Unsupervised Learning in Recurrent Neural Networks*

Magdalena Klapper-Rybicka[1],
Nicol N. Schraudolph[2], and Jürgen Schmidhuber[3]

[1] Institute of Computer Science, University of Mining and Metallurgy, al. Mickiewicza 30, 30-059 Kraków, Poland
mklapper@uci.agh.edu.pl

[2] Institute of Computational Sciences, Eidgenössische Technische Hochschule (ETH), CH-8092 Zürich, Switzerland
nic@inf.ethz.ch

[3] Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), Galleria 2, CH-6928 Manno, Switzerland
juergen@idsia.ch

Technical Report IDSIA-17-00
November 28, 2000; revised May 1, 2001

Abstract. While much work has been done on unsupervised learning in feedforward neural network architectures, its potential with (theoretically more powerful) recurrent networks and time-varying inputs has rarely been explored. Here we train Long Short-Term Memory (LSTM) recurrent networks to maximize two information-theoretic objectives for unsupervised learning: Binary Information Gain Optimization (BINGO) and Nonparametric Entropy Optimization (NEO). LSTM learns to discriminate different types of temporal sequences and group them according to a variety of features.

1 Introduction

Unsupervised detection of input regularities is a major topic of research on feed-forward neural networks (FFNs), e.g., [1–33]. Most of these methods derive from information-theoretic objectives, such as maximizing the amount of preserved information about the input data at the network's output.

Typical real-world inputs, however, are not static but sequential, full of temporally extended features, statistical regularities, and redundancies. FFNs therefore necessarily ignore a large potential for compactly encoding data. This motivates our research on unsupervised feature detection with recurrent neural networks (RNNs), whose computational abilities in theory exceed those of FFNs

* Condensed version to appear in: *Proc. Intl. Conf. Artificial Neural Networks* (ICANN 2001, Vienna, Austria). Lecture Notes in Computer Science, Springer Verlag, Berlin.

by far. Hitherto little work has been done, however, on this topic. The few exceptions [30, 34–39] include (a) toy applications of predictability minimization [7, 40] to the problem of encoding input sequences through RNNs in a nonredundant fashion [35, 36], (b) unsupervised RNN-based compression of input sequences to facilitate subsequent processing by another RNN [37–39], and (c) an illustrative toy application of the recent redundancy minimization technique LOCOCODE [31] to sequence-processing RNNs [30].

One of the reasons for the conventional focus on FFNs may be the relative maturity of this architecture, and the algorithms used to train it. Compared to FFNs, traditional RNNs [41–43] are notoriously difficult to train, especially when the interval between relevant events in the input sequence exceeds about 10 time steps [44–46]. Recent progress in RNN research, however, has overcome some of these problems [46, 47], and may pave the way for a fresh look at unsupervised sequence learning.

Here, for the first time, we will plug certain information-theoretic objectives into a recent RNN architecture called Long Short-Term Memory (LSTM), which dramatically outperforms other RNNs on a wide variety of *supervised* sequence learning tasks. We will begin with a brief description of LSTM, then describe two unsupervised learning methods: Binary Information Gain Optimization [48, 49] and Nonparametric Entropy Optimization [49, 50]. We will then present examples of LSTM performance with each unsupervised objective function applied to both artificial and real-world data sequences.

2 Unsupervised training of LSTM

Long Short-Term Memory (LSTM) is a novel efficient type of recurrent neural network (RNN) architecture [46] whose advantages over other RNN models have been demonstrated in various areas: learning regular and context free languages [51], predicting continual time series [47], motor control and rhythm detection [52].

While previous work explored LSTM training based on clearly defined objective targets, in this paper we will show that LSTM can also handle unsupervised problems. We used two unsupervised learning algorithms, Binary Information Gain Optimization (BINGO) and Nonparametric Entropy Optimization (NEO), to train the LSTm network to discriminate between sets of temporal sequences, and cluster them into groups. The network is expected to autonomously detect some relevant aspect of the input sequences, according to the particular learning rule used. The learning algorithm strongly determines which aspects of the input sequences the network considers to be important; this can be formalized as a maximization of *relevant* information about the input data at the network's output [49].

2.1 LSTM architecture

The basic unit of an LSTM network is the *memory block* containing one or more *memory cells* and three adaptive, multiplicative gating units shared by all

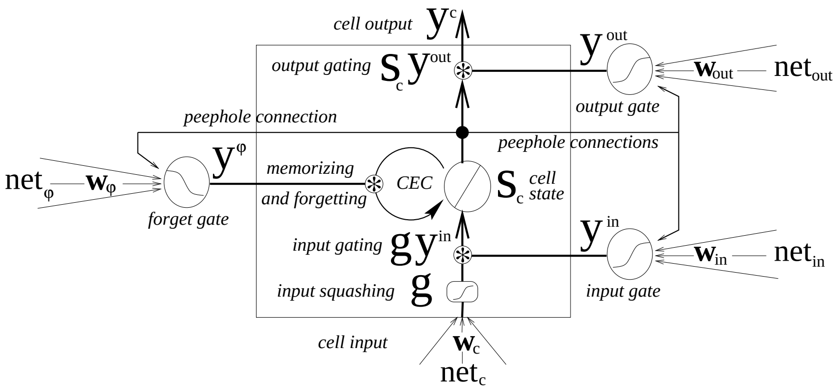

Fig. 1. LSTM memory block with one cell. From [52].

cells in the block (Fig. 1). Each memory cell has at its core a recurrently self-connected linear unit (CEC) whose activation is called the cell *state*. The CECs enforce *constant* error flow and overcome a fundamental problem plaguing previous RNNs: they prevent error signals from decaying quickly as they propagate "back in time". The adaptive gates control input and output to the cells and learn to reset the cell's state once its contents are out of date. Peephole connections connect the CEC to the gates. All errors are cut off once they leak out of a memory cell or gate, although they do serve to change the incoming weights. The effect is that the CECs are the only part of the system through which errors can flow back forever, while the surrounding machinery learns the nonlinear aspects of sequence processing. This permits LSTM to bridge huge time lags (1000 steps and more) between relevant events, while traditional RNNs already fail to learn in the presence of 10-step time lags, despite requiring more complex update algorithms. See [52] for a detailed description.

2.2 Binary Information Gain Optimization

The BINGO algorithm clusters unlabeled data with linear adaptive discriminants stressing the gaps between clusters [48, 49]. The method maximizes the information gained from observing output of a single-layer network of logistic nodes, interpreting their activity as stochastic binary variables. The resulting weight update formula (see [48, 49] for complete derivation) for node i is given by

$$\Delta \boldsymbol{w}_i \;\propto\; f'(y_i)\, \boldsymbol{x}\, (y_i - \hat{y}_i)\,, \tag{1}$$

where $f'(y_i)$ is the derivative of the logistic squashing function f, $\boldsymbol{x}$ the presynaptic input, and $(y_i - \hat{y}_i)$ the difference between actual and estimated network output. We used a linear second-order estimator for multiple outputs:

$$\hat{\boldsymbol{y}} \;=\; \boldsymbol{y} + (Q_{\boldsymbol{y}} - 2I)(\boldsymbol{y} - \bar{\boldsymbol{y}})\,, \tag{2}$$

where $\bar{\boldsymbol{y}}$ denotes the average of $\boldsymbol{y}$ over the data, and $Q_{\boldsymbol{y}}$ its autocorrelation. The binary information gain is maximal (namely, 1 bit/node) when the network's outputs are uncorrelated, and approach '1' (resp. '0') for half the data points. Thus BINGO seeks to identify independent dichotomies in the data.

2.3 Nonparametric Entropy Optimization

NEO, in contrast to parametric unsupervised learning techniques, is a differential learning rule that optimizes signal entropy by way of kernel density estimation, so no additional assumption about a density's smoothness or functional form is necessary [49, 50]. The nonparametric *Parzen window* density estimate $\hat{p}(y)$ is given by:

$$\hat{p}(\boldsymbol{y}) = \frac{1}{|T|} \sum_{\boldsymbol{y}_j \in T} K_\sigma(\boldsymbol{y} - \boldsymbol{y}_j)\,, \tag{3}$$

where T is a sample of points $\boldsymbol{y}_j$ and K_σ is a kernel function, in our case an isotropic Gaussian with variance σ^2. The kernel width σ that best regularizes the density estimate can be found by maximizing the log-likelihood:

$$\hat{L} = \sum_{\boldsymbol{y}_i \in S} \log \sum_{\boldsymbol{y}_j \in T} K_\sigma(\boldsymbol{y}_i - \boldsymbol{y}_j) - |S| \log |T|\,, \tag{4}$$

whose derivative with respect to the kernel width is given by:

$$\frac{\partial}{\partial \sigma} \hat{L} = \sum_{\boldsymbol{y}_i \in S} \frac{\sum_{\boldsymbol{y}_j \in T} \frac{\partial}{\partial \sigma} K_\sigma(\boldsymbol{y}_i - \boldsymbol{y}_j)}{\sum_{\boldsymbol{y}_j \in T} K_\sigma(\boldsymbol{y}_i - \boldsymbol{y}_j)}\,. \tag{5}$$

The maximum likelihood kernel makes a second sample S derived from $p(\boldsymbol{y})$ most likely under the estimated density $\hat{p}(\boldsymbol{y})$ computed from the sample T.[1] A nonparametric estimate of the empirical entropy (given optimal kernel shape) of a neural network's output $\boldsymbol{y}$ can then be calculated as:

$$\hat{H}(\boldsymbol{y}) = -\frac{1}{|S|} \sum_{\boldsymbol{y}_i \in S} \log \sum_{\boldsymbol{y}_j \in T} K_\sigma(\boldsymbol{y}_i - \boldsymbol{y}_j) + \log |T|\,, \tag{6}$$

and minimized by adjusting the neural network's weights $\boldsymbol{w}$:

$$\frac{\partial}{\partial \boldsymbol{w}} \hat{H}(\boldsymbol{y}) = -\frac{1}{|S|} \sum_{\boldsymbol{y}_i \in S} \frac{\sum_{\boldsymbol{y}_j \in T} \frac{\partial}{\partial \boldsymbol{w}} K_\sigma(\boldsymbol{y}_i - \boldsymbol{y}_j)}{\sum_{\boldsymbol{y}_j \in T} K_\sigma(\boldsymbol{y}_i - \boldsymbol{y}_j)}\,. \tag{7}$$

Low $\hat{H}(\boldsymbol{y})$ is achieved by clustering the data. See [49, 50] for further details.

[1] To make efficient use of avaliable training data, we let $\boldsymbol{y}_j$ in the inner sums of equations (5) and (7) range over $T = S \setminus \{\boldsymbol{y}_i\}$ for each $\boldsymbol{y}_i \in S$ in the outer sum [49].

Table 1. The artificial data (from left to right): symbol used in diagrams, mean and variance of the random sequences, and range of outputs produced by the NEO network.

Sym.	*Mean*	*Var.*	*Output range*
$\circ$	0.4	0.3	12.93 – 13.68
$+$	0.4	0.1	11.43 – 12.49
$*$	0.2	0.2	7.50 – 7.69
$\triangle$	0.7	0.1	29.16 – 30.18

Sym.	*Pitch*		*# of*	*Output range*
	Note	*[Hz]*	*cycles*	*($\times 10^6$)*
$\circ$	A3	440	1	-3.57 – -3.54
$+$	A4	880	2	-3.40 – -3.35
$*$	E5	1320	3	-3.47 – -3.43
$\triangle$	F$^\sharp$5	1480	3.4	-3.66 – -3.62

Table 2. The real data (from left to right): symbol used in diagrams, base frequency of the clarinet sounds in musical notation and in Hertz, number of cycles in a sequence, and range of outputs produced by the NEO network.

3 Experimental Setup

The LSTM networks were applied to unsupervised discrimination of groups of temporal sequences. Two types of data were used: artificial (random sequences) and real (fragments of clarinet sounds). For each data type two experiments were run, using the BINGO and NEO objective functions, respectively.

3.1 Training sets

Artificial data sequences were generated by independently drawing random values from a Gaussian distribution with mean and variance as shown in Table 1. For each of four distributions a group of five 100-pattern long sequences was generated, producing a training set of 20 sequences. The mean and variance values were chosen such a way that the ranges of all four groups of sequences overlapped one another. The means of two groups (symbols $\circ$ and $+$ in Table 1) were set to the same value in order to force the network to distinguish between them based only on variance information.

In the real data case we used sequences constructed of 100-sample fragments of clarinet sounds at four different pitches, sampled at 44.1kHz (Table 2). As with the artificial data, the fragments of five different sounds were selected for each of four pitches, for a training set of 20 sequences. In this case the network should discover an entirely different feature of the training set: groups of sounds are no longer distinguished by mean and variance, but by the number of cycles in the sequence, corresponding to the sound's pitch. Two groups (symbols $*$ and $\triangle$ in Table 2) have very similar pitch and are thus hard to distinguish from the small number of samples we supplied.

3.2 Network architecture and training

The LSTM networks trained with the BINGO algorithm consisted of two single-cell memory blocks and two output units with sigmoidal activation function.

For each sequence the values of the two outputs at the end of the sequence can be interpreted as a point in the unit square. The networks were expected to group these points (corresponding to the groups of sequences), and spread them towards the four corners of the square, equivalent to assigning a unique two-bit binary code to each group.

For the NEO algorithm LSTM networks with just one single-cell memory block appeared sufficient to learn both tasks. The single linear output unit produced just one real value for each sequence and the networks were supposed to cluster the output at the end of each sequence, according to the group it belonged to.

After training with the artificial data the network was expected to discriminate between groups of the sequences by detecting their mean and (especially in case of groups with equal mean) variance. For the clarinet sounds, where mean and variance were no longer informative, the networks had to discover the quite different feature of varying pitch (number of cycles) between groups of sequences.

In both experiments the most recent LSTM network model with forget gates and peephole-connections was used (Fig. 1) [46, 47, 52]. The network was trained until the error dropped below an empirically chosen threshold, or until further training did not noticeably improve the result. In order to make variance information accessible to the network, a second input was added, whose value at any given time was set to the square of the first (main) input [53].

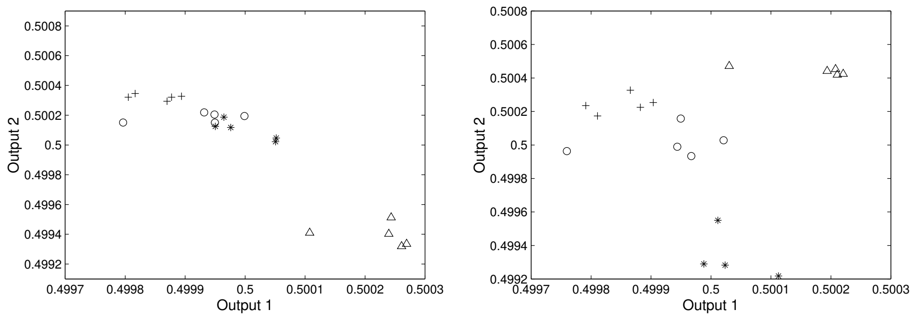

Fig. 2. Network output for artificial data and BINGO objective function. Left: beginning of training, right: after 860 epochs.

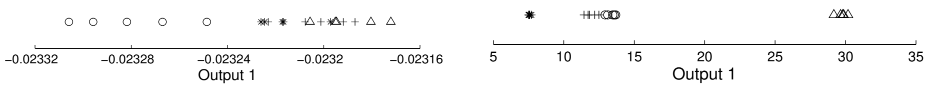

Fig. 3. Network output for artificial data and NEO objective function. Left: beginning of training, right: after 14 epochs.

4 Results

For both real and artificial data, with application of either of the two previously described unsupervised training algorithms, the LSTM networks were able to distinguish between four groups of sequences. Figures 2, 3, 4, and 5 present the results of BINGO and NEO training with artificial data and real clarinet sounds, respectively. For each experiment two diagrams are presented, showing the network output before and after training.

4.1 Training with the BINGO algorithm

Artificial data (Fig. 2). From the very beginning of training, points corresponding to the group marked $\triangle$ are well separated. The other groups overlap each other, and their correlation coefficient is high. After 860 training epochs all four groups are clearly separated, and spread out across the unit square. The two groups with identical mean remain close to each other, with one occupying a more central location.

Clarinet sounds (Fig. 4). At the beginning of training all points are distributed along a line, with a correlation coefficient close to one, and they do not form distinct clusters. After 690 epochs, however, all four groups of points are well separated.

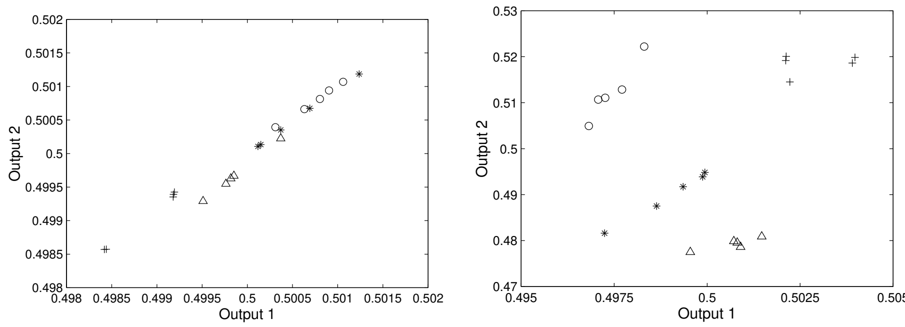

Fig. 4. Network output for clarinet sounds and BINGO objective function. Left: beginning of training, right: after 690 epochs.

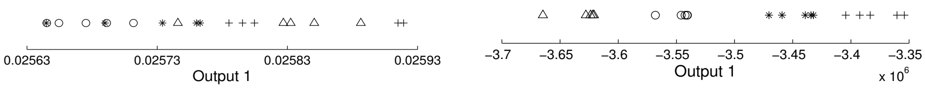

Fig. 5. Network output for clarinet sounds and NEO objective function. Left: beginning of training, right: after 6910 epochs.

4.2 Training with the NEO algorithm

Artificial data (Fig. 3). Before training the outputs for the group marked $\circ$ are separated, though spread over a comparatively large range. The other three groups overlap each other. After just 14 epochs of training all groups are well clustered, with very small range compared to the gaps between them (Table 1). Again the two groups with identical mean remain close to each other.

Clarinet sounds (Fig. 5). The initial distributions of outputs for all groups overlap heavily. After training, however, the network's outputs form four clearly separated groups, though they are not as dense as in case of the artificial data (Table 1). The result was obtained only after almost 7000 epochs, suggesting that in this experiment the correct discriminants were harder to find than for the artificial data set.

5 Conclusion

Previous work presented many successful applications of Long Short-Term Memory (LSTM) networks to various supervised tasks, where LSTM definitely outperforms other recurrent neural network architectures. In this paper we showed that in conjunction with appropriate objective functions, LSTM can also solve unsupervised problems. We combined LSTM with two unsupervised learning methods, Binary Information Gain Optimization and Nonparametric Entropy Optimization, and showed that the resulting system performs unsupervised discrimination and classification of temporal sequences. Our experiments with both artificial and real data showed a remarkable ability of unsupervised LSTM to cluster temporal sequences according to a variety of reasonable features.

Combining the advantages of the LSTM recurrent network model with various objective functions for unsupervised learning should result in promising techniques for numerous real-world tasks involving unsupervised detection of input sequence features spanning extended time periods.

Acknowledgement

This research was supported by the Stefan Batory Foundation, and by the Swiss National Science Foundation under grants 2100–49'144.96 and 2000–052'678.97/1.

References

1. H. B. Barlow, *Understanding natural vision*. Springer-Verlag, Berlin, 1983.
2. H. B. Barlow, T. P. Kaushal, and G. J. Mitchison, "Finding minimum entropy codes," *Neural Computation*, vol. 1, no. 3, pp. 412–423, 1989.
3. D. J. Field, "Relations between the statistics of natural images and the response properties of cortical cells," *Journal of the Optical Society of America*, vol. 4, pp. 2379–2394, 1987.

4. R. Linsker, "Self-organization in a perceptual network," *IEEE Computer*, vol. 21, pp. 105–117, 1988.
5. C. Jutten and J. Herault, "Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, no. 1, pp. 1–10, 1991.
6. S. Becker, "Unsupervised learning procedures for neural networks," *International Journal of Neural Systems*, vol. 2, no. 1 & 2, pp. 17–33, 1991.
7. J. Schmidhuber, "Learning factorial codes by predictability minimization," *Neural Computation*, vol. 4, no. 6, pp. 863–879, 1992.
8. J. Schmidhuber and D. Prelinger, "Discovering predictable classifications," *Neural Computation*, vol. 5, no. 4, pp. 625–635, 1993.
9. N. Schraudolph and T. J. Sejnowski, "Unsupervised discrimination of clustered data via optimization of binary information gain," in *Advances in Neural Information Processing Systems* (S. J. Hanson, J. D. Cowan, and C. L. Giles, eds.), vol. 5, pp. 499–506, Morgan Kaufmann, San Mateo, 1993.
10. A. N. Redlich, "Redundancy reduction as a strategy for unsupervised learning," *Neural Computation*, vol. 5, pp. 289–304, 1993.
11. P. Comon, "Independent component analysis – a new concept?," *Signal Processing*, vol. 36, no. 3, pp. 287–314, 1994.
12. G. Deco and L. Parra, "Nonlinear features extraction by unsupervised redundancy reduction with a stochastic neural network," Tech. Rep. ZFE ST SN 41, Siemens AG, 1994.
13. Z. Li, "A theory of the visual motion coding in the primary visual cortex," *Neural Computation*, vol. 8, no. 4, pp. 705–730, 1995.
14. M. C. Mozer, "Discovering discrete distributed representations with iterative competitive learning," in *Advances in Neural Information Processing Systems 3* (R. P. Lippmann, J. E. Moody, and D. S. Touretzky, eds.), pp. 627–634, San Mateo, CA: Morgan Kaufmann, 1991.
15. P. Földiák, "Forming sparse representations by local anti-Hebbian learning," *Biological Cybernetics*, vol. 64, pp. 165–170, 1990.
16. P. Földiák and M. P. Young, "Sparse coding in the primate cortex," in *The Handbook of Brain Theory and Neural Networks* (M. A. Arbib, ed.), pp. 895–898, The MIT Press, Cambridge, Massachusetts, 1995.
17. G. Palm, "On the information storage capacity of local learning rules," *Neural Computation*, vol. 4, no. 2, pp. 703–711, 1992.
18. R. S. Zemel and G. E. Hinton, "Developing population codes by minimizing description length," in *Advances in Neural Information Processing Systems 6* (J. D. Cowan, G. Tesauro, and J. Alspector, eds.), pp. 11–18, San Mateo, CA: Morgan Kaufmann, 1994.
19. D. J. Field, "What is the goal of sensory coding?," *Neural Computation*, vol. 6, pp. 559–601, 1994.
20. E. Saund, "Unsupervised learning of mixtures of multiple causes in binary data," in *Advances in Neural Information Processing Systems 6* (J. D. Cowan, G. Tesauro, and J. Alspector, eds.), pp. 27–34, Morgan Kaufmann, San Mateo, CA, 1994.
21. P. Dayan and R. Zemel, "Competition and multiple cause models," *Neural Computation*, vol. 7, pp. 565–579, 1995.
22. B. A. O. D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, pp. 607–609, 1996.
23. R. S. Zemel, *A minimum description length framework for unsupervised learning*. PhD thesis, University of Toronto, 1993.

24. S. Amari, A. Cichocki, and H. Yang, "A new learning algorithm for blind signal separation," in *Advances in Neural Information Processing Systems* (D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, eds.), vol. 8, The MIT Press, Cambridge, MA, 1996.
25. A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, no. 6, pp. 1129–1159, 1995.
26. J.-P. Nadal and N. Parga, "Redundancy reduction and independent component analysis: Conditions on cumulants and adaptive approaches," *Neural Computation*, vol. 9, no. 7, pp. 1421–1456, 1997.
27. G. E. Hinton and Z. Ghahramani, "Generative models for discovering sparse distributed representations," *Philosophical Transactions of the Royal Society* **B**, vol. 352, pp. 1177–1190, 1997.
28. G. Yang and S. Amari, "Adaptive online learning algorithms for blind source separation: Maximum entropy and minimum mutual information," *Neural Computation*, vol. 9, no. 7, pp. 1457–1482, 1997.
29. P. Pajunen, "Blind source separation of natural signals based on approximate complexi y minimization," in *Proceedings of the First International Workshop on Independent Component Analysis and Signal Separation, Aussois, France* (J.-F. Cardoso, C. Jutten, and P. Loubaton, eds.), pp. 267–270, 1999.
30. S. Hochreiter and J. Schmidhuber, "Feature extraction through LOCOCODE," *Neural Computation*, vol. 11, no. 3, pp. 679–714, 1999.
31. S. Hochreiter and J. Schmidhuber, "Source separation as a by-product of regularization," in *Advances in Neural Information Processing Systems 12* (M. Kearns, S. A. Solla, and D. Cohn, eds.), pp. 459–465, MIT Press, Cambridge MA, 1999.
32. S. Hochreiter and J. Schmidhuber, "LOCOCODE performs nonlinear ICA without knowing the number of sources," in *Proceedings of the First International Workshop on Independent Component Analysis and Signal Separation, Aussois, France* (J.-F. Cardoso, C. Jutten, and P. Loubaton, eds.), pp. 149–154, 1999.
33. A. Hyvarinen, P. Hoyer, and E. Oja, "Sparse code shrinkage: Denoising by maximum likelihood extimation," in *Advances in Neural Information Processing Systems 12* (M. Kearns, S. A. Solla, and D. Cohn, eds.), MIT Press, Cambridge MA, 1999.
34. P. Tiňo, M. Stancik, and L. Beňušková, "Building predictive models on complex symbolic sequences with a second-order recurrent BCM network with lateral inhibition," in *Proc. Int. Joint Conf. Neural Networks*, vol. 2, pp. 265–270, 2000.
35. S. Lindstädt, "Comparison of unsupervised neural networks for redundancy reduction," 1993. Master's thesis, Dept. of Comp. Sci., University of Colorado at Boulder.
36. S. Lindstädt, "Comparison of two unsupervised neural network models for redundancy reduction," in *Proc. of the 1993 Connectionist Models Summer School* (M. C. Mozer, P. Smolensky, D. S. Touretzky, J. L. Elman, and A. S. Weigend, eds.), pp. 308–315, Hillsdale, NJ: Erlbaum Associates, 1993.
37. J. Schmidhuber, "Learning unambiguous reduced sequence descriptions," in *Advances in Neural Information Processing Systems 4* (J. E. Moody, S. J. Hanson, and R. P. Lippman, eds.), pp. 291–298, San Mateo, CA: Morgan Kaufmann, 1992.
38. J. Schmidhuber, "Learning complex, extended sequences using the principle of history compression," *Neural Computation*, vol. 4, no. 2, pp. 234–242, 1992.
39. J. Schmidhuber, "Neural predictors for detecting and removing redundant information," in *Adaptive Behavior and Learning* (H. Cruse, J. Dean, and H. Ritter, eds.), Kluwer, 1999.

40. N. N. Schraudolph, M. Eldracher, and J. Schmidhuber, "Processing images by semi-linear predictability minimization," *Network: Computation in Neural Systems*, vol. 10, no. 2, pp. 133–169, 1999.
41. A. J. Robinson and F. Fallside, "The utility driven dynamic error propagation network," Tech. Rep. CUED/F-INFENG/TR.1, Cambridge University Engineering Department, 1987.
42. P. J. Werbos, "Generalization of backpropagation with application to a recurrent gas market model," *Neural Networks*, vol. 1, 1988.
43. R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent networks," Tech. Rep. ICS Report 8805, Univ. of California, San Diego, La Jolla, 1988.
44. S. Hochreiter, " Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut für Informatik, Lehrstuhl Prof. Brauer, Technische Universität München," 1991. See www7.informatik.tu-muenchen.de/~hochreit.
45. Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.
46. S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
47. F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000.
48. N. N. Schraudolph and T. J. Sejnowski, "Unsupervised discrimination of clustered data via optimization of binary information gain," in *Advances in Neural Information Processing Systems* (S. J. Hanson, J. D. Cowan, and C. L. Giles, eds.), vol. 5, pp. 499–506, Morgan Kaufmann, San Mateo, CA, 1993.
49. N. N. Schraudolph, *Optimization of Entropy with Neural Networks.* PhD thesis, University of California, San Diego, 1995.
50. P. A. Viola, N. N. Schraudolph, and T. J. Sejnowski, "Empirical entropy manipulation for real-world problems," in *Advances in Neural Information Processing Systems* (D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, eds.), vol. 8, pp. 851–857, The MIT Press, Cambridge, MA, 1996.
51. F. A. Gers and J. Schmidhuber, "Long Short-Term Memory learns simple context free and context sensitive languages," *IEEE Transactions on Neural Networks*, 2001 (forthcoming).
52. F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proc. Int. Joint Conf. Neural Networks*, (Como, Italy), IEEE Computer Society Press, 2000.
53. G. W. Flake, "Square unit augmented, radially extended, multilayer perceptrons," in *Neural Networks: Tricks of the Trade* (G. B. Orr and K.-R. Müller, eds.), vol. 1524 of *Lecture Notes in Computer Science*, pp. 145–163, Berlin: Springer Verlag, 1998.