Classification with Restricted Boltzmann Machines

Projects in Machine Learning and AI

Fritjof Wolf
Katarzyna Tarnowska

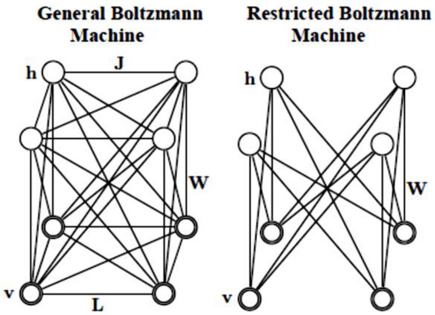
Technische Universitat Berlin

2015

Table of Contents

Boltzman Machine and Restricted Boltzmann Machine

- Recurrent neural network
- Hidden layer and visible layer
- Symmetric weights
- Stochastic binary neurons
- Generative Model
- In a Restricted Boltzmann Machine the joints between hidden units and also between visible units are disconnected

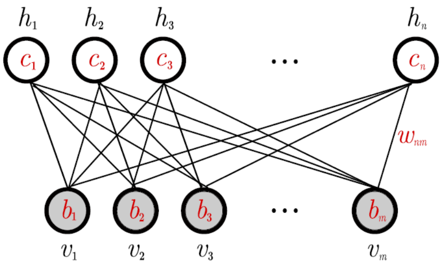


Boltzman Machine

- Energy function depends on model parameter
- Probability depends on weights and state of the other neurons
- Unsupervised learning
- Used to model probability distribution:
  - Apply random input
  - Run the model for some time to generate sample from learned distribution
- First used as an feature extractor

Restricted Boltzmann Machines

- Complete bipartite graph
- Stochastic neural network:
  - nodes - neurons
  - edges - synaptic connections



Source: A.Fischer, Ch.Igel: Training Restricted Boltzmann Machines: An Introduction

Mathematical description of the model

Energy function

$$\mathbf{E}(\mathbf{v},\mathbf{h}) = \sum_{i=1}^{V} \frac{(v_i - b_i^v)^2}{2\sigma_i^2} - \sum_{j=1}^{H} b_j^h h_j - \sum_{i=1}^{V}\sum_{j=1}^{H} \frac{v_i}{\sigma_i} h_j w_{ij} \tag{1}$$

Probability of (v,h)

$$p(\mathbf{v},\mathbf{h}) = \frac{e^{-E(v,h)}}{\sum_x \sum_k e^{-E(x,k)}} \tag{2}$$

Conditional distributions

$$p(h|v) = \sum_i p(h_i|v) \tag{3}$$

$$p(h_j = 1|v) = \mathrm{sigm}(c_j + \sum_i W_{ji} x_i) \tag{4}$$

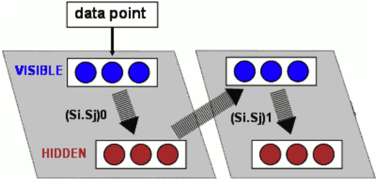
Contrastive Divergence

- Problem: Log likelihood gradient is hard to compute
- Run Markov chain to approximate the model distribution
- One step of Gibbs Sampling is sufficient

Remark

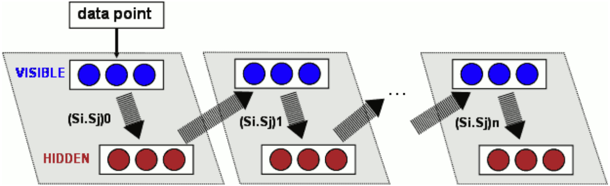
Training a RBM is performed by algorithm known as "Contrastive Divergence Learning"

Contrastive Divergence - reconstruction step



- Get one data point from data set
- Use values of the data to set state of visible units - Si
- Compute Sj for each hidden neuron based on Si
- Compute (Si.Sj)0
- Reconstruction: on visible units compute Si using the Sj
- Compute state of hidden neurons Sj again using Si
- Use Si and Sj to compute (Si.Sj)1

Contrastive Divergence in n steps - whole algorithm



- For each data point in data set:
  - perform reconstruction in n-steps
  - Accumulate CDpos = CDpos + (Si.Sj)0
  - Accumulate CDneg = CDneg + (Si.Sj)n
- Compute average CDpos and CDneg (divide by nr of points)
- Compute gradient CD = CDpos - CDneg
- Update weights and biases W" = W + alpha*CD
- Repeat for whole dataset in number of epochs (iterations)

Using RBM for classification

Three approaches (Hinton):

- Use the hidden features learned by the RBM as the inputs for some standard discriminative method
- Train a separate RBM on each class
- Train a joint density model using a single RBM (that has two sets of visible units - y for label and x for data)

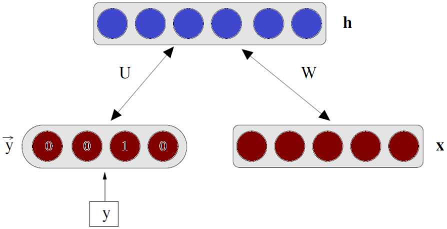
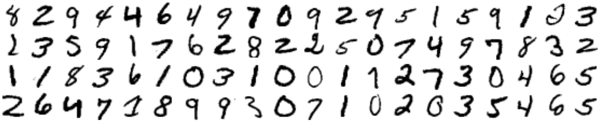


Table of Contents

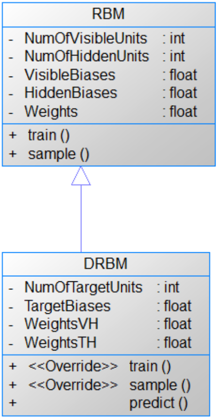
Tools

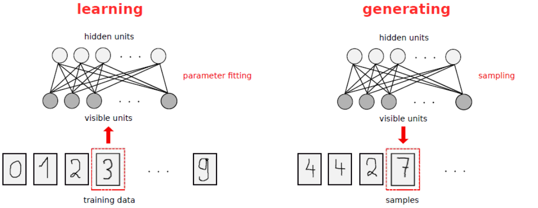
Data loading and preprocessing module



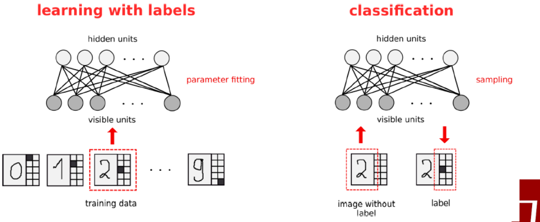
- MNIST - handwritten digit images
- Raw data consists of greyscale normalized images (28x28 pixels, pixel is number 0-255) and their labels (0..9)
- Dataset divided into training (50000), validation (10000) and test (10000) subsets
- Loaded optionally from cPickle file or CSV
- Data-specific, help functions implemented (binarization, transformations, scaling, visualizations)

Generative and Discriminative models of RBM



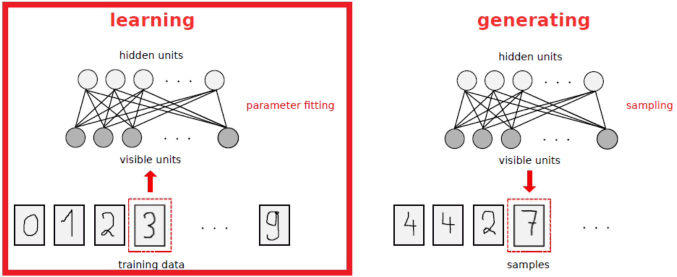


Source: A.Fischer, Ch.Igel: Training Restricted Boltzmann Machines: An Introduction



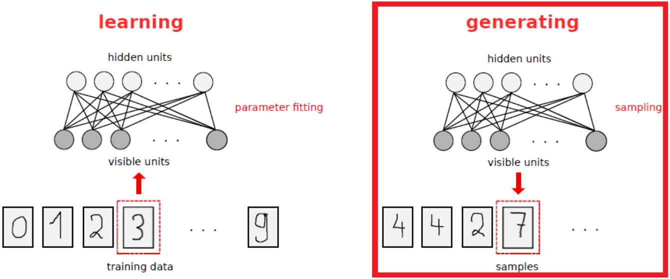
Source: A.Fischer, Ch.Igel: Training Restricted Boltzmann Machines: An Introduction

Generative model - train()



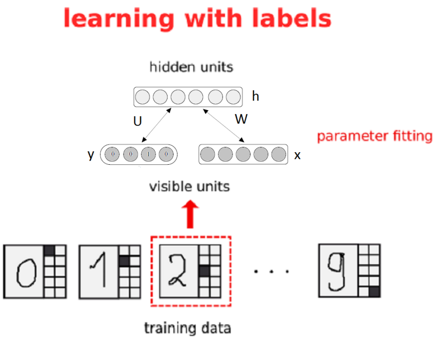
- Fitting RBM parameters so that to model distribution of the training data
- Iteratively performs one step of Contrastive Divergence (using Gibbs sampling) on data subset of one-class
- Learns until specified error threshold between data and reconstruction is reached

Generative model - sample()



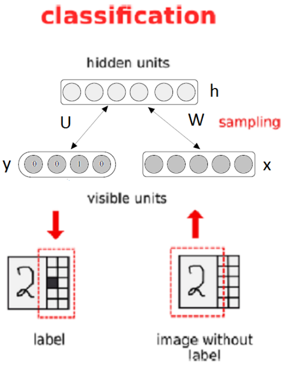
- Trained RBM used to generate samples from learned distribution
- Shows reconstructed image for the specified digit

Discriminative model - train()



- DRBM models a joint distribution of inputs (x) and target classes (y)
- Two sets of visible units and two weight matrices: between x and h (W) and between y and h (U)
- Train() performs n-step Constrastive Divergence for a mini-batch

Discriminative model - predict()



- Fix the visible variables corresponding to the image
- Sample target variables corresponding to the labels in chosen number of iterations
- For each datapoint in testset return probabilities of each class
- Choose the label class with highest probability
- Compare with original labels
- Count wrong predictions and compute accuracy

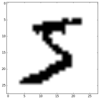
Table of Contents

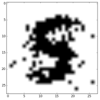
Testing methodology and assumptions

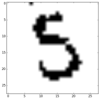
- Reducing to binary problem (binarization threshold = 0.5)
- Parameters possible to test:
  - size of training set,
  - size of test set,
  - learning rate,
  - initial weight distribution,
  - momentum,
  - l2 penaltization,
  - number of steps for contrastive divergence,
  - size of hidden units,
  - number of epochs for training,
  - number of iterations for sampling,
  - error threshold for traning,
  - random state

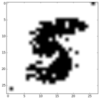
Testing generative model

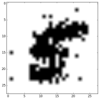
Experiments on different sizes of hidden units

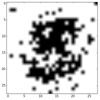

Original

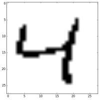

hidden=200

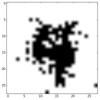

hidden=300

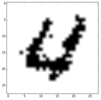

hidden=400

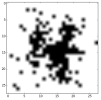

hidden=500

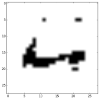

hidden=700

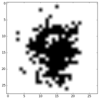

Original


hidden=100


hidden=200


hidden=300


hidden=400

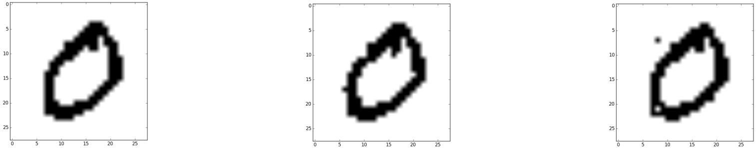

hidden=500

Remark

Different digit classes have different optimal hyperparameters

Testing reconstruction with DRBM

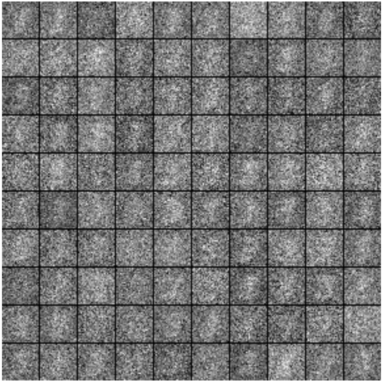
- Learning in mini-batches improved performance
- Momentum parameter for weight update other than 0.0 worsened results
- Optimized results for reconstruction after 500 epochs of training were good:

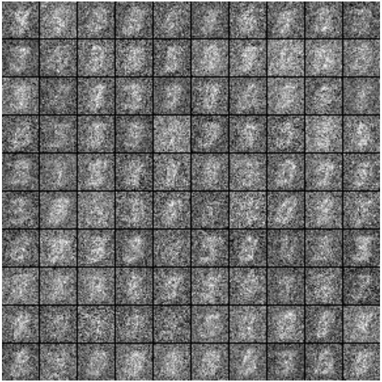


Original image momentum=0.0 momentum=0.5

- For 500-epoch training MSE falls below 1.0 - in about 30 minutes (on 50 train size)

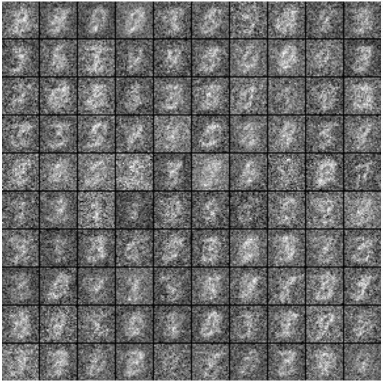
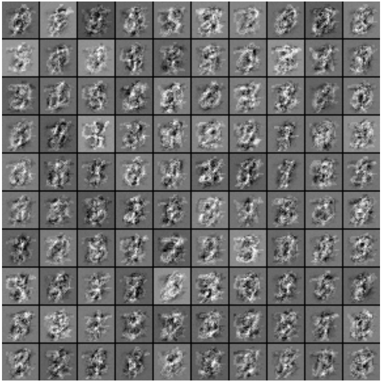
Monitoring progress of learning





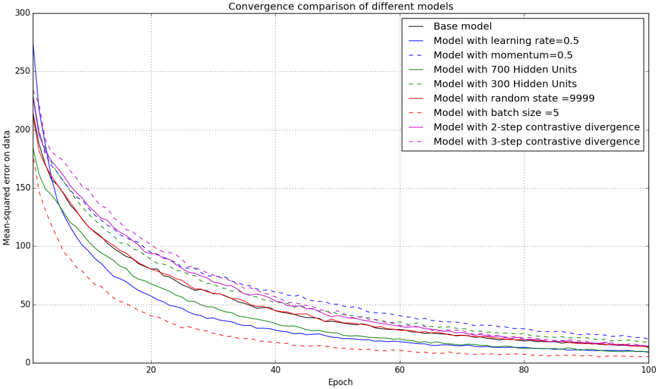
Learned weights after 1 and 5 iterations

Monitoring progress of learning

Learned weights after 10 and 500 iterations

Monitoring progress of learning
Reconstruction error for 100 epochs

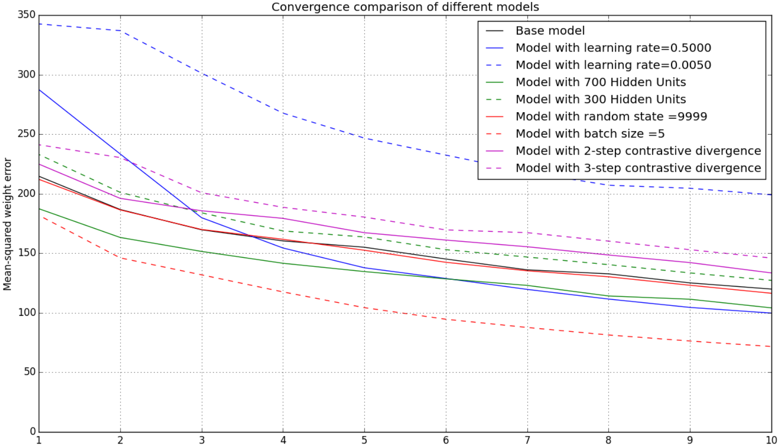


Remark

The reconstruction error on the training set falls rapidly and consistently at the start of learning and then more slowly.

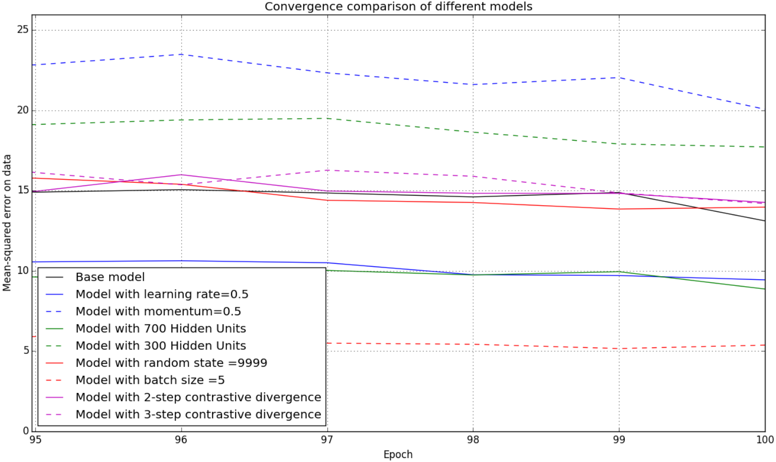
Model selection
Reconstruction error for first 10 epochs

Base model: lr = 0.01, hidden units=500, random state =1234, batch size=10, 1-step constrastive divergence, no momentum



Model selection

Reconstruction error for last 5 epochs

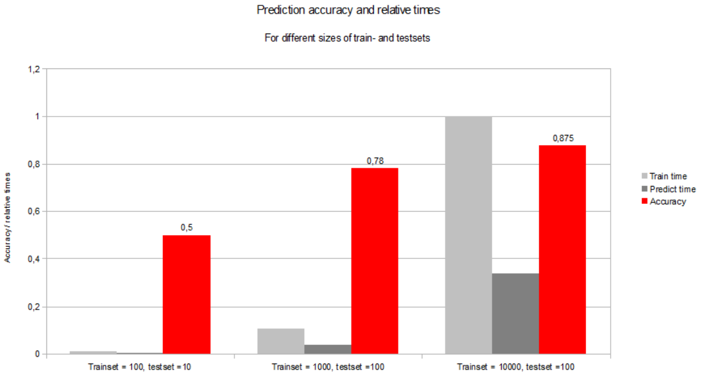


Model selection

Conclusions

- Learning in smaller mini-batches and increasing number of hidden units (from 400 to 700) improved reconstruction
- However, these changes resulted in longer train time
- Higher learning rate (0.5 instead of 0.05) caused reconstruction error to drop more sharply
- For larger train datasets higher learning rate caused instability (after some time of drop error started to increase)
- Different random states do not change reconstruction error significantly
- 1-step contrastive divergence is optimal

Testing RBM for classification (I)



- Classification accuracy depends on are train- and test sets size
- For larger data sets train and prediction times become prohibitive for personal computing

Testing RBM for classification(II)

- Optimal hyperparameters for training phase were chosen
- Performing 100 percent accurate classification on training data could be achieved even on small sets
- Classification accuracy on new data achieved so far is 95 percent(MNIST 50000 trainset and 10000-validation set)
- Better results are expected given greater computing power

Conclusion

Implementation and tests on synthetic data within this project proved that RBMs can be effectively used as standalone classifiers.

Table of Contents

Plans for further work

- Test-against-all-labels prediction approach
- Optimizing algorithms for best performance
- Testing on gaussian values
- Another dataset, possibly CIFAR

Literature I

Hugo Larochelle, Yoshua Bengio.
Classification using Discriminative Restricted Boltzmann Machines.
Proceedings of the 25th International Conference on Machine Learning, 2008.

Geoffrey Hinton.
A Pratical Guide to Training Restricted Boltzmann Machines.
UTML TR 2010-003, 2010.

Miguel A. Carreira-Perpin Geoffrey E. Hinton
On Contrastive Divergence Learning
Artificial Intelligence and Statistics, 2005.

Thank you