

Temporal Coherence in Energy-based Deep Learning Machines for Action Recognition

Adepu Ravi Sankar

A Thesis Submitted to
Indian Institute of Technology Hyderabad
In Partial Fulfillment of the Requirements for
The Degree of Master of Technology



Department of Computer Science and Engineering

June 2014

Declaration

I declare that this written submission represents my ideas in my own words, and where ideas or words of others have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources that have thus not been properly cited, or from whom proper permission has not been taken when needed.

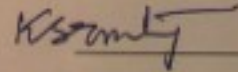
(Signature)

(Adepu Ravi Sankar)

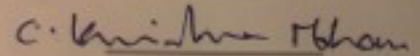
(Roll No.)

Approval Sheet

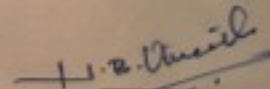
This Thesis entitled Temporal Coherence in Energy-based Deep Learning Machines for Action Recognition by Adepu Ravi Sankar is approved for the degree of Master of Technology in Computer Science and Engineering from IIT Hyderabad



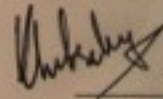
(Dr. K. Sri Rama Murty) Examiner
Dept. of Electrical Engineering
IITH



(Dr. C. Krishna Mohan) Examiner
Dept. Computer Science and Engineering
IITH



(Dr. Vineeth N. Belasubramanian) Adviser
Dept. of Computer Science and Engineering
IITH



(Dr. Subrahmanyam Kalyanasundaram) Chairman
Dept. of Computer Science and Engineering
IITH

Acknowledgements

I would like to thank Dr.Jayaram Balasubramaniam and Dr.K Sri Rama Murty for their invaluable discussions on Deep Learning.

Dedication

I would like to dedicate this thesis to Miss.Gadde Vinuthna who is my inspiration always.

Abstract

Deep Learning, a sub-area of machine learning, has become a buzz word in recent days due to its great successes in many applications of machine learning, including speech processing, computer vision and natural language processing. Deep learning became famous in the initial days through the successful application of Convolutional Neural Networks as well as Energy-based Models -or Restricted Boltzmann Machines (RBMs) - on handwritten digit recognition. While the last decade has seen the growing use of convolution-based deep learning methods for image analysis, limited work has been done in adapting deep learning to video analysis. Existing methods have largely extended the ideas based on convolution applied to images into the video analysis setting. The primary deep learning approaches that have been proposed so far explicitly for video sequences are the 3D Convolutional Neural Networks and the Convolutional Gated RBM.

Our work in this thesis extends the Convolutional Gated RBM (CGRBM) by leveraging the idea of temporal coherence in video sequences to modify the weight update rule of CGRBM. A novel idea of using similar and dissimilar images as alternate training inputs to modulate the energy curve of Energy Based Models (RBMs) so as to form deep valleys at points of similar images and to construct big peaks at the location of dissimilar images is being proposed. Experiments conducted on the NORB and KTH datasets reveal that the modified learning algorithm learns the image transformations between two images just like optical flow and can produce optical flow-like relationship between two input images without using any hand engineered features. Using the learned features for recognition using standard classifiers (such as Support Vector Machines) also demonstrated promise on the KTH dataset. The proposed method can also be generalized to other application settings by defining similarity appropriately, such as an object's similarity with a rotated version, or a person's similarity to images of the same person showing different expressions.

Contents

Declaration	ii
Approval Sheet	iii
Acknowledgements	iv
Abstract	vi
Nomenclature	viii
1 Introduction to Deep Learning	1
1.1 History of Deep Learning	1
1.2 Introduction to Energy Based Models	2
1.3 Action Recognition from videos	3
1.4 Thesis Motivation	3
2 Related Work	5
2.1 Background	5
2.1.1 Convolutional Neural Networks	5
2.1.2 Restricted Boltzmann Machines (RBMs)	6
2.1.3 Free Energy in RBM	8
2.1.4 Energy Minimization training of RBM using Contrastive Divergence	8
2.1.5 Update Rule	10
2.1.6 CD Algorithm	10
2.1.7 Gaussian Bernoulli RBM	10
2.2 Related Work: Deep Learning for Action Recognition	13
2.2.1 3D CNN for Action Recognition	13
2.2.2 Convolutional Gated RBM (CGRBM) for Action Recognition	15
3 Temporal Coherence in Energy-Based Deep Models: Proposed Methodology	17
3.1 Modelling temporal sequences of videos	17
4 Experimental Results	19
4.1 Experimental Setup	19
4.2 Experimental Results on NORB Data set	19
4.3 Experimental Results on KTH Data set	20

5	Conclusions and Future Work	22
5.1	Where does our approach work and why?	22
5.2	Directions for Future Work	22
5.3	Conclusions	22
	References	23

Chapter 1

Introduction to Deep Learning

Deep learning is a sub-area of machine learning which, inspired by models of the brain, has provided methods to learn features at different layers in a hierarchical architecture for potential use in various real-world applications. In its early years, Deep Learning (DL) was not so successful due to the requirement of huge computational power given the presence of deep hierarchical layers. Machine learning algorithms are typically classified as either discriminative models or generative models. Deep learning methods fall under the generative category as they learn the underlying probability distribution in the data and generate the same distribution upon successful training.

1.1 History of Deep Learning

The artificial neural networks research community first introduced the concept of Deep Learning through the use of feedforward neural networks or Multi Layer Perceptrons (MLPs) which are trained using backpropagation. Over the years, these networks have lost their popularity due to the presence of spurious local minima and corresponding non-convex objective functions, which make the models get trapped in local minima [1]. This motivated the development of other methods in machine learning based on shallow architectures such as Support Vector Machines (SVMs) and Conditional Random Fields (CRFs) for which global optima can be calculated very efficiently.

A new class of deep architectures named Deep Belief Networks (DBNs) was introduced by Hinton et al in 2006. DBNs are composed of Restricted Boltzmann Machines (which are described in Chapter 2 of this thesis) as a basic building block; and are trained in a greedy layer-wise manner and fine-tuned using backpropagation. The success of DBNs for real-world tasks such as handwritten digit recognition spearheaded the re-entry of deep neural networks into mainstream machine learning applications.

Deep learning methods can broadly be categorized into energy-based models (such as RBMs), and non-energy based models (such as convolutional backpropagation networks). We now discuss energy-based models in more details, considering this is the focus of this thesis.

1.2 Introduction to Energy Based Models

The key idea of Energy Based Models (EBMs) is to associate a scalar energy function value to every configuration of variables. Training is carried out by finding the values of weight variables that minimize the energy. This is achieved by associating low energy to correct variables and high energy to incorrect values. The overall loss function is defined as minimizing the energy of overall training data and labels. EBMs can be used to replace traditional probabilistic models such as graphical models. In fact, many probabilistic models can be defined as a sub-category of EBMs where the normalization term satisfies certain conditions.

Let us consider two sets of variables X and Y to be modelled using EBMs. The energy function models the measure of goodness of possible configurations of X and Y . In a classification context, the energy function can be viewed as a measure of nearness between inputs vectors X and output vectors Y . Given that the energy function is defined as $E(X, Y)$, the output Y is then given by:

$$Y^* = \operatorname{argmin}_{Y \in \mathcal{Y}} E(X, Y)$$

Finding Y^* is not easy if the size of set Y is large. So, we use an inference procedure to find Y^* which is the minima of $E(X, Y)$. However, there may be many local minimas for the energy curve. Several inference procedures can be used to infer based on the models internal structure. Depending on the nature of the energy function, E , we can opt for many optimization techniques viz. simulated annealing, gradient descent, linear programming etc to learn values of the variables in the network. For instance, we can use a gradient-based optimization approach if the energy surface is smooth. The trained model can be used to predict Y or classify Y or to calculate the conditional estimate of $p(Y | X)$.

In the classification setting, training an EBM attributes to finding an \mathcal{E} that produces the most relevant Y for a given X , which is defined as:

$$\mathcal{E} = \{E(W, X, Y) : W \in \mathcal{W}\}$$

A simple interpretation of the above function is to find a set of weights W in a neural network that minimizes the error for a given input X and corresponding Y . The model is trained using a set of inputs $S = \{(X^i, Y^i) : i = 1..n\}$, where X^i is the i^{th} input training sample and Y^i is the corresponding output label. The overall loss function is defined as

$$\mathcal{L}(E, S) = \frac{1}{n} \sum_{i=1}^n L(Y^i, E(W, \mathcal{Y}, X^i)) + R(W)$$

The above loss function is the average loss function and $R(W)$ is the regularization parameter.

The minimization criterion of EBMs can be interpreted in the following way. The value of the energy for favoured inputs should be low when compared to non-favoured inputs (as illustrated in the adjoining figure). When the EBM is trained on a given set of $\{(X^i, Y^i)\}$, the energy value of valid Y^i is pushed down and the energy of the invalid \bar{Y}^i is pushed up.

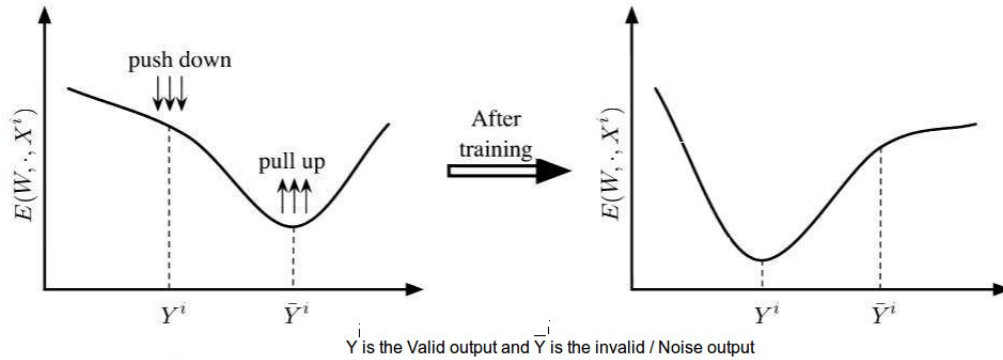


Figure 1.1: Interpretation of energy curve of EBMs [2]

1.3 Action Recognition from videos

The ability to classify the action of a person in a video is referred to as human action recognition. This action recognition has wide range of applications ranging from surveillance systems to robotics. A vast amount of research has been done in this area. The figure below gives an overview of the various approaches that have been attempted to tackle the problem of human action recognition.

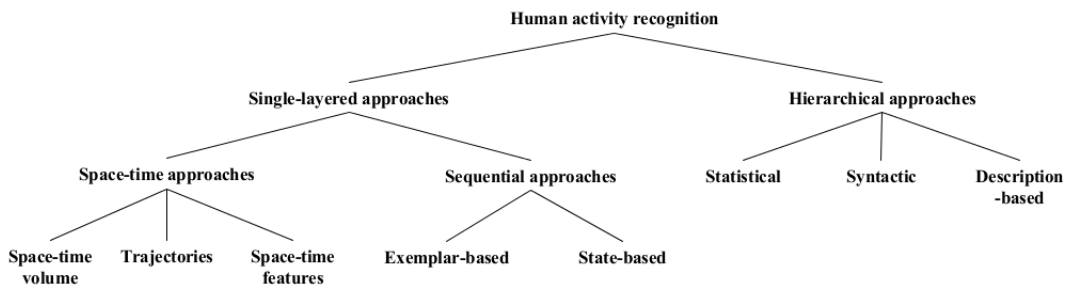


Figure 1.2: Hierarchical based approach for Action Recognition

As shown in Figure 1.2, action recognition models can be broadly classified into two types, viz. single-layered approaches and hierarchical Approaches [3]. Space-time approaches consider time as the third dimension for the input data. Sequential approaches take previous frames also into consideration to model the current frame. Hidden Markov Models (HMMs) are the best example for these kinds of models. Syntactic models use context-free grammar to model human actions. For more information, the interested reader is requested to refer [3].

1.4 Thesis Motivation

As evident from the previous section, almost all approaches for action recognition are based on handcrafted features (such as Scale-Invariant Feature Transform or Histogram of Gradients). These features are very helpful for image analysis, however ignore temporal correlation in videos. In problems such as action recognition, the temporal information plays a crucial role in determining the type of action that was performed, as the state of present action is dependent on one or more previous

states. Only a very few approaches like HMMs consider temporal information in the model, but are based on handcrafted features which may or may not suit a given application.

On the other hand, the features learned using RBMs (and similar EBMs) have been understood to be very rich in recent work, as they capture nuances of the input image in a particular application that cannot be modelled easily using handcrafted features. However, these methods have predominantly been used only with images. We hypothesize that when these underlying features are together combined with temporal information in a video, the representation can be very useful for classification problems such as action recognition. The RBMs model the joint probability of the input distribution with the labels. Typical RBMs only take one frame at a time, but do not consider information about previous frames in a video. In this work, we have sought to make this contribution - to infuse the idea of temporal coherence in EBMs, by building on top of a recently proposed deep architecture called the Gated RBM.

Chapter 2

Related Work

2.1 Background

In this chapter we will discuss methods to achieve deep learning using Convolutional Neural Networks and Restricted Boltzmann Machines. Since the contribution of this thesis is related to Energy Based Models, inference procedure and weight update rules are also presented for these models.

2.1.1 Convolutional Neural Networks

Convolutional neural networks(CNN) are a type of feed forward Deep network that showed success full results on classifying MNIST data set by Yann lecunn [4]. CNNs automate the process of feature constructing by acting directly upon raw inputs. CNNs result in learning complex features that cannot be learned using hand crafted features through alternating convolution and pooling layers.

Convolutional Neural Networks are based on the idea of replicated features. In images if one feature detector is used at one location it is highly likely that the same feature detector could be used at another location in the image. This replication of feature detector across the position greatly reduces the number of free parameters to be learned. The replicated features achieve equivariance. Pooling achieves small amount of translational invariance by averaging neighbouring replicated detectors to give a single output. Pooling helps us to reduce the number of inputs to the next layer thus allowing us to have many more different feature maps.

Figure 2.1 is an example of Convolutional Neural network by Le Cunn et al.

The above architecture consists of 7 layers. An image of size 32x32 pixels is fed as input to the network. Layer C1 is a convolution layer with 6 feature maps with convolution kernel of size 5x5. Layer S2 is a sub-sampling layer where each unit in S2 is connected to a 2x2 non overlapping neighbourhood of its previous convolution layer C1. The output of sub-sampling is passed through sigmoid activation function. Layer C3 is a convolutional layer with 16 feature maps using convolutional kernel of size 5x5. Layer S4 is a sub-sampling layer with 16 feature maps. C5 is a again a convolution layer with 120 feature maps which when convolved with 5x5 kernel will give a feature map of size 1x1, which amounts to a fully-connected layer between S4 and C5.

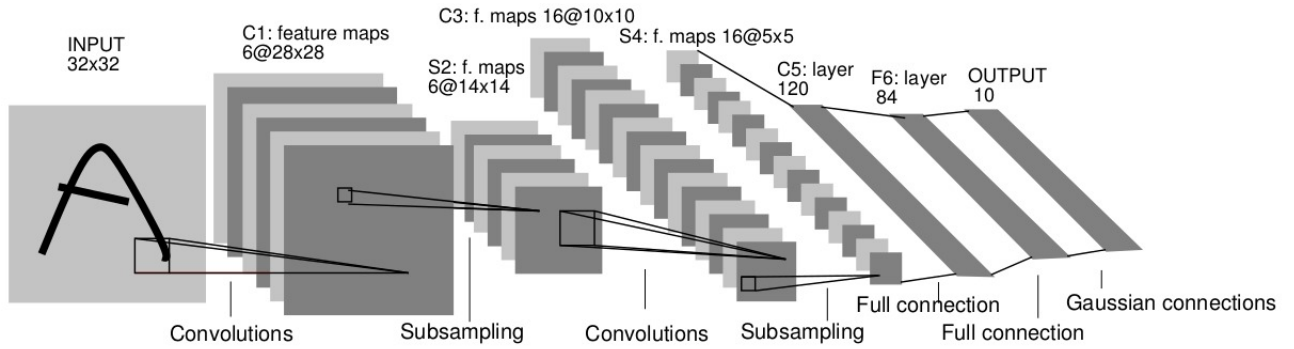


Figure 2.1: Architecture of LeNet-5

2.1.2 Restricted Boltzmann Machines (RBMs)

RBMs follow unsupervised learning i.e they use only inputs $x^{(t)}$ for learning and automatically extract meaningful features for learning. They leverage the availability of unlabelled data. An RBM is an undirected graphical model that defines a distribution over some input vector x , and models the distribution of the input vectors x using a layer of binary hidden units H .

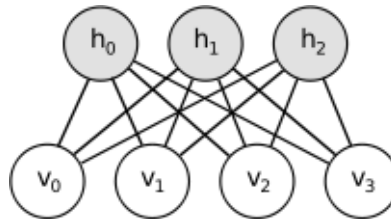


Figure 2.2: Restricted Boltzmann Machine

The energy of RBM is defined as:

$$\begin{aligned}
 E(x, h) &= -h^T W x - c^T x - b^T h \\
 &= \sum_j \sum_k W_{j,k} h_j x_k - \sum_k c_k x_k - \sum_j b_j h_j
 \end{aligned}$$

where j, k are number of neurons in hidden and visible layers.

The probability of observing a particular configuration is given by:

$$p(x, h) = e^{-E(x, h)/Z}$$

where Z is the sum of energy of all combinations of visible and hidden, but is intractable given the combinatorial expansion of possible configuration.

In the Markov network view (given below), the notation is simply an alternative to the representation as the product of factors.

$$\begin{aligned}
 p(x, h) &= e^{-E(x, h)/Z} \\
 &= e^{(h^T W x + c^T x + b^T h)/Z} \\
 &= e^{h^T W x / Z} e^{c^T x / Z} e^{b^T h / Z}
 \end{aligned}$$

Inference in RBM: Since $p(x,h)$ is intractable due to presence of Z , we will infer using the conditional inference using either $P(x|h)$ or $p(h|x)$

In RBM the conditional inference $p(h|x)$ or $p(x|h)$ is very easy to compute. $p(h | x) = \prod_j p(h_j | x)$

Proof:

$$= \frac{\exp(h^T W x + c^T x + b^T h)/Z}{\sum_{h' \in \{0,1\}^H} \exp(h'^T W x + c^T x + b^T h')/Z}$$

As $c^T x$ is independent of summation in both numerator and denominator it cancels out.

$$\begin{aligned} & \frac{\exp(\sum_j h_j W_j x + b_j h_j)}{\sum_{h'_1 \in \{0,1\}^H} \dots \sum_{h'_H \in \{0,1\}^H} \exp(\sum_j h'_j W_{ij} x + b_j h'_j)} \\ &= \frac{\prod_j \exp(h_j W_j x + b_j h_j)}{\sum_{h'_1 \in \{0,1\}^H} \dots \sum_{h'_H \in \{0,1\}^H} \prod_j \exp(\sum_j h'_j W_{ij} x + b_j h'_j)} \\ &= \frac{\prod_j \exp(h_j W_j x + b_j h_j)}{\sum_{h'_1 \in \{0,1\}^H} \exp(h'_1 W_1 x + b_1 h'_1) \dots \sum_{h'_H \in \{0,1\}^H} \exp(h'_H W_H x + b_H h'_H)} \\ &= \frac{\prod_j \exp(h_j W_j x + b_j h_j)}{\prod_j (\sum_{h'_j \in \{0,1\}^H} \exp(h'_j W_j x + b_j h'_j))} \end{aligned}$$

If we expand the summation which is over 0,1 in the denominator, when h'_j is 0, it gives e^0 which is 1 and when h'_j is 1, it gives $\exp(b_j + W_j x)$

$$\begin{aligned} &= \frac{\prod_j \exp(h_j W_j x + b_j h_j)}{\prod_j (1 + \exp(b_j + W_j x))} \\ &= \prod_j \frac{\exp(h_j W_j x + b_j h_j)}{1 + \exp(b_j + W_j x)} \\ &= \prod_j p(h_j | x) \end{aligned}$$

Now by above theorem we have:

$$p(h_j = 1 | x) = \prod_j \frac{\exp(W_j x + b_j)}{1 + \exp(b_j + W_j x)}$$

Multiply and divide the above equation by $\exp(-b_j - w_j x)$ we get,

$$p(h_j = 1 | x) = \text{sigm}(b_j + W_j x)$$

2.1.3 Free Energy in RBM

We will see how RBM models the marginal distribution of $p(x)$ only, i.e How RBM make certain types of inputs more likely and others less likely.

$p(x)$ is given by,

$$\begin{aligned} p(x) &= \sum_{h \in \{0,1\}^H} p(x, h) \\ &= \sum_{h \in \{0,1\}^H} \exp(-E(x, h))/Z \end{aligned}$$

Lets transform the above exponential sum to linear sum

$$\begin{aligned} p(x) &= \sum_{h \in \{0,1\}^H} \exp(h^T W x + c^T x + b^T h)/Z \\ &= \exp(c^T x) \sum_{h_1 \in \{0,1\}} \dots \sum_{h_H \in \{0,1\}} \exp(h_j W_j x + b_j h_j)/Z \\ &= \exp(c^T x) \left(\sum_{h_1 \in \{0,1\}} \exp(h_1 W_1 x + b_1 h_1)/Z \right) \dots \left(\sum_{h_H \in \{0,1\}} \exp(h_H W_H x + b_H h_H)/Z \right) \end{aligned}$$

After expanding with the summation term,

$$\begin{aligned} &= \exp(c^T x) (1 + \exp(b_1 + w_1 x)) \dots (1 + \exp(b_H + w_H x))/Z \\ &= \exp(c^T x) \exp(\log(1 + \exp(b_1 + w_1 x))) \dots \exp(\log(1 + \exp(b_H + w_H x)))/Z \\ &= \exp(c^T x + \sum_{j=1}^H \log(1 + \exp(b_j + w_j x)))/Z \\ p(x) &= \exp(c^T x + \sum_{j=1}^H \text{softplus}(b_j + w_j x))/Z \end{aligned}$$

From the above equation we can see, $p(x)$ is directly correlated to Network parameters namely W, b, c . Our aim is to find those set of W, b, c which can maximize $p(x)$.

2.1.4 Energy Minimization training of RBM using Contrastive Divergence

Let the problem be treated as empirical minimization problem, without Regularization. The loss function is defined in the following way [5]:

$$\frac{1}{T} \sum l(f(x)^{(t)}) = \frac{1}{T} \sum_t -\log(p(x^{(t)}))$$

where t is from 1 to T , set of all inputs. We would like to proceed by using Stochastic gradient descent

$$\frac{\partial -\log p(x^t)}{\partial \theta} = E_h \left[\frac{\partial E(x^{(t)}, h)}{\partial \theta} \mid x^{(t)} \right] - E_{x,h} \left[\frac{\partial E(x, h)}{\partial \theta} \right]$$

The first term in the above equation is called +ve phase and the second term is -ve phase. The +ve phase is expectation over hidden of the partial derivative of the energy w.r.t θ given the input observation. The second term is expectation over x, h of our model, which is generally intractable. Instead of calculating the second term we approximate the value of the second term by using a technique called contrastive divergence.

Contrastive Divergence:

1. Replace the expectation by a point estimate at \tilde{x}
2. Obtain \tilde{x} by Gibb's sampling
3. Start sampling chain at $x^{(t)}$

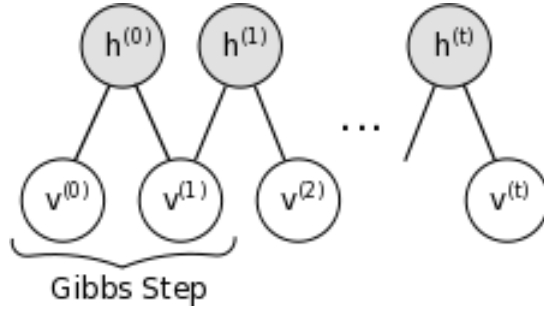


Figure 2.3: Gibb's Sampling

Perform the Gibbs sample for K -steps

Training Algorithm :

Calculating Derivative of

$$\frac{\partial E(x, h)}{\partial \theta}$$

Calculating this for $\theta = W_{jk}$, the same follows when $\theta = b_j, c_k$

$$\begin{aligned} \frac{\partial E(x, h)}{\partial W_{jk}} &= \frac{\partial}{\partial W_{jk}} \left(- \sum_{jk} W_{jk} h_j x_k - \sum_k c_k x_k - \sum_j b_j h_j \right) \\ &= - \frac{\partial}{\partial W_{jk}} \sum_{jk} W_{jk} h_j x_k \\ \nabla_W E(x, h) &= -h x^T \end{aligned}$$

Derivative of $E_h \left[\frac{\partial E(x, h)}{\partial \theta} \mid x \right]$ for $\theta = W_{jk}$

$$\begin{aligned} E_h \left[\frac{\partial E(x, h)}{\partial W_{jk}} \mid x \right] &= E_h[-h_j x_k \mid x] \\ &= \sum_{h_j \in \{0,1\}} -h_j x_k p(h_j \mid x) \end{aligned}$$

$$\begin{aligned}
&= -x_k p(h_j = 1 | x) \\
&= h(X)X^T
\end{aligned}$$

where $h(X)$ is defined as

$$\begin{aligned}
&\begin{pmatrix} p(h_1 = 1 | x) \\ \vdots \\ p(h_H = 1 | x) \end{pmatrix} \\
&= \text{sigm}(b + Wx)
\end{aligned}$$

2.1.5 Update Rule

Given $x^{(t)}$ and \tilde{x} the learning rule with $\theta = W$ becomes

$$W \Leftarrow W - \alpha(\nabla W)$$

$$W \Leftarrow W - \alpha(E_h[\nabla_W E(x^{(t)}, h) | x^{(t)}] - E_{x,h}[\nabla_w E(x, h)])$$

$$W \Leftarrow W - \alpha(E_h[\nabla_W E(x^{(t)}, h) | x^{(t)}] - E_h[\nabla_w E(\tilde{x}, h) | \tilde{x}])$$

$$W \Leftarrow W + \alpha(h(x^{(t)})x^{(t)T} - h(\tilde{x})\tilde{x}^T)$$

2.1.6 CD Algorithm

1. For each training example $x^{(t)}$

- (i) Generate a negative sample \tilde{x} using K-steps of gibbs sampling, starting at $x^{(t)}$
- (ii) Update the parameters,

$$W \Leftarrow W + \alpha(h(x^{(t)})x^{(t)T} - h(\tilde{x})\tilde{x}^T)$$

$$b \Leftarrow b + \alpha(h(x^{(t)}) - h(\tilde{x}))$$

$$c \Leftarrow c + \alpha(x^{(t)} - \tilde{x})$$

2. Go back to step 1, until stopping criteria

2.1.7 Gaussian Bernoulli RBM

We have seen how RBMs deal with Binary type of data. But the real world is typically contains all real valued data. To model real(Un bounded) valued data we use Gaussian Bernoulli RBM(GBRBM). The Energy function will get modified as

$$E(x, h) = -h^T Wx - c^T x - b^T h + \frac{1}{2}x^T x$$

The only change from normal RBM to GBRBM comes while evaluation $p(x | h)$ which is now a Gaussian distribution with mean $\mu = c + W^T h$ and identity co-variance matrix. Giving normalized data as input is recommended which involves subtracting each data by mean and dividing by

standard deviation.

2.1.7.1 Convolutional RBM (CRBM)

CRBM's exploit spatially local correlation by enforcing a local connectivity pattern between neurons of adjacent layers. Different convolution operators are used to detect different types of features. The idea of CRBMs is derived from Convolutional Neural network (CNN), where there are alternating convolutional and subsampling layers. The architecture of CRBM is as below [6], CRBM consists

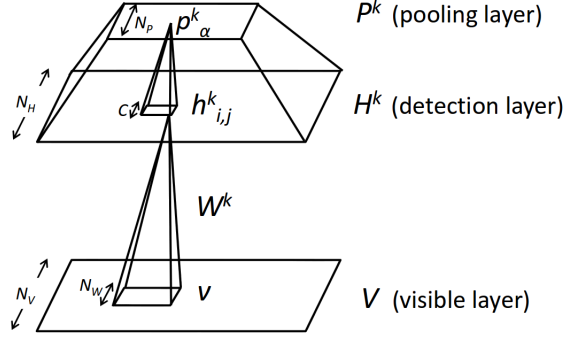


Figure 2.4: Convolutional Restricted Boltzmann Machine

of an input layer (V) and a hidden layer (H). Input layer is of size $N_v \times N_v$ and the k-hidden layers each of size $N_h \times N_h$, each Hidden layer is associated with visible by a kernel of size N_w where $N_w = N_v - N_h + 1$. The energy for a CRBM is defined in the following way:

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

where

$$A \bullet B = \text{tr} A^T B$$

and $*$ represents convolution of an $m \times m$ array with an $n \times n$ array may result in an $(m + n - 1) \times (m + n - 1)$ array or an $(m - n + 1) \times (m - n + 1)$ array

As with standard RBMs (Section 2.1.4), we can perform block Gibbs sampling using the following conditional distributions:

$$p(h_{ij} = 1|v) = \text{sigm}((\tilde{W}^k * v)_{ij} + b_k)$$

$$p(v_{ij} = 1|h) = \text{sigm}((\sum_k W^k * h^k)_{ij} + c)$$

2.1.7.2 Gated RBM

Gated RBMs are used to model the stream of observations through the use of hidden variables which capture the dependencies between adjacent data in a stream. Given below is the architecture of a Gated RBM [7]. A Gated RBM consists of an input layer X, an output layer Y and a hidden layer H. The dependencies among these three layers are modelled by using a three dimensional tensor W_{ijk}

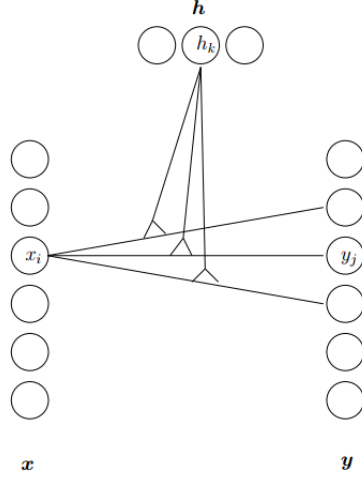


Figure 2.5: Gated RBM

The energy function that is used to capture the dependencies between input, output and hidden is given by $E(y, h; x) = -\sum_{ijk} W_{ijk} x_i y_j h_k$

The joint distribution $p(h, y | x)$ is defined from $E(y, h; x)$ as

$$p(h, y | x) = \frac{1}{Z(x)} \exp(-E(y, h; x))$$

where

$$Z(x) = \sum_{y, h} \exp(-E(y, h; x))$$

The distribution over output given input is obtained by marginalizing over hidden as: $p(y | x) = \sum_h p(y, h | x)$ In gated RBM, only the hidden and output units are inferred for learning, and the inputs are always kept constant. The sampling of hidden units is done using the following expression:

$$p(h_k = 1 | x, y) = \frac{1}{1 + \exp(-\sum_{ij} W_{ijk} x_i y_j)}$$

Similarly the output units are sampled using:

$$p(y_j = 1 | x, h) = \frac{1}{1 + \exp(-\sum_{ik} W_{ijk} x_i h_k)}$$

Learning occurs in Gated RBM by maximizing the conditional log likelihood L , where,

$$L = \frac{1}{N} \sum_{\alpha} \log p(y^{\alpha} | x^{\alpha})$$

for all N training samples (x^{α}, y^{α}) The gradient of L is given by difference of two expectations as below:

$$\frac{\partial L}{\partial W} = \sum_{\alpha} \left\langle \frac{\partial E(y^{\alpha}, h; x^{\alpha})}{\partial W} \right\rangle_h - \left\langle \frac{\partial E(y, h; x^{\alpha})}{\partial W} \right\rangle_{y, h}$$

2.1.7.3 Factored RBM

A factored RBM is simply an extension of Gated RBM. Gated RBMs are used to model the pairwise interactions between input and output using a three-dimensional tensor W_{ijk} . But this tensor has too many parameters to be learned. Hence, the W_{ijk} in a Gated RBM can be approximated by factoring W_{ijk} as W_{if}, W_{jf}, W_{kf} where f is the number of factors [8]. This factoring reduces the number of parameters from $O(n^3)$ to $O(n^2)$ if the size of f is comparable to other units. Given below is the architecture of a Factored RBM [9],

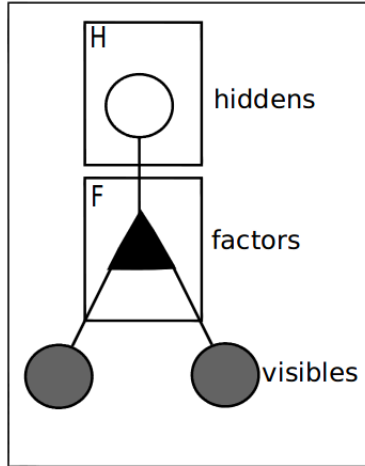


Figure 2.6: Factored RBM : The triangle in the middle represents factors

The energy equation of Factored RBM becomes : $-E(v, h) = \sum_f (\sum_i v_i W_{if}) (\sum_j v_j W_{jf}) (\sum_k h_k W_{kf})$
 The learning happens just like in gated RBM by maximizing the log likelihood, with gradient calculated as

$$\frac{\partial L}{\partial w} = \langle \frac{\partial E}{\partial w} \rangle_h - \langle \frac{\partial E}{\partial w} \rangle_{y,h}$$

2.2 Related Work: Deep Learning for Action Recognition

This section explains various methods that are used in deep learning methods for action recognition in videos. We will discuss two main approaches, viz. 3D Convolutional Neural Networks (CNNs) approach and Convolutional Gated RBM (CGRBM) approach for action recognition. There are many other approaches proposed for the action recognition problem that are not based on deep learning, but they are not discussed here as our focus is on the possibility of infusing the concept of temporal coherence in deep learning architectures.

2.2.1 3D CNN for Action Recognition

2D CNNs are capable of only modelling the spatial features in a given input data. But a video comprises of temporal information which plays a crucial role in detecting actions in a video. To model this, 2D CNNs are extended as 3D CNNs with temporal data as the third dimension. The figure below explains how 3D convolution is performed. To summarize, 3D convolution is performed

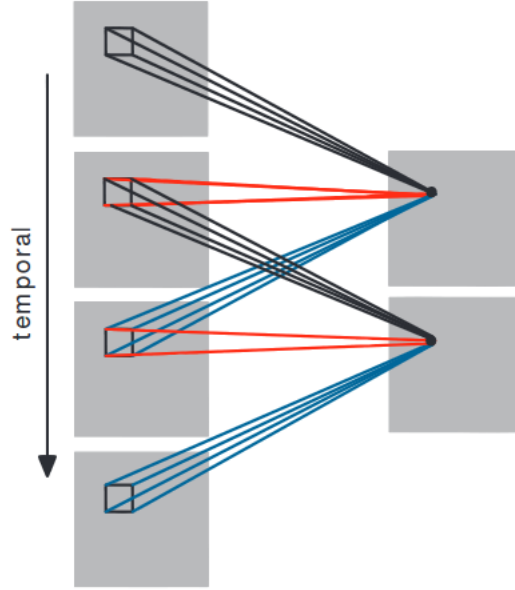


Figure 2.7: 3D convolution

by using a convolutional kernel of three dimensions, which convolve consecutive images from a given video. The value at position (x, y, z) on the j^{th} feature map in the i^{th} layer is defined as:

$$v_{ij}^{xyz} = \tanh(b_{ij} + \sum_m \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} \sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(z+r)})$$

where R_i is the size of the 3D kernel along the temporal dimension. Figure 2.8 shows an example of 3D CNN [10]

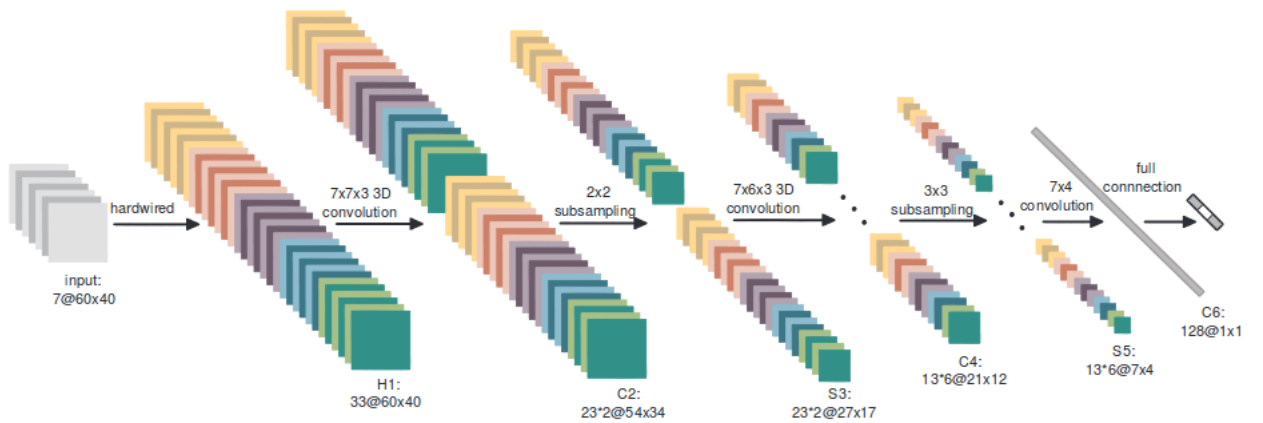


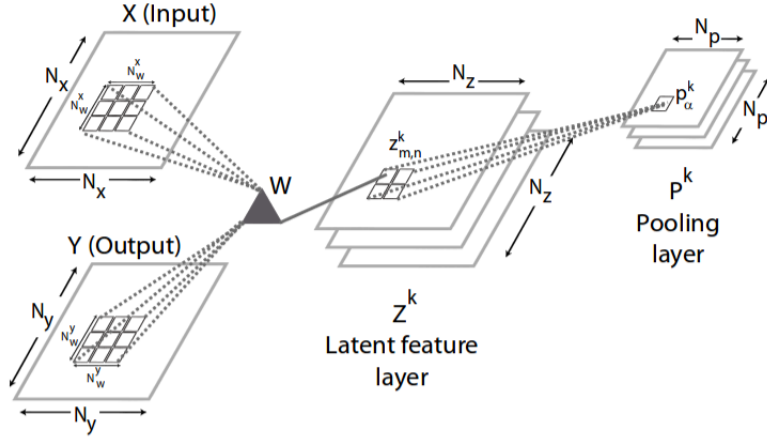
Figure 2.8: 3D convolution

The input to 3D CNN is 7 images of size 60x40, which are hardwired to give 33 images of same size in Layer H1. A 3D kernel of shape 7x7x3 is applied on all these images resulting in 23*2@54x34 images in layer named C2. A kernel of size 7x7 is then applied on each image and by considering

three images in the temporal direction. The layer C2 undergoes subsampling, resulting in reducing the size of image by half in both axis resulting in $23*2@27*17$ in layer S3. This alternating of 3D convolution and subsampling continues for layers C4 and S5, and a 2D convolution is finally applied between layers S5 and C6 resulting in scalar quantities which are fully connected to output neurons. The number of output neurons is equal to the number of actions in our data set used. This 3D CNN gave state-of-the-art performance on one of the most difficult data sets for action recognition, viz. the TRECVID dataset.

2.2.2 Convolutional Gated RBM (CGRBM) for Action Recognition

This approach in deep learning is a combination of RBMs and 3D CNNs for action recognition in videos. As the gated RBM suffers from the problem of learning too many parameters i.e of $O(n^3)$, this approach uses the advantage of weight-sharing through convolution. The architecture of a Convolutional Gated RBM is shown in Figure 2.2.2. The Gated Factored RBM is similar to the Gated



RBM, except for the change that the three way interaction between input, output and hidden are performed using convolution operations. The input and output images are of size $N_x \times N_x$ and $N_y \times N_y$ respectively. The number of feature maps are K , resulting in $KN_z \times N_z$ feature maps.

The energy of a CGRBM is defined as [11]:

$$E(y, z; x) = - \sum_{k=1}^K \sum_{m,n=1}^{N_z} \sum_{r,s=1}^{N_y} z_{m,n}^k \gamma(x)_{r,s,m,n}^k y_{m+r-1,n+s-1} - \sum_{k=1}^K b_k \sum_{m,n=1}^{N_z} z_{m,n}^k - c \sum_{i,j=1}^{N_y} y_{i,j}$$

The filter $\gamma(x)_{r,s,m,n}^k = \sum_{u,v}^{N_w^x} W_{r,s,u,v}^k x_{m+u-1,n+v-1}$ The probability of hidden units, output being active is given by:

$$p(z_{m,n}^k = 1 | x, y) = \text{sigm}\left(\sum_{r,s=1}^{N_w^y} \gamma(x)_{r,s,m,n}^k y_{m+r-1,n+s-1}\right) + b_k$$

$$p(y_{i,j} = 1 | x, h) = \text{sigm}\left(\sum_{k=1}^K \sum_{r,s=1}^{N_w^y} \tilde{\gamma}(x)_{r',s',i+r-1,j+s-1} z_{i+r-1,j+s-1}^k + c\right)$$

where $r' = N_w^y - r + 1$ and $s' = N_w^y - s + 1$ and also flips the matrix in horizontal and vertical direction.

In the next chapter, we describe the proposed method to incorporate temporal coherence in videos, building on top of the Convolutional Gated RBM.

Chapter 3

Temporal Coherence in Energy-Based Deep Models: Proposed Methodology

3.1 Modelling temporal sequences of videos

The CGRBM models the joint energy of two input images. In a given video, two consecutive frames in a video will be very similar. We can use this property to model the energy curve modeled by the CGRBM in such a way that the similar images will be at lower energy states and we intentionally pull up the joint energy for dissimilar images.

The overall loss function for CGRBM is:

$$\mathcal{L}(\theta) = -\frac{1}{N}E(y, z; x)$$

and the update rule is defined as:

$$\theta = \theta - \lambda \frac{\partial L(\theta, x, y, z)}{\partial \theta}$$

, where λ is the learning rate.

Our proposed approach works in the following manner. When the CRBM is given inputs as current data X and past data Y , where X, y are two input images, the CGRBM models the joint energy of these two images and updates its weights so as to reduce its energy. To leverage the property of temporal coherence in modelling the joint energy of consecutive images in CGRBM, the loss function is redefined as:

$$\mathcal{L}_{coh}(x, y, \theta) = \begin{cases} \operatorname{argmin}_{\theta} E(y, z; x), & \text{similar} \\ \operatorname{argmax}_{\theta} E(y, z; x), & \text{otherwise} \end{cases}$$

The proposed learning algorithm for CGRBM that is based on the above loss function is given be-

Data: Image pairs $(X_i, Y_i, type_i)$, $i=1..n$
Result: Set of Weights which minimizes the joint energy for Consecutive image pairs and maximizes for other
Initialize Weights and biases of the network to Uniform random numbers;
while *epochs* **do**
 read an image pair x_i, y_i ;
low. **if** $type_i == 'similar'$ **then**
 Do Contrastive Divergence to decrease the joint energy of $E(y,z;x)$;
 else
 Do Contrastive Divergence to increase the joint energy of $E(y,z;x)$;
 end
end
Algorithm 1: Proposed Learning Algorithm for Deep Energy Based Models with Temporal Coherence

We use Gibbs sampling to perform Contrastive Divergence to leverage the property of conditional independence of nodes in a given layer.

Data: Image pair $x_i, y_i, type_i$
Result: Updated Weights of the network
Sample the hidden units(Z) conditional on Past(X),Data(Y);
Sample the data units(Y) conditional on Hidden,Past(X); /* End of positive phase */
Compute the positive grads of W,hidbias,visbias;
while *number of Gibb's iterations* **do**
 Sample the hidden units(Z) conditional on Past(X),Data(Y);
 Sample the data units(Y) conditional on Hidden(Z) ; /* End of Negative Phase */
end
Compute the negative grads of W,hidbias,visbias;
if $type_i == 'similar'$ **then**
 $gradW = posgradW - neggradW$;
else
 $gradW = -posgradW + neggradW$;
end

Algorithm 2: Contrastive Divergence Algorithm for the Proposed Method

The proposed method above framework can also be generalized to other coherence settings, beyond temporal coherence. The same framework can be used to any type of data with similarity defined as per the data. For video data, we have taken past frames as temporally coherent, and frames from other actions as non-coherent, but this can be easily changed as evident.

Chapter 4

Experimental Results

4.1 Experimental Setup

We had experimented on the following setup of CGRBM. While modelling the NORB dataset, the input size is resized to 22x22 and filters of size 9x9x9x9 are used with number of feature maps as 1. The temporally coherent frame for NORB is defined as an image that is rotated 20 degree. In the initial epochs the ratio is similar to dis similar frames is kept at 2:1 and after few epochs its kept at 1:1.

For KTH the input size of an image is kept at 80x60 and the number of filters are kept at 9x9x9x9. A subsampling of 4x4 is used for probabilistic max-pooling. The ratio of similar to dis similar is kept at 9:1. A total of 76 video clips from each action is given as training set and the rest 24 from each action are given as test set.

4.2 Experimental Results on NORB Data set

The modified algorithm is applied on small NORB data set [12].The database consists of objects from five different categories: animals, human figures, airplanes, trucks, and cars. These objects are viewed under different lightening, elevations and Azimuth settings. We have considered objects differing by Azimuth and trained a convolutional Gated RBM and the CGRBM learns the transformations between the images just like optical flow.

The quiver plots of the CGRBM transformations looks similar to Optical flow between the two images that differ by certain Azimuth. A sample car objects are taken and given to CRBM.

Figure 4.2 is the transformation the CGRBM has learnt,

The norb data set size is down sampled to 32x32 size and a filter of size 9x9 is used to do convolution operations. A pooling of size 2x2 is used at probabilistic max pooling. The ratio of dis similar to similar is given as 1:3. If the ration is given as 1:1 , there is a danger of disturbing the entire energy curve with many ridges and valleys to the energy curve.

The ratio can be increased after few epochs of training as the shape of the curve stabilizes after few iterations



Figure 4.1: Input to CGRBM

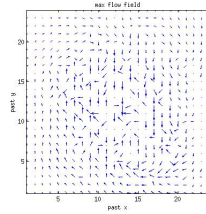


Figure 4.2: Transformations learnt by CGRBM

4.3 Experimental Results on KTH Data set

KTH is an Human Action Recognition data set [13]. It contains action of six different types performed by 25 subjects in four scenarios.

The following is the confusion matrix for KTH dataset. The ratio of train to test is 3:1

Action	Walking	Jogging	Running	Boxing	Waving	Clapping
Walking	21	0	1	1	0	2
Jogging	0	20	4	0	0	0
Running	0	3	21	0	0	0
Boxing	0	0	0	21	2	1
Waving	0	0	0	1	21	2
Clapping	0	0	0	1	3	20

An accuracy of 85.52% is achieved using the KTH dataset.

The CGRBM is trained on KTH data set and the following are the hidden inferences obtained after training the CGRBM for one full epoch.

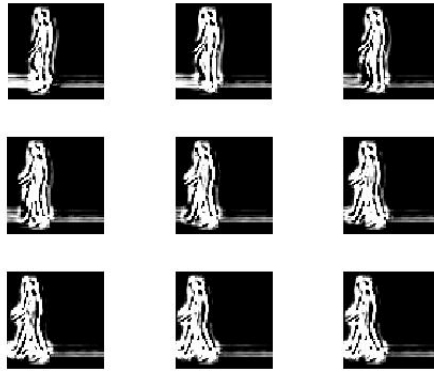


Figure 4.3: Hidden Inferences of action walking

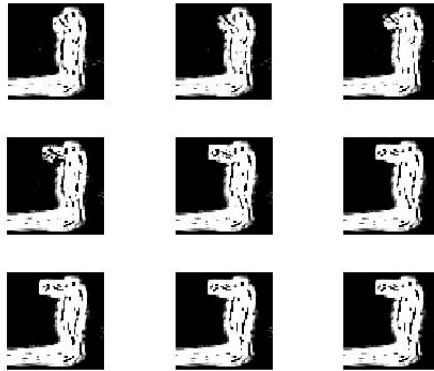


Figure 4.4: Hidden Inferences of action boxing

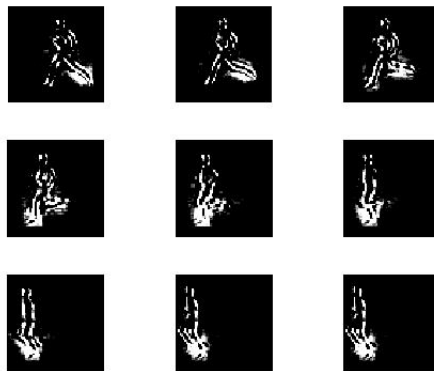


Figure 4.5: Hidden Inferences of action running

Chapter 5

Conclusions and Future Work

5.1 Where does our approach work and why?

Our approach of modified learning algorithm works for data which has a temporal coherence embedded in it. Several types of data like stock market prices, weather ratings, speech data, video data, etc where the current instance of data is dependent on the previous instances can be modelled by our approach. Our approach would be of great interest to those type of data that has temporal coherence embedded into it as the CGRBM models the transformation in a given data as the learning itself is happening considering the previous data into account.

5.2 Directions for Future Work

This work can be extended to include more past data while modelling the current instance of data. This can help us to model the data much better.

5.3 Conclusions

A novel idea of inclusion of similar and dissimilar data during training is introduced and the learning algorithm is modified so that it could better model the Joint energy curve. The proposed learning approach is applied on NORB data set and KTH data set and the results are studied and analysed.

References

- [1] L. Deng. A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing* 3.
- [2] Y. Lecun, S. Chopra, R. Hadsell, F. J. Huang, G. Bakir, T. Hofman, B. Schlkopf, A. Smola, and B. T. (eds. A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press, 2006 .
- [3] J. Aggarwal and M. Ryoo. Human Activity Analysis: A Review. *ACM Comput. Surv.* 43, (2011) 16:1–16:43.
- [4] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-Based Learning Applied to Document Recognition. In *Proceedings of the IEEE*, volume 86. 1998 2278–2324.
- [5] G. E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation* 14, (2002) 1771–1800.
- [6] 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 International Conference on Machine Learning*. 2009 609–616.
- [7] R. Memisevic and G. Hinton. Unsupervised Learning of Image Transformations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 2007 .
- [8] M. A. Ranzato, A. Krizhevsky, and G. E. Hinton. Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images 2010.
- [9] M. Ranzato, A. Krizhevsky, and G. E. Hinton. Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images. *Journal of Machine Learning Research - Proceedings Track* 9, (2010) 621–628.
- [10] S. Ji, W. Xu, M. Yang, and K. Yu. 3D Convolutional Neural Networks for Human Action Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, (2013) 221–231.
- [11] G. W. Taylor, R. Fergus, Y. Lecun, and C. Bregler. Convolutional Learning of Spatio-temporal Features.
- [12] Y. LeCun, F. J. Huang, and L. Bottou. Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR'04*. IEEE Computer Society, Washington, DC, USA, 2004 97–104.

- [13] C. Schudt, I. Laptev, and B. Caputo. Recognizing Human Actions: A Local SVM Approach. In Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 3 - Volume 03, ICPR '04. IEEE Computer Society, Washington, DC, USA, 2004 32–36.