Recurrent Networks and Natural Language: Exploiting Self-organization

Igor Farkas (ifarkas@coli.uni-sb.de)
Matthew W. Crocker (crocker@coli.uni-sb.de)
Department of Computational Linguistics and Phonetics
Saarland University, Saarbrücken, D-66041, Germany

Abstract

Prediction is believed to be an important cognitive component in natural language processing. Within connectionist approaches, Elman's simple recurrent network has been used for this task with considerable success, especially on small scale problems. However, it has been appreciated for some time that supervised gradient-based learning models have difficulties with scaling up, because their learning becomes very time-consuming for larger data sets. In this paper, we explore an alternative neural network architecture that exploits self-organization. The prediction task is effectively split into separate stages of self-organized context representation and subsequent association with the next-word target distribution. We compare various prediction models and show, in the task of learning a language generated by stochastic context-free grammar, that self-organization can lead to higher accuracy, faster training, greater robustness and more transparent internal representations, when compared to Elman's network.

Introduction

Recurrent neural networks have been traditionally used in various tasks that involve time-dependent data. The best known architecture is the Simple Recurrent Network (SRN; Elman, 1990) that has been employed in a variety of tasks, including language learning by prediction (e.g. Elman, 1991; Servan-Schreiber et al., 1991; Rohde and Plaut, 1997; Christiansen and Chater, 1999). Supervised learning of temporal dependencies by prediction typically involves error gradient learning algorithms of which various forms have been proposed (see Pearlmutter, 1995, for overview). Despite their considerable success, the supervised learning approaches are difficult to scale up to realistic tasks due to learning complexity. One common aspect of these methods is that via error back-propagation they optimize the internal states of a recurrent network to a particular task. In the prediction task this implies that both internal states and predictions are optimized using the same learning mechanism.

Here we explore an alternative avenue along which we split the whole task into two subtasks and treat them independently. Hence, we first optimize internal states, and then we associate these with desired predictions. Optimizing internal states consists in building temporal context representations and since it is not driven by supervision, it can potentially benefit from self-organization. Self-organized temporal context learning is expected not only to facilitate the learning process but has also been argued to have a greater biological plausibility. There have been a number of unsupervised methods proposed during the last decade (see overview in Barreto et al., 2003; Hammer et al., 2004a). Here we focus on two models, namely Recursive Self-Organizing Map (RecSOM; Voegtlin, 2002) and feedforward SardNet (James and Miikkulainen, 1995) that represent, in a sense, complementary approaches to representation of the temporal context. RecSOM has been shown to demonstrate a rich repertoire of dynamic behavior when trained on a complex symbolic sequence such as natural language text (Tiňo and Farkaš, 2005). Similarly, it has been shown that SardNet, when added as a parallel input preprocessing module to a supervised recurrent network, enhances the processing capacity of a neural network in a shift-reduce parsing task (Mayberry and Miikkulainen, 1999).

Once the context representations are optimized with a chosen self-organizing module, we associate them with desired predictions using a supervised learning module. We tested two such modules. One is a simple counting method that builds independent prediction distributions for all units in the map. The other is a single-layer perceptron trained by the error delta rule.

Simulation methods

Self-organization of temporal context

For temporal context learning, we explored two basic self-organizing modules – RecSOM and SardNet – as well as a combination of the two, which we called RecSOM-sard. We describe them all in more detail below.

Recursive Self-Organizing Map The architecture of the RecSOM model is shown in Figure 1 (without the top layer). Each map neuron $i \in \{1, 2, ..., N\}$ has two weight vectors associated with it: $\mathbf{w}_i \in \mathcal{R}^n$ linked with an n-dimensional input $\mathbf{s}(t)$, and $\mathbf{c}_i \in \mathcal{R}^N$ linked with the context $\mathbf{y}(t-1) = (y_1(t-1), y_2(t-1), ..., y_N(t-1))$, containing map activations $y_i(t-1)$ from the previous time step. The output of a neuron i at time t is computed as $y_i(t) = \exp(-d_i(t))$, where

$$d_i(t) = \alpha \cdot \|\mathbf{s}(t) - \mathbf{w}_i\|^2 + \beta \cdot \|\mathbf{y}(t-1) - \mathbf{c}_i\|^2$$

with $\|\cdot\|$ denoting the Euclidean norm. Parameters $\alpha > 0$ and $\beta > 0$ respectively influence the effect of the input and the context upon a neuron's profile. Both weight

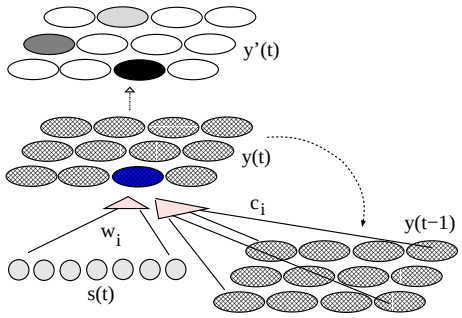

Figure 1: RecSOMsard architecture. The bottom part (without the top layer) represents Recursive SOM. Solid lines represent trainable connections, dashed line represents one-to-one copy of the activity vector $\mathbf{y}$. In RecSOMsard, $\mathbf{y}$ is transformed to $\mathbf{y}$' in the top layer by a mechanism described in the text.

vectors can be updated using the same form of learning rule (Voegtlin, 2002):

$$\begin{aligned}\Delta\mathbf{w}_i &= \gamma \cdot h_{ik} \cdot (\mathbf{s}(t) - \mathbf{w}_i),\\ \Delta\mathbf{c}_i &= \gamma \cdot h_{ik} \cdot (\mathbf{y}(t-1) - \mathbf{c}_i),\end{aligned}$$

where k is an index of the best matching unit ('winner') at time t, (i.e. the unit with the highest activation $y_k(t)$) and $0 < \gamma < 1$ is the learning rate. Neighborhood function h_{ik} is a Gaussian on the distance $d(i,k)$ of units i and k in the map: $h_{ik} = \exp\{-d(i,k)^2/\sigma^2(t)\}$. Parameter σ linearly decreases in time to allow for forming topographic representation of input sequences.

As a result of self-organization, RecSOM units learn to topographically represent temporal contexts (subsequences). Specifically, it has been shown that the context-based input representations typically become organized in a Markovian manner: Subsequences sharing a common suffix are mapped close to each other (Tiňo et al., 2005). At the same time, a more complex input sequence can lead to more complex, non-Markovian behavior (Tiňo and Farkaš, 2005). Learning in recursive self-organizing networks (such as RecSOM) has been shown to approximate stochastic gradient descent driven by input data (Hammer et al., 2004b). Due to recurrency, the process of weight optimization can be thought of as temporally enhanced vector quantization.

SardNet The alternative model, SardNet (Sequential Activation Retention and Decay Network; James and Miikkulainen, 1995), has an architecture and mechanism very similar to the standard SOM (Kohonen, 1990). Unlike RecSOM, it does not have a recursive architecture, so it learns to unambiguously represent the sequences as distributed activation patterns over the map. For each input, the winner is assigned the activation of 1.0 and the activations of all other units representing previous inputs in the current sequence are decayed via $y_i \leftarrow \kappa y_i$ using a preset decay factor $0 \ll \kappa < 1$. Once the unit is activated, it is removed from competition and cannot represent later input in the current sequence. Forcing other (neighboring) units to participate in the representation allows each unit to represent different inputs depending on the context, which leads to an efficient representation of sequences, and also generalizes well to new sequences.

RecSOMsard This combined model has the RecSOM architecture and processing, but adds on a SardNet-like output preprocessing to be fed to a prediction module (see Figure 1). RecSOM outputs are replaced during processing by spatially distributed representations of the context. In each iteration, the winner's activation y_k in RecSOM is transformed to a focused Gaussian profile $y'_i = \exp\{-d(i,k)^2/\sigma_y^2\}$ centered arround the winner k, and previous activations in the top layer are decayed via $y' \leftarrow \lambda y'$ (as in SardNet). At boundaries between sequences, all activations y' are reset to zero. This SardNet feature establishes that the activation vector $\mathbf{y}(t)$ with mostly unimodal shape is transformed to a distributed activation vector $\mathbf{y}'(t)$ whose number of peaks equals the position of a current word in a sentence (see Figure 2). In this manner, the context in RecSOMsard becomes represented both spatially (due to SardNet) and temporally (due to RecSOM). [1]

Figure 2: RecSOMsard activation patterns during a sentence processing.

Next-word prediction modules

We employed two next-word supervised prediction modules built on top of the outputs of the trained context-representation module.

W-module The Winner-based prediction module uses the matrix U (of size $N \times n$) of counters, which are selectively updated during a single epoch through the training data set, after the training has been completed. For each input s_t, the winner i is found and its counter $U(i, s_{t+1})$ corresponding to the next word s_{t+1} is incremented by one. At the end of the epoch, the rows of U are normalized using L_1-norm to be interpretable as unit's prediction probability vectors. Hence, in W-module the prediction in each step is retrieved locally, from the winner's probability distribution vector.

P-module The Perceptron-based prediction module uses n softmax outputs, which allows for the outputs to be interpreted as prediction probabilities. In softmax, the output of the perceptron unit j is computed as $z_j = \exp(net_j)/\sum_l \exp(net_l)$, where the total input of unit j is $net_j = \sum_i v_{ji} y_i$. Perceptron weights are updated via the delta rule as

$$\Delta v_{ji} = \eta(t) y_i (d_j - z_j)$$

with targets d_j being one-hot encoded, and learning rate $\eta(t)$ linearly decreasing to zero. Unlike W-module, P-

[1] the latter being the case, because each winner in RecSOM best matches the current input in a particular temporal context

module involves a true learning algorithm, so it requires multiple epochs of training, rather than one.

Models

Using a complete cross-design of the above context-learning and next-word prediction modules, we end up with six possible models exploiting the self-organization in the context learning stage. Not all of these models can be expected to work well, such as SardNet combined with W-module (SardNet-W), in which temporal information is lost. All models were evaluated for comparison and were compared with Elman's SRN as well as statistical bigram and trigram models (Stolcke, 2002). The SRN was matched in architecture with RecSOM and hence, was a two-layer network using N hidden units with sigmoidal activation function but like the P-module, it had n softmax output units. The SRN was trained using online stochastic gradient descent to minimize cross-entropy between outputs and corresponding targets. Error was not back propagated through time, only through the current time step.

Training data

We used SLG (Rohde, 2002) to generate sentences constructed by a moderately complex stochastic context-free grammar specified in Table 1. The choice of the grammar was motivated by the use of child-directed sentences (following Christiansen and Dale, 2001) enriched with recursive structures, as used in earlier works on SRN in next-word prediction tasks (Elman, 1993; Rohde and Plaut, 1997). Our grammar included three primary sentence types: declarative, interrogative, and imperative. Each type consisted of a variety of common utterances. Declarative sentences most frequently appeared as transitive or intransitive verb constructions, but also included predication using the copula. Interrogative sentences were composed of wh-questions and questions formed by using auxiliary verbs. Imperatives were the simplest class of sentences, appearing as intransitive or transitive verb phrases. We controlled subject-verb agreement, as well as appropriate determiners accompanying nouns. The sentences obeyed a number of semantic constraints, similar to those used in Rohde and Plaut (1997). Regarding recursive sentences, we did not follow the 'starting small' scenario, hypothesized by Elman (1993), as we believe it would not be beneficial to the networks (Rohde and Plaut, 1997). Sentences longer than 16 words were discarded in generating the corpus, but these were so rare ($< 0.1\%$) that their loss should have negligible effects. The lexicon contained 72 words including the end-of-sentence marker.

Parameters

Using the above grammar, we generated 10,000 sentences. Each model was run 10 times, using a different training set comprising a randomly chosen subset of 50% of total sentences (the complementary subset was used for testing). On average, the training set contained 30867 words and testing set 23148 words (we removed duplicate sentences from the testing sets). In each model, we experimented with a number of parameters to obtain the best performance. Each input to the networks contained a localist representation of a word. Hence, all neural network models had $n = 72$ input units. RecSOM(sard) and SardNet contained $N = 18 \times 18$ neurons and their weights were randomly initialized within the interval [0.4; 0.6]. Other parameters for models with self-organization were as follows: $\alpha = 3, \beta = 0.6, \gamma = 0.1, \sigma : 8 \rightarrow 0.5, \sigma_y = 1, \kappa = 0.9$ and $\lambda = 0.8$. Weights of the P-module were trained for 8 epochs with $\eta : 0.3 \rightarrow 0.03$. RecSOM output activations and SRN context-layer activations were not reset between sentences, since sentence boundaries were clearly detectable by end-of-sentence marker. The weights in the SRN were intialized within the interval [-0.1; 0.1], it had 324 hidden units, a learning rate set to 0.05, and no momentum. Higher learning rate for the SRN (to speed up convergence) had a destabilizing effect and mostly resulted in a somewhat higher testing error. On the other hand, lower learning rates did not lead to further decrease of the testing error (during 100 epochs of training). Varying the size of the hidden-layer did not lead to significant improvement either. All models were trained for maximum 10 epochs, except for the SRN that needed at least 30 epochs to sufficiently converge. [2]

Table 1: The stochastic context-free grammar used to generate training corpora.

S → Declar (.75) | Interrog (.2) | Imper (.05);
Declar → SP VP (.85) | NP-Adj (.1) | That-NP (.05);
SP → NP | NP RC;
RC → who VI | who VT SP | who SP VT;
NP-Adj → NP is/are/were Adj;
That-NP → that/those is/are/were NP;
Interrog → Wh-Qn (.65) | Aux-Qn (.35);
Wh-Qn → where/who is/are/were NP | where/who/what do/does NP VP;
Aux-Qn → do/does NP VP | do/does NP wanna VP | is/are/were NP Adj;
Imper → VP;
VP → VI | VT NP; NP → ART ADJ N;
ART → "" | a | the; ADJ → "" (.5) | Adj;

Results

First, we evaluated each trained model $\mathcal{M}$ in terms of its normalized negative log-likelihood (NNL) using the testing data. It is computed as

$$\mathrm{NNL}_{\mathcal{M}}(S_{tst}) = \frac{-1}{L_{tst}-1} \sum_{t=1}^{L_{tst}-1} \log_n P_{\mathcal{M}}(s_{t+1}|C_t)$$

where $C_t = s_1 s_2 ... s_t$ is the context at time t, and L_{tst} is the length of the testing sequence. The higher the predictive probabilities assigned to the actual next symbols are, the smaller the NNL is. The NNL measure can be viewed as a quantification of a statistical average

[2] In terms of actual implementation, one cycle for training SRN took somewhat longer than that of any self-organizing module coupled with a prediction module.

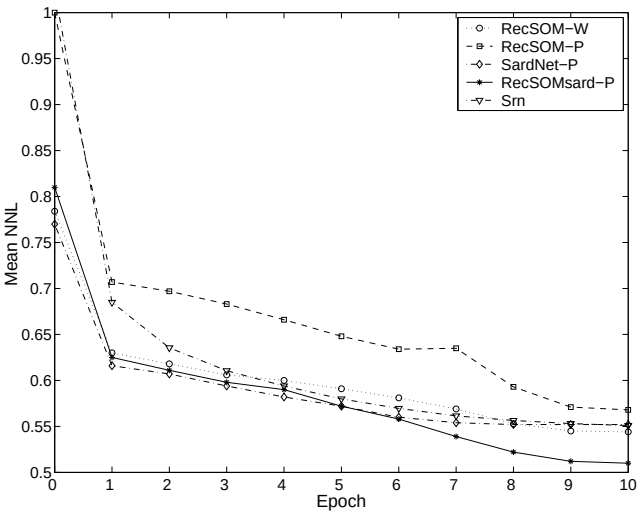

Figure 3: Mean NNLs for various prediction models. SRN was actually trained for 30 epochs, but is aligned with other curves to facilitate comparison. Error bars around means (not shown) were below 0.01.

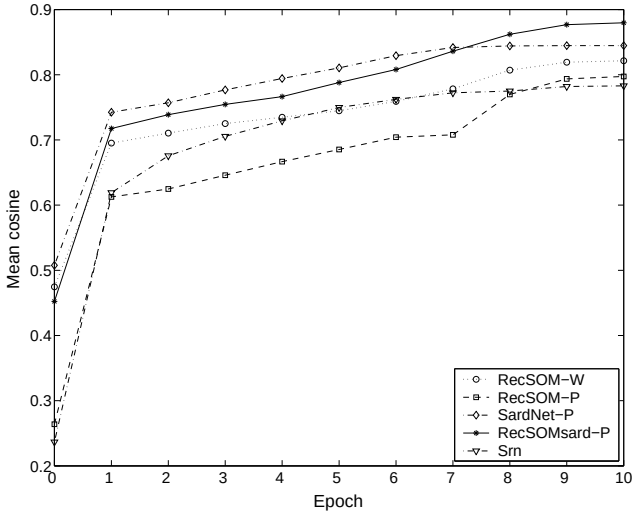

Figure 4: Mean cosines between actual and optimal prediction distribution vectors for various prediction models. The SRN was actually trained for 30 epochs, but is aligned with other curves to facilitate comparison. Error bars around means (not shown) were below 0.06.

of 'surprise' experienced by the model upon seeing the sequence.

The NNL is a standard measure used in prediction, but due to the ambiguity in grammar, more candidates should be predicted (which is not captured by the NNL). For this reason, we also used an alternative measure that assesses the entire prediction vectors. One such measure is the mean *cosine* between the actual and the optimal prediction vectors (calculated from the grammar).

Figure 3 shows the mean NNL for selected models, as a function of training epochs. It is evident that all models learn, but to different degrees of accuracy. Several observations can be made: First, RecSOMsard-P model achieves the best performance, which may be both due to its spatio-temporal representation of the context and due to the P-module (the means difference of the paired t-test was significant at $p<10^{-9}$ compared to RecSOM-W). This claim is supported by the second observation, that the NNL of RecSOMsard-P further decreases during the last 3 epochs of training which occurs when the neighborhood radius σ drops sufficiently (below 1) to allow for fine-tuning of RecSOM units. This is also observed in RecSOM-P whose map units have recurrent connections, but not in case of feedforward SardNet-P. Third, the SRN has comparable performance to that of RecSOM-W and SardNet-P (mutual mean differences n.s.), while RecSOM-P has significantly the poorest accuracy (e.g. $p<10^{-6}$ compared to SardNet-P).

The corresponding Figure 4 displays the mean cosines for these models and is quite consistent with the previous figure. However, the cosines lead to at least two new observations: The first relates to the individual contributions of both processing stages in the models that can lead to higher accuracy. According to the NNL, both RecSOM-W and SardNet-P have similar performance, but in terms of cosines SardNet-P is significantly better than RecSOM-W ($p < 10^{-8}$). This suggests that, for the purpose of prediction, the spatially distributed representation of the context in SardNet may be more important than localized temporal representations in RecSOM-W. However, respective comparison of Sardnet and RecSOMsard (linked with either prediction module) suggests that also temporal information encoded locally in RecSOM units is exploited in the complete combined model, hence leading to its best performance (both $p<10^{-9}$).

Table 2: Mean NNLs and cosines (in parentheses) on testing data for models with self-organization.

	RecSOM	SardNet	RecSOMsard
W	0.544 (0.821)	0.560 (0.783)	0.543 (0.820)
P	0.568 (0.797)	0.552 (0.845)	0.510 (0.880)

The second observation resulting from cosines is that the SRN makes the least accurate prediction vectors (cos = 0.785) which makes it worse than most models using self-organization (cf. Table 2). We acknowledge that in principle, the SRN may be able to achieve better performance by varying different parameters: e.g. using fewer hidden nodes in combination with BPTT learning algorithm (Werbos, 1990), or using more hidden layers of units could be beneficial (as e.g. in Elman, 1993; Rohde and Plaut, 1997).

It is also interesting to compare a self-organizing module with the SRN in terms of the structure of internal (state-space) representations. The hidden-layer of the SRN has been shown to form distributed representations with nice structural properties (as shown originally by Elman) that mostly lead to Markovian behaviour: in-

put sequences sharing common suffixes are mapped together in the hidden-layer and hence are likely to lead to similar predictions. The state-space vizualization of the SRN is typically achieved either by clustering techniques (showing a dendrogram), or by examining a few dimensions via PCA. In contrast, the state-space representation in SardNet and RecSOMsard is very transparent and hence directly comprehensible in high-dimensional map space (see Figure 2). Markovian behavior applies here as well, because sequences sharing common suffixes have a high overlap, and are hence spatially close (modulated by proper setting of σ_y in RecSOMsard).

Table 3: Mean correlation coefficients between predictions of three selected models and optimal predictions given by the grammar. Error bars around means were below 0.014.

RecSOMsard-P	SRN	3-gram
0.881	0.822	0.785

In terms of the NNLs, the bigram and trigram models performed surprisingly very well (0.561 and 0.519, respectively). However, the accuracy of n-grams drops when we look at the similarity between the predictions and the optimal predictions, [3] computed as correlation coefficients, see Table 3. Trigram has significantly the lowest accuracy among the three models (all $p's < 10^{-6}$).

While we leave a systematic analysis of errors types for the different models for future research, we did observe difficulty for all models with the prediction of long-distance dependencies. This is not suprising, given that all models were observed to be driven by Markovian dynamics.

Lesioning test To test the robustness of the models, we randomly lesioned (deactivated) a subset of the middle layer units (i.e. map units, or hidden-layer units in the SRN). In each run, we randomly lesioned the network only once. Figure 5 suggests, showing the mean NNLs for the selected models, that sparse representation in the maps lends itself to higher robustness than fully distributed representation in the hidden layer of the SRN. Among the self-organizing models, prediction models with P-modules always yield higher robustness compared to their counterpart with W-modules (To preserve clarity, the other models with W-modules are not included, but their NNL increase was quite similar to that of RecSOM-W.) Consistently with previous figures, RecSOMsard-P model is the best which is due to the sparseness of representation over a number of participating units (and their nearest neighbors) within a sentence.

Discussion

In this paper we illustrated the benefits of a combined architecture in terms of better prediction accuracy, faster training, greater robustness and better transparency of

[3] We did not compute cosines for n-grams, because SRILM package does not have that feature.

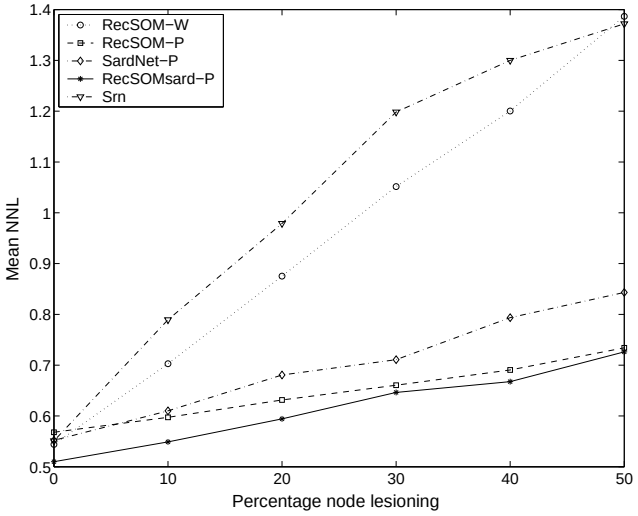

Figure 5: Mean NNLs of various models after lesioning. Error bars around means (not shown) were below 0.05 for maps, below 0.25 for the SRN.

internal representations. The benefits of combined architectures is not new in neural network research. Earlier experiments with feedforward architectures showed, that, for example, radial-basis-function networks whose first layer is unsupervised, typically need shorter training, albeit they may require more hidden units for the same accuracy, compared to fully supervised two-layer perceptron (Tarassenko and Roberts, 1994).

We can distinguish three approaches in total, regarding the optimization of context representations. The first is the above mentioned supervised approach (as in the SRN), where the adaptation of the recurrent weights is driven by output targets. The unsupervised approach, explored in this paper, optimizes the recurrent weights independently from the predictions. As a third option, there exist recent models with no temporal context learning, such as the so called 'liquid state' machines (Maass et al., 2002), 'echo state' networks (Jaeger, 2001), or prediction fractal machines (Parfitt et al., 2000). In these models, the recurrent part is not trained, but with suitably preset parameters (weights) and due to the so called architectural bias (Tiňo et al., 2004) it is able to generate contextual representations with nice structural properties. However, the exploration of these models is at an early stage, and in addition, we have shown (Tiňo et al., 2006) that learning the recurrent weights in RecSOM leads to temporal representations with significantly deeper contexts, when compared to an untrained recurrent model based on affine contractions (as used in PFM; Parfitt et al., 2000).

Our preliminary results in this paper shed light on the benefits of self-organization in learning data with temporal structure, as exemplified on a word-prediction task. Nevertheless, exploiting self-organization is not limited to with word prediction; in principle it is also applicable to other language tasks, such as the case-role assignment. As a matter of fact, even more complicated architec-

tures than ours have recently been shown to profit from self-organization in incremental nonmonotonic parsing task (Mayberry and Miikkulainen, 2003; Mayberry and Crocker, 2004).

Acknowledgments

Igor Farkaš was supported by the Alexander von Humboldt Foundation and by Slovak Grant Agency for Science. He was on leave from the Department of Applied Informatics, Comenius University in Bratislava, and Institute of Measurement Science, Slovak Academy of Sciences. Matthew Crocker gratefully acknowledges the financial support of the German Research Foundation (SFB-378, project "Alpha"). Both authors are thankful to Marty Mayberry for fruitful discussions, and three anonymous reviewers for constructive comments.

References

Barreto, G., Araújo, A., & Kremer, S. (2003). A taxanomy of spatiotemporal connectionist networks revisited: The unsupervised case. *Neural Computation*, 15, 1255-1320.

Christiansen, M. & Chater, N. (1999). Toward a connectionist model of recursion in human linguistic performance. *Cognitive Science*, 23(2), 157-205.

Christiansen, M. & Dale, R. (2001). Integrating distributional, prosodic and phonological information in a connectionist model of language acquisition. In *Proc. of the 23rd Annual Conf. of the Cognitive Science Society* (pp. 220-225). Mahwah, NJ: Lawrence Erlbaum.

Elman, J. (1990). Finding structure in time. *Cognitive Science*, 14,179-211.

Elman, J. (1991). Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, 7, 195-225.

Elman, J. (1993). Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1), 71-99.

Hammer, B., Micheli, A., Sperduti, A., & Strickert, M. (2004a). Recursive self-organizing network models. *Neural Networks*, 17(8-9), 1061-1085.

Hammer, B., Micheli, A., Strickert, M., & Sperduti, A. (2004b). A general framework for unsupervised processing of structured data. *Neurocomputing*, 57, 3-35.

Jaeger, H. (2001). Short term memory in echo state networks. (Tech. Rep. GMD Report 152). German National Research Center for Information Technology.

James, D. & Miikkulainen, R. (1995). SardNet: a self-organizing feature map for sequences. In *Advances in NIPS, 7* (pp. 577-584). MIT Press.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, 78(9), 1464-1480.

Maass, W., Natschläger, T., & Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11), 2531-2560.

Mayberry, M. & Crocker, M. (2004). Generating semantic graphs through self-organization. In *Proceedings of the AAAI Symposium on Compositional Connectionism in Cognitive Science*, Washington, DC: Erlbaum.

Mayberry, M. & Miikkulainen, R. (1999). Using a sequential SOM to parse long-term dependencies. In *Proc. of the 21st Annual Conf. of the Cognitive Science Society*, Hillsdale, NJ. Erlbaum.

Mayberry, M. & Miikkulainen, R. (2003). Incremental nonmonotonic parsing through semantic self-organization. In *Proc. of the 25th Annual Conf. of the Cognitive Science Society*, Mahwah, NJ. Erlbaum.

Parfitt, S., Tiňo, P., & Dorffner, G. (2000). Graded grammaticality in prediction fractal machines. In *Advances of NIPS, 12*. MIT Press.

Pearlmutter, B. (1995). Gradient calculations for dynamic recurrent neural networks: A survey. *IEEE Trans. on Neural Networks*, 6(5), 1212-1228.

Rohde, D. (2002). The simple language generator: Encoding complex languages with simple grammars. http://tedlab.mit.edu/~dr/SLG.

Rohde, D. & Plaut, D. (1997). Simple recurrent networks and natural language: How important is starting small? In *Proc. of the 19th Annual Conf. of the Cognitive Science Society* (pp. 656-661). Hillsdale, NJ: Erlbaum.

Servan-Schreiber, D., Cleeremans, A., & McClelland, J. (1991). Graded state machines: The representation of temporal contingencies in simple recurrent networks. *Machine Learning*, 7(2-3), 161-193.

Stolcke, A. (2002). SRILM – an extensible language modeling toolkit. In *Proc. International Conf. on Spoken Language Processing* (pp. 901-904). Denver, CO.

Tarassenko, L. & Roberts, S. (1994). Supervised and unsupervised learning in radial basis function classifiers. *IEE Proceedings – Visual Image Signal Processing*, 141(4), 210-216.

Tiňo, P. & Farkaš, I. (2005). On non-markovian topographic organization of receptive fields in Recursive Self-Organizing Map. In *Advances in Natural Computation* (pp. 676-685). Lecture Notes in Computer Science, Springer.

Tiňo, P., Farkaš, I., & van Mourik, J. (2005). Recursive Self-Organizing Map as a contractive iterative function system. In *Intelligent Data Engineering and Automated Learning* (pp. 327-334). Lecture Notes in Computer Science, Springer.

Tiňo, P., Farkaš, I., & van Mourik, J. (2006). Dynamics and topographic organization in recursive self-organizing map. Accepted to Neural Computation.

Tiňo, P., Čerňanský, M., & Beňušková, Ľ. (2004). Markovian architectural bias of recurrent neural networks. *IEEE Transactions on Neural Networks*, 15, 6-15.

Voegtlin, T. (2002). Recursive self-organizing maps. *Neural Networks*, 15(8-9), 979-992.

Werbos, P. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78, 1550-1560.