A Hierarchy of Recurrent Networks for Speech Recognition

Benjamin Schrauwen
Electronics and Information Systems Department
Ghent University
B-9000 Ghent, Belgium
benjamin.schrauwen@ugent.be

Lars Büsing
Institute for Theoretical Computer Science
Graz University of Technology
A-8010 Graz, Austria
lars@igi.tugraz.at

Abstract

Generative models for sequential data based on directed graphs of Restricted Boltzmann Machines (RBMs) are able to accurately model high dimensional sequences as recently shown. In these models, temporal dependencies in the input are discovered by either buffering previous visible variables or by recurrent connections of the hidden variables. Here we propose a modification of these models, the Temporal Reservoir Machine (TRM). It utilizes a recurrent artificial neural network (ANN) for integrating information from the input over time. This information is then fed into a RBM at each time step. To avoid difficulties of recurrent network learning, the ANN remains untrained and hence can be thought of as a random feature extractor. Using the architecture of multi-layer RBMs (Deep Belief Networks), the TRMs can be used as a building block for complex hierarchical models. This approach unifies RBM-based approaches for sequential data modeling and the Echo State Network, a powerful approach for black-box system identification. The TRM is tested on a spoken digits task under noisy conditions, and competitive performances compared to previous models are observed.

1 Introduction

Restricted Boltzmann Machines (RBMs) are rather simple generative models for which an efficient maximum likelihood learning algorithm, contrastive divergence (CD), exists (see [1]). RBMs turned out to be suitable building blocks for more complex and very powerful probabilistic models obtained by stacking them “vertically”, yielding a hierarchical model called a Deep Belief Network (DBN, see [2]). The DBN essentially benefits from the capability of a RBM to learn a simpler representation of its visible variables (its input). This simplified representation is appropriate to serve as the visible variables in the next higher layer in the hierarchy of the DBN. Beyond modeling static, i. e. iid. data, RBMs have also been used for building models of sequential data, i. e. of data which features an inherent temporal dimension (e. g. videos, speech recordings). Conditional RBMs (CRBMs), where previous visible variables (from a time window of fixed size) determine the biases for the variables of the current time slice, have been shown in [3] to accurately model motion capture data of humans. In [4], Recurrent Temporal RBMs (RTRBMs), which integrate information over time via recurrent connections of the hidden variables, can generate faithful video sequences of bouncing balls illustrating their modeling power for high dimensional time series.

In contrast to these RBM-based *probabilistic* models for time series, the Echo State Network (ESN), introduced in [5], a powerful technique for black-box system identification, classification etc., is rooted mainly in *dynamical systems theory*. An ESN can be interpreted as computing random features of the input sequence via a randomly connected recurrent artificial neural network (ANN). These features are then fed into a simple memoryless readout (linear classifier, linear regressor) which is trained in a supervised way. For several tasks (chaotic attractor learning etc., see [5, 6]) ESNs are state-of-the-art and unsurpassed in accuracy while being very efficient to train.

In this paper we propose a probabilistic model, the Temporal Reservoir Machine (TRM), which operates on sequential data and aims at unifying the advantages of RBM-based models and ESNs. It utilizes a random recurrent network as a feature extractor and models the time series by a directed graph of RBMs. The architecture is tested on the classification of spoken digits.

2 Existing Models

Basic building block: Restricted Boltzmann Machine

The Restricted Boltzmann Machine [1] is an undirected generative model involving the visible (observable) random variables (RVs) V, which may be real or binary, and the hidden (latent) variables H, which are commonly restricted to be binary. The RBM is defined by the joint distribution (for real visible RVs V and binary hidden RVs H):

$$p(V,H) = Z^{-1} \exp\left(-\|V\|^2/2 + H^\top W^{\mathrm{HV}} V + B_H^\top H + B_V^\top V\right),$$

where Z is the so-called partition function ensuring normalization and $X^\top$ denotes the transposed of a vector X. B_H and B_V are the biases of the visible and hidden RVs and W^{HV} denotes the weight matrix from the visible to the hidden RVs. Due to the bilinear form of the exponent of $p(V,H)$, the conditional distributions $p(H|V)$ and $p(V|H)$ both factorize and hence make inference in a RBM easy as “explaining away” does not occur. Learning in RBMs is most commonly done by contrastive divergence (CD), an algorithm which efficiently approximates the gradient of the training data likelihood.

RBM-based models for sequential data

We use the following notation. The N_V real-valued visible variables at time t are denoted as $V_t \in \mathbb{R}^{N_V}$ and the corresponding N_H binary hidden variables as $H_t \in \{0,1\}^{N_H}$. The visible variables from the time step τ up to t are concatenated into a matrix $V_\tau^t = (V_1, \ldots, V_t)$; H_τ^t is defined analogously for the hidden variables.

RBMs have been used as building blocks in directed graphical models for time series modeling. The basic idea is that at every time step a RBM models the corresponding visible and hidden variables. Visible or hidden RVs of previous time steps influence the distribution of the current RVs by determining their biases B_H and B_V via directed connections. Conditional RBMs (CRBMs), introduced for time series modeling in [3], were shown to be able to accurately model human motion capture data. In CRBMs only directed connections from the visible variables V_{t-m}^t to V_t and H_t are considered, where $m \in \mathbb{N}$ is called the model order. The graphical representation of this model is shown in Fig. 1 panel **A**. Hence in CRBMs the memory depth, the window of temporal information integration, is explicitly given by m. If the data time series exhibits dependencies on long time scales, a high model order has to be chosen resulting in a large number of weights to be learned (linear in m).

A different model using RBMs, termed Recurrent Temporal RBM (RTRBM), was presented in [4]. In the RTRBM only connections from H_{t-1} to H_t are considered. To facilitate learning in this model, the activations of the hidden units are determined by a deterministic mean field update. The recurrent weights from H_{t-1} to H_t are then learned by backpropagation through time (BPTT), which was introduced in the context of recurrent neural networks in [7]. RTRBMs do not have a fixed memory depth, they implicitly integrate information over time in the activations of the hidden units, enabling the model to learn long temporal relationships with a limited number of parameters. However, as known from literature, BPTT often suffers from slow convergence due to local minima and bifurcations in the network phase space.

Echo State Networks and Reservoir Computing

Echo State Networks (ESNs), introduced in [5], represent an approach for utilizing recurrent ANNs for online/any-time computations (regression, classification, prediction/generation) on time series based on the following principle. The input sequence is fed into a recurrent ANN which is not trained for the specific task at hand and hence can be regarded as randomly structured wrt. this task. A memoryless linear readout (or a linear classifier) is then trained on the current network state in order to carry out the target computation. In contrast to learning in recurrent networks with BPTT, learning in ESNs only involves linear regression and is therefore cheap, robust and not plagued by local minima. The "random preprocessing" in time performed by an ESN is a very powerful tool, especially for prediction of time series based on chaotic attractors. In this context it has been shown that appropriately generated networks for ESNs can store a large amount of information about past inputs, allowing ESNs to model temporal dependencies on long time scales (see [8]). Further, the high performance of ESNs indicates that, for some tasks, random features (at least when generated in a suitable way) are superior to features learned in a supervised way especially if little training data/time is available. The basic idea of ESNs has been generalized to a framework termed Reservoir Computing (RC): The recurrent ANN of an ESN is replaced by any non-autonomous dynamical system (the so-called reservoir) which implements a filter of its input. The target function is then approximated by training a readout device on the output of this filter (the current system state).

3 Temporal Reservoir Machine

Guided by the advantageous and disadvantageous of previous approaches for time series modeling summarized briefly above, we introduce a modification of these models called Temporal Reservoir Machine (TRM). The TRM is aimed at overcoming the limitations of a fixed model order (memory depth) of the CRBM and at the same time circumventing the problems associated with learning via BPTT which occur in the RTRBM, by using strategies from RC.

Definition of the TRM

The basic idea of the TRM is the following. In addition to the visible and hidden RVs we introduce bias variables B_t for each time step t. These B_t determine the biases of the visible and hidden variables V_t, H_t of the RBM at time step t via a linear mapping. Further, B_t is given by a deterministic function of the previous visible variables V_1^{t-1}. Thus the bias RVs B_t contain information of the past input data V_1^{t-1} that can be used by the RBM at time slice t for modeling the course of the time series. We assume $B_t \in [-1,1]^{N_B}$, where N_B is the dimension of B_t.

Inspired by the ESN, we consider here the setup where B_t is given by the activation of a recurrent ANN that receives the visible variables V_t as input at time t. We call this network the reservoir of the TRM. In the spirit of the ESN approach, the reservoir weight matrix W^{BB} (the recurrent connections from B_{t-1} to B_t) and the input matrix W^{BV} (the connections from V_t to B_t) are randomly generated and they remain untrained. Only the connections from the reservoir to the visible and hidden variables W^{VB}, W^{HB} (the connections from B_t to V_t and from B_t to H_t respectively) are adapted (see below). Thus B_t can be interpreted as a collection of random features of the history of the visible variables V_1^{t-1}. A full RBM is kept at each time step for modeling the joint distribution of H_t and V_t given the bias RVs B_t. Formally the TRM is defined for a time series of length T in the following way:

$$\begin{aligned}
p(V_1^T, H_1^T, B_1^T) &= \prod_{t=1}^{T} p(V_t, H_t, B_t | V_1^{t-1}, H_1^{t-1}, B_1^{t-1}) \\
&= \prod_{t=1}^{T} p(V_t, H_t | B_t) p(B_t | V_{t-1}, B_{t-1}) \\
p(V_t, H_t | B_t) &:= Z^{-1} \exp\left(-\|V_t\|^2/2 + H_t^{\mathsf{T}} W^{\mathrm{HV}} V_t + H_t^{\mathsf{T}} W^{\mathrm{HB}} B_t + V_t^{\mathsf{T}} W^{\mathrm{VB}} B_t\right) \\
p(B_t | V_{t-1}, B_{t-1}) &:= \delta(B_t - f(V_{t-1}, B_{t-1})),
\end{aligned}$$

The function $f(V_{t-1}, B_{t-1})$ represents the update of the recurrent network (the reservoir) which determines the deterministic update for B_t (formulated here with Dirac's δ).

If the update function (the dynamical system) f is chosen such that B_t is a simple delay line with memory depth m (that stores the last m visible variables V_{t-m}^{t-1}) then the TRM architecture is equivalent the CRBM as applied to sequential data in [3]. It is of course possible to generalize the TRM model by replacing f (now implementing an ANN) with any input/output system that generates suitable features.

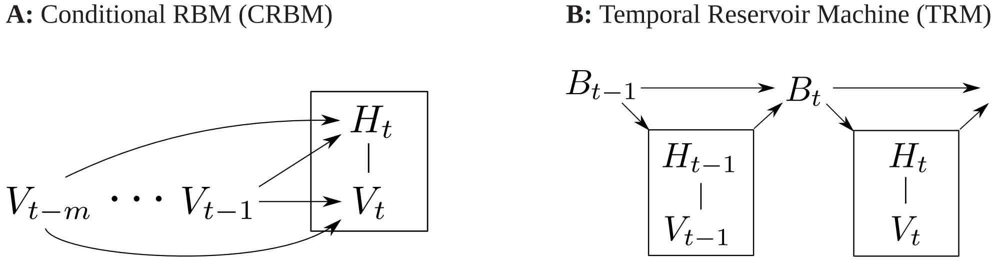

Figure 1: **A:** The graphical structure of the CRBM. Previous visible variables V_{t-m}^{t-1} are buffered and determine the biases of the visible and hidden RVs at the current time step. **B:** The graphical structure of the TRM. A deterministic recurrent network with state B_t integrates information from past visible V_1^{t-1} and sets the biases for the hidden variables H_t of the RBM at time t.

In this paper the update function f will be of the following form:

$$f(V_{t-1}, B_{t-1}) \quad := \quad (1-\lambda)B_{t-1} + \lambda \cdot \tanh\left(W^{\mathrm{BB}} B_{t-1} + W^{\mathrm{BV}} V_{t-1}\right),$$

where $\tanh$ denotes the hyperbolic tangent applied elementwise to its input. The parameter λ is called leak rate and it has a strong influence the memory depth (large λ favors memory on longer time scales). There exists a number of powerful heuristics in the RC literature concerning an appropriate distribution for the weight matrices W^{BB} and W^{BV}. Most commonly, these matrices are generated probabilistically with carefully chosen spectral properties. In the experiments reported in this paper the entries of W^{BB} and W^{BV} are iid. according to $\mathcal{N}(0,1)$ and all weights with an absolute value smaller than unity are set to zero. Further, the matrix W^{BB} is then scaled such that its spectral radius is given by a parameter σ. The matrix W^{BV} is scaled by a parameter σ_{BV}

Learning in a TRM

Learning the weights W^{HB} and W^{VB} from the recurrent network state B_t to the visible and hidden variables V_t and H_t in a TRM is done by maximum likelihood learning. The log likelihood $\mathcal{L}$ of the training data V_1^T for the TRM takes a simple factorized form. Let $B(V)_1^T$ denote the values of the variables B_1^T at step t that result (deterministically) from visible variables V_1^T via the update function f. The total likelihood $\mathcal{L}$ factorizes into likelihoods $\mathcal{L}_t$ for every time step t:

$$\begin{aligned} \mathcal{L} \quad &= \quad \log p(V_1^T) = \log\left(\sum_{H_1^T, B_1^T} p(V_1^T, H_1^T, B_1^T)\right) = \log\left(\sum_{H_1^T} p(V_1^T, H_1^T, B(V)_1^T)\right) \\ &= \quad \sum_{t=1}^{T} \log\left(\sum_{H_t} p(V_t, H_t | B(V)_t)\right) = \sum_{t=1}^{T} \mathcal{L}_t. \end{aligned}$$

The log likelihood $\mathcal{L}_t$ for each time step t is that of an ordinary RBM with biases $W^{\mathrm{VB}} B(V)_t$, $W^{\mathrm{HB}} B(V)_t$ of the visible and hidden RVs. Hence learning is similar to learning in RBMs and can be made stepwise at each time step:

$$\begin{aligned} \Delta W^{\mathrm{HV}} \quad &\propto \quad \sum_{t=1}^{T} \left(\langle H_t \otimes V_t \rangle_0 - \langle H_t \otimes V_t \rangle_\infty\right) \\ \Delta W^{\mathrm{VB}} \quad &\propto \quad \sum_{t=1}^{T} \left(\langle V_t \rangle_0 - \langle V_t \rangle_\infty\right) \otimes B(V)_t \\ \Delta W^{\mathrm{HB}} \quad &\propto \quad \sum_{t=1}^{T} \left(\langle H_t \rangle_0 - \langle H_t \rangle_\infty\right) \otimes B(V)_t, \end{aligned}$$

where $\otimes$ denotes the outer product. The operators $\langle \cdot \rangle_0$ and $\langle \cdot \rangle_\infty$ are the expectation operators of the RBM with clamped visible variables to the training data and of the RBM equilibrium distribution respectively. The latter is approximated by a sample that is obtained after n-step Gibbs sampling, yielding the so called contrastive divergence learning CD-n which has been successfully applied in numerous studies.

Sampling from a TRM

Ancestral sampling from the TRM can be done in a straight-forward manner:

Sampling:

```
1.)  initialize $B_1$
2.)  for $t \in \{1, \ldots, T\}$ do:

       • sample $V_t, H_t \sim p(V_t, H_t | B_t)$, RBM sampling step
       • update $B_{t+1} = f(V_t, B_t)$, network update step
```

Normally, the reservoir is initialized randomly, e. g. B_1 is drawn uniformly from $[-1, 1]^{N_B}$. The "`RBM sampling step`" is done by prolonged Gibbs sampling analogously to sampling in RBMs.

Inference in a TRM

Exact inference in the TRM is tractable, since $p(H_1^T | V_1^T) = \prod_{t=1}^{T} p(H_t | V_t, B_t(V))$ factorizes as the bias variables B_1^T are deterministic. The following algorithm sums up inference in the TRM:

Inference:

```
1.)  initialize $B_1$
2.)  for $t \in \{1, \ldots, T\}$ do:

       • for $i \in \{1, N_H\}$ evaluate
         $p((H_t)_i = 1 | V_t, B_t) = \sigma(\sum_j W_{ij}^{\mathrm{HV}} (V_t)_j + \sum_j W_{ij}^{\mathrm{HB}} (B_t)_j)$, inference step
       • update $B_{t+1} = f(V_t, B_t)$, network update step
```

Here $\sigma(x) = (1 + \exp(-x))^{-1}$ denotes the logistic function.

Multi-layer TRMs

Mimicking the hierarchical structure of a DBN, TRMs can be stacked yielding a multi-layer model for sequential data analogous to the CRBM. Learning is performed layer-wise ("greedy"), i. e. at each layer a full TRM is trained using the inferred distribution over the binary hidden variables of the next lower layer as real-valued input data. The rational behind this hierarchical structure is that higher levels are thought to be advantageous for representing more abstract features and statistical relationships on longer time scales. Sampling from the multi-layer model is then performed by sampling from the undirected TRM at the top layer and a down-pass using directed connections.

Reducing the dimension of the ESN input

In general, input is fed into an ESN via a randomly generated input matrix, denoted here as W^{BV}. This works well if the input dimension is reasonably low. If however the input dimension is high, especially if there is a lot of redundancy across input dimensions, such a random input mapping will drastically degrade the short-term memory of the ESN. Intuitively speaking, this is because redundant information is pushing the ESN dynamics in random directions, thereby destroying memory of past inputs. Therefore directly applying an ESN as the reservoir in higher layers of a TRM, where the visible layers often have a high dimension (300 to 1000 in our experiments, see below), will lead to a poor modeling performance because all memory of previous visibles is quickly lost.

A straightforward solution for solving this problem is to apply standard dimensionality reduction techniques such as Principle Component Analysis to first significantly reduce the dimension of the visibles before feeding them into the ESN. As we will show in the experiments below, this procedure can drastically increase the performance.

4 Speech recognition experiments

Here we present results of computer experiments with the TRM where we illustrate the classification of spoken digits with the TRM. Training was done with CD-1 and a momentum of 0.9 was used.

The learning rates were set to $0.2 \cdot 10^{-3}$ and $5 \cdot 10^{-3}$ for the weights W^{VB} and W^{HB} respectively in the first layer and to $0.2 \cdot 10^{-2}$ and $5 \cdot 10^{-2}$ in the second layer.

The spoken digits data set was constructed by taking the isolated digits from the TI46 data set (which is a subset of the well known TIDIGITS corpus) and concatenating them into a continuous stream with random pauses between the digits. The data consists of 10 utterances of 10 digits by 5 different female speakers, resulting in 500 spoken digits in total. Training was performed on 300 randomly drawn clean utterances and the data was labeled up to the time step level. For testing we used the remaining 200 utterances and added 5 dB of babble noise from the NOISEX noise data set[1]. This generates a high-noise condition using noise which has the same spectral properties as the digits which need to be recognized. During testing we assumed that the segmentation of the data was given.

We pre-processed the speech stream using the Lyon Passive Ear model [9]. The model consists essentially of a filter bank which closely resembles the selectivity of the human ear to certain frequencies, followed by a series of half-wave rectifiers and adaptive gain controllers both modeling the hair cell response in the human ear. After the pre-processing, the data is down-sampled to 94Hz, and the number of frequency bands is 39. The Lyon Passive Ear model has previously been used as pre-processing in a RC-based speech recognition setup [10]. The more classic MFCC speech front-end which is the standard for HMM-based speech processing is not well suited for ESN-based speech recognition as the MFCC front-end has been specifically engineered to circumvent some of the problems of HMMs, and this is poorly matched to the way ESNs process spatiotemporal data.

We trained a two-layer TRM with 300 hidden units and a reservoir of 600 units in both layers. We added 10 binary visibles (and an additional silence label unit) to the second visible layer which are used as label nodes, similar to the setup in [2]. After training the first layer in an unsupervised way, the second layer is trained to jointly model the data and the labels. We trained for 150 epochs without fine-tuning by a wake-sleep or backpropagation algorithm. The word error rate (WER) when sampling (using 30 steps of Gibbs sampling) from the TRM while clamping the lowest layer to the input data is 14% (with a STD of 6%, best of 4.7%, averaged over 20 runs). When using the model fully deterministically and training a linear readout function on the 300 hidden units of the top layer, a WER of 13% (with a STD of 4%, best of 5.6%, averaged over 20 runs) is obtained. Learning the same architecture, but where the reservoir ANN is replaced by a delay line of order three yielding a CRBM, results in a sampling-based WER of 70.2%, and with a linear deterministic classification in a WER of 38.7%.

We also compare the results to standard ESNs, which have shown improved noise robustness over HMMs in previous work [11]. Applying a well-tuned ESN for classification with a network size of 600 neurons to the pre-processed data, yields a WER of 9.1% (averaged over 20 runs). For all experiments with the TRM and the ESN the spectral radius was set to $\sigma = 0.9$, the leak rate to $\lambda = 0.8$ and the input scaling to $\sigma_{\mathrm{BV}} = 0.5$.

When classifying the digit stream (without pre-processing with Lyon Passive Ear model) using a standard MFCC-HMM classifier [2], we get a WER of 36.6%; but note that the HMM classifier was applied on the unsegmented digit stream.

When we compare single layer vs. two-layer models for smaller TRMs (120-node reservoir and 300 hidden units per layer), the classification performance with the Hinton-like classification is clearly better for the two layer variant (WER 26.3%, STD 6.0%) than for the single layer (55.2%, STD 6.8%) TRM.

When comparing to a CRBM of order 10 in the first layer and order 3 in the second layer (300 hidden units in each layer) it slightly more parameters than the reported TRM and performs significantly worse: with a linear readout: WER 65.3%, STD 8.2%; with Hinton-style classification: WER: 77.6%, STD 10.0%.

[1] Available online at *http://spib.rice.edu/spib/select_noise.html*

[2] For this we trained a HMM using the HTK toolkit on data from the AURORA corpus. The AURORA corpus is the TIDIGITS corpus together with standardized noise condition for training and testing. We trained using a “multi-condition” setup which means that training occurs using several noisy conditions. This classifier is thus trained on more data and is optimized for noise robustness.

When applying the PCA dimensionality reduction to the input of the ESN in the second TRM layer, we find that the 40 principal components capture almost all variance. Feeding these 40 latent variables to the ESN instead of the 300 hiddens allows the ESN to have a much longer short-term memory. When training a two layer TRM using this technique, we get a WER of 7.0% with a linear readout, and 11.7% using a Hinton-style classification. This dimensionality reduction will become even more important in larger TRM models with much wider hidden layers.

These results show the power of multi-layer TRMs for modeling and classifying pre-processed data streams of spoken digits. Especially the comparison to the CRBM, which has a simple delay line instead of a recurrent network, emphasizes the usefulness of filtering the data through a recurrent (untrained) ANN.

5 Discussion

We proposed a probabilistic generative model for sequential data, the TRM, that extends previously published models based on directed graphs of RBMs. The TRM takes advantage of two insights that emerged in a branch of the recurrent ANN literature, namely the Echo State Network / Reservoir Computing literature. First, randomly generated, untrained recurrent ANNs can store a considerable amount of information from the input history and this information can easily be extracted by a simple, memoryless linear readout. Second, such random ANNs (if generated appropriately) also compute interesting transformations of the input, i. e. useful features, that are adequate for computations (e. g. classification) and/or for being fed into higher stages of an architecture. This is consistent with the observation (see e. g. [12]) that random (untrained) feature extractors can be more powerful than feature extractors which are trained in a supervised way, given little labeled training data is available. The TRM makes use of these insights by filtering previous visible variables through a random recurrent network and making this transformed data available to the undirected model at the current time step. The simulation results reported in this paper on classification of spoken digits, where TRMs outperform CRBMs and ESNs, underline the benefits of combining the probabilistic RBM framework with the random recurrent network approach of Echo State Networks.

The TRM model has a couple of favorable properties. Compared the the CRBM (which can be interpreted as a special case of the TRM) the “memory depth” is not restricted, hence the TRM can more easily model statistical relationships which extend over long time scales. Furthermore, in contrast to the RTRBM the TRM circumvents learning with BPTT, an algorithm that is able to produced good results but which may suffer from slow convergence due to vanishing gradients and bifurcations in the network phase space. Instead of learning the recurrent weights, the TRM uses manifold random features of the input history (that may well be redundant to some extend) which are made available to the probabilistic model at the current time step. It is thus clear that the TRM is not the “smallest” model for time series which uses RBMs. Rather, the TRM trades model size against training time by introducing the bias variables B_t.

The multi-layer TRM is a possible architecture for implementing a hierarchical model based on the Echo State Network / Reservoir Computing idea. Previous attempts to construct such a model consisting of multiple layers of ESNs (or other RC models) failed due to the lack of efficient principles for learning hidden variables/features in an unsupervised way which can be passed from lower to higher layers (personal communication from a collaborating lab). This problem is tackled in the TRM by utilizing the powerful probabilistic RBM for which a greedy, fast and accurate maximum likelihood learning algorithm exists. The encouraging simulation results of the TRM provided in this article recommend the TRM architecture as a possible solution to the previously encountered problems. Nevertheless, an extensive evaluation of the capabilities and the limitations of the TRM model has to be provided and is the subject of current work.

A obvious possible extension of the TRM can be obtained by replacing the recurrent network implementing the biases B_t with a more general dynamical system (e. g. a suitable filter bank or other input-output systems). This modeling freedom poses an interesting interface where task specific knowledge and educated guesses about the nature of the problem can be implemented into the model, comparable to a well made choice kernel in support vector machines. This will be in the focus of future research.

References

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

[2] G. E. Hinton, S. Osindero, and Y Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[3] G. W. Taylor, G. E. Hinton, and S. Roweis. Modeling human motion using binary latent variables. In *Advances in Neural Information Processing Systems*, volume 19, Cambridge, MA, 2007. MIT Press.

[4] I. Sutskever, G.E. Hinton, and G. W. Taylor. The recurrent temporal restricted boltzmann machine. In *Advances in Neural Information Processing Systems*, volume 21, Cambridge, MA, 2009. MIT Press.

[5] H. Jäger. The ”echo state” approach to analyzing and training recurrent neural networks. GMD Report 148, German National Research Center for Information Technology, 2001.

[6] H. Jäger and H. Haas. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science*, 304:78–80, 2004.

[7] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[8] H. Jäger. Short term memory in echo state networks. GMD Report 152, German National Research Center for Information Technology, 2002.

[9] R.F. Lyon. A computational model of filtering, detection and compression in the cochlea. In *Proceedings of the IEEE ICASSP*, pages 1282–1285, May 1982.

[10] D. Verstraeten, B. Schrauwen, D. Stroobandt, and J. Van Campenhout. Isolated word recognition with the liquid state machine: a case study. *Information Processing Letters*, 95(6):521–528, 2005.

[11] Mark D. Skowronski and John G. Harris. Automatic speech recognition using a predictive echo state network classifier. *Neural Networks*, 20(3):414–423, 2007.

[12] M. Ranzato, F. Huang, Y. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proc. Computer Vision and Pattern Recognition Conference (CVPR’07)*. IEEE Press, 2007.