Department of Computer Science
University of Toronto
http://learning.cs.toronto.edu

6 King's College Rd, Toronto
M5S 3G4, Canada
fax: +1 416 978 1455

November 29, 2011

UTML TR 2011–002

Data Normalization in the Learning of Restricted Boltzmann Machines

Yichuan Tang,
and **Ilya Sutskever**
Department of Computer Science, University of Toronto

Abstract

In practice, training Restricted Boltzmann Machines with Contrastive Divergence and other approximate maximum likelihood methods works well on data with black backgrounds. However, when using inverted images for training, learning is typically much worse. In this paper, we propose a very simple yet very effective solution to this problem. The new algorithm requires the addition of only three(!) lines of code to existing RBM learning algorithms.

Data Normalization in the Learning of Restricted Boltzmann Machines

Yichuan Tang,
and **Ilya Sutskever**
Department of Computer Science, University of Toronto

1 Introduction

The Restricted Boltzmann Machine (RBM) [5, 2] is a popular type of unsupervised model for binary data. Practical ways to train it are discussed in [3]. We first demonstrate the problem of training RBMs on data that are non-sparse. We then provide a very simple way of resolving this problem, improving RBM learning.

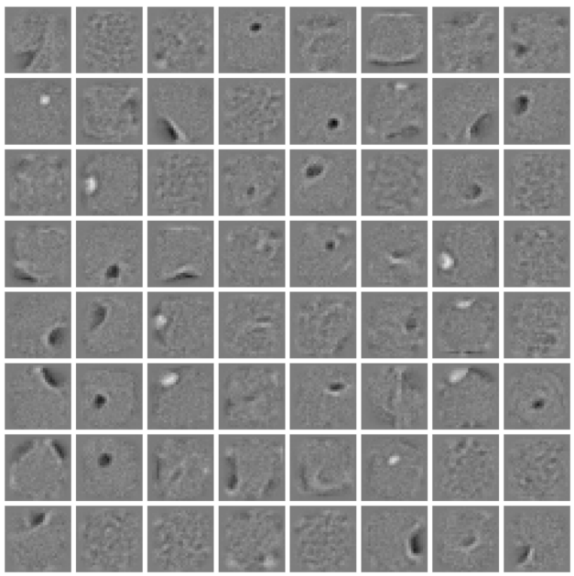
2 The Problem

RBMs work well when their input data vectors are sparse:

$$\left(\frac{1}{N}\sum_{n}^{N} v_i^n\right) \ll 1.0 \tag{1}$$

The standard MNIST digits are an example since they are white digits on a black background. However, when the input data mean is near 1.0, learning is considerably worse. We demonstrate this with the MNIST digits. In Fig. 1, we learned an 500 hidden nodes RBM on the MNIST digits.

(a) MNIST training images.

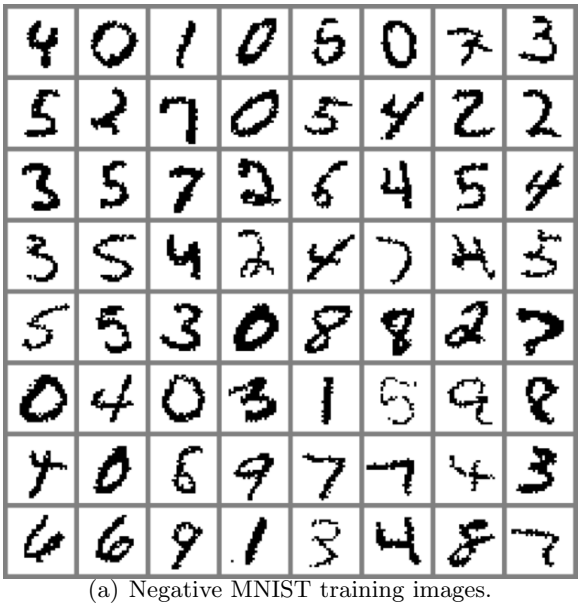


(b) Filters

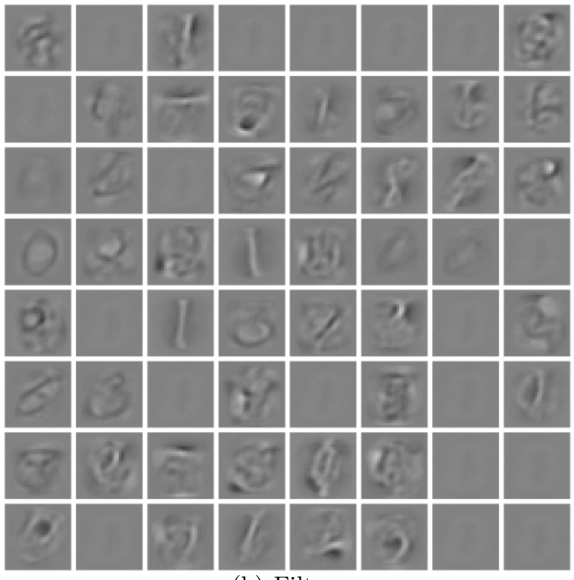
Figure 1: MNIST digits and learned filters from a 500 hidden nodes RBM using Fast Persistent Contrastive Divergence (FPCD).

The filters learned are typical for an RBM and the average log-probability estimated using Annealed Importance Sampling (AIS) are -96 nats for both the training and test set.

We can create a new set of input data by using the negative images of MNIST. This transformation preserves structure in the underlying data density. Therefore, we expect the RBMs to learn equally well on the new dataset. Using the same training hyper-parameters as before, we obtain the filters in Fig. 2 (b).



(a) Negative MNIST training images.



(b) Filters

Figure 2: Negative MNIST digits and learned filters from a 500 hidden nodes RBM using FPCD. Notice some filters have not been used by the model.

However, we note that the filters learned are very different from the previous filters. Quantitatively, the log-probability estimated using AIS are -110 nats for both the training and test sets. The difference of the 14 nats shows that the RBM learning using approximate maximum likelihood is *prejudiced* against input data with large means.

3 The Solution

In most machine learning algorithms, it is considered to be standard protocol to remove the mean of data before training:

$$z_i = v_i - \frac{1}{N}\sum_n^N v_i^n \tag{2}$$

$$= v_i - \mu_i \tag{3}$$

A ensuing whitening step can also be applied. Removing the mean usually allows the optimization to be easier. For example, for a one layer neural net with linear outputs and a mean squared error objective, it can be shown that the Hessian matrix of the objective is a function of the correlation matrix of the input data [4]. Data distribution with a non-zero mean would increase the eigenvalue of the Hessian of the objective function, affecting optimization detrimentally.

Following this principle, we wish to learn an RBM on data with zero-mean, namely the variables $\mathbf{z} = \mathbf{v} - \boldsymbol{\mu}$. Note that our original visibles are $\mathbf{v} \in \{0,1\}^{N_v}$ and hidden $\mathbf{h} \in \{0,1\}^{N_h}$. The new zero-meaned variables $\mathbf{z}$ is still discrete, but its domain have been shifted by $\boldsymbol{\mu}$. For example, the domain of v_i is $\{0,1\}$, while the domain of z_i is $\{-0.3, 0.7\}$, if $\mu_i = 0.3$.

In this case, the energy function is:

$$E_{RBM}(\mathbf{z}, \mathbf{h}; \theta) = -\sum_{i}^{N_v} b_i z_i - \sum_{j}^{N_h} c_j h_j - \sum_{i,j}^{N_v, N_h} W_{ij} z_i h_j \tag{4}$$

$$= -\sum_{i}^{N_v} b_i (v_i - \mu_i) - \sum_{j}^{N_h} c_j h_j - \sum_{i,j}^{N_v, N_h} W_{ij} (v_i - \mu_i) h_j \tag{5}$$

The posterior distribution $p(h_j = 1|\mathbf{z})$ has the same form as the posterior in a standard RBM:

$$p(h_j = 1|\mathbf{z}) = \frac{1}{1 + \exp(-\mathbf{z}^\mathsf{T} \mathbf{W}_{(:,j)} - c_j)} \tag{6}$$

The conditional distribution of the visibles z_j given the hidden states is:

$$p(z_i = 1 - \mu_i|\mathbf{h}) = \frac{\exp((1 - \mu_i)\phi_i)}{\exp((1 - \mu_i)\phi_i) + \exp(-\mu_i \phi_i)}, \quad where\ \phi_i = \mathbf{W}_{(i,:)}\mathbf{h} + b_i \tag{7}$$

The probability of the other state of z_i is $p(z_i = 0 - \mu_i|\mathbf{h}) = 1 - p(z_i = 1 - \mu_i|\mathbf{h})$. We can simplify the above conditional distribution further by multiplying by $\frac{\exp((1-\mu_i)\phi_i)}{\exp((1-\mu_i)\phi_i)}$:

$$p(z_i = 1 - \mu_i|\mathbf{h}) = \frac{1}{1 + \exp(-\phi_i)} \tag{8}$$

$$p(z_i = 1 - \mu_i|\mathbf{h}) \equiv p(v_i = 1|\mathbf{h}) \tag{9}$$

The equivalence of the two conditional distributions $p(z_i = 1-\mu_i|\mathbf{h})$ and $p(v_i = 1|\mathbf{h})$ means that we can implement this solution by easily modifying existing algorithms with minimal intrusion. Specifically, when sampling z_i given $\mathbf{h}$, we first sample v_i just like in the standard RBM (using Eq. 8). We can then obtain the value of z_i by subtracting the mean from v_i: $z_i = v_i - \mu_i$. We stress that this operation is proper due to the equivalence of the two probability distributions in Eq. 9.

After learning the parameters $\{\mathbf{W}, \mathbf{b}, \mathbf{c}\}$ of the RBM defined over $\mathbf{z}$ and $\mathbf{h}$, we can obtain an RBM over the original visibles $\mathbf{v}$:

$$E_{RBM}(\mathbf{z}, \mathbf{h}; \theta) = -\sum_{i}^{N_v} b_i (v_i - \mu_i) - \sum_{j}^{N_h} c_j h_j - \sum_{i,j}^{N_v, N_h} W_{ij} (v_i - \mu_i) h_j \tag{10}$$

$$= -\sum_{i} b_i v_i - \sum_{j} c_j h_j - \sum_{i,j} W_{ij} v_i h_j + \sum_{i,j} W_{ij} \mu_i h_j + \sum_{i} b_i \mu_i \tag{11}$$

$$= -\sum_{i} b_i v_i - \sum_{i,j} W_{ij} v_i h_j - \sum_{j} (c_j - \sum_{i} W_{ij} \mu_i) h_j + const. \tag{12}$$

The energy in Eq. 12 is equal (up to an additive constant) to the energy of an RBM over $\mathbf{v}$ with weights: $\mathbf{W}$, visible biases: $\mathbf{b}$, and hidden biases: $\mathbf{c} - \boldsymbol{\mu}^\mathsf{T}\mathbf{W}$.

We give a summarization of the Contrastive Divergence algorithm to learn parameters on zero-meaned data in Alg. 1. The three additions needed are in **bold**.

Algorithm 1 Contrastive Divergence Training of RBM on zero-meaned data.

0: Randomly initialize the weights and biases to be small.
1: **Subtract data mean from all training data vectors: $\mathbf{z} = \mathbf{v} - \boldsymbol{\mu}$**
  for $t = 1 : NumberEpochs$ **do**
    for $n = 1 : NumberDataSamples$ **do**
      Positive Phase:
2: Compute hidden activations and sample using Eq. 6.
3: Calculate the MLE gradient of the positive phase.
      Negative Phase:
4: Compute reconstructions $\hat{v}_i$ and sample using Eq. 8.
5: **Compute reconstructions $\hat{\mathbf{z}}$: $\hat{\mathbf{z}} = \hat{\mathbf{v}} - \boldsymbol{\mu}$**
6: Compute hidden activations and sample using Eq. 6.
7: Calculate the MLE gradient of the negative phase.
8: Approx. gradient = positive phase gradient − negative phase gradient.
9: Update the parameters.
    end for
  end for
10: **Modify hidden biases: $\mathbf{c}_{new} \leftarrow \mathbf{c} - \boldsymbol{\mu}^{\mathsf{T}}\mathbf{W}$**

4 Experiments

To demonstrate the advantage of RBM learning using zero-meaned data, we trained a 500 hidden nodes RBM with FPCD as well as using the zero-mean (ZM) normalization technique. Learning rate is fixed at 0.01, the weights are initialized from a Gaussian distribution with a standard deviation of 0.01, the visible and hidden biases are initialized to be 0.0. A weight penalty of 0.0002 is used for the visible to hidden weights. Training was run for 30 and 1000 epochs with 600 mini-batches per epoch. We presented the AIS estimated log-probability (nats) in Table 1.

Dataset	FPCD-30	ZM-30	FPCD-1000	ZM-1000
MNIST	-96 (-96)	-94 (-94)	-84(-81)	-84 (-81)
Neg. MNIST	-110(-110)	-96 (-96)	-87(-85)	-84 (-81)

Table 1: AIS est. log-probability in nats. Average training log-probs are in parenthesis. FPCD-30 means the model was trained for 30 epochs using Fast Persistent Contrastive Divergence.

For Negative MNIST, the model learned by FPCD was significantly worse than the equivalent model from MNIST. However, we see that it makes no difference for the ZM algorithm. For learning using 30 epochs, ZM is better than the standard FPCD on the original MNIST data. When training for 1000 epochs (probably achieving convergence), ZM training achieves the same log-prob for both the MNIST and Negative MNIST. In Fig. 3, we plot the learned filters of Negative MNIST using the ZM algorithm.

5 Discussions

The problem of learning Negative MNIST have been previously addressed by [1]. In that paper, RBM's gradients are modified to be a weighted average of the gradients of all possible input data flipping. The weighting was engineered according to a sparseness preference. Parallel tempering and learning rate adaptation using AIS were also employed. In contrast, this work has shown that by simply subtracting the mean of the data prior to learning the parameters, we can achieve similar boost

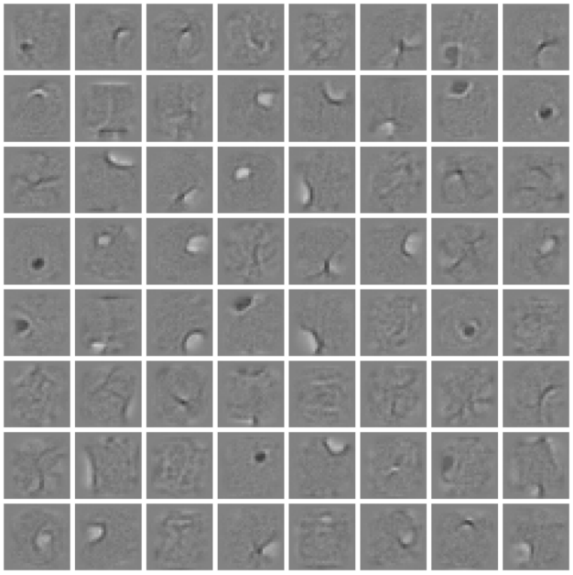


Figure 3: Filters learned by FPCD on zero-meaned Negative MNIST.

in learning performance. In addition, the extra incurred computation cost is one matrix subtraction for every step of Contrastive Divergence, which is negligible compared to the other matrix-matrix multiplications that are required.

References

[1] K. Cho, T. Raiko, and A. Ilin. Enhanced gradient and adaptive learning rate for training restricted boltzmann machines. In *Proceedings of the International Conference on Machine Learning (ICML 2011)*, 2011.

[2] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800, 2002.

[3] G. E. Hinton. A practical guide to training restricted boltzmann machines. Technical report, 2010. `http://www.cs.toronto.edu/~hinton/absps/guideTR.pdf`.

[4] Y. LeCun, L. Bottou, G. Orr, and K. Muller. Efficient backprop. In G. Orr and Muller K., editors, *Neural Networks: Tricks of the trade*. Springer, 1998.

[5] P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, volume 1, chapter 6, pages 194–281. MIT Press, Cambridge, 1986.