CLASSIFYING IMAGES USING DEEP LEARNING ARCHITECTURES

A thesis submitted in partial fulfillment
of the requirements for the degree of

INTEGRATED MASTERS OF TECHNOLOGY

in

Mathematics and Computing

by

RAGHAV GOYAL

Entry No. 2010MT50612

Under the guidance of

PROF. B. CHANDRA

Department of Mathematics
Indian Institute of Technology Delhi.
New Delhi - 110016, India

June 2015.

Certificate

This is to certify that the thesis titled **CLASSIFYING IMAGES USING DEEP LEARNING ARCHITECTURES** being submitted by **Raghav Goyal**, 2010MT50612 for the award of **Integrated Masters of Technology** in **Mathematics and Computing** is a record of bonafide work carried out by him under my guidance and supervision at the **Department of Mathematics**.

To the best of my knowledge, the results embodied in this thesis have not been submitted in part of full to other University or Institute for the award of any degree or diploma

DR. B. CHANDRA
Professor
Department of Mathematics
Indian Institute of Technology, Delhi

Abstract

Deep Learning is an area of Machine Learning which consist of algorithms to model high level representation of data by use of multiple hidden layers. Restricted Boltzmann Machine (RBM) were developed to model input data distribution and were used as feature extractors for various classification algorithms. Deep Belief Networks (DBN) are stacked representation of RBM's which are pre-trained greedily to initialize multi layered neural network which is then fine tuned using back propagation.

Restricted Boltzmann Machine can be modified by augmenting an output layer above hidden layer known as Classification RBM (ClassRBM). It is then used to model input distribution along with its class labels. This architecture based on supervised learning can also be used as a standalone classifier. In this work, we pretrained Deep Belief Network greedily in unsupervised manner with ClassRBM as the top layer and fine-tuned to obtain better accuracy over traditional DBN. We argue that the better performance comes from pre training of weights between stacks of RBM and output layer which were randomly initialized previously for fine-tuning. Also, we introduce Convolutional Classification Restricted Boltzmann Machine, which is an extension of ClassRBM to incorporate convolutional aspect and obtained better classification accuracy over ClassRBM.

Acknowledgments

I would like to take this opportunity to express my deepest and sincere gratitude to our thesis supervisor **Dr. B. Chandra** for her valuable suggestions, discussions and continuous support throughout the course of the project. Based on her supervision, I learned a lot during the duration of this Major project.

RAGHAV GOYAL

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Deep Learning is an area of machine learning which consists of algorithms that uses multiple hidden layers in the neural network architecture to model high level representation of data. These algorithms learn feature hierarchies at high levels from lower level features which is then used in feature extraction. Deep learning architectures consistently outperform other classification methods and are comparable to the state-of the art techniques. These are used in various fields including Computer Vision, Natural Language Processing and Speech Recognition.

The breakthrough in the training of deep architectures was achieved when Hinton et al (2006) [4] demonstrated algorithm to train the Deep Belief Networks composed of Restricted Boltzmann Machines by greedy layer wise unsupervised pre-training followed by supervised fine tuning. This work significantly reduced the running time for training of these algorithms.

In Deep Belief Networks, each layer is pre-trained with an unsupervised learning algorithm, learning a nonlinear transformation of its input (the output of the previous layer). [2] This unsupervised pre-training initializes the network for a final training phase where the deep architecture is fine-tuned with respect to a supervised training criterion with gradient-based optimisation.

Classification RBM [8] is an extension of RBM in which an output layer is added over hidden layer to model distribution of input data together with its labels. In this work, DBN is pre-trained by replacing its top RBM layer with ClassRBM layer maintaining same number of units and layers in network. This DBN after fine tuning was found to perform better over traditional DBN. The performance is attributed to pre training of weights between stacks of RBM and output layer which were then used to be initialized randomly for fine tuning.

Real images are typically of large size approximately 200×200 pixels. Since, DBN were unable to incorporate spatial properties of image in model. Convolutional Deep Belief Networks [10] were introduced which provides translational invariance and weight sharing for image classification tasks.

Consequently, we extended the model of ClassRBM to incorporate convolutional aspect to give Convolutional ClassRBM. It was found to perform better over its one-dimensional counterpart. The performance is believed to come from using spatial properties of images through convolution operations.

Also, performance of above architectures on different datasets with state-of-the-art deep learning architectures was compared: Convolutional Neural Network [9], Deep Convolutional Neural Network [7] to complete the work.

In the following chapters, the architecture of Restricted Boltzmann Machine and the Deep Belief Networks is first discussed. Further, Convolutional RBM, ClassRBM, Convolutional ClassRBM are discussed followed by the results.

Chapter 2

Restricted Boltzmann Machine

Restricted Boltzmann Machine is a type of stochastic artificial neural network consisting of two layers. Basic structure of RBM has connections between the layer of hidden and the layer of visible variables but not between the variables of the same layer. In terms of probability, this means that the hidden variables are independent given the state of the visible variables and vice versa.

A binary RBM consists of m visible units $V = (V_1, \cdots, V_m)$ representing the observable data and n hidden units $H = (H_1, \cdots, H_n)$ to capture the dependencies between the observed variables.

$$E(v, h) = -\sum_{i=1}^{m}\sum_{j=1}^{n} v_i W_{i,j} h_j - \sum_{i=1}^{m} b_i v_i - \sum_{j=1}^{n} c_j h_j \tag{2.1}$$

where W_{ij} is a real valued weight between visible unit v_i and hidden unit h_j and b_i and c_j are real valued bias terms associated with i^{th} visible and j^{th} hidden unit respectively.

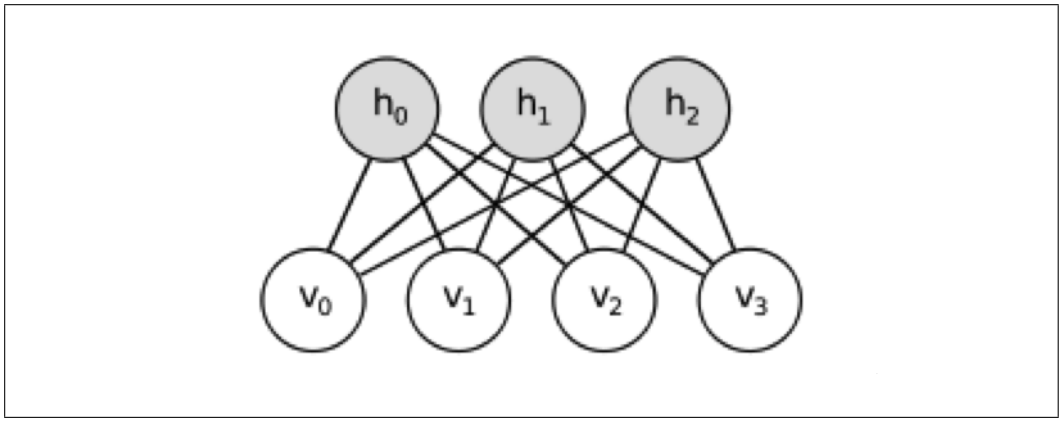

Figure 2.1: Graphical model of the structure of a Restricted Boltzmann Machine

The joint probability distribution of the hidden and visible units can be represented as:

$$p(h|v) = \prod_{j=1}^{n} p(h_j|v) \tag{2.2}$$

$$p(v|h) = \prod_{i=1}^{m} p(v_i|h) \tag{2.3}$$

In particular, the units of a binary hidden layer are independent Bernoulli random variable as follows:

$$p(h_j = 1|v) = \sigma(\sum_{i} W_{ij} v_i + c_j) \tag{2.4}$$

and the visible units are independent Bernoulli random variable as follows:

$$p(v_i = 1|h) = \sigma(\sum_{j} W_{ij} h_j + b_i) \tag{2.5}$$

where σ is the sigmoid function represented as:

$$s\sigma(s) = \frac{1}{1 + exp(-s)} \tag{2.6}$$

The Restricted Boltzmann Machine follow the energy model whereby they are trained to maximise the log likelihood of the training visible data. Since, the derivative of log-likelihood of data does not yield closed solution. The algorithm most commonly used to train RBMs, is the Contrastive Divergence Algorithms proposed by Hinton et al (2006) [4]. It approximates the term with expectation over data distribution using Gibbs Sampling.

2.1 Contrastive Divergence Algorithm

The following is the RBM update procedure for binary visible data [1]. It can be readily adapted for real data using gaussian units. The algorithm describes 1 step of block gibbs sampling. However, it can be extended for any k, by repeating positive and negative phase required number of times to obtain final sample.

Algorithm 1 Training algorithm for RBM

Input: training sample x_i and learning rate λ

Notation: W are weights between visible and hidden layer, b is bias for visible layer, c is bias for hidden layer and $\sim$ is used to denote sampled from distribution

Positive phase
$x^0 = x_i$, $\widehat{h^0} = sigm(c + Wx^0)$

Negative phase
$h^0 \sim p(h|x^0)$, $x^1 \sim p(x|h^0)$
$\widehat{h^1} = sigm(c + Wx^1)$

Update
$W = W + \lambda(\widehat{h^0}(x^0)^T - \widehat{h^1}(x^1)^T)$
$c = c + \lambda(\widehat{h^0} - \widehat{h^1})$
$b = b + \lambda(x^0 - x^1)$

Chapter 3

Deep Belief Networks

A Deep Belief Network (DBN) is a probabilistic, generative model made up of multiple layer of hidden units. It is trained greedily layer wise as introduced by Hinton et al (2006)[4]. RBM being a two layered model is itself limited in what it represents. By stacking layers of RBM and training in a greedy manner, a Deep Belief Network is obtained. In a DBN, each layer comprises a set of binary or real-valued units. Two adjacent layers have a full set of connections between them, but no two units in the same layer are connected.

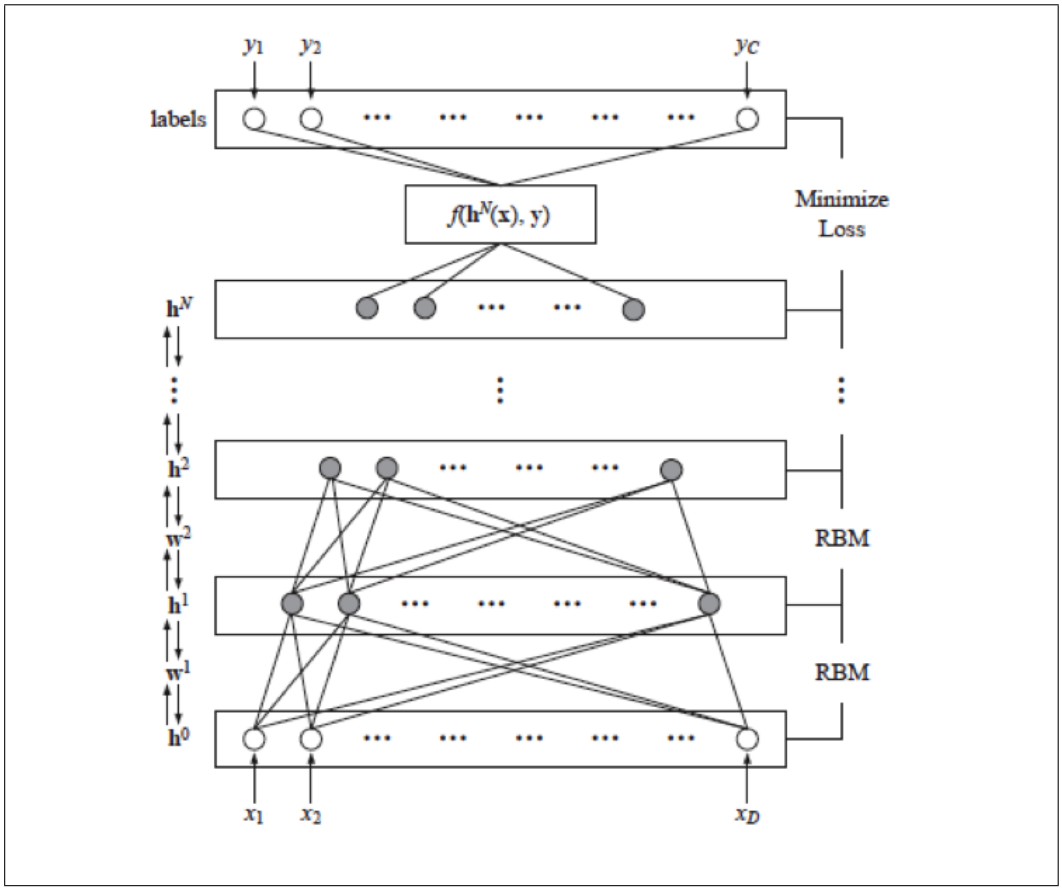

Figure 3.1: [11] A fully interconnected Deep Belief Network with one input layer h^0 and N hidden layers $h^1, h^2, \cdots, h^N$ and one label layer at top.

In general, the probability of k^th hidden layer, h^k is written as a function of states of previous hidden layer h^{k-1} & weights, W^{k-1} between them,

$$p(h^k = 1|h^{k-1}) = \sigma(W^{k-1}h^{k-1} + c^k) \tag{3.1}$$

where, W^{k-1} are weights between s^{th} hidden layer h^{k-1} and h^k and c^k is the bias of layer h^k

Greedy layer wise training of DBN includes,

1. Training the first layer as an RBM that models input $x = h^0$ as its visible layer.
2. Obtaining input using first layer that will be used as data for second layer, by using $p(h^1 = 1|h^0)$
3. Train the second layer as RBM , taking the transformed data as training examples for visible layer of that RBM
4. Propagate upward by iterating through layers using steps 2 & 3 repeatedly
5. Finally, fine tune the weights, W via supervised gradient descent. We intend to minimize the loss function, $f(h^N(x), y)$ over whole labeled dataset by performing gradient descent of negative log-likelihood of this loss function. In general multi-layered perceptron, the loss function is taken to be mean squared error.

Chapter 4

Convolutional Restricted Boltzmann Machine

Restricted Boltzmann Machine and Deep Belief Networks ignore the 2D structure of images. To incorporate this fact, Convolutional Restricted Boltzmann Machine [10] were introduced which gives significantly improved results. In this section, architecture of Convolutional Restricted Boltzmann Machine is discussed which forms the core unit of Convolutional Deep Belief Networks (CDBN).

Intuitively, the Convolutional RBM (CRBM) is same as RBM, but the weights between hidden and visible layer is shared among all locations in an image. The visible layer consists of $N_V \times N_V$ array of binary units. The hidden layer consists of K groups where each group is an $N_H \times N_H$ array of binary units, resulting in $N_H^2 K$ hidden units. Each of the K groups is associated with a $N_W \times N_W$ filter, the filter weights are shared across all hidden units within the group. In addition, each hidden group has a bias b_k and all visible units share a single bias c.
The Energy function for this model is defined as:

$$E(v,h) = -\sum_{k=1}^{K}\sum_{i,j=1}^{N_H}\sum_{r,s=1}^{N_W} h_{i,j}^k W_{r,s}^k v_{i+r-1,j+s-1} - \sum_{k=1}^{K} b_k \sum_{i,j=1}^{N_H} h_{ij}^k - c\sum_{i,j=1}^{N_V} v_{ij} \quad (4.1)$$

In terms of convolution, this energy equation simplifies to:

$$E(v,h) = -\sum_{k=1}^{K} h^k \bullet (\tilde{W}^k * v) - \sum_{k=1}^{K} b_k \sum_{i,j=1} h_{ij}^k - c\sum_{i,j=1} v_{ij} \quad (4.2)$$

Here $*$ represents convolution of two matrices and $\bullet$ denotes element wise product followed by summation. $A \bullet B = trA^T B$. $\tilde{A}$ represents the flipping

of A horizontally and vertically. As with standard RBMs, Gibbs sampling forms the basis of inference and learning algorithms.

$$P(h_{ij}^k = 1|v) = \sigma((\tilde{W^k} * v)_{ij} + b_k) \tag{4.3}$$

$$P(v_{ij} = 1|h) = \sigma\left((\sum_k W^k * H^k)_{ij} + c\right) \tag{4.4}$$

To learn high level representations, CRBM are stacked in a similar way to Deep Belief Networks fomulating Convolutional Deep Belief Networks (CDBN) based on probabilistic max-pooling. Each unit in a pooling layer computes the maximum activation of the units in a small region of the detection layer. Shrinking the representation with max-pooling allows higher-layer representations to be invariant to small translations of input and reduces the computational burden.

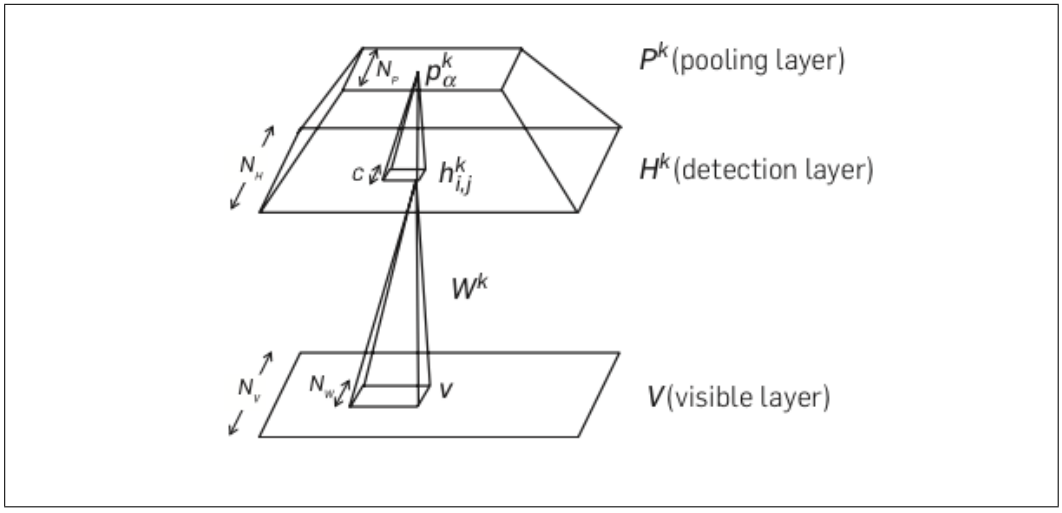

Figure 4.1: [10] Convolutional RBM with probabilistic max-pooling

The detection and pooling layers both have K group of units and each group in pooling layer has $N_P \times N_P$ binary units. The detection layer H^k is partitioned into blocks of size $C \times C$, and each block α is connected to exactly one binary unit p_α^k in the pooling layer with $N_p = N_H/C$. The detection units in the block B_α and the pooling unit p_α are connected where at most of the detection units may be on and the pooling unit is on if and only if a

detection unit is on.

The energy function of probabilistic max-pooling CRBM is:

$$E(v,h) = -\sum_{k}\sum_{i,j}\left(h^k_{i,j}(\tilde{W}^k * v)_{ij} + b_k h^k_{i,j}\right) - c\sum_{i,j} v_{i,j} \tag{4.5}$$

subject to

$$\sum_{(i,j)\in B_\alpha} h^k_{i,j} \leq 1, \forall k, \alpha \tag{4.6}$$

CDBN are defined as hierarchical generative model for full-sized images. In a similar fashion to DBN, CDBN consists of several max-pooling CRBMs stacked together to form the architecture of CDBN. The network defines an energy function by summing the energy function for all individual pair of layers and the training is done with the greedy, layer-wise procedure.

Algorithm 2 Training algorithm for CRBM:

Input: Training sample x_i and learning rate λ
Positive phase
$x^0 = x_i$, $\widehat{h^{0k}_{ij}} = \sigma((\tilde{W}^k * x^0)_{ij} + b^k_{ij})$

Negative phase
$h^0 \sim p(h|x^0)$, $x^1 \sim p(x|h^0)$
$\widehat{h^{1k}_{ij}} = \sigma((\tilde{W}^k * x^1)_{ij} + b^k_{ij})$

Update
$W^k = W^k + \lambda(x^0 * \widetilde{\widehat{h}}^0 - x^1 * \widetilde{\widehat{h}}^1)$
$b^k = b^k + \lambda((\widehat{h}^0)^k - (\widehat{h}^1)^k)$
$c = c + \lambda(\sum_{ij}(x^0_{ij} - x^1_{ij}))$

Chapter 5

Classification Restricted Boltzmann Machine

Introduced earlier, Restricted Boltzmann Machines are generative models which are trained in unsupervised manner to model input distribution. They find applications as feature extractors for another classification algorithm or initializing deep networks, particularly Deep Belief Networks [5].

Classification RBM or ClassRBM [8] is an architecture of RBM augmented with output layer such that there are no interactions between input & output layer and hidden layer is fully connected to both of them.

The joint probability configuration of this model with hidden units $\mathbf{h}$, observed input units $\mathbf{x}$ & observed output units $\mathbf{y} = (1_{y=i})^C$ is,

$$p(y, \mathbf{x}, \mathbf{h}) \propto e^{-E(y,\mathbf{x},\mathbf{h})} \tag{5.1}$$

where energy is given by,

$$E(y, \mathbf{x}, \mathbf{h}) = -\mathbf{h}^T\mathbf{W}\mathbf{x} - \mathbf{b}^T\mathbf{x} - \mathbf{c}^T\mathbf{h} - \mathbf{d}^T\mathbf{y} - \mathbf{h}^T\mathbf{U}\mathbf{y} \tag{5.2}$$

with parameters $\Theta = (\mathbf{W}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{U})$ and $\mathbf{y} = (1_{y=i})^C$ for C classes.

From the conditional independence property of this architecture, it can be seen that,

$$p(\mathbf{x}|\mathbf{h}) = \prod_i p(x_i|\mathbf{h}) \tag{5.3}$$

$$p(\mathbf{h}|y, \mathbf{x}) = \prod_j p(h_j|y, \mathbf{x}) \tag{5.4}$$

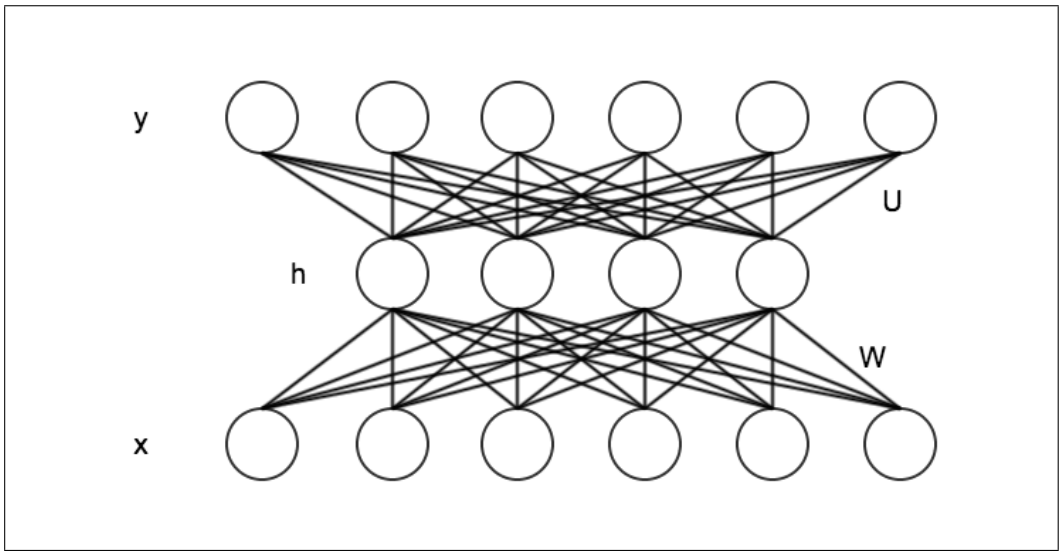

Figure 5.1: Classification Restricted Boltzmann Machine

$$p(y|\mathbf{h}) = \frac{e^{d_y + \sum_j U_{jy} h_j}}{\sum_y e^{d_y + \sum_j U_{jy} h_j}} \tag{5.5}$$

where $p(y|\mathbf{h})$ is a softmax function. Also,

$$p(x_i = 1|\mathbf{h}) = \sigma(\sum_j W_{ji} h_j + b_i) \tag{5.6}$$

$$p(h_j = 1|y, \mathbf{x}) = \sigma(\sum_i W_{ji} x_i + c_j + U_{jy}) \tag{5.7}$$

where σ is the sigmoid function represented as:

$$\sigma(s) = \frac{1}{1 + exp(-s)} \tag{5.8}$$

Now, the likelihood of observed data is maximised i.e $p(\mathbf{x}, y)$ where

$$p(\mathbf{x}, y) = \sum_{\mathbf{h}} p(\mathbf{x}, \mathbf{h}, y) \tag{5.9}$$

The gradient of log-likelihood for i^{th} sample with respect to model parameter θ is given by,

$$\frac{\partial logp(y_i, x_i)}{\partial \theta} = -E_{h|y_i,x_i}\left[\frac{\partial}{\partial \theta}E(y_i, x_i, h)\right] + E_{y,x,h}\left[\frac{\partial}{\partial \theta}E(y, x, h)\right] \tag{5.10}$$

which can be seen as difference of expectation w.r.t model and data respectively. The first term is tractable in case RBM's while second term is not. Hence, expectation w.r.t. data over all possible values of **x**, y is approximated by taking sample using Gibbs Sampling. Also popularly known as Contrastive Divergence.

Algorithm 3 Contrastive Divergence Updates

Input:
training pair(y_i, x_i) and learning rate λ

Positive phase
$y^0 = y_i$, $x^0 = x_i$, $\widehat{h^0} = sigm(c + Wx^0 + U\vec{y^0})$

Negative phase
$h^0 \sim p(h|y^0, x^0)$, $y^1 \sim p(y|h^0)$, $x^1 \sim p(x|h^0)$
$\widehat{h^1} = sigm(c + Wx^1 + U\vec{y^1})$

Update
$W = W + \lambda(\widehat{h^0}(x^0)^T - \widehat{h^1}(x^1)^T)$
$U = U + \lambda(\widehat{h^0}(y^0)^T - \widehat{h^1}(y^1)^T)$
$c = c + \lambda(\widehat{h^0} - \widehat{h^1})$
$b = b + \lambda(x^0 - x^1)$
$d = d + \lambda(y^0 - y^1)$

For classification, $p(y|\mathbf{x})$ is calculated and subsequently y is sampled from this distribution.

$$p(y|\mathbf{x}) = \frac{e^{d_y}\prod_{j=1}^{n}(1 + e^{c_j + U_{jy} + \sum_i W_{ji}x_i})}{\sum_y e^{d_y}\prod_{j=1}^{n}(1 + e^{c_j + U_{jy} + \sum_i W_{ji}x_i})} \tag{5.11}$$

Chapter 6

Convolutional Classification Restricted Boltzmann Machine

Introduced in Chapter 5, Classification Restricted Boltzmann Machine architecture is based on RBM which assumes input data to be 1-dimensional. Since, convolution is known to exploit spatial properties of input data, specifically images. Hence, the same configuration can be extended for Convolutional Restricted Boltzmann Machine model. Earlier Convolutional RBM was introduced by [10].

The model consists of a visible input layer $\mathbf{x}$, k hidden layers $\mathbf{h}$ and output layer $\mathbf{y}$. There are no connections between input & output layer. The hidden layer, which is a 2-dimensional layer is fully connected to output layer and is connected to input layer through a weight filter convolution.

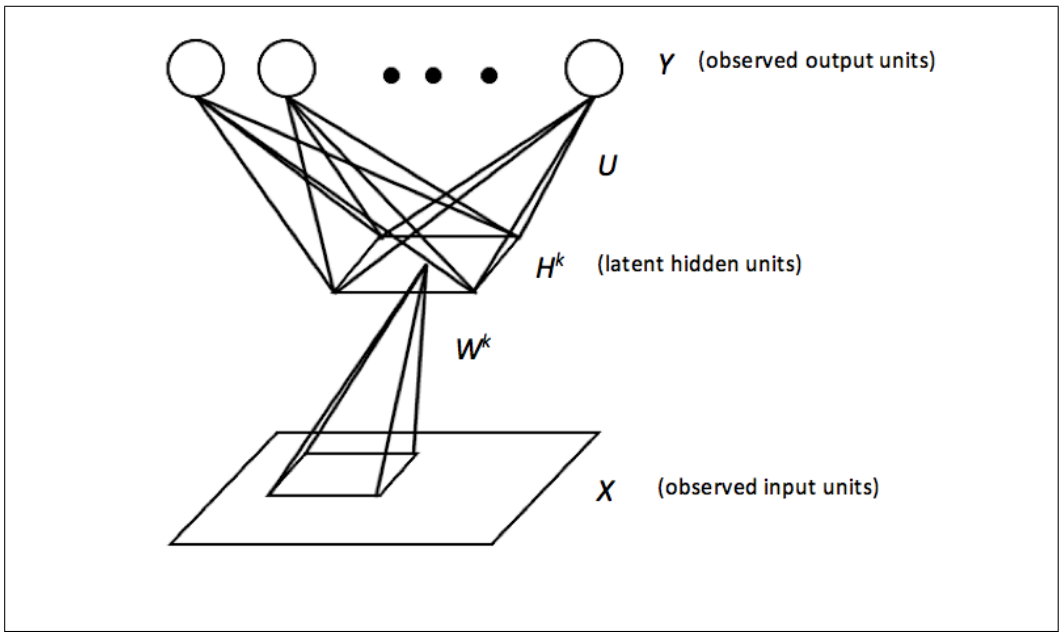

Figure 6.1: Convolutional ClassRBM where only k^{th} hidden layer group is demonstrated for simplicity

As illustrated in the figure, W^k is the weight filter specific to k^{th} hidden group. The joint probability configuration of model is given by,

$$p(y, \mathbf{x}, \mathbf{h}) \propto e^{-E(y,\mathbf{x},\mathbf{h})} \tag{6.1}$$

where energy is,

$$\begin{aligned} E(\mathbf{x}, \mathbf{h}, \mathbf{y}) = -\sum_{k=1}^{K} h^k \bullet (\tilde{W^k} * \mathbf{x}) - \sum_{k=1}^{K} \sum_{i,j} b_{ij}^k h_{ij}^k - c \sum_{i,j} x_{ij} \\ - \sum_z d_z y_z - \sum_z y_z \sum_k (h^k \bullet U^{kz}) \end{aligned} \tag{6.2}$$

Here, $*$ is used to denote convolution and $\bullet$ to denote element-wise product followed by summation. Also, K is the number of groups of weight filters or hidden layers, b_{ij}^k is the bias of $(i,j)^{th}$ hidden unit for k^{th} group, c is the bias for visible input image, d_z is the bias for z^{th} output unit & U^{kz} are weight connections between k^{th} hidden group and z^{th} output unit.

Following the conditional independence property,

$$p(\mathbf{x}|\mathbf{h}) = \prod_{i,j} p(x_{ij}|\mathbf{h}) \tag{6.3}$$

$$p(\mathbf{h}|\mathbf{x}, y) = \prod_{i,j,k} p(h_{ij}^k|\mathbf{x}, y) \tag{6.4}$$

Also, the conditional distributions are specified as,

$$p(x_{ij} = 1|\mathbf{h}) = \sigma((\sum_k W^k * h^k)_{ij} + c) \tag{6.5}$$

$$p(h_{ij}^k = 1|y, \mathbf{x}) = \sigma((\tilde{W^k} * x)_{ij} + b_{ij}^k + \sum_z U_{ij}^{kz} y_z) \tag{6.6}$$

$$p(y_z = 1|\mathbf{h}) = \frac{\exp(d_z + \sum_k h^k \bullet U^{kz})}{\sum_z \exp(d_z + \sum_k h^k \bullet U^{kz})} \tag{6.7}$$

For training the model, the likelihood of data is maximised. By analogy from previous RBM, the expectation w.r.t. data is intractable and approximated by Gibbs Sampling, also known as Contrastive Divergence.

Algorithm 4 Contrastive Divergence Updates

Input:
training pair(y_i, x_i) and learning rate λ

Positive phase
$y^0 \leftarrow y_i$, $x^0 \leftarrow x_i$, $\widehat{h_{ij}^{0k}} \leftarrow \sigma((\tilde{W}^k * x^0)_{ij} + b_{ij}^k + \sum_z U_{ij}^{kz} y_z^0)$

Negative phase
$h^0 \sim p(h|y^0, x^0)$, $y^1 \sim p(y|h^0)$, $x^1 \sim p(x|h^0)$
$\widehat{h_{ij}^{1k}} \leftarrow \sigma((\tilde{W}^k * x^1)_{ij} + b_{ij}^k + \sum_z U_{ij}^{kz} y_z^1)$

Update
$W^k = W^k + x^0 * \widetilde{\widehat{h}^0} - x^1 * \widetilde{\widehat{h}^1}$
$U^{kz} = U^{kz} + (\widehat{h}^0)^k y_z^0 - (\widehat{h}^1)^k y_z^1$
$b^k = b^k + (\widehat{h}^0)^k - (\widehat{h}^1)^k$
$c = c + \sum_{ij}(x_{ij}^0 - x_{ij}^1)$
$d_z = d_z + y_z^0 - y_z^1$

$$p(y_z = 1|\mathbf{x}) = \frac{e^{d_z} \prod_{i,j,k}(1 + e^{b_{ij}^k + U_{ij}^{kz} + (\tilde{W}^k * x)_{ij}})}{\sum_y e^{d_z} \prod_{i,j,k}(1 + e^{b_{ij}^k + U_{ij}^{kz} + (\tilde{W}^k * x)_{ij}})} \tag{6.8}$$

Chapter 7

Results

Dataset used in the results is MNIST [9]. It is a popularly used benchmark dataset consisting of handwritten digits from 0 to 9. It is divided into 60,000 training samples and 10,000 test samples

The results are divided into two parts. The first contains classification performance of DBN with ClassRBM layer at the top & second contains Convolutional ClassRBM and its performance comparison with ClassRBM.

DBN + ClassRBM is an architecture of DBN with top layer RBM replaced with Classification RBM. Therefore, unlike traditional DBN which pre trains in an unsupervised manner, this architecture additionally uses label information in pre-training step due to presence of ClassRBM. Hence, when fine-tuning the architecture using back propagation, the initialisation of weights between output layer and second-last layer is already obtained in pre-training step.

The following table shows comparison where DBN + ClassRBM is a one hidden layered network with 784 input units, 1000 hidden units & 10 output units. The code is implemented in Theano. It's result is compared with DBN, which is one hidden layered neural network pre-trained as an unsupervised RBM [1].

Table 7.1: Comparison of classification errors on MNIST dataset

Model	Error
DBN	1.41%
DBN+ClassRBM	**1.37%**

As mentioned by [1], differences of 0.1% classification error on this dataset is statistically significant.

The Weight filters learned for the DBN+ClassRBM architecture after pre-training step can be observed below,

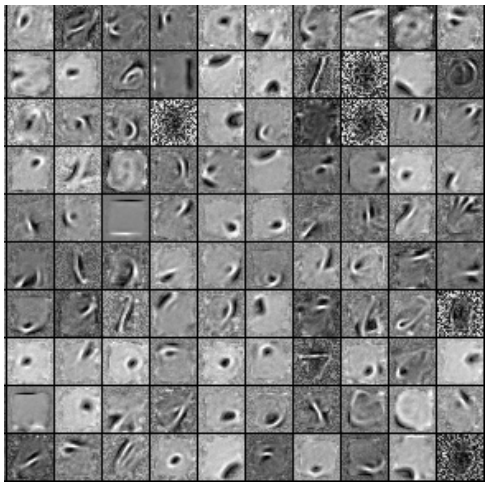

Figure 7.1: Few weight filters between input layer & hidden layer after pre training for DBN+ClassRBM

Figure 7.2: Weight filters between output & hidden layer after pre training for DBN+ClassRBM

The second part of results compares the performance of Classification RBM and Convolutional Classification RBM as a standalone classifier. Wherein, the classification is obtained by sampling from posterior distribution described for the models in equations 5.11 and 6.8 respectively.

ClassRBM is Classification RBM described in [8] with one hidden layer with 6000 hidden units.
ConvClassRBM is convolutional classification RBM with one hidden layer consisting of 40 groups of hidden units. Size of each weight filter is 12×12. Each input sample is now an image of 26×26 size.

Table 7.2: Comparison of classification performance on MNIST dataset

Model	Error
ClassRBM	3.39%
ConvClassRBM	**2.9%**

Chapter 8

Conclusion

Deep Learning architectures have achieved almost human precision in classifying real world images, with Deep Convolutional Networks being state-of-the art. Deep Belief Networks, stacked representation of the probabilistic Restricted Boltzmann Machine was trained greedily followed by supervised back propagation. The architecture of Classification Restricted Boltzmann Machine can be used as the topmost layer in a DBN to classify images and has shown significant improvements over the original Deep Belief Networks where the supervised training of the topmost layer helps in learning the weights more efficiently.

On the other hand, Convolutional ClassRBM is shown to perform better than its one dimensional counterpart. The reason being that the latter exploits spatial properties of image via translational invariance and weight sharing as shown in related literature.

Bibliography

[1] Bengio, Yoshua, et al. "Greedy layer-wise training of deep networks." Advances in neural information processing systems 19 (2007): 153.

[2] Erhan, Dumitru, et al. "Why does unsupervised pre-training help deep learning?." The Journal of Machine Learning Research 11 (2010): 625-660.

[3] Fischer, Asja, and Christian Igel. "Training restricted Boltzmann machines: An introduction."Pattern Recognition47.1 (2014): 25-39.

[4] Hinton, Geoffrey. "A practical guide to training restricted Boltzmann machines." Momentum 9.1 (2010): 926.

[5] Hinton, Geoffrey E., Simon Osindero, and Yee-Whye Teh. "A fast learning algorithm for deep belief nets." Neural computation 18.7 (2006): 1527-1554.

[6] Krizhevsky, Alex, and Geoffrey Hinton. "Learning multiple layers of features from tiny images." Computer Science Department, University of Toronto, Tech. Rep (2009).

[7] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

[8] Larochelle, Hugo, and Yoshua Bengio. "Classification using discriminative restricted Boltzmann machines." Proceedings of the 25th international conference on Machine learning. ACM, 2008.

[9] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998d). "Gradient-based learning applied to document recognition". Proceedings of the IEEE, 86(11), 22782324

[10] Lee, Honglak, et al. "Unsupervised learning of hierarchical representations with convolutional deep belief networks."Communications of the ACM54.10 (2011): 95-103.

[11] Zhou, Shusen, Qingcai Chen, and Xiaolong Wang. "Discriminative Deep Belief networks for image classification." Image Processing (ICIP), 2010 17th IEEE International Conference on. IEEE, 2010.