Classification using Discriminative Restricted Boltzmann Machines

Hugo Larochelle and Yoshua Bengio

Presented by: Bhargav Mangipudi

December 2, 2016

Outline

Restricted Boltzmann Machines

- Restricted Boltzmann Machines are generative stochastic models that can model a probability distribution over its set of inputs using a set of hidden (or latent) units.

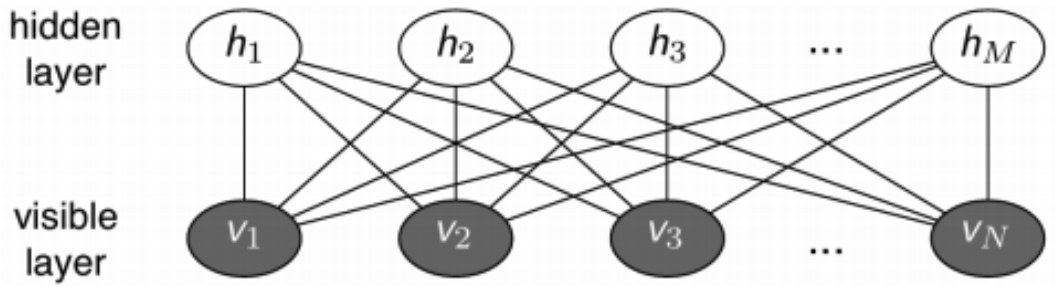


Figure 1: Restricted Boltzmann Machine

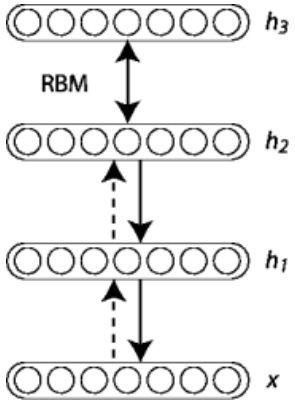
- They are represented as a bi-partitie graphical model where the *visible* layer is the observed data and the *hidden* layer models latent features.

RBM as feature extractors

- RBMs were predominantly used in an unsupervised setting to extract better *features* from the input data.
- By trying to reconstruct the input vector, we can make the RBM learn a different representation of the input data. This representation is then used in a higher classification algorithm.
- By using features extracted from RBM and using them in an upstream task, we introduce large number of hyper-parameters: Parameters for the RBM/DBNs + Parameters from the upstream classification procedure.

Using RBM for NN/DBN initialization

- RBMs have been used as a *building block* in Deep Belief Networks.



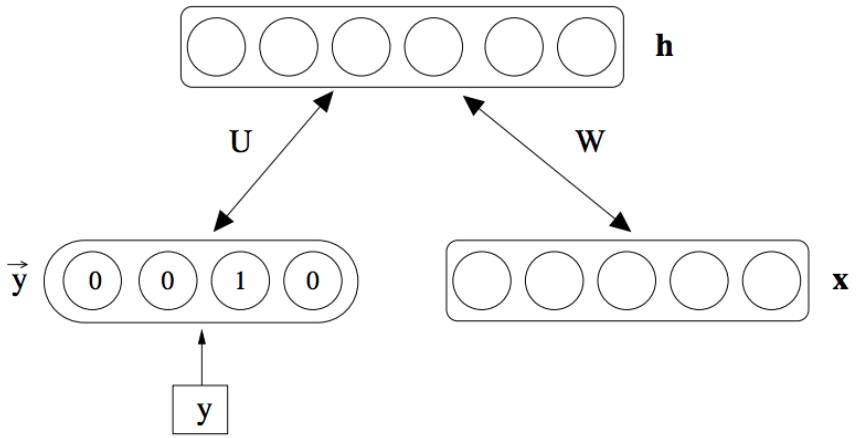
- Pre-training: The stacked RBMs are trained in a layer-by-layer procedure in a **greedy** manner by fixing the previously computed layer weights.
- Discriminative fine-tuning: When used for a classification task, fine-tuning of the weights is performed by adding a final layer of output nodes and propagating the gradients of the loss.
- This was one of the earliest methods to tractably train deep neural network architectures.

Main Ideas

- RBMs can be used as a self-contained framework for non-linear classification tasks.
- Introduces an objective function to model discriminative training of RBMs.
- Extension to support semi-supervised learning setting.
- Experimentation to show competitive results with other techniques.

Modeling Classification using RBMs

- Assume that the training set is represented as $\mathcal{D}_{train} = \{(x_i, y_i)\}$ where x_i represents the input vector for the *i-th* example and $y_i \in \{1, \ldots C\}$ is the corresponding target class.
- The labels y_i is encoded as a one-hot representation ($\vec{y}$) over the set of C classes and treated as observed variables.



- Thus, we have $\mathcal{P}(y, x, h) \propto e^{-E(y,x,h)}$
  where $E(y, x, h) = -h^T W x - b^T x - c^T h - d^T \vec{y} - h^T U \vec{y}$
  with parameters $\mathbf{\Theta} = (\mathbf{W}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{U})$.

Modeling Classification using RBMs - Continued

Thus, we see the following properties based on the structure and modeling

$$\mathcal{P}(x|h) = \prod_i \mathcal{P}(x_i|h)$$

$$\mathcal{P}(x_i = 1|h) = \text{sigm}(b_i + \sum_j W_{j,i} h_j)$$

$$\mathcal{P}(y|h) = \frac{e^{d_y + \sum_j U_{j,y} h_j}}{\sum_{y*} e^{d_{y*} + \sum_j U_{j,y*} h_j}}$$

Similarly for the hidden units, we get:

$$\mathcal{P}(h|y,x) = \prod_j \mathcal{P}(h_j|y,x)$$

$$\mathcal{P}(h_j = 1|y,x) = \text{sigm}(c_j + U_{j,y} + \sum_i W_{j,i} x_i)$$

Training RBMs

- To train a generative model on the training set, we minimize the negative log-likelihood:

$$\mathcal{L}_{gen}(\mathcal{D}_{train}) = -\sum_{i=1}^{|\mathcal{D}_{train}|} log\ \mathcal{P}(y_i, x_i)$$

- Calculating the exact gradient of $log\ \mathcal{P}(y_i, x_i)$ for any parameter $\theta \in \Theta$, we get:

$$\frac{\partial log\ \mathcal{P}(y_i, x_i)}{\partial \theta} = -\mathbb{E}_{\mathbf{h}|\mathbf{y_i},\mathbf{x_i}}\left[\frac{\partial}{\partial \theta}\mathbf{E}(\mathbf{y_i}, \mathbf{x_i}, \mathbf{h})\right] + \mathbb{E}_{\mathbf{y},\mathbf{x},\mathbf{h}}\left[\frac{\partial}{\partial \theta}\mathbf{E}(\mathbf{y}, \mathbf{x}, \mathbf{h})\right]$$

- Gradient can be estimated using *Contrastive Divergence*

Training RBMs - Continued

- After learning the model's parameters iteratively by stochastic gradient descent with a suitable learning rate, we can perform classification by estimating:

$$p(y|x) = \frac{e^{d_y} \prod_{j=1}^{n} (1 + e^{(c_j + U_{j,y} + \sum_i W_{j,i} x_i)})}{\sum_{y*} e^{d_{y*}} \prod_{j=1}^{n} (1 + e^{(c_j + U_{j,y*} + \sum_i W_{j,i} x_i)})}$$

- While training in an unsupervised setting, the features representations learned by the hidden layer is not guaranteed to be useful for the classification task.

Discriminative Restricted Boltzmann Machines

- For the discriminative setting, we directly optimize $\mathcal{P}(y|x)$ instead of the joint-distribution for $p(y, x)$. We get the following log-likelihood expression:

$$\mathcal{L}_{disc}(\mathcal{D}_{train}) = -\sum_{i=1}^{|\mathcal{D}_{train}|} log\mathcal{P}(y_i|x_i)$$

- Training Procedure:

$$\frac{\partial log\mathcal{P}(y_i|x_i)}{\partial\theta} = \sum_j \text{sigm}(o_{y,j}(x_i))\frac{\partial o_{y,j}(x_i)}{\partial\theta} - \sum_{j,y*} \text{sigm}(o_{y*,j}(x_i))\mathcal{P}(y^*|x_i)\frac{\partial o_{y*,j}(x_i)}{\partial\theta}$$

$$\text{where } o_{y,j}(x) = c_j + \sum_k W_{j,k}x_k + U_{j,y}$$

- This gradient can be computed efficiently using stochastic gradient descent optimization.

Hybrid Discriminative Restricted Boltzmann Machines

- Observation: Smaller training sets tend to favor generative learning and bigger training sets favor discriminative learning.
- We adopt a *hybrid* generative/discriminative approach by combining the respective objective terms.

$$\mathcal{L}_{hybrid}(\mathcal{D}_{train}) = \mathcal{L}_{disc}(\mathcal{D}_{train}) + \alpha\mathcal{L}_{gen}(\mathcal{D}_{train})$$

- α is a hyper-parameter controlling the generative criterion.
- $\mathcal{L}_{disc}(\mathcal{D}_{train})$ and $\mathcal{L}_{gen}(\mathcal{D}_{train})$ are calculated using the techniques described above.

Extension for Semi-Supervised Learning

- Motivation: Labeled/supervised data is not readily available.
- Assume $\mathcal{D}_{unlab}$ represents the set of unlabelled data.

$$\mathcal{L}_{unsup}(\mathcal{D}_{unlab}) = -\sum_{i=1}^{|\mathcal{D}_{unlab}|} log\mathcal{P}(x_i)$$

- Training:

$$\frac{\partial log\ \mathcal{P}(x_i)}{\partial\theta} = -\mathbb{E}_{\mathbf{h},\mathbf{y_i}|\mathbf{x_i}}\left[\frac{\partial}{\partial\theta}\mathbf{E}(\mathbf{y_i},\mathbf{x_i},\mathbf{h})\right] + \mathbb{E}_{\mathbf{y},\mathbf{x},\mathbf{h}}\left[\frac{\partial}{\partial\theta}\mathbf{E}(\mathbf{y},\mathbf{x},\mathbf{h})\right]$$

- The first term is different from the $\mathcal{L}_{gen}$ gradient estimation. This can efficiently calculated by taking an expectation w.r.t $\mathcal{P}(y|x_)$ of the term from LL_{gen} or sampling for a single y_i and doing a point-estimate.

Extension for Semi-Supervised Learning - Continued

- Semi-supervised setting:

$$\mathcal{L}_{semi-sup}(\mathcal{D}_{train}, \mathcal{D}_{unlab}) = \mathcal{L}_{TYPE}(\mathcal{D}_{train}) + \beta \mathcal{L}_{unsup}(\mathcal{D}_{unlab})$$

  where $TYPE \in \{gen, disc, hybrid\}$

- Thus, online training by stochastic gradient descent corresponds to applying two gradient updates: one for the objective $\mathcal{L}_{TYPE}$ and the other one for the unlabelled data objective $\mathcal{L}_{unsup}$.

Character (Digit) Recognition - MNIST Dataset

- MNIST Dataset for classifying images of digits.

Model	Error
RBM ($\lambda = 0.005$, $n = 6000$)	3.39%
DRBM ($\lambda = 0.05$, $n = 500$)	1.81%
RBM+NNet	1.41%
HDRBM ($\alpha = 0.01$, $\lambda = 0.05$, $n = 1500$)	1.28%
Sparse HDRBM (idem + $n = 3000$, $\delta = 10^{-4}$)	**1.16%**
SVM	1.40%
NNet	1.93%

- $\lambda =$ learning rate and $n =$ number of training iterations.
- Likelihood objective for HDRBM (after hyper-parameter tuning):

$$\mathcal{L}_{hybrid}(\mathcal{D}_{train}) = \mathcal{L}_{disc}(\mathcal{D}_{train}) + 0.01\mathcal{L}_{gen}(\mathcal{D}_{train})$$

- Sparse HDRBM - Small value δ is subtracted from the biases $\mathbf{c}$ in the hidden layer after each update.

Document Classification - 20-newsgroup Dataset

- Task: Classify newsgroup documents into 20 target classes.
- Features used: **5000** most common works across the data set as binary input features.

Model	Error
RBM ($\lambda = 0.0005$, $n = 1000$)	24.9%
DRBM ($\lambda = 0.0005$, $n = 50$)	27.6%
RBM+NNet	26.8%
HDRBM ($\alpha = 0.005$, $\lambda = 0.1$, $n = 1000$)	**23.8%**
SVM	32.8%
NNet	28.2%

Document Classification - 20-newsgroup Dataset - HDRBM analysis

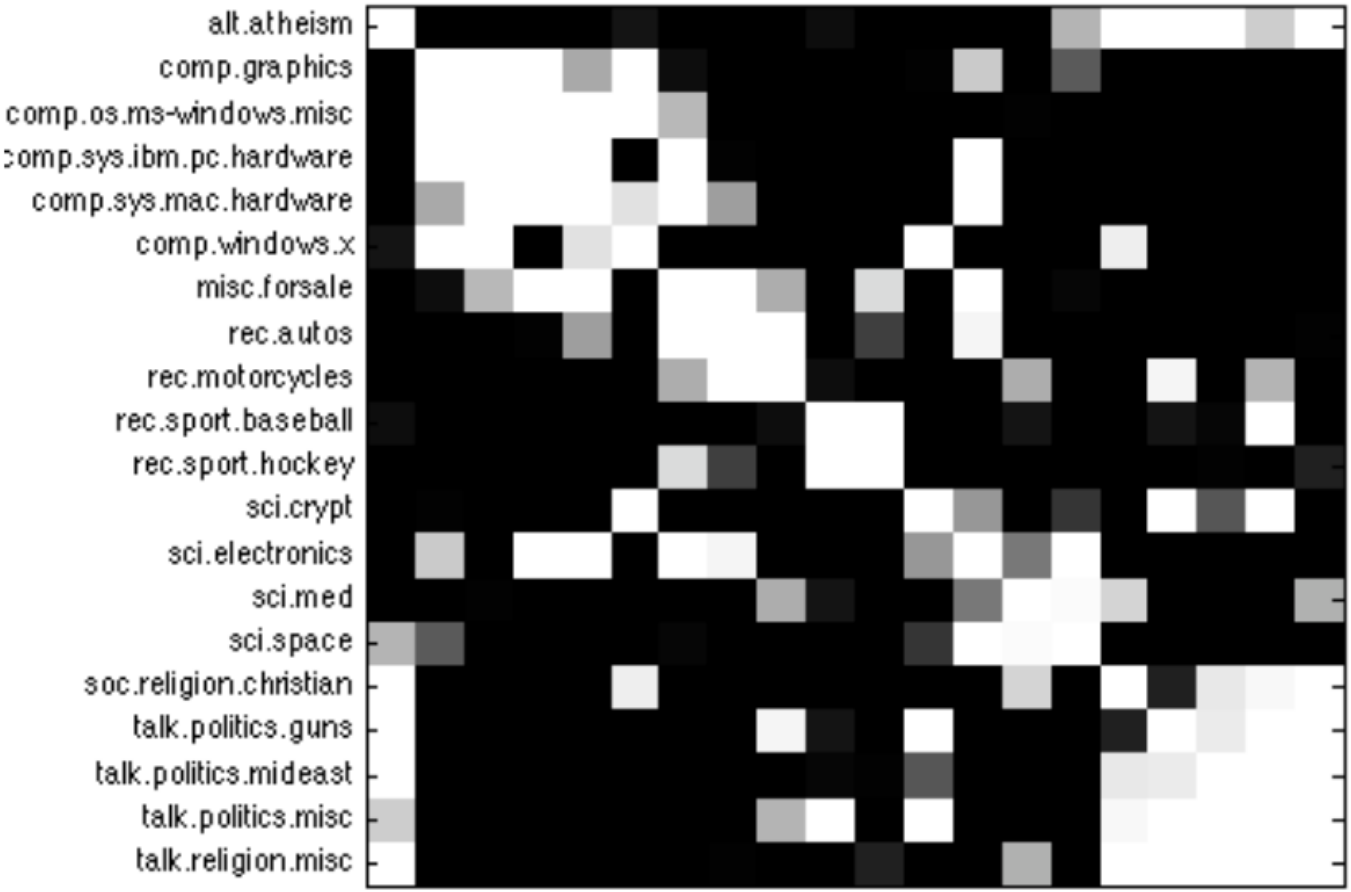


Figure 2: Similarity Matrix for weights $U_{\cdot,y_j}$

Semi-supervised learning results

- Compare the semi-supervised setting against other non-parametrized semi-supervised learning algorithms.

Model	MNIST	MNIST-BI	20-news
HDRBM	9.73%	42.4%	40.5%
Semi-sup HDRBM	8.04%	**37.5%**	**31.8%**
NP-Gauss	10.60%	66.5%	85.0%
NP-Trunc-Gauss	**7.49%**	61.3%	82.6%

Figure 3: Percentage error across different datasets for semi-supervised learning

Further Work


Louradour, Jérôme and Larochelle, Hugo
Classification of Sets using Restricted Boltzmann Machines.
UAI 2011


Larochelle, Hugo and Mandel, Michael and Pascanu, Razvan and Bengio, Yoshua
Learning algorithms for the classification restricted boltzmann machine.
JMLR 2012


van der Maaten, Laurens
Discriminative restricted Boltzmann machines are universal approximators for discrete data.
Technical Report EWI-PRB TR 2011001, Delft University of Technology

Classification of Sets using RBMs

- In this scenario, each input instance x_i is represented by a set of vectors $x_i^{(0)}$, $x_i^{(1)} \ldots x_i^{(s)}$. For example: Each input instance could be different snippets of a document (mail) or different regions of an image. The is still modelled a multi-class classification task.
- One common approach to the problem is to use *multiple-instance learning (MIL)* model, where of the instance segments are individually classified assigned either the majority label or a conservative voting mechanism.
- The drawback with *MIL* is that it treats each instance segment equally and this implicit assumption might not always hold. If each segment can only provide *partial* class information, we won't capture that correctly in MIL. This paper tackles the problem by adding redundant hidden units (layers) to the RBM architecture.

ClassSetRBM - Mutually exclusive hidden units (XOR)

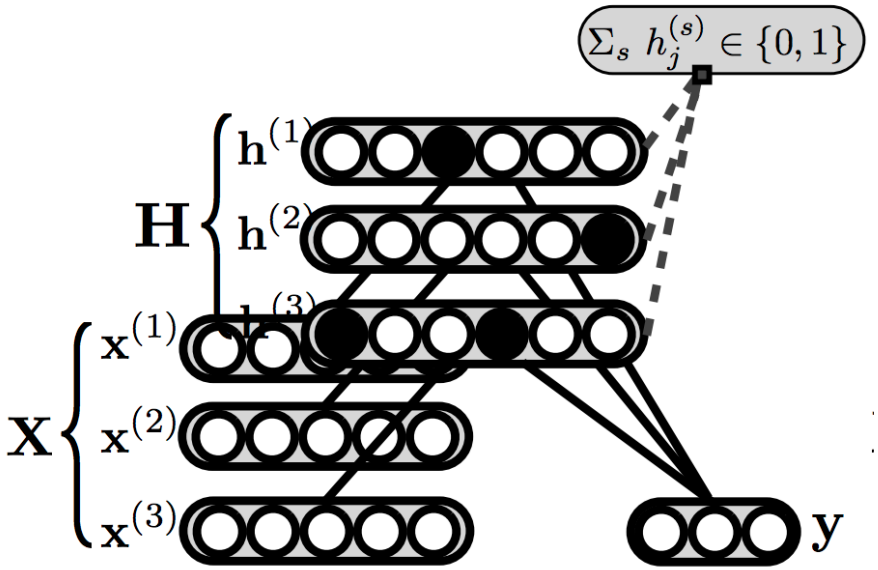


Figure 4: *ClassSetRBM*^{XOR} architecture

- X represents the set of instance segments for each data instance, i.e. $|X| =$ number of segments per training instance.

ClassSetRBM - Mutually exclusive hidden units (XOR) - Continued

- In this scenario, all hidden units are mutually exclusive across different segments, i.e.

$$\sum_{x=1}^{|X|} h_j^{(s)} \in \{0,1\} \;\; \forall j = 1 \ldots H$$

- Other distributions are slightly modified to accommodate the redundant hidden layers.

$$\mathcal{P}(y = y_i|X) = \frac{exp(-F^{XOR}(X, y_i))}{\sum_{y*} exp(-F^{XOR}(X, y*))}$$

where free energy $F^{XOR}(X, y) = -d^T y - \sum_{j=1}^{H} softplus(softmax_j(X) + U_{j,y}))$

and $softmax_j(X) = log(\sum_{s=1}^{|X|} exp(c_j + W_j x^{(s)}))$

ClassSetRBM - Redundant Evidence (OR)

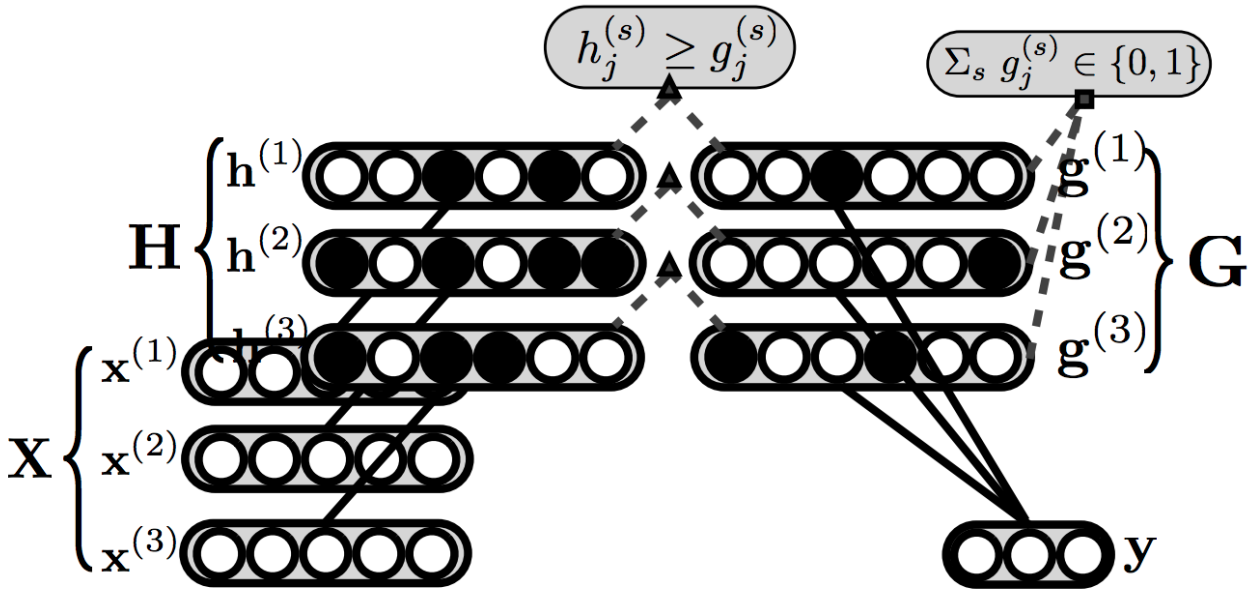


Figure 5: *ClassSetRBM*^{OR} architecture.

- In the paper, they argue that the assumption of mutual exclusivity over hidden unit may be too strong and try to relax it by adding additional "hidden" layer copies G connected only to y and by removing the direct connection from H to y.

Thank You!

Questions?