Stopping Criteria in Contrastive Divergence: Alternatives to the Reconstruction Error

David Buchaca Prats DAVIDBUCHACA@GMAIL.COM
Departament de Llenguatges i Sistemes Informàtics,
Universitat Politècnica de Catalunya,
Barcelona, Spain

Enrique Romero Merino EROMERO@LSI.UPC.EDU
Departament de Llenguatges i Sistemes Informàtics,
Universitat Politècnica de Catalunya,
Barcelona, Spain

Ferran Mazzanti Castrillejo FERRAN.MAZZANTI@UPC.EDU
Departament de Física i Enginyeria Nuclear,
Universitat Politècnica de Catalunya,
Barcelona, Spain

Jordi Delgado Pin JDELGADO@LSI.UPC.EDU
Departament de Llenguatges i Sistemes Informàtics,
Universitat Politècnica de Catalunya,
Barcelona, Spain

Abstract

Restricted Boltzmann Machines (RBMs) are general unsupervised learning devices to ascertain generative models of data distributions. RBMs are often trained using the Contrastive Divergence learning algorithm (CD), an approximation to the gradient of the data log-likelihood. A simple reconstruction error is often used to decide whether the approximation provided by the CD algorithm is good enough, though several authors (Schulz et al., 2010; Fischer & Igel, 2010) have raised doubts concerning the feasibility of this procedure. However, not many alternatives to the reconstruction error have been used in the literature. In this manuscript we investigate simple alternatives to the reconstruction error in order to detect as soon as possible the decrease in the log-likelihood during learning.

1. Introduction

Learning algorithms for deep multi-layer neural networks have been known for a long time (Rumelhart et al., 1986), though none of them have been widely used to solve large scale real-world problems. In 2006, Deep Belief Networks (DBNs) (Hinton et al., 2006) came out as a real breakthrough in this field, since the learning algorithms proposed ended up being a feasible and practical method to train these networks, with spectacular results (Hinton & Salakhutdinov, 2006; Larochelle et al., 2009; Lee et al., 2009; Le et al., 2012). DBNs have Restricted Boltzmann Machines (RBMs) (Smolensky, 1986) as their building blocks.

RBMs are topologically constrained Boltzmann Machines (BMs) with two layers, one of hidden and another of visible neurons, and no intra-layer connections. This property makes working with RBMs simpler than with regular BMs, and in particular the stochastic computation of the log-likelihood gradient may be performed more efficiently by means of Gibbs sampling (Bengio, 2009).

In 2002, the *Contrastive Divergence* learning algorithm (CD) was proposed as an efficient training method for product-of-expert models, from which RBMs are a special case (Hinton, 2002). It was observed that using CD to train

Proceedings of the 2^{nd} International Conference on Learning Representations, Banff, Canada, 2014. Copyright 2014 by the author(s).

RBMs worked quite well in practice. This fact was important for deep learning since some authors suggested that a multi-layer deep neural network is better trained when each layer is pre-trained separately as if it were a single RBM (Hinton & Salakhutdinov, 2006; Bengio et al., 2007; Larochelle et al., 2009). Thus, training RBMs with CD and stacking up RBMs seems to be a good way to go when designing deep learning architectures.

However, the picture is not as nice as it looks. CD is not a flawless training algorithm. Despite CD being an approximation of the true log-likelihood gradient (Bengio & Delalleau, 2009), it is biased and it may not converge in some cases (Carreira-Perpiñán & Hinton, 2005; Yuille, 2005; MacKay, 2001). Moreover, it has been observed that CD, and variants such as Persistent CD (Tieleman, 2008) or Fast Persistent CD (Tieleman & Hinton, 2009) can lead to a steady decrease of the log-likelihood during learning (Fischer & Igel, 2010; Desjardins et al., 2010). Therefore, the risk of learning divergence imposes the requirement of a stopping criterion. The two main methods used to decide when to stop the learning process are *reconstruction error* and *Annealed Importance Sampling* (AIS) (Neal, 1998). Reconstruction error is easy to compute and it has been often used in practice, though its adequacy remains unclear (Fischer & Igel, 2010). AIS seems to work better than reconstruction error in some cases, though it my also fail (Schulz et al., 2010).

In this paper we propose an alternative stopping criteria for CD and show its preliminary results. These criteria are based on the computation of two probabilities that do not require from the knowledge of the partition function of the system. The early detection of the decrease of the likelihood allows to overcome the reconstruction error faulty observed behavior.

2. Learning in Restricted Boltzmann Machines

2.1. Energy-based Probabilistic Models

Energy-based probabilistic models define a probability distribution from an energy function, as follows:

$$P(\boldsymbol{x},\boldsymbol{h}) = \frac{e^{-\text{Energy}(\boldsymbol{x},\boldsymbol{h})}}{Z} , \tag{1}$$

where $\boldsymbol{x}$ stand for visible variables and $\boldsymbol{h}$ are hidden variables (typically binary) introduced to increase the expressive power of the model. The normalization factor Z is called partition function and reads

$$Z = \sum_{\boldsymbol{x},\boldsymbol{h}} e^{-\text{Energy}(\boldsymbol{x},\boldsymbol{h})} . \tag{2}$$

Since only $\boldsymbol{x}$ is observed, one is only interested in the marginal distribution

$$P(\boldsymbol{x}) = \frac{\sum_{\boldsymbol{h}} e^{-\text{Energy}(\boldsymbol{x},\boldsymbol{h})}}{Z} , \tag{3}$$

but the evaluation of the partition function Z is computationally prohibitive since it involves an exponentially large number of terms.

The energy function depends on several parameters θ, that are adjusted at the learning stage. This is done by maximizing the likelihood of the data. In energy-based models, the derivative of the log-likelihood can be expressed as

$$\begin{aligned} -\frac{\partial \log P(\boldsymbol{x};\theta)}{\partial\theta} = &\; E_{P(\boldsymbol{h}|\boldsymbol{x})}\left[\frac{\partial \text{Energy}(\boldsymbol{x},\boldsymbol{h})}{\partial\theta}\right] \\ & - E_{P(\tilde{\boldsymbol{x}})}\left[E_{P(\boldsymbol{h}|\tilde{\boldsymbol{x}})}\left[\frac{\partial \text{Energy}(\tilde{\boldsymbol{x}},\boldsymbol{h})}{\partial\theta}\right]\right] , \end{aligned} \tag{4}$$

where the first term is called the positive phase and the second term is called the negative phase. Similar to (3), the exact computation of the derivative of the log-likelihood is usually unfeasible because of the second term in (4), which comes from the derivative of the partition function.

2.2. Restricted Boltzmann Machines

Restricted Boltzmann Machines are energy-based probabilistic models whose energy function is:

$$\text{Energy}(\boldsymbol{x},\boldsymbol{h}) = -\boldsymbol{b}^t\boldsymbol{x} - \boldsymbol{c}^t\boldsymbol{h} - \boldsymbol{h}^t\boldsymbol{W}\boldsymbol{x} . \tag{5}$$

RBMs are at the core of DBNs (Hinton et al., 2006) and other deep architectures that use RBMs to unsupervised pre-training previous to the supervised step (Hinton & Salakhutdinov, 2006; Bengio et al., 2007; Larochelle et al., 2009).

The consequence of the particular form of the energy function is that in RBMs both $P(\boldsymbol{h}|\boldsymbol{x})$ and $P(\boldsymbol{x}|\boldsymbol{h})$ factorize. In this way it is possible to compute $P(\boldsymbol{h}|\boldsymbol{x})$ and $P(\boldsymbol{x}|\boldsymbol{h})$ in one step, making possible to perform Gibbs sampling efficiently (Geman & Geman, 1984) that can be the basis of the computation of an approximation of the derivative of the log-likelihood (4).

2.3. Contrastive Divergence

The most common learning algorithm for RBMs uses an algorithm to estimate the derivative of the log-likelihood of a Product of Experts model called CD (Hinton, 2002).

The algoritmh for CD$_n$ estimates the derivative of the log-likelihood as

$$\begin{aligned} -\frac{\partial \log P(\boldsymbol{x}_1;\theta)}{\partial\theta} \simeq &\; E_{P(\boldsymbol{h}|\boldsymbol{x}_1)}\left[\frac{\partial \text{Energy}(\boldsymbol{x}_1,\boldsymbol{h})}{\partial\theta}\right] \\ & - E_{P(\boldsymbol{h}|\boldsymbol{x}_{n+1})}\left[\frac{\partial \text{Energy}(\boldsymbol{x}_{n+1},\boldsymbol{h})}{\partial\theta}\right] . \end{aligned} \tag{6}$$

where $\boldsymbol{x}_{n+1}$ is the last sample from the Gibbs chain starting from $\boldsymbol{x}_1$ obtained after n steps:

$$\boldsymbol{h}_1 \sim P(\boldsymbol{h}|\boldsymbol{x}_1)$$

$$\boldsymbol{x}_2 \sim P(\boldsymbol{x}|\boldsymbol{h}_1)$$

$$\ldots$$

$$\boldsymbol{h}_n \sim P(\boldsymbol{h}|\boldsymbol{x}_n)$$

$$\boldsymbol{x}_{n+1} \sim P(\boldsymbol{x}|\boldsymbol{h}_n) \ .$$

For binary RBMs, $E_{P(\boldsymbol{h}|\boldsymbol{x})}\left[\frac{\partial \text{Energy}(\boldsymbol{x},\boldsymbol{h})}{\partial \theta}\right]$ can be easily computed.

Several alternatives to CD$_n$ are Persistent CD (PCD) (Tieleman, 2008), Fast PCD (FPCD) (Tieleman & Hinton, 2009) or Parallel Tempering (PT) (Desjardins et al., 2010).

2.4. Monitoring the Learning Process in RBMs

Learning in RBMs is a delicate procedure involving a lot of data processing that one seeks to perform at a reasonable fast speed in order to be able to handle large spaces with a huge amount of states. In doing so, drastic approximations that can only be understood in a statistically averaged sense are performed (section 2.3).

One of the most relevant points to consider at the learning stage is to find a good way to determine whether a good solution has been found or not, and so to determine when should the learning process stop. One of the most widely used criteria for stopping is the reconstruction error, which is a measure of the capability of the network to produce an output that is consistent with the data at input. Since RBMs are probabilistic models, the reconstruction error of a data point $\boldsymbol{x}^{(i)}$ is computed as the probability of $\boldsymbol{x}^{(i)}$ given the expected value of $\boldsymbol{h}$ for $\boldsymbol{x}^{(i)}$:

$$R(\boldsymbol{x}^{(i)}) = P\left(\boldsymbol{x}^{(i)}|E\left[\boldsymbol{h}|\boldsymbol{x}^{(i)}\right]\right) \ , \tag{7}$$

which is the equivalent of the sum-of-squares reconstruction error for deterministic networks.

Some authors have shown that it may happen that learning induces an undesirable decrease in likelihood that goes undetected by the reconstruction error (Schulz et al., 2010; Fischer & Igel, 2010). It has been studied (Fischer & Igel, 2010) that the reconstruction error defined in (7) usually decreases monotonically. Since no increase in the reconstruction error takes place during training there is no apparent way to detect the change of behavior of the log-likelihood for CD$_n$.

3. Proposed Stopping Criteria

The proposed stopping criteria are based on the monitorization of the ratio of two probabilities: the probability of the data (that should be high) and the probability of points in the input space whose probability should be low. More formally, it can be defined as:

$$\xi = \frac{P(X)}{P(Y)} = \prod_{i=1}^{N} \frac{P(\boldsymbol{x}^{(i)})}{P(\boldsymbol{y}^{(i)})} \ , \tag{8}$$

where X stands for the complete training set of N samples and Y is a suitable artificially generated data set. The data set Y can be generated in different ways (see below).

The idea behind ξ comes from the fact that the standard gradient descent update rule used during learning requires from the evaluation of two terms: the *positive* and *negative* phases. The positive phase tends to decrease the energy (hence increase the probability) of the states related to the training data, while the negative phase tends to increase the energy of the whole set of states with the corresponding decrease in probability. In this way, if Y is selected so as to have low probability, the numerator in ξ is expected to increase while the denominator is expected to decrease during the learning process, making ξ maximal when learning is achieved.

Most relevant to the discussion is the fact that, being a ratio of probabilities computed at every step of the Markov chain built on-the-fly, the partition functions Z involved in $P(X)$ and $P(Y)$ cancel out in ξ. In other words, the computation of ξ can be equivalently defined as

$$\xi = \frac{P(X)}{P(Y)} = \prod_{i=1}^{N} \frac{\sum_{\boldsymbol{h}} e^{-\text{Energy}(\boldsymbol{x}^{(i)},\boldsymbol{h})}}{\sum_{\boldsymbol{h}} e^{-\text{Energy}(\boldsymbol{y}^{(i)},\boldsymbol{h})}} \ . \tag{9}$$

The particular topology of RBMs allows to compute $\sum_{\boldsymbol{h}} e^{-\text{Energy}(\boldsymbol{x},\boldsymbol{h})}$ efficiently. This fact dramatically decreases the computational cost involved in the calculation, which would otherwise become unfeasible in most real-world problems where RBMs could been successfully applied.

While $P(\boldsymbol{x}^{(i)})$ in ξ is directly evaluated from the data in the training set, the problem of finding suitable values for Y still remains. In order to select a point $\boldsymbol{y}^{(i)}$ with low probability, one may seek for zones of the space distant from $\boldsymbol{x}^{(i)}$, thus representing the complementary of the features to be learnt. This point should not be difficult to find. On the one hand, in small spaces one can enumerate the states. On the other hand, in large spaces with a small training set X the probability that a state picked up at random does not belong to X should be large. A second possibility is, for fixed $\boldsymbol{x}^{(i)}$, to suitably change the values of the hidden units during learning in such a way that they differ significantly from the values they should take during data reconstruction. We expect that, once learning is done, the reconstruction vectors should be independent of the value of the hidden units. However, this may not be the case while the system is still learning, as the basins of attraction of the energy functional depend explicitly on the values of the weights and bias terms, which can change significantly. This is in fact the main idea behind the stopping criteria proposed in this work, that we shall exploit in the following.

With all that in mind, two different alternatives have been explored:

i) $\boldsymbol{y}^{(i)} = E[\boldsymbol{x}|\boldsymbol{h}_s]$, where $\boldsymbol{h}_s$ is a random vector whose components are drawn from the uniform distribution in [0,1].

ii) $\boldsymbol{y}^{(i)} = E[\boldsymbol{x}|\boldsymbol{h}_s]$, where $\boldsymbol{h}_s = 1 - \boldsymbol{h}_1^{(i)}$, i.e., the complementary of the first hidden vector obtained in the Gibbs chain for $\boldsymbol{x}^{(i)}$.

Regarding the first alternative, random hidden vectors are expected to lead to regions of low reconstruction probability, at least while the system is still learning. In the second alternative, we expect that if a good reconstruction of $\boldsymbol{x}^{(i)}$ is achieved for a certain value of $\boldsymbol{h}_1^{(i)}$ (see Eq. (7)), the opposite should happen when $1 - \boldsymbol{h}_1^{(i)}$ is used instead.

Other related possibilities like monitoring the average value $E[\boldsymbol{h}|x_1^{(i)}]$ and using its complementary instead of $1 - h_1^{(i)}$ have also been explored and yield similar results to the ones shown in the following.

Notice that the reconstruction error only gathers information from the training set X, while the proposed estimator ξ in equation (8) samples also states from the rest of the input space.

4. Experiments

We performed several experiments to explore the aforementioned alternatives defined in section 3 and compare the behavior of the estimator ξ to that of the actual *log-likelihood* and the reconstruction error in a couple of problems.

The first problem, denoted *Bars and Stripes* (BS), tries to identify vertical and horizontal lines in 4×4 pixel images. The training set consists in the whole set of images containing all possible horizontal or vertical lines (but not both), ranging from no lines (blank image) to completely filled images (black image), thus producing $2 \times 2^4 - 2 = 30$ different images (avoiding the repetition of fully back and fully white images) out of the space of 2^{16} possible images with black or white pixels. The second problem, named *Labeled Shifter Ensemble* (LSE), consists in learning 19-bit states formed as follows: given an initial 8-bit pattern, generate three new states concatenating to it the bit sequences 001, 010 or 100. The final 8-bit pattern of the state is the original one shifting one bit to the left if the intermediate code is 001, copying it unchanged if the code is 010, or shifting it one bit to the right if the code is 100. One thus generates the training set using all possible $2^8 \times 3 = 768$ states that can be created in this form, while the system space consists of all possible 2^{19} different states one can build with 19 bits. These two problems have already been explored in (Fischer & Igel, 2010) and are adequate in the current context since, while still large, the dimensionality of space allows for a direct monitorization of the partition function and the log-likelihood during learning.

In the following we discuss the learning processes of both problems with single RBMs, employing the Contrastive Divergence algorithm CD$_n$ with $n = 1$ and $n = 10$ as described in section 2.3. In all cases, binary visible and hidden units were used. In the BS case the RBM had 16 visible and 8 hidden units, while in the LSE problem these numbers were 19 and 10, respectively. Every simulation was carried out for a total of 50000 epochs, with measures being taken every 50 epochs. Moreover, every point in the subsequent plots was the average of ten different simulations starting from different random values of the weights and bias. Other parameters affecting the results that were changed along the analysis are the learning rate (LR) involved in the weight and bias update rules and a weight decay parameter (WD) that prevents weights from achieving large values that would saturate the sigmoid functions present in the analytical expressions associated to binary RBMs.

We present the results for the two problems at hand, showing for each instance analyzed three different plots corresponding to the actual log-likelihood of the problem, $log(\xi)$ (ξ as defined in (9)) and the logarithm of the reconstructed error (7), all three quantities monitored during the learning process.

Figure 1 shows results for the BS problem using the alternatives i) and ii) defined in section 3 using CD$_1$ with LR=0.01 and WD=0. The left panel corresponds to alternative i) and the right panel corresponds to alternative ii). As can be seen, the log-likelihood increases very rapidly, reaches a maximum and then starts decreasing, thus indicating that further learning only worsens the model. In both cases, though, the log probability of the reconstruction converges towards a constant value (very near 0, indicating high probabilities for the reconstructed data), giving the false impression that going on with the learning process will neither improve nor worsen the predictions produced by the network. Interestingly enough, though, the middle

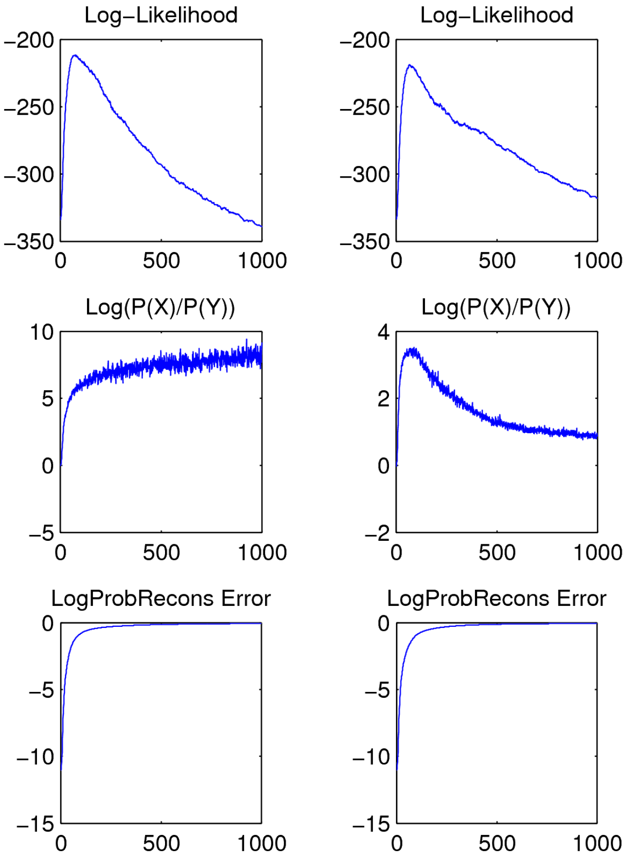

Figure 1. Log-likelihood, $\log(\xi)$ and log-probability of the reconstruction (upper, middle and lower panels) for the BS problem. Left and right columns correspond to options i) and ii) when choosing values for the hidden units using CD$_1$ with LR=0.01 and WD=0.

plot on the right panel indicate that ii) is able to capture the increasing and decreasing behavior of the log-likelihood, a feature that i) seems to miss. At this point it looks like ii) is a better estimator of optimal log-likelihood than the reconstruction error. This same behaviour is seen in figure 2 where a weight decay value WD=0.001 is employed.

The LSE problem yields somewhat similar results under the same learning and monitoring conditions. The log-likelihood, $log(\xi)$ and log-reconstruction error are shown as before in the upper, middle and lower panels of figure 3, with options i) and ii) on the left and right, respectively. In this case the learning rate has been set to LR=0.001 (otherwise the log-likelihood of the problem decreases monotonically). In this case, however, both estimators i) and ii) are able to find the region where the log-likelihood is maximal, decreasing similarly to the later when this point is surpassed.

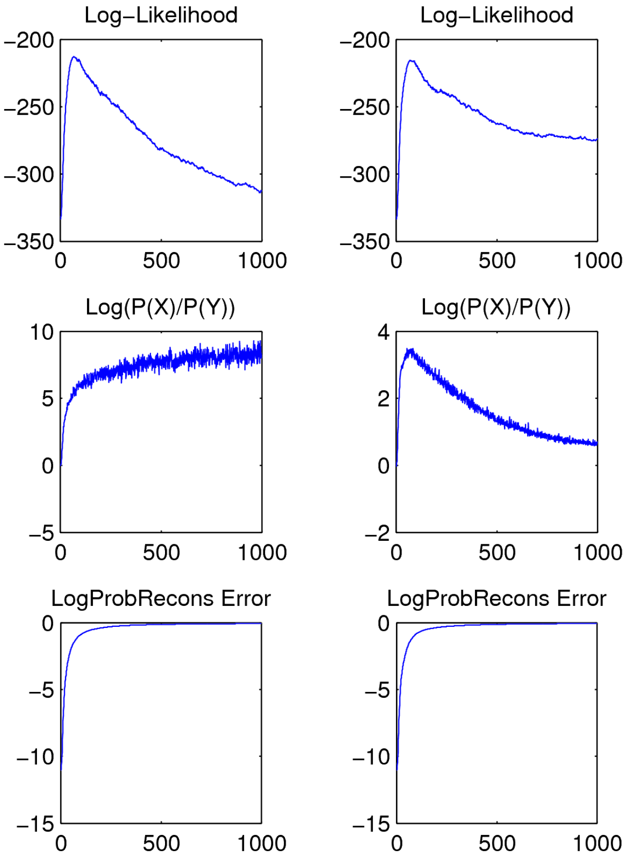

Figure 2. Same as in figure 1 but with WD=0.001.

These results seem to indicate that estimator ii) is more robust than estimator i). Still, these two are better than the reconstruction error which always present a similar pattern, both for the BS and LSE problems, with a transient regime that always stabilizes to a plateau that apparently has little to do with the actual behavior of the log-likelihood.

All these results have been obtained in the CD$_1$ approximation. Since it is known that CD$_n$ with increasing n can lead to better learning results because of the increased statistical independence of the input and output values generated, estimators i) and ii) can also be used in this case. We have checked their performance using CD$_{10}$ on the same two problems at hand. Results for the LSE problem using CD$_{10}$, LR=0.01 and WD=0 are shown in the left and right panels of figure 4 for estimators i) and ii), respectively. In this case, none of the estimators is able to detect the region

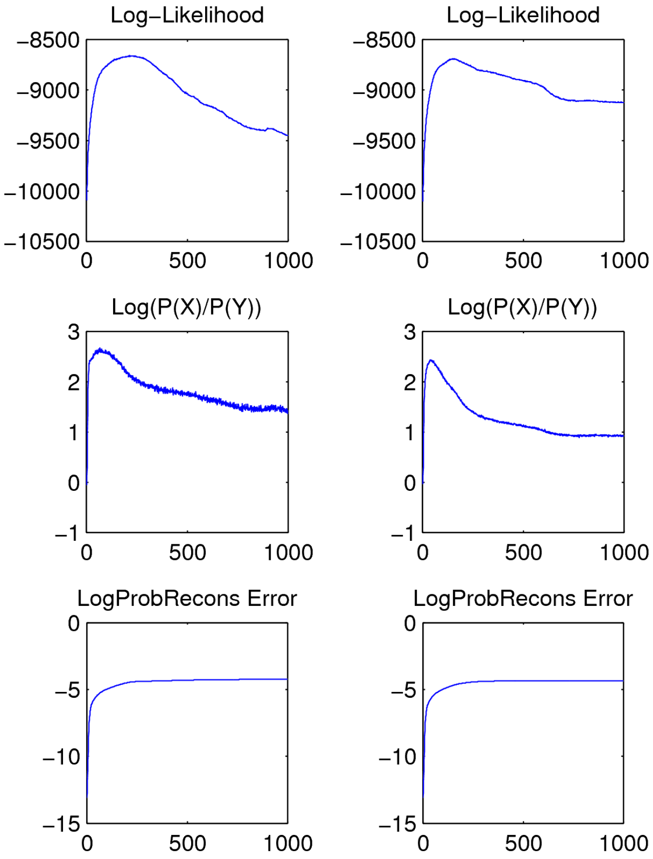

Figure 3. Results for the LSE problem as reported in figure 1 for CD_1, LR=0.001 and WD=0.001.

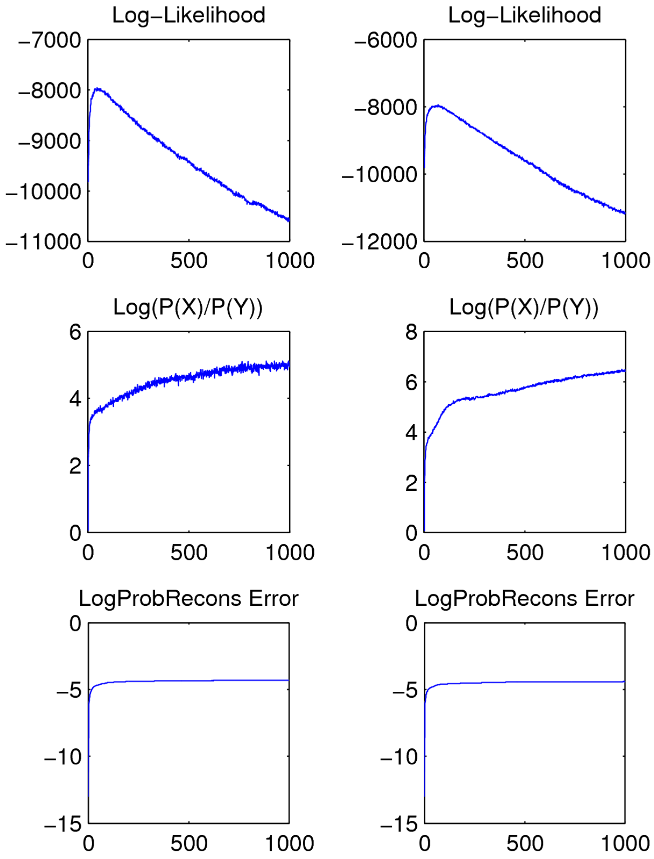

Figure 4. Results for the LSE problem as reported in figure 1 for CD_{10}, LR=0.01 and WD=0.

of maximal likelihood, stressing that none of these shall be used as a test to stop the learning algorithm. However, the reconstruction error has a similar behavior, thus indicating that it is not a good testing quantity either. Similar results for the BS problem are obtained when using CD_{10}. A possible explanation can be related to the fact that the Markov chain involved in the process tends to lose memory with increasing number of steps. Therefore, ξ is computed with more independent data in CD_{10} than in CD_1. Anyway, the behavior of the proposed criteria with CD_{10} should be further studied.

As a final remark, we note that for the BS problem the trained RBM stopped using the proposed criteria is able to qualitatively generate samples similar to those in the training set. We show in figure 5 the complete training set (two upper rows) and the same number of generated samples obtained from the RBM stopped after 3000 epochs in the training process using CD_1 as discussed above, corresponding to the maximum value of the proposed criterion ii), which coincides with the optimal value of the log-likelihood (two lower rows in the same figure).

5. Conclusion

Based on the fact that learning tries to increase the contribution of the relevant states while decreasing the rest, two new estimators based on the ratio of two probabilities have been proposed and discussed as an alternative to the reconstruction error. It has been shown that the better one, obtained by replacing the value of the (binary) hidden units $\boldsymbol{h}$ by $1 - \boldsymbol{h}$, can at some point be able to monitor the actual behavior of the log-likelihood of the model without additional computational cost. This estimator works well for CD_1 but for CD_{10}, which is considered to yield better learning results at the expense however of a linear increase in computational cost. We believe that the use of the estimator presented here in CD_1 learning problems provides a

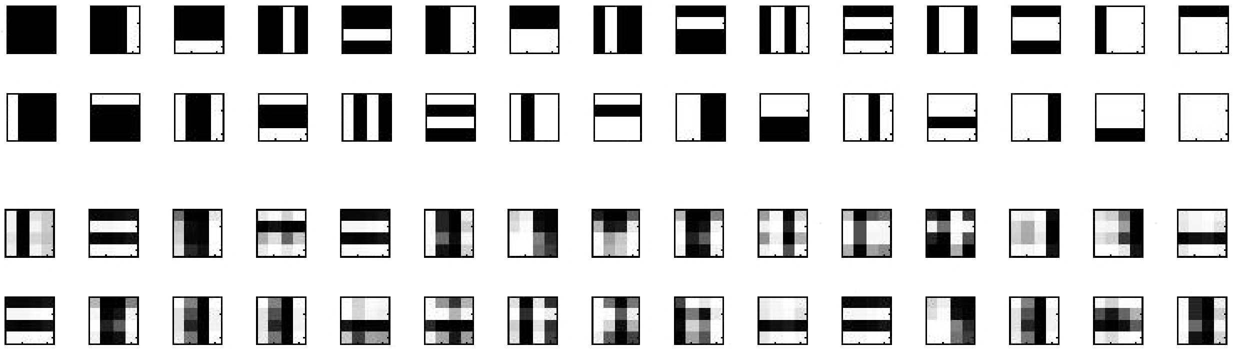

Figure 5. Training data (two upper rows) and generated samples (two lower rows) for the BS problems after 3000 epochs in the training process using CD_1.

faster stopping criteria for the learning algorithm that can yield results compatible in quality to those obtained in standard CD_n learning for moderate n. Future work along this line will be carried out in an attempt to formalize that statement.

Acknowledgments

JD: This work was partially supported by MICINN project TIN2011-27479-C04-03 (BASMATI) and by SGR2009-1428 (LARCA).

FM: This work has been supported by grant No. FIS2011-25275 from DGI (Spain) and Grant No. 2009-SGR1003 from the Generalitat de Catalunya (Spain).

ER: This research is partially funded by Spanish research project TIN2012-31377.

References

Bengio, Y. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009.

Bengio, Y. and Delalleau, O. Justifying and Generalizing Contrastive Divergence. *Neural Computation*, 21(6): 1601–1621, 2009.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. Greedy Layer-wise Training of Deep Networks. In *Advances in Neural Information Processing (NIPS'06)*, volume 19, pp. 153–160. MIT Press, 2007.

Carreira-Perpiñán, M. A. and Hinton, G. E. On Contrastive Divergence Learning. In *International Workshop on Artificial Intelligence and Statistics*, pp. 33–40, 2005.

Desjardins, G., Courville, A., Bengio, Y., Vincent, P., and Delalleau, O. Parallel Tempering for Training of Restricted Boltzmann Machines. In *13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 145–152, 2010.

Fischer, A. and Igel, C. Empirical Analysis of the Divergence of Gibbs Sampling Based Learning Algorithms for Restricted Boltzmann Machines. In *International Conference on Artificial Neural Networks (ICANN)*, volume 3, pp. 208–217, 2010.

Geman, S. and Geman, D. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 6(6):721–741, nov 1984.

Hinton, G. E. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14:1771–1800, 2002.

Hinton, G. E. and Salakhutdinov, R. R. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, 2006.

Hinton, G. E., Osindero, S., and Teh, Y. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18(7):1527–1554, 2006.

Larochelle, H., Bengio, Y., Lourador, J., and Lamblin, P. Exploring Strategies for Training Deep Neural Networks. *Journal of Machine Learning Research*, 10:1–40, 2009.

Le, Q. V., Ranzato, M. A., Monga, R., Devin, M., Chen, K., Corrado, G. S., and Ng, A. Y. Building High-level Features Using Large Scale Unsupervised Learning. In *29th International Conference on Machine Learning*, 2012.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations. In *International Conference on Machine Learning*, pp. 77, 2009.

MacKay, D. J. C. Failures of the one-step learning algorithm, 2001. Unpublished Technical Report.

Neal, R. M. Annealed Importance Sampling, 1998. Technical Report 9805, Dept. Statistics, University of Toronto.

Rumelhart, David E., Hinton, Geoffrey E., and Williams, R. J. Learning Internal Representations by Error Propagation. In Rumelhart, D. E., McClelland, J. L., and the PDP research group. (eds.), *Parallel distributed processing: Explorations in the microstructure of cognition, Volume 1: Foundations*. MIT Press, 1986.

Schulz, H., Müller, A., and Behnke, S. Investigating Convergence of Restricted Boltzmann Machine Learning. In *NIPS 2010 Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.

Smolensky, P. Information Processing in Dynamical Systems: Foundations of Harmony Theory. In Rumelhart, D. E. and McClelland, J. L. (eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition (vol. 1)*, pp. 194–281. MIT Press, 1986.

Tieleman, T. Training Restricted Boltzmann Machines using Approximations to the Likelihood Gradient. In *25th International Conference on Machine Learning*, pp. 1064–1071, 2008.

Tieleman, T. and Hinton, G. E. Using Fast Weights to Improve Persistent Contrastive Divergence. In *26th International Conference on Machine Learning*, pp. 1033–1040, 2009.

Yuille, A. The Convergence of Contrastive Divergence. In *Advances in Neural Information Processing Systems (NIPS'04)*, volume 17, pp. 1593–1600. MIT Press, 2005.