Training Restricted Boltzmann Machines with Multi-Tempering: Harnessing Parallelization

Philemon Brakel, Sander Dieleman, and Benjamin Schrauwen

Department of Electronics and Information Systems, Ghent University,
Sint-Pietersnieuwstraat 41, 9000 Gent, Belgium
{philemon.brakel,sander.dieleman,benjamin.schrauwen}@elis.ugent.be

Abstract. Restricted Boltzmann Machines (RBM's) are unsupervised probabilistic neural networks that can be stacked to form Deep Belief Networks. Given the recent popularity of RBM's and the increasing availability of parallel computing architectures, it becomes interesting to investigate learning algorithms for RBM's that benefit from parallel computations. In this paper, we look at two extensions of the parallel tempering algorithm, which is a Markov Chain Monte Carlo method to approximate the likelihood gradient. The first extension is directed at a more effective exchange of information among the parallel sampling chains. The second extension estimates gradients by averaging over chains from different temperatures. We investigate the efficiency of the proposed methods and demonstrate their usefulness on the MNIST dataset. Especially the weighted averaging seems to benefit Maximum Likelihood learning.

Keywords: Markov Chain Monte Carlo, Restricted Boltzmann Machines, Neural Networks, Machine Learning

1 Introduction

Since the recent popularity of deep neural architectures for learning [2], Restricted Boltzmann Machines (RBM's; [6, 5]), which are the building blocks of Deep Belief Networks [7], have been studied extensively. An RBM is an undirected graphical model with a bipartite connection structure. It consists of a layer of visible units and a layer of hidden units and can be trained in an unsupervised way to model the distribution of a dataset. After training, the activations of the hidden units can be used as features for applications such as classification or clustering. Unfortunately, the likelihood gradient of RBM's is intractable and needs to be approximated.

Most approximations for RBM training are based on sampling methods. RBM's have an independence structure that makes it efficient to apply Gibbs sampling. However, the efficiency of Gibbs sampling depends on the rate at which independent samples are generated. This property is known as the *mixing rate*. While Gibbs samplers will eventually generate samples from the true underlying

distribution they approximate, they can get stuck in local modes. This is especially problematic for distributions that contain many modes that are separated by regions where the probability density is very low.

In this paper, we investigate two methods for improving both the mixing rate of the sampler and the quality of the gradient estimates at each sampling step. These two methods are extensions for the so-called Replica Exchange method and were recently proposed for statistical physics simulations [1]. The first extension allows every possible pair of replicas to swap positions to increase the number of sampling chains that can be used in parallel. The second extension is to use a weighted average of the replicas that are simulated in parallel. The weights are chosen in a way that is consistent with the exchange mechanism.

2 Restricted Boltzmann Machines

An RBM defines an energy function that depends on the joint configuration of a set of visible variables $\mathbf{v}$ and a set of hidden variables $\mathbf{h}$. In an RBM where all variables are binary, this energy function is given by

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i=1}^{N_h}\sum_{j=1}^{N_v} W_{ij} h_i v_j - \sum_{i=1}^{N_h} h_i a_i - \sum_{j=1}^{N_v} v_j b_j \ , \tag{1}$$

where N_h and N_v are, respectively, the number of hidden and the number of visible units. The symbols W, a and b denote trainable weight and bias parameters. This function can be used to define a Gibbs probability distribution of the form $p(\mathbf{v}) = \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}/Z$, where Z is the partition function which is given by $Z = \sum_{\mathbf{h},\mathbf{v}} e^{-E(\mathbf{v},\mathbf{h})}$.

The gradient of this likelihood function is given by

$$\frac{\partial \ln p(\mathbf{v})}{\partial \theta} = -\sum_{\mathbf{h}} p(\mathbf{h}|\mathbf{v}) \frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \theta} + \sum_{\mathbf{v},\mathbf{h}} p(\mathbf{v}, \mathbf{h}) \frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \theta} \ , \tag{2}$$

where θ is an element in the set of parameters $\{\mathbf{W}, \mathbf{a}, \mathbf{b}\}$. The first term of this gradient can be evaluated analytically in RBM's but the second term needs to be approximated. This second term is the gradient of the partition function and will be referred to as the *model expectation*.

3 Training RBM's

The most commonly used training method for RBM's is the Contrastive Divergence (CD; [6]) algorithm. During training, a Gibbs sampler is initialized at a sample from the data and run for a couple of iterations. The last sample of the chain is used to replace the intractable model expectation. This strategy assumes that many of the low energy configurations, that contribute most to the model expectation, can be found near the data. However, it is very likely that

there are many other valleys of low energy. Furthermore, the algorithm does not necessarily optimize the likelihood function at all.

In Persistent Contrastive Divergence learning [13], (PCD) a Markov chain is updated after every parameter update during training and used to provide samples that approximate the model expectation. The difference with normal CD is that the chain is not reset at a data point after every update, but keeps on running so it can find low energy regions that are far away from the data. Given infinite training time, this algorithm optimizes the true likelihood. However, as training progresses and the model parameters get larger, the energy landscape becomes more rough. This will decrease the size of the steps the chain takes and increase the chance that the chain gets stuck in local modes of the distribution.

To obtain better mixing rates for the sampling chains in PCD, the Fast PCD algorithm was proposed [12]. This algorithm uses a copy of the model that is trained using a higher learning rate to obtain samples. The training itself is in this case pushing chains out of local modes. Unfortunately, the training algorithm is now not necessarily converging to the true likelihood anymore.

Another way to improve the mixing rate is Replica Exchange Monte Carlo [11], also referred to as Parallel Tempering (PT). Recently, PT has been applied to RBM training as well [4]. This algorithm runs various chains in parallel that sample from replicas of the system of interest that operate under different temperatures. Chains that operate at lower temperatures can escape from local modes by jumping to locations of similar energy that have been proposed by chains that operate at higher temperatures. A serial version of this idea has also been proposed for training RBM's [9].

One downside of PT for training RBM's is that the number of parallel sampling chains that can be used by this algorithm is limited. One can use many chains in PT to cover more temperatures. This will cause more swaps between neighbouring chains to be accepted because they are closer together. However, it will also take more sequential updates before a certain replica moves back and forth between the lowest and the highest temperatures. Another disadvantage of PT is that only the chain with the lowest temperature is actually used to gather statistics for the learning algorithm.

4 Multi-Tempering

To increase the number of parallel chains that PT can effectively use, we propose Multiple Replica Exchange methods for RBM training. These methods have already been shown to work well in statistical physics [3, 1]. To prevent the use of very different names for similar algorithms, we will refer to this method as Multi-Tempering (MT). Since MT is a modification of PT Markov Chain Monte Carlo, it is necessary to describe the original algorithm in some more detail.

The idea behind PT is to run several Markov chains in parallel and treat this set of chains as one big chain that generates samples from a distribution with augmented variables. Transition steps in this combined chain can now also include possible exchanges among the sub chains. Let $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_M\}$ be

the state of a Markov chain that consists of the states of M sub chains that operate under inverse temperatures $\{\beta_1, \cdots, \beta_M\}$, where $\beta_1 = 1$ and indicative of the model we want to compute expectations for. The combined energy of this system is given by $E(\mathbf{X}) = \sum_{i=1}^{M} \beta_i E(\mathbf{x}_i)$. The difference in total energy that results from switching two arbitrary sub chains with indices i, j, is given by

$$E(\hat{\mathbf{X}}(i,j)) - E(\mathbf{X}) = (\beta_i - \beta_j)(E(\mathbf{x}_j) - E(\mathbf{x}_i)) \ , \tag{3}$$

where $\hat{\mathbf{X}}(\cdot)$ denotes the new state of the combined chain that results from the exchange indicated by its arguments[1]. If i and j are selected uniformly and forced to be neighbours, the Metropolis-Hastings acceptance probability is given by $r_{ij} = \exp(E(\mathbf{X}) - E(\hat{\mathbf{X}}(i,j)))$. This is the acceptance criterion that is used in standard Parallel Tempering.

In Multi-Tempering [1], index i is selected uniformly and index j is selected with a probability that is based on the difference in total energy the proposed exchange would cause:

$$p(j|i) = \frac{r_{ij}}{\sum_{j'=1}^{M} r_{ij'}} \ . \tag{4}$$

The acceptance probability is now given by

$$A(i,j) = \min \left\{ \frac{\sum_{j'} e^{-E(\hat{\mathbf{X}}(i,j'))}}{\sum_{k} e^{-E(\hat{\mathbf{X}}(i,j,k))}} \right\} \ . \tag{5}$$

5 Using a weighted average of the chains

Given the selection probabilities $p(j|i)$ from Equation 4 and the acceptance probabilities $A(i,j|\mathbf{X})$, one can compute a weighted average to estimate the gradient of the intractable likelihood term. This average is given by

$$\langle g \rangle_1 = \sum_{j=1}^{M} \left[(1 - A\,(i,j))\, g(\mathbf{x}_1) + A(i,j) g(\mathbf{x}_j) \right] p(j|i) \ , \tag{6}$$

where $g(\cdot)$ is short for $\frac{\partial E(\cdot)}{\partial \theta}$. This extension is originally called Information Retrieval but this term might lead to confusion in a Machine Learning context. We will refer to this version of the algorithm as Multi-Tempering with weighed averaging (MTw).

6 Experiments

All experiments were done on the MNIST dataset. This dataset is a collection of $70,000$ 28×28 grayscale images of handwritten digits that has been split into a

[1] So $\hat{\mathbf{X}}(i,j,k)$ would mean that i is first swapped with j and subsequently, the sample at position j is swapped with the one at position k.

train set of 50000 images and test and validation sets of each 10000 images. The pixel intensities were scaled between 0 and 1 and interpreted as probabilities from which binary values were sampled whenever a datapoint was required.

First, is was investigated how the MT and the PT algorithms behave with different numbers of parallel chains by looking at the rate at which replicas travel from the highest temperature chain to the one with the lowest temperature. Ten RBM's with 500 hidden units were trained with PCD using a linearly decaying learning rate with a starting value .002 for 500 epochs. Subsequently, both sampling methods were run for 10000 iterations and the number of times that a replica was passed all the way from the highest to the lowest temperature chain was counted. This experiment was done for different numbers of parallel chains. The inverse temperatures were uniformly spaced between .8 and 1. In preliminary experiments, we found that almost no returns from the highest to the lowest temperature occurred for any algorithm for much larger intervals.

The second experiment was done to get some insight in the mixing rates of the sampling methods and their success at approximating the gradient of the partition function. A small RBM with 15 hidden units was trained on the MNIST dataset using the PCD algorithm. The different sampling methods were now run for 20000 iterations while their estimates of the gradient were compared with the true gradient which had been computed analytically . Because the success of the samplers partially depends on their random initialization, we repeated this experiment 10 times.

Finally, to see how the different sampling algorithms perform at actual training, a method called annealed importance sampling (AIS) [8, 10] was used to estimate the likelihood of the data under the trained models. PCD, PT, MT and MTw were each used to train 10 RBM models on the train data for 500 epochs. Each method used 100 chains in parallel. The inverse temperatures for the Tempering methods were linearly spaced between .85 and 1 as we expected a slightly more conservative temperature range would be needed to make PT competitive. We used no weight decay and the order of magnitude of the starting learning rates was determined using a validation set. The learning rate decreased linearly after every epoch.

7 Results and Discussion

Fig. 1 displays the results of the first experiment. The number of returns is a lot higher for MT at the start and seems to go down at a slightly slower rate than for PT. This allows a larger number of chains to be used before the number of returns becomes negligible.

As Fig. 2 shows, the MT estimator was most successful at approximating the gradient of the partition function of the RBM with 15 hidden units. To our surprise, the MT estimator also performed better than the MTw estimator. However, it seems that the algorithms that used a single chain to compute the expectations (MT and PT), fluctuate more than the ones that use averages (MTw and PCD).

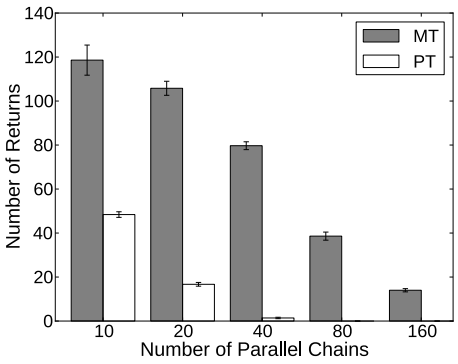


Fig. 1: Number of returns for parallel tempering and multiple replica exchange as a function of the number of parallel chains that are used.

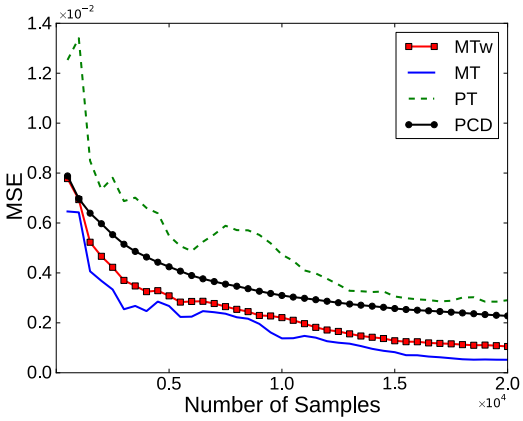
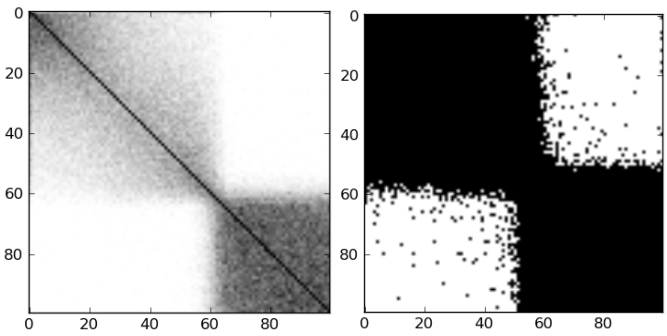


Fig. 2: Mean Square Error (MSE) between the approximated and the true gradients of the partition function of an RBM with 15 units as a function of the number of samples.

Table 1: Means and standard deviations of the AIS estimates of the likelihood of the MNIST test set for different training methods. Means are based on 10 experiments with different random initializations.

Epochs	MTw	MT	PT	PCD
250	−82.25(10.33)	−92.59(7.79)	−93.48(11.54)	−94.43(1.71)
500	−65.09(7.66)	−83.74(6.76)	−84.18(7.79)	−80.45(11.36)

Table 1 displays the AIS estimates of the likelihood for the MNIST test set for each of the training methods. MTw outperforms all other methods on this task. The standard deviations of the results are quite high and MT, PT and PCD don't seem to differ much in performance. The fact that MT and PT use only a single chain to estimate the gradient seems to be detrimental. This is not in line with the results for the gradient estimates for the 15 unit RBM. It could be that larger RBM's benefit more from the higher stability of gradient estimates that are based on averages than small RBM's. The results suggest that PCD with averaged parallel chains is preferable to Tempering algorithms that use only a single chain as estimate due to its relative simplicity but that MTw is an interesting alternative.



(a) Matrix of exchange frequencies cut off at 100.

(b) Binarized matrix of exchanges.

Fig. 3: Plot of inter chain replica exchanges for MT.

During MT training, we also recorded the transition indices for further inspection. There are clearly many exchanges that are quite large as can be seen in Fig. 3a, which shows a matrix in which each entry $\{i, j\}$ represents the number of times that a swap occurred between chains i and j. While there seems to be a bottleneck that is difficult to cross, it is clear that some particles still make it to the other side once in a while. In Fig. 3b, one can see that occasionally some very large jumps occur that span almost the entire temperature range.

8 Conclusion

We proposed two methods to improve Parallel Tempering training for RBM's and showed that the combination of the two methods leads to improved performance on learning a generative model of the MNIST dataset. We also showed that the MTw algorithm allows more chains to be used in parallel and directly improves the gradient estimates for a small RBM. While the weighted average didn't seem

to improve the mixing rate, it seemed to stabilize training. For future work, it would be interesting to see how the sampling algorithms compare when the RBM's are used for pre-training a Deep Belief Network.

Acknowledgments. The work presented in this paper is funded by the EC FP7 project ORGANIC (FP7-231267).

References

1. Athènes, M., Calvo, F.: Multiple-replica exchange with information retrieval. Chemphyschem 9(16), 2332–9 (2008)
2. Bengio, Y.: Learning deep architectures for AI. Foundations and Trends in Machine Learning 2(1), 1–127 (2009), also published as a book. Now Publishers, 2009.
3. Brenner, P., Sweet, C.R., VonHandorf, D., Izaguirre, J.A.: Accelerating the replica exchange method through an efficient all-pairs exchange. The Journal of Chemical Physics 126(7), 074103 (2007)
4. Desjardins, G., Courville, A.C., Bengio, Y., Vincent, P., Delalleau, O.: Tempered markov chain monte carlo for training of restricted boltzmann machines. Journal of Machine Learning Research - Proceedings Track 9, 145–152 (2010)
5. Freund, Y., Haussler, D.: Unsupervised Learning of Distributions on Binary Vectors Using Two Layer Networks. Tech. rep., Santa Cruz, CA, USA (1994)
6. Hinton, G.E.: Training Products of Experts by Minimizing Contrastive Divergence. Neural Computation 14(8), 1771–1800 (2002)
7. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Computation 18(7), 1527–1554 (2006)
8. Neal, R.M.: Annealed importance sampling. Statistics and Computing 11, 125–139 (1998)
9. Salakhutdinov, R.: Learning in markov random fields using tempered transitions. In: Bengio, Y., Schuurmans, D., Lafferty, J.D., Williams, C.K.I., Culotta, A. (eds.) NIPS. pp. 1598–1606. Curran Associates, Inc. (2009)
10. Salakhutdinov, R., Murray, I.: On the quantitative analysis of Deep Belief Networks. In: McCallum, A., Roweis, S. (eds.) Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008). pp. 872–879. Omnipress (2008)
11. Swendsen, R.H., Wang, J.S.: Replica Monte Carlo Simulation of Spin-Glasses. Physical Review Letters 57(21), 2607–2609 (Nov 1986)
12. Tieleman, T., Hinton, G.: Using Fast Weights to Improve Persistent Contrastive Divergence. In: Proceedings of the 26th international conference on Machine learning. pp. 1033–1040. ACM New York, NY, USA (2009)
13. Tieleman, T.: Training restricted Boltzmann machines using approximations to the likelihood gradient. In: Proceedings of the International Conference on Machine Learning (2008)