Vision Processing, 2009. CIMSVP '09. IEEE Symposium on*, 2009, pp. 61-67.

[48] X. Pang and P. Werbos, "Neural Network Design for J Function Approximation in Dynamic Programming," vol. 1, ed: arXiv:adap-org/9806001, June 1998.

[49] A. M. Schäfer and H. G. Zimmermann, "Recurrent neural networks are universal approximators," in *Artificial Neural Networks–ICANN 2006*: Springer, 2006, pp. 632-640.

[50] K. H. Pribram, *Origins: Brain and self organization*. Psychology Press, 1994.

[51] P. J. Werbos, *The roots of backpropagation: from ordered derivatives to neural networks and political forecasting*. John Wiley & Sons, 1994.

[52] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cognitive modeling,* vol. 5, p. 3, 1988.

[53] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE,* vol. 78, no. 10, pp. 1550-1560, 1990.

[54] M. Alam, L. Vidyaratne, and K. M. Iftekharuddin, "Novel hierarchical Cellular Simultaneous Recurrent neural Network for object detection," in *Neural Networks (IJCNN), 2015 International Joint Conference on*, 2015, pp. 1-7.

[55] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted Boltzmann machines for collaborative filtering," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 791-798: ACM.

[56] R. Salakhutdinov and G. Hinton, "Deep boltzmann machines," in *Artificial Intelligence and Statistics*, 2009, pp. 448-455.

[57] R. M. Neal, "Connectionist learning of belief networks," *Artificial intelligence,* vol. 56, no. 1, pp. 71-113, 1992.

[58] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," *arXiv preprint arXiv:1401.4082,* 2014.

[59] A. Mnih and K. Gregor, "Neural variational inference and learning in belief networks," *arXiv preprint arXiv:1402.0030*, 2014.

[60] Z. Gan, R. Henao, D. E. Carlson, and L. Carin, "Learning Deep Sigmoid Belief Networks with Data Augmentation," in *AISTATS*, 2015, vol. 38, pp. 268-276.

[61] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[62] D. P. Kingma and M. Welling, "Stochastic gradient VB and the variational auto-encoder," in *Second International Conference on Learning Representations, ICLR*, 2014.

[63] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, "Improved variational inference with inverse autoregressive flow," in *Advances in Neural Information Processing Systems*, 2016, pp. 4743-4751.

[64] I. Goodfellow *et al.*, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672-2680.

[65] E. L. Denton, S. Chintala, and R. Fergus, "Deep generative image models using a￼ laplacian pyramid of adversarial networks," in *Advances in neural information processing systems*, 2015, pp. 1486-1494.

[66] X. Huang, Y. Li, O. Poursaeed, J. Hopcroft, and S. Belongie, "Stacked generative adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, vol. 2, p. 4.

[67] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[68] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of Machine Learning Research*, vol. 11, no. Dec, pp. 3371-3408, 2010.

[69] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 1096-1103: ACM.

[70] J. Kiefer and J. Wolfowitz, "Stochastic estimation of the maximum of a regression function," *The Annals of Mathematical Statistics*, vol. 23, no. 3, pp. 462-466, 1952.

[71] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural networks: Tricks of the trade*: Springer, 2012, pp. 437-478.

[72] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121-2159, 2011.

[73] M. D. Zeiler, "ADADELTA: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[74] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[75] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929-1958, 2014.

[76] P. Baldi and P. Sadowski, "The dropout learning algorithm," *Artificial intelligence*, vol. 210, pp. 78-122, 2014.

[77] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[78] R. Salakhutdinov and G. E. Hinton, "Deep Boltzmann Machines," in *AISTATS*, 2009, vol. 1, p. 3.

[79] R. Salakhutdinov and I. Murray, "On the quantitative analysis of deep belief networks," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 872-879: ACM.

[80] L. K. Saul, T. Jaakkola, and M. I. Jordan, "Mean field theory for sigmoid belief networks," *Journal of artificial intelligence research*, vol. 4, no. 1, pp. 61-76, 1996.

[81] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Machine learning*, vol. 37, no. 2, pp. 183-233, 1999.

[82] E. P. Xing, M. I. Jordan, S. Russell, and A. Y. Ng, "Distance metric learning with application to clustering with side-information," in *Advances in neural information processing systems*, 2002, pp. 505-512.

[83] K. Q. Weinberger, J. Blitzer, and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," in *Advances in neural information processing systems*, 2005, pp. 1473-1480.

[84] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, "Information-theoretic metric learning," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 209-216: ACM.

[85] Y. Ying and P. Li, "Distance metric learning with eigenvalue optimization," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 1-26, 2012.

[86] C. J. Langmead, "A randomized algorithm for learning mahalanobis metrics: application to classification and regression of biological data," 2005.

[87] J. K. Kruschke and W. Vanpaemel, "Bayesian estimation in hierarchical models," *The Oxford handbook of computational and mathematical psychology*, pp. 279-299, 2015.

[88] G. M. Allenby and P. E. Rossi, "Hierarchical bayes models," *The handbook of marketing research: Uses, misuses, and future advances*, pp. 418-440, 2006.

[89] A. Sanborn, N. Chater, and K. A. Heller, "Hierarchical learning of dimensional biases in human categorization," in *Advances in neural information processing systems*, 2009, pp. 727-735.

[90] R. Salakhutdinov, J. Tenenbaum, and A. Torralba, "One-shot learning with a hierarchical nonparametric bayesian model," in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, 2012, pp. 195-206.

[91] A. M. Andrew, "Object Recognition By Computer: The Role Of Geometric Constraints by W. Eric, L. Grimson, with contributions from Tomás Lazano-Pérez and Daniel P. Huttenlocher, MIT Press, Cambridge, Mass., 1990, Hard cover, xv+ 512 pp.(£ 40.50)," *Robotica*, vol. 10, no. 05, pp. 475-475, 1992.

[92] S. Ullman and R. Basri, "Recognition by linear combinations of models," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 10, pp. 992-1006, 1991.

[93] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.

[94] R. Brunelli and T. Poggio, "Face recognition: Features versus templates," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 10, pp. 1042-1052, 1993.

[95] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision*, vol. 4, pp. 51-52, 2001.

[96] T. Poggio and S. Ullman, "Vision: are models of object recognition catching up with the brain?," *Annals of the New York Academy of Sciences*, vol. 1305, no. 1, pp. 72-82, 2013.

[97] M. A. Ranzato, J. Susskind, V. Mnih, and G. Hinton, "On deep generative models with applications to recognition," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 2011, pp. 2857-2864: IEEE.

[98] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *The Journal of Machine Learning Research*, vol. 11, pp. 3371-3408, 2010.

[99] P. Dayan and L. F. Abbott, *Theoretical neuroscience*. Cambridge, MA: MIT Press, 2001.

[100] Y. Taigman, M. Yang, M. A. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, 2014, pp. 1701-1708: IEEE.

[101] Y. Sun, D. Liang, X. Wang, and X. Tang, "Deepid3: Face recognition with very deep neural networks," *arXiv preprint arXiv:1502.00873*, 2015.

[102] Y. Tang and A.-r. Mohamed, "Multiresolution deep belief networks," in *International Conference on Artificial Intelligence and Statistics*, 2012, pp. 1203-1211.

[103] M. Kan, S. Shan, H. Chang, and X. Chen, "Stacked progressive auto-encoders (spae) for face recognition across poses," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, 2014, pp. 1883-1890: IEEE.

[104] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS workshop on deep learning and unsupervised feature learning*, 2011, vol. 2011, no. 2, p. 5: Granada, Spain.

[105] C. J. Langmead, "A randomized algorithm for learning mahalanobis metrics: application to classification and regression of biological data," in *APBC*, 2006, pp. 217-226: World Scientific.

[106] T. Kanade, J. F. Cohn, and Y. Tian, "Comprehensive database for facial expression analysis," in *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, 2000, pp. 46-53: IEEE.

[107] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, 2010, pp. 94-101: IEEE.

[108] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137-154, 2004.

[109] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *International conference on artificial intelligence and statistics*, 2010, pp. 249-256.

[110] P. Liu, S. Han, Z. Meng, and Y. Tong, "Facial expression recognition via a boosted deep belief network," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, 2014, pp. 1805-1812: IEEE.

[111] M. Liu, S. Li, S. Shan, R. Wang, and X. Chen, "Deeply Learning Deformable Facial Action Parts Model for Dynamic Expression Analysis," in *Computer Vision--ACCV 2014*: Springer, 2014, pp. 143-157.

[112] M. Liu, S. Li, S. Shan, and X. Chen, "Au-aware deep networks for facial expression recognition," in *Automatic Face and Gesture Recognition (FG), 2013 10th IEEE International Conference and Workshops on*, 2013, pp. 1-6: IEEE.
[113] T. E. de Campos, B. R. Babu, and M. Varma, "Character Recognition in Natural Images."
[114] M. Alam, L. S. Vidyaratne, and K. M. Iftekharuddin, "Sparse Simultaneous Recurrent Deep Learning for Robust Facial Expression Recognition," *IEEE Transactions on Neural Networks and Learning Systems,* vol. PP, no. 99, pp. 1-12, 2018.
[115] M. Alam, L. Vidyaratne, and K. M. Iftekharuddin, "Efficient feature extraction with simultaneous recurrent network for metric learning," in *2016 International Joint Conference on Neural Networks (IJCNN)*, 2016, pp. 1195-1201.
[116] M. Alam, L. Vidyaratne, T. Wash, and K. Iftekharuddin, "Deep SRN for robust object recognition: A case study with NAO humanoid robot," in *SoutheastCon, 2016*, 2016, pp. 1-7: IEEE.
[117] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771-1800, 2002.
[118] G. W. Taylor, G. E. Hinton, and S. T. Roweis, "Modeling human motion using binary latent variables," *Advances in neural information processing systems,* vol. 19, p. 1345, 2007.
[119] I. Sutskever and G. E. Hinton, "Learning Multilevel Distributed Representations for High-Dimensional Sequences," in *AISTATS*, 2007, vol. 2, pp. 548-555.
[120] R. Salakhutdinov, "Learning deep generative models," University of Toronto, 2009.
[121] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science,* vol. 313, no. 5786, pp. 504-507, 2006.
[122] W. Liu, H. Liu, D. Tao, Y. Wang, and K. Lu, "Multiview Hessian regularized logistic regression for action recognition," *Signal Processing,* vol. 110, pp. 101-107, 2015/05/01/ 2015.
[123] W. Liu, Z.-J. Zha, Y. Wang, K. Lu, and D. Tao, "$ p $-Laplacian regularized sparse coding for human activity recognition," *IEEE Transactions on Industrial Electronics,* vol. 63, no. 8, pp. 5120-5129, 2016.
[124] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009, pp. 609-616: ACM.
[125] R. Sarikaya, G. E. Hinton, and A. Deoras, "Application of deep belief networks for natural language understanding," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP),* vol. 22, no. 4, pp. 778-784, 2014.
[126] A.-r. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 20, no. 1, pp. 14-22, 2012.
[127] R. Salakhutdinov and H. Larochelle, "Efficient Learning of Deep Boltzmann Machines," in *AISTATs*, 2010, vol. 9, pp. 693-700.
[128] N. Srivastava and R. R. Salakhutdinov, "Multimodal learning with deep boltzmann machines," in *Advances in neural information processing systems*, 2012, pp. 2222-2230.
[129] Z. Song, R. Henao, D. Carlson, and L. Carin, "Learning sigmoid belief networks via Monte Carlo expectation maximization," in *Artificial Intelligence and Statistics*, 2016, pp. 1347-1355.

[130] Z. Gan, C. Li, R. Henao, D. E. Carlson, and L. Carin, "Deep temporal sigmoid belief networks for sequence modeling," in *Advances in Neural Information Processing Systems*, 2015, pp. 2467-2475.
[131] J. Walker, C. Doersch, A. Gupta, and M. Hebert, "An uncertain future: Forecasting from static images using variational autoencoders," in *European Conference on Computer Vision*, 2016, pp. 835-851: Springer.
[132] A. Deshpande, J. Lu, M.-C. Yeh, and D. A. Forsyth, "Learning diverse image colorization," *CoRR, abs/1612.01958,* vol. 1, 2016.
[133] X. Yan, J. Yang, K. Sohn, and H. Lee, "Attribute2image: Conditional image generation from visual attributes," in *European Conference on Computer Vision*, 2016, pp. 776-791: Springer.
[134] C. Ledig *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," *arXiv preprint*, 2016.
[135] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," *arXiv preprint arXiv:1605.05396,* 2016.
[136] R. A. Yeh, C. Chen, T. Y. Lim, A. G. Schwing, M. Hasegawa-Johnson, and M. N. Do, "Semantic image inpainting with deep generative models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5485-5493.
[137] M. Koivisto, H. Railo, A. Revonsuo, S. Vanni, and N. Salminen-Vaparanta, "Recurrent processing in V1/V2 contributes to categorization of natural scenes," *Journal of Neuroscience,* vol. 31, no. 7, pp. 2488-2492, 2011.
[138] B. M. Marlin, K. Swersky, B. Chen, and N. de Freitas, "Inductive Principles for Restricted Boltzmann Machine Learning," in *AISTATS*, 2010, pp. 509-516.
[139] B. M. Lake, R. R. Salakhutdinov, and J. Tenenbaum, "One-shot learning by inverting a compositional causal process," in *Advances in neural information processing systems*, 2013, pp. 2526-2534.
[140] Y. Bengio, L. Yao, and K. Cho, "Bounding the test log-likelihood of generative models," *arXiv preprint arXiv:1311.6184,* 2013.
[141] I. Murray and R. R. Salakhutdinov, "Evaluating probabilities under high-dimensional latent variable models," in *Advances in neural information processing systems*, 2009, pp. 1137-1144.
[142] K. Cho, T. Raiko, and A. Ilin, "Enhanced gradient for training restricted Boltzmann machines," *Neural computation,* vol. 25, no. 3, pp. 805-831, 2013.
[143] Y. Burda, R. Grosse, and R. Salakhutdinov, "Importance weighted autoencoders," *arXiv preprint arXiv:1509.00519,* 2015.
[144] Y. Burda, R. B. Grosse, and R. Salakhutdinov, "Accurate and conservative estimates of MRF log-likelihood using reverse annealing," in *AISTATS*, 2015.
[145] M. Alam, L. Vidyaratne, and K. Iftekharuddin, "Novel deep generative simultaneous recurrent model for efficient representation learning," *Neural Networks,* 2018.
[146] L. B. Smith, S. S. Jones, B. Landau, L. Gershkoff-Stowe, and L. Samuelson, "Object name learning provides on-the-job training for attention," *Psychological Science,* vol. 13, no. 1, pp. 13-19, 2002.
[147] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "One-shot learning with memory-augmented neural networks," *arXiv preprint arXiv:1605.06065,* 2016.
[148] O. Vinyals, C. Blundell, T. Lillicrap, and D. Wierstra, "Matching networks for one shot learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 3630-3638.

[149] Y. Duan *et al.*, "One-Shot Imitation Learning," *arXiv preprint arXiv:1703.07326*, 2017.

[150] J. Burgess, J. R. Lloyd, and Z. Ghahramani, "One-Shot Learning in Discriminative Neural Networks," *arXiv preprint arXiv:1707.05562*, 2017.

[151] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 4, pp. 594-611, 2006.

[152] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML Deep Learning Workshop*, 2015, vol. 2.

[153] S. Blaes and T. Burwick, "Few-shot learning in deep networks through global prototyping," *Neural Networks*, vol. 94, pp. 159-172, 2017.

[154] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical Networks for Few-shot Learning," *arXiv preprint arXiv:1703.05175*, 2017.

[155] Z. Li, F. Zhou, F. Chen, and H. Li, "Meta-SGD: Learning to Learn Quickly for Few Shot Learning," *arXiv preprint arXiv:1707.09835*, 2017.

[156] B. Hariharan and R. Girshick, "Low-shot Visual Recognition by Shrinking and Hallucinating Features," *arXiv preprint arXiv:1606.02819*, 2016.

[157] X. Wang, C. Chen, Y. Cheng, and Z. J. Wang, "Zero-shot image classification based on deep feature extraction," *IEEE Transactions on Cognitive and Developmental Systems*, 2016.

[158] A. Anderson, K. Shaffer, A. Yankov, C. D. Corley, and N. O. Hodas, "Beyond fine tuning: A modular approach to learning on small data," *arXiv preprint arXiv:1611.01714*, 2016.

[159] R. Salakhutdinov, J. B. Tenenbaum, and A. Torralba, "Learning with hierarchical-deep models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1958-1971, 2013.

[160] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332-1338, 2015.

[161] S. Bengio, Y. Bengio, J. Cloutier, and J. Gecsei, "On the optimization of a synaptic learning rule," in *Preprints Conf. Optimality in Artificial and Biological Neural Networks*, 1992, pp. 6-8: Univ. of Texas.

[162] J. Schmidhuber, "Learning to control fast-weight memories: An alternative to dynamic recurrent networks," *Neural Computation*, vol. 4, no. 1, pp. 131-139, 1992.

[163] M. Andrychowicz *et al.*, "Learning to learn by gradient descent by gradient descent," in *Advances in Neural Information Processing Systems*, 2016, pp. 3981-3989.

[164] K. Li and J. Malik, "Learning to optimize," *arXiv preprint arXiv:1606.01885*, 2016.

[165] D. Ha, A. Dai, and Q. V. Le, "Hypernetworks," *arXiv preprint arXiv:1609.09106*, 2016.

[166] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," 2016.

[167] C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," *arXiv preprint arXiv:1703.03400*, 2017.

[168] L. Fe-Fei, "A Bayesian approach to unsupervised one-shot learning of object categories," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, 2003, pp. 1134-1141: IEEE.

[169] A. Maas and C. Kemp, "One-shot learning with bayesian networks," 2009: Cognitive Science Society.

[170] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum, "One shot learning of simple visual concepts," in *Proceedings of the Cognitive Science Society,* 2011, vol. 33, no. 33.

[171] D. M. Blei, T. L. Griffiths, and M. I. Jordan, "The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies," *Journal of the ACM (JACM),* vol. 57, no. 2, p. 7, 2010.

[172] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Citeseer2009.

[173] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "EMNIST: an extension of MNIST to handwritten letters," *arXiv preprint arXiv:1702.05373,* 2017.

[174] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *IEEE transactions on pattern analysis and machine intelligence,* vol. 30, no. 11, pp. 1958-1970, 2008.

APPENDICES

APPENDIX A. AVERAGING AND CONVERGENCE PROPERTY OF DSRN

This section shows that the proposed DSRN satisfies several desirable properties that are common for models with dropout learning.

1) AVERAGING PROPERTY OF DSRN WITH DROPOUT LEARNING

The analysis of [76] shows that when dropout is applied to a neural network with non-linear activation functions, the expected outcome of the dropout network can be observed by finding the normalized weighted geometric mean (NWGM) of the actual output of the network. Therefore, in our case the expected outcome of the DSRN with dropout is obtained by computing NWGM of (15) and can be written as,

$$E(y_{t+1}^{h}) \approx NWGM\left(y_{t+1}^{h}\right) \tag{A1}$$

$$NWGM\left(y_{t+1}^{h}\right) = \sigma\left[E\left(S_{t+1}^{h}\right)\right] \tag{A2}$$

where, $E\left(S_{t+1}^{h}\right) = \sum_{l<h}\left[W_e^h . E\left(y_T^l\right) + R^h . E(y_t^h)\right]p^l$. The fundamental assumption for (A2) is the independence of the random variables from the activity of the units of the DSRN.

2) CONVERGENCE OF DSRN WITH DROPOUT LEARNING

Following [76], we show the convergence of DSRN with dropout learning. This is performed by showing that the expected gradient of the dropout network is equal to the gradient of the ensemble network to a certain extent with the addition of a complex and adaptive weight decay term. Here, an ensemble network means the collection of all possible sub-networks obtained from the dropout network. Let us consider the following quadratic error function,

$$E = \frac{1}{2}(tar - y_T^F)^2 \tag{A3}$$

where, tar indicates target value for supervised training. From here onwards, we use E_D and E_{ENS} to denote the error function associated with the dropout network and ensemble network respectively.

Now, the output of the dropout DSRN is given by (15). To avoid any confusion the output of ensemble S-DSRN is described with different notations I and V as follows,

$$V_{t+1}^{h} = \sigma\left(I_{t+1}^{h}\right) = \sigma\left(\sum_{l<h}\left[W_e^{h}.V_T^{l} + R^{h}.V_t^{h}\right]p^{l}\right)$$

$$with\ V_T^{0} = x\ and\ V_t^{0} = 0 \qquad (A4)$$

Since the core of DSRN is a recurrent network, we apply BPTT rather than regular backpropagation based gradient descent technique. The gradients of the feed-forward W_e^{h} and recurrent R^{h} weights are obtained by applying the chain rule on the two weights separately. The gradient of the ensemble network in terms recurrent weights R^{h} with $\sigma(b) = \frac{1}{1+e^{-b}}$ is computed as follows,

$$\frac{\partial E_{ENS}}{\partial R^{h}} = \frac{\partial E_{ENS}}{\partial I_t^{h}}\frac{\partial I_t^{h}}{\partial R^{h}}, \qquad (A5)$$

where, the backpropagated error $\frac{\partial E_{ENS}}{\partial I_t^{h}}$ through the recurrent weights is obtained recursively as follows,

$$\frac{\partial E_{ENS}}{\partial I_t^{h}} = \sum_{l>h}\sum_{i>t}\frac{\partial E_{ENS}}{\partial I_i^{l}}R^{h}p^{h}\sigma'\left(I_i^{l}\right), \qquad (A6)$$

where, σ' denotes the derivative of the activated units. The error deltas at the top layer $h = F$ is computed by,

$$\frac{\partial E_{ENS}}{\partial I_t^{F}} = -(tar - V_T^{F}). \qquad (A7)$$

The second term of (A5) is given as,

$$\frac{\partial I_t^{h}}{\partial R^{h}} = p^{l}V_t^{h}, \qquad (A8)$$

Similarly, error for the dropout DSRN is given as,

$$\frac{\partial E_D}{\partial R^h} = \frac{\partial E_D}{\partial S_t^h}\frac{\partial S_t^h}{\partial R^h}, \qquad (A9)$$

with,

$$\frac{\partial E_D}{\partial S_t^h} = \sum_{l>h}\sum_{i>t}\frac{\partial E_D}{\partial S_i^l} R^h \delta^h \sigma'\left(S_i^l\right), \qquad (A10)$$

$$\frac{\partial E_D}{\partial S_t^F} = -(tar - y_T^F) \quad and \qquad (A11)$$

$$\frac{\partial S_t^h}{\partial R^h} = \delta^l y_t^h \qquad (A12)$$

The expected gradient error for dropout S-DSRN is computed as follows,

$$\begin{aligned} E\left(\frac{\partial E_D}{\partial R^h}\right) &= E\left(\frac{\partial E_D}{\partial S_t^h}\middle|\delta^h = 1\right) p^l E\left(y_t^h\right) \\ &\approx E\left(\frac{\partial E_D}{\partial S_t^h}\middle|\delta^h = 1\right) p^l V_t^h \\ &= E\left(\frac{\partial E_D}{\partial S_t^h}\middle|\delta^h = 1\right) \frac{\partial I_t^h}{\partial R^h} \quad [from\ (A8)]. \qquad (A13) \end{aligned}$$

From (A13), we can see that the second term is actually same as the second term of (A5) which provides the error gradient of the ensemble DSRN in terms of recurrent weights, R^h. To show the equality between the first term of (A5) and (A13) we take the expectation of (A10). The resulting term is similar to the right-hand side term of (A6) which is the first term of (A5). Therefore, from (A5) and (A13) it can be concluded that the gradient of the error for ensemble DSRN ($\frac{\partial E_{ENS}}{\partial R^h}$) and that for the dropout DSRN ($E\left(\frac{\partial E_D}{\partial R^h}\right)$) are related and similar (in terms of R^h). This in turn suggests that the convergence of DSRN with dropout learning is retained. Similar analysis can be performed to show the convergence of DSRN in terms of feed-forward weights, W_e^h.

APPENDIX B. LEARNING AND INFERENCE DETAILS OF D-SRBN

This section shows the learning and inference details of the proposed D-SRBN model. We first show the detailed equations for the generative and recognition model of the D-SRBN and equations to compute the variational lower bound.

For $t = 1, \dots, T$ and $l = 2, \dots, L$ we consider $h^{(1)} \in \{0,1\}^{J_1}$ and $h_t^{(l)} \in \{0,1\}^{J_l}$. The parameters θ of the model are defined as $W_r^{(l)} \in \mathbb{R}^{\mathrm{J_l \times J_{(l-1)}}}$, $W_f^{(l)} \in \mathbb{R}^{\mathrm{J_l \times J_l}}$, $W_g^{(l)} \in \mathbb{R}^{\mathrm{J_{(l-1)} \times J_l}}$ and $b^{(l)} \in \mathbb{R}^{\mathrm{J_l}}$. The generative model of the D-SRBN is expressed as,

$$p\left(h_{j_l t}^{(l)} = 1\right) = \sigma\left({w_{f j_l}^{(l)}}^T h_{t-1}^{(l)} + {w_{g j_l}^{(l)}}^T h_{\mathrm{T}}^{(l-1)} + {b_{j_l}}^{(l)}\right),$$
$$for\ l = L\ and\ t = 1\ to\ T; \qquad \text{(B1)}$$

$$p\left(h_{j_l t}^{(l)} = 1\right) = \sigma\left({w_{r j_l}^{(l)}}^T h_t^{(l+1)} + {w_{f j_l}^{(l)}}^T h_{t-1}^{(l)} + {w_{g j_l}^{(l)}}^T h_{\mathrm{T}}^{(l-1)} + {b_{j_l}}^{(l)}\right),$$
$$for\ 1 < l < L\ and\ t = 1\ to\ T\ with\ h_T^{(l=1)} = v; \qquad \text{(B2)}$$

and,

$$p\left(h_{j_l}^{(l)} = 1\right) = \sigma\left({w_{r j_{(l+1)}}^{(l+1)}}^T h_t^{(l+1)} + {b_{j_l}}^{(l)}\right), \qquad \text{(B3)}$$
$$for\ l = 1\ and\ t = 1\ to\ T;$$

The recognition model is specified as,

$$q(h_{j_l t}^{(l)} = 1) = \sigma\left({u_{g j_l}^{(l)}}^T h_T^{(l-1)} + {u_{f j_l}^{(l)}}^T h_{t-1}^{(l)} + {c_{j_l}}^{(l)}\right),$$
$$for\ l = 2, \dots, L\ and\ t = 1, \dots, T\ with\ h_T^{(l=1)} = v; \qquad \text{(B4)}$$

where, the recognition model parameters φ are specified as $U_f^{(l)} \in \mathbb{R}^{\mathrm{J_l \times J_l}}$, $U_g^{(l)} \in \mathbb{R}^{\mathrm{J_{(l-1)} \times J_l}}$ and $c^{(l)} \in \mathbb{R}^{\mathrm{J_l}}$. Additionally, $h_0^{(l)}$ needed for $p\left(h_1^{(l)}\right)$, $p\left(h_1^{(l)} \middle| h_1^{(l+1)}\right)$ and $q\left(\mathrm{h}_1^{(\mathrm{l})} \middle| \mathrm{h}_T^{(\mathrm{l-1})}\right)$ are defined as zero vectors.

In order to utilize the NVIL algorithm [12] we calculate the variational lower bound as follows,

$$£ = \sum_{t=1}^{T} \mathbb{E}_{q_\varphi\left(H^{(L)}\middle|h^{(1)}\right)}[e_t]; \tag{B5}$$

where, e_t is expressed as,

$$e_t = \sum_{l=2}^{L}[\sum_{j_l=1}^{J_l}(\Psi_{j_l t}{}^{(l)} h_t^{(l)} - \log\left(1 + \exp\left(\Psi_{j_l t}{}^{(l)}\right)\right)] + \sum_{j_l=1\,\&\,l=1}^{J_l}(\Psi_{j_l t}{}^{(l)} h^{(l)} - \\ \log\left(1 + \exp\left(\Psi_{j_l t}{}^{(l)}\right)\right) - \sum_{l=2}^{L}[\sum_{j_l=1}^{J_l}(\beta_{j_l t}{}^{(l)} h_t^{(l)} - \log(1 + \exp(\beta_{j_l t}{}^{(l)}))]; \tag{B6}$$

where,

$$\Psi_{j_l t}{}^{(l)} = w_{f j_l}^{(l)^T} h_{t-1}^{(l)} + w_{g j_l}^{(l)^T} h_{\mathrm{T}}^{(l-1)} + b_{j_l}{}^{(l)}, \\ for\ l = L\ and\ t = 1, \dots, T; \tag{B7}$$

$$\Psi_{j_l t}{}^{(l)} = w_{r j_l}^{(l)^T} h_t^{(l+1)} + w_{f j_l}^{(l)^T} h_{t-1}^{(l)} + w_{g j_l}^{(l)^T} h_{\mathrm{T}}^{(l-1)} + b_{j_l}{}^{(l)}, \\ for\ 1 < l < L\ and\ t = 1, \dots, T\ with\ h_T^{(l=1)} = v; \tag{B8}$$

$$\Psi_{j_l t} h^{(l)} = w_{r j_{(l+1)}}^{(l+1)^{\ T}} h_t^{(l+1)} + b_{j_l}{}^{(l)}, \\ for\ l = 1\ and\ t = 1, \dots, T; \tag{B9}$$

and,

$$\beta_{j_l t}{}^{(l)} = u_{g j_l}^{(l)^T} h_{\mathrm{T}}^{(l-1)} + u_{f j_l}^{(l)^T} h_{t-1}^{(l)} + c_{j_l}{}^{(l)}, \\ for\ l = 2, \dots, L\ and\ t = 1, \dots, T\ with\ h_T^{(l=1)} = v. \tag{B10}$$

Subsequently, for parameter updates the gradients of the D-SRBN model are computed using chain rule of backpropagation through time (BPTT).

VITA

Mahbubul Alam
Department of Electrical and Computer Engineering
Old Dominion University
Norfolk, VA 23529

EDUCATION

- Old Dominion University, Norfolk, Virginia, USA GPA: 4.0

  Ph.D. Candidate, Electrical and Computer Engineering, August 2012 - Present
- Jahangirnagar University, Dhaka, Bangladesh GPA: 3.96

  M.S., Computer Science and Engineering, December 2011
- Jahangirnagar University, Dhaka, Bangladesh GPA: 3.95

  B.S., Computer Science and Engineering, November 2008

ACADEMIC EXPERIENCE

- Old Dominion University, Norfolk, Virginia, USA
  - Graduate Research Assistant, August 2012 – Present
  - Teaching Assistant, January 2013 - December 2015
- Jahangirnagar University, Dhaka, Bangladesh

  Lecturer, Computer Science and Engineering, July 2009 - August 2012

HONORS AND AWARDS

- Best undergrad student of the year Gold Medal, Jahangirnagar University, 2010.
- Highest grade in B.S., M.S. and Ph.D.
- Ranked No. 1 in the world in BRATS 2017: Brain Tumor Segmentation Challenge (Survival prediction category).

- 8th Place (among 100 participating teams) in ACM ICPC Dhaka Regional programming challenge 2007.
- Top 10 finalists in the Old Dominion University inaugural 3-Minute Thesis (3MT) presentation competition, 2017.

PATENT

- Image Recognition using Deep Simultaneous Recurrent Learning (Pending, filing date: January 2018).

GRANT PROPOSAL

- Wrote a few sections of the NSF grant proposal “EAGER: Real-Time: Real-Time Learning and Prediction of Road Inundations using Image Data for Recurrent Nuisance Flooding” (P.I. - Khan M. Iftekharuddin).
    - Total amount: $300k
    - Proposed start date: January 15, 2019

PROFESSIONAL SERVICES

- Reviewing Activity
    - Serving regularly as a reviewer for the IEEE Transactions on Neural Networks and Learning Systems.
    - Served as a reviewer for the IEEE Transactions on Emerging Topics in Computational Intelligence.
    - Served as a reviewer for the IEEE Journal of Biomedical and Health Informatics.
    - Served as a reviewer for the IEEE SoutheastCon 2016 conference.

- Mentoring
    - Mentored several undergraduate research students from the department of Electrical and Computer Engineering and Computer Science at Old Dominion University.
    - Mentored visiting undergraduate research students from different U.S. Universities awarded by the NSF Research Experiences for Undergraduates (REU) program.

LIST OF PUBLICATIONS

JOURNALS

- M. Alam, L. Vidyaratne and K. M. Iftekharuddin, "Sparse Simultaneous Recurrent Deep Learning for Robust Facial Expression Recognition," IEEE Trans. on Neural Networks and Learning Systems (Impact Factor: 7.982), vol. 29, No. 10, pp. 4905-4916, October 2018.
- M. Alam, L. Vidyaratne and K. M. Iftekharuddin, "Novel Deep Generative Simultaneous Recurrent Model for Efficient Representation Learning," Neural Networks, Elsevier (Impact Factor: 7.197), vol. 107, pp. 12-22, November 2018.
- M. Alam, L. Vidyaratne and K. M. Iftekharuddin, "One-shot Image Classification using Deep Recurrent Generative Hierarchical Bayesian Model," IEEE Trans. on Neural Networks and Learning Systems (Impact Factor: 7.982), Under Review.
- M. Alam, M. D. Samad, L. Vidyaratne, A. Glandon and K. M. Iftekharuddin, "Computational Intelligence in Speech and Vision Systems A Survey on Emerging Research Trends," IEEE Transaction on Emerging Topics in Computational Intelligence, Under Review.

CONFERNCES

- M. Alam, L. Vidyaratne and K. M. Iftekharuddin, "Efficient Learning of Data Distribution using Simultaneous Recurrent Belief Network," 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, 2018, pp. 1-6.
- M. Alam, L. Vidyaratne, and K. M. Iftekharuddin, "Efficient Feature Extraction with Simultaneous Recurrent Network for Metric Learning," In Neural Networks (IJCNN), International Joint Conference on, IEEE, pp. 1195-1201, 2016.
- M. Alam, L. Vidyaratne, T. Wash, and K. M. Iftekharuddin, "Deep SRN for Robust Object Recognition: A case study with NAO humanoid robot," In SoutheastCon, IEEE, pp. 1-7, 2016.
- M. Alam, L. Vidyaratne, and K. M. Iftekharuddin, "Novel Hierarchical Cellular Simultaneous Recurrent Neural Network for Object Detection," In Neural Networks (IJCNN), International Joint Conference on, IEEE, pp. 1-7, 2015.
- M. Alam, L. Vidyaratne, and K. M. Iftekharuddin, "Cellular Recurrent Deep Network for Image Registration," In SPIE Optical Engineering+ Applications, International Society for Optics and Photonics, pp. 95981B-95981B, 2015.
- L. Vidyaratne, M. Alam, Z. Shboul and K. M. Iftekharuddin, "Deep Learning and Texture Based Semantic Label Fusion for Brain Tumor Segmentation," Accepted, In SPIE Medical Imaging, Houston, Texas, February 10-15, 2018.
- M. Z. Alom, M. Alam, T. M. Taha, and K. M. Iftekharuddin, "Object Recognition using Cellular Simultaneous Recurrent Networks and Convolutional Neural Network," In Neural Networks (IJCNN), International Joint Conference on, IEEE, pp. 2873-2880, 2017.
- L. Vidyaratne, M. Alam, and K. M. Iftekharuddin, "Constrained versus Unconstrained

Learning in Generalized Recurrent Network for Image Processing," In Neural Networks (IJCNN), International Joint Conference on, IEEE, pp. 3310-3317, 2017.

- C. V. Dolph, M. Alam, Z. Shboul, M. D. Samad, and K. M. Iftekharuddin, "Deep Learning of Texture and Structural Features for Multiclass Alzheimer's Disease Classification," In Neural Networks (IJCNN), International Joint Conference on, IEEE, pp. 2259-2266, 2017.
- K. M. Iftekharuddin, M. Alam, and L. Vidyaratne, "Contemporary Deep Recurrent Learning for Recognition," In Pattern Recognition and Tracking XXVIII, International Society for Optics and Photonics, vol. 10203, p. 1020302, 2017.
- S. Bakas et. al, "Pre-conference Proceedings of the International Multimodal Brain Tumor Segmentation (BraTS) Challenge 2017," Pre-conference, Publisher: BraTS 2017.
- Z. Shboul, L. Vidyaratne, M. Alam, and K. M. Iftekharuddin, "Glioblastoma and Survival Prediction," Brain Lesion (BrainLes) workshop, a satellite event of the International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI), Springer, Cham, pp. 358-368, February 17 2018.
- A. Glandon, M. Ullah, L. Vidyaratne, M. Alam, C. Xin and K. M. Iftekharuddin, "Prediction of Spatial Spectrum in Cognitive Radio using Cellular Simultaneous Recurrent Networks," In Neural Networks (IJCNN), International Joint Conference on, IEEE, Accepted, Rio, Brazil, Jul 8 - Jul 13, 2018.
- D. Bussey, A. Glandon, L. Vidyaratne, M. Alam and K. M. Iftekharuddin, "Convolutional Neural Network Transfer Learning for Robust Face Recognition in NAO Humanoid Robot," IEEE Symposium Series on Computational Intelligence (IEEE SSCI), Honolulu, Hawaii, pp.1-7, 2017.
- L. Vidyaratne, A. Glandon, M. Alam, and K. M. Iftekharuddin. "Deep Recurrent Neural

Network for Seizure Detection," In Neural Networks (IJCNN), International Joint Conference on, IEEE, pp. 1202-1207, 2016.

- L. Vidyaratne, M. Alam, J. K. Anderson, and K. M. Iftekharuddin, "Improved Training of Cellular SRN using Unscented Kaiman Filtering for ADP," In Neural Networks (IJCNN), International Joint Conference on, IEEE, pp. 993-1000, 2014.
- J. B. Flora, M. Alam, and K. M. Iftekharuddin. "Improvements to Vehicular Traffic Segmentation and Classification for Emissions Estimation using Networked Traffic Surveillance Cameras." Proceedings of SPIE, vol. 9216, p. 92160K, 2014.