Systematicity in sentence processing with a recursive self-organizing neural network

Igor Farkaš[1]   and Matthew W. Crocker[2]

1- Dept. of Aplied Informatics, Comenius University
Mlynská dolina, 842 48 Bratislava, Slovak Republic

2- Dept. of Computational Linguistics, Saarland University
Saarbrücken, 66041, Germany

Abstract.   As potential candidates for human cognition, connectionist models of sentence processing must learn to behave systematically by generalizing from a small traning set. It was recently shown that Elman networks and, to a greater extent, echo state networks (ESN) possess limited ability to generalize in artificial language learning tasks. We study this capacity for the recently introduced recursive self-organizing neural network model and show that its performance is comparable with ESNs.

1   Introduction

To be considered a viable approach to human cognition, connectionist models must be able to account for the systematicity in human language. This potential capability was questioned by Fodor and Pylyshyn [1] and is still a matter of debate. Hadley [2] noted that systematicity is a matter of learning and generalization, and made a distinction betweeen weak and strong systematicity. A network is weakly systematic if it can process sentences with novel combinations of words, but these words are in the syntactic positions they also occurred in during training (e.g. the network trained on sentences *boy loves girl* and *dog chases cat* can also process *dog chases girl*). Strong systematicity, on the other hand, does require generalization to new syntactic positions (e.g. ability to process *dog chases boy*, provided that *boy* never appeared as object during training).

According to Hadley, connectionist models were at best weakly systematic. But van der Velde [3], using simulations with the Elman's simple recurrent network (SRN) in a next-word prediction task, claimed that even weak systematicity lies beyond the abilities of connectionist models. However, Frank [4] extended these simulations and showed that even SRN showed some generalization performance whose limitations arose from overfitting in large networks. Furthermore, he demonstrated that generalization could be improved by employing an alternative architecture – the echo-state network (ESN; [5]) that requires less training and does not suffer from overfitting.

We investigated the potential benefit of self-organization in learning context representations by experimenting with various recursive self-organizing modules, coupled with two types of a single-layer prediction module [6]. In a next-word prediction language task we showed that the best performance was achieved by the so called RecSOMsard module (explained below) coupled with a simple

perceptron. In this paper, we investigate the weak syntactic systematicity of the
RecSOMsard-based model and compare the performance with ESN [4].

2   Input data

The sentences were constructed using the grammar in Table 1, which subsumes
the grammar used in [3]. The language consists of three sentence types: sim-
ple sentences with an N-V-N structure, and two types of complex sentences,
namely, right-branching sentences with an N-V-N-*who*-V-N structure and centre-
embedded sentences with an N-*who*-N-V-V-N structure. Complex sentence types
represent commonly used English sentences such as *boy loves girl who walks dog*
and *girl who boy loves walks dog*, respectively.

S → Simple (.2) | Right (.4) | Centre (.4)	N → N$_1$ | N$_2$ | N$_3$ | N$_4$
Simple → N V N .	V → V$_1$ | V$_2$ | V$_3$ | V$_4$
Right → N V N *who* V N .	N$_x$ → n$_{x1}$ | n$_{x2}$ | ...
Centre → N *who* N V V N .	V$_x$ → v$_{x1}$ | v$_{x2}$ | ...

Table 1: Grammar used for generating training and test sentences.

Content words (nouns and verbs) are divided into four groups $N_1, ..., N_4$ and
$V_1, ..., V_4$. If W denotes the lexicon size (i.e. the total number of word types
in the language), each group has $(W\text{-}2)/8$ nouns and the same number of verbs
(*who* and "." are also considered words). Hence, for $W=18$ we have four nouns
and four verbs per group. The training set consisted of all sentences in which all
content words were taken from the same group, i.e. simple sentences had the form
"n_{xi} v_{xj} n_{xk} ." (analogically for complex sentences). The range of indices i, j, k
depends on W. In contrast to training sentences, each test sentence contained
content words from as many different groups as possible (as the most complicated
case, pursued in [3, 4]), i.e. each simple sentence had the form "n_{xi} v_{yj} n_{zk} .",
where $x \neq y \neq z$. This training scheme ensured that the proportion of training
sentences relative to all possible sentences remained very small ($\sim$0.4%).

3   RecSOMsard-P2 model

Our model consists of two modules that can be trained separately: a context-
learning RecSOMsard and a two-layer perceptron (P2). Adding a hidden layer
of units in the prediction module was shown to enhance prediction accuracy [4]
and hence is used also here for consistency.

The architecture of the RecSOMsard module is shown in Figure 1a. It is
based on RecSOM that has an extra top layer attached to it. Each RecSOM unit
$i \in \{1, 2, ..., N\}$ has two weight vectors associated with it: $\mathbf{w}_i \in \mathcal{R}^W$ linked with
an W-dimensional input $\mathbf{s}(t)$, and $\mathbf{c}_i \in \mathcal{R}^N$ linked with the context $\mathbf{y}(t-1) =
(y_1(t-1), y_2(t-1), ..., y_N(t-1))$.

The output of a unit i at time t is computed as $y_i(t) = \exp(-d_i(t))$, where
$d_i(t) = \alpha\|\mathbf{s}(t) - \mathbf{w}_i\|^2 + \beta\|\mathbf{y}(t-1) - \mathbf{c}_i\|^2$ with $\|\cdot\|$ denoting the Euclidean norm.
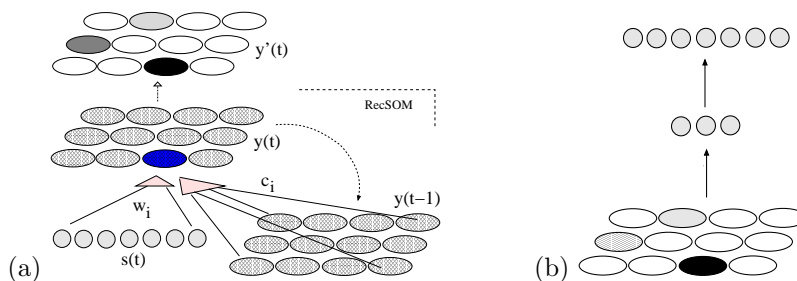
Fig. 1: (a) RecSOMsard architecture. The bottom part (without the top layer) represents RecSOM whose activity vector $\mathbf{y}$ is transformed to $\mathbf{y}$' by a mechanism described in the text. (b) A two-layer perceptron with inputs $\mathbf{y}$'.

Parameters $\alpha > 0$ and $\beta > 0$ respectively influence the effect of the input and the context upon a unit's profile. Both weight vectors can be updated using the same form of rule: $\Delta\mathbf{w}_i = \gamma h_{ik}(t)(\mathbf{s}(t) - \mathbf{w}_i)$, and $\Delta\mathbf{c}_i = \gamma h_{ik}(t)(\mathbf{y}(t-1) - \mathbf{c}_i)$, where k is an index of the winner (i.e. the unit with the highest $y_k(t)$), and $0 < \gamma < 1$ is the learning rate [7]. Radius of the neighborhood function $h_{ik}(t)$ linearly decreases in time to allow for forming topographic representation of input sequences. RecSOM units self-organize to topographically represent temporal contexts (subsequences) in a Markovian manner. At the same time, a more complex sequence can lead to non-Markovian behavior [8]. For an overview of recursive self-organizing networks see [10].

The RecSOMsard module contains a SardNet-like [9] output postprocessing to be fed to a prediction module. In each iteration, the winner's activation y_i in RecSOM is transformed to a sharp Gaussian profile y_i' centered arround the winner, and previous activations in the top layer are decayed via $y' \leftarrow \lambda y'$ (as in SardNet). At boundaries between sequences, all y_i' are reset to zero. This transforms the activation vector $\mathbf{y}(t)$ with mostly unimodal shape into a distributed activation vector $\mathbf{y}'(t)$ whose number of peaks equals the position of a current word in a sentence. This way, the context in RecSOMsard becomes represented both spatially (due to SardNet) and temporally (due to RecSOM).

Training the networks. Using localist encodings of words, networks were trained on the next word prediction task (i.e. one word at a time). All training sentences were concatenated in random order. The ratio of complex to simple sentences was 4:1. For each model and lexicon size, 10 networks were trained for $\sim$300,000 iterations and differed only in initial weights, randomly set to small values. Some RecSOMsard parameters were fixed: λ=0.9, γ=0.1, the effect of others was investigated (N, α, β). The perceptron, having 10 hidden units with logistic activation function (as in [4]), was trained by online back-propagation (without momentum), with learning rate set to linearly decrease from 0.1 to 0.01. Cross-entropy was used as an error function, therefore the perceptron output units had softmax activation functions, i.e. $a_i = \mathrm{e}^{net_i} / \sum_j \mathrm{e}^{net_j}$, where net_j is the total input activation received by output unit j.
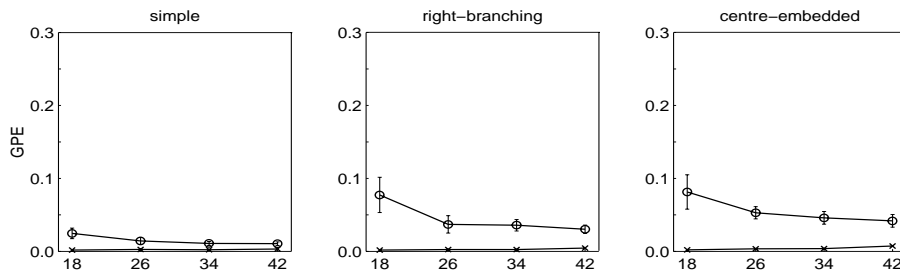
Fig. 2: Mean GPE measure for the three sentence types as a function of lexicon size W. Error bars were negligible for training data denoted by 'x' and hence are not shown. The lines marked with 'o' refer to the testing data.
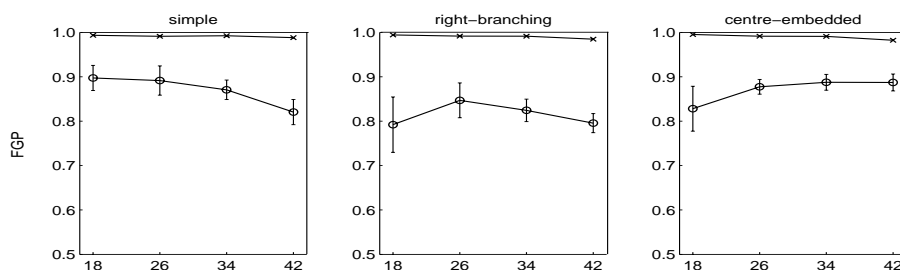

Fig. 3: Mean FGP measure for the three sentence types as a function of lexicon size W ('x' = training data, 'o' = testing data).

4   Results

We rated the network performance using two measures. One is the grammatical prediction error (GPE) defined as the ratio of the sum of non-grammatical output activations $a(\neg G) = \sum_{i \notin G} a_i$ and the sum of total output activation $a(\neg G) + a(G)$ [3]. Since we have softmax output units, GPE $= a(\neg G)$. Frank argues that GPE lacks the baseline (that would correspond to the network with no generalization) and that this problem is overcome in the alternative measure he introduced (see [4] for details) to quantify the generalization (we will call it Frank's generalization performance, FGP). FGP is based on comparing network's $a(G)$ with the predictions $b(G)$ of a bigram statistical model. The marginal cases result as follows: If $a(G) = 1$, then FGP $= 1$ (perfect generalization); if $a(G) = 0$ (completely non-grammatical predictions), then FGP $= -1$; a non-generalizing network with $a(G) = b(G)$ (i.e. behaving as a bigram model) would yield FGP $= 0$. Hence, positive FGP score measures degree of generalization.

We ran extensive simulations that can be divided into three stages:
(1) *Effect of N*: We looked for a reasonable number of map units N (using map radii 9, 10, 12, 14, 16) and found that beyond $N = 12 \times 12 = 144$ the test performance stopped to significantly improve. Hence, for all subsequent simulations we used this network size.

(2) *Effect of α and β*: We found that the mean GPE (e.g. its slope as a function of W) depended on specific values of map parameters. At the same time, we found the following conditions resulting in best performance: $0.8 \leq \alpha \leq 1.5$, $0.2 \leq \beta \leq 0.6$ and $0.4 \leq \alpha - \beta \leq 1.1$. Figure 2 shows the mean of mean GPEs (computed for 8 concrete α-β pairs from the above intervals) as a function of W. Training error was negligible, and test error that remains below 10% can be seen to reduce with larger lexicon. Similar dependence was observed in [4] in terms of increasing FGP, both in case of SRN and ESN models. In our model (Figure 3), in terms of FGP this trend is only visible for centre-embedded sentences. Also, our FGP values are a little bit worse than those of ESN [4], but still clearly indicating the presence of generalization.
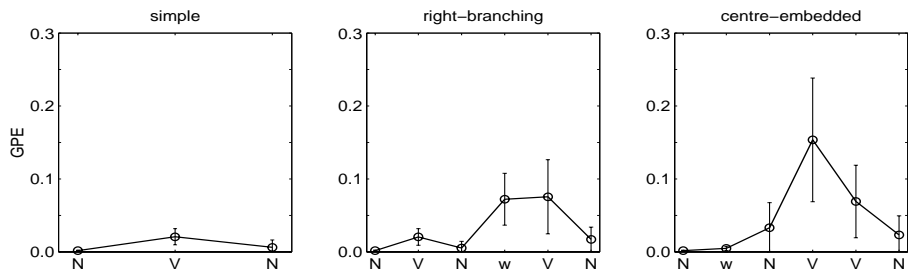

Fig. 4: Mean GPE for the three sentence types, averaged over test sentences and all lexicon sizes ($N = 144, \alpha = 0.8, \beta = 0.4$).
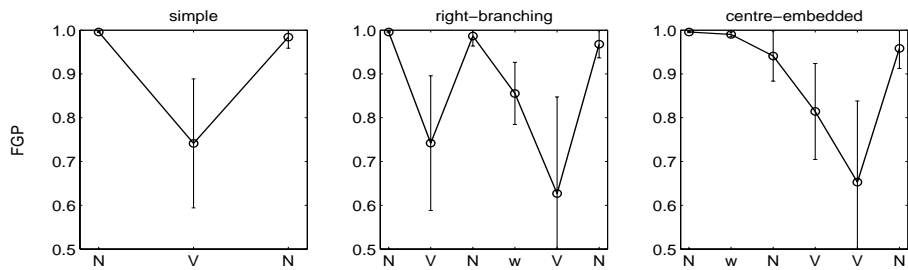

Fig. 5: Mean FGP for the three sentence types, averaged over test sentences and all lexicon sizes ($N = 144, \alpha = 0.8, \beta = 0.4$).

(3) *Mean word predictions:* Next, we took the model (given by N, α, β) with the lowest mean GPE and calculated its performance for individual inputs (words) in tested sentences, averaged over all lexicon sizes (N). Again, GPE on training data was close to 0. Results for test data are shown in Figure 4. The corresponding performance in terms of FGP in Figure 5 stays above 0.6. Compared to ESN [4], this falls between the best ESN model (for best W and N) whose FGP$\geq$0.8 and the mean FGP performance (averaged over W and N) which drops to 0.5 for most difficult cases in complex sentences. In our model, the most difficult predictions were similar for all word positions, and can be seen

from both figures. Unlike ESN, end-of-sentence markers were predicted very well by our model. Both figures suggest that the RecSOMsard-P2 network is also capable of generalization. The two measures appear to be inversely related, but differ in the fact that only FGP depends on bigram performance (which could explain different peak positions in the two graphs for complex sentences with centre-embedding).

5   Conclusion

In the context of weak systematicity, we showed that RecSOMsard-P2, like ESN, considerably avoids making non-grammatical predictions (quantified by GPE measure), and by doing that, it displays some generalization (quantified by positive FGP). A more rigorous comparison study with more complex data would be required in order to find out to what extent self-organization has its merits in learning context representations, as opposed to untrained weights used in ESN.

This type of work aims to shed light on the question how purely unbiased, distributional information can inform the learning the systematic syntactic knowledge in a variety of neural net architectures and training scenarios (SRN, ESN, RecSOMsard). This observation undermines the claim made by Fodor and Pylylshyn (or some of their supporters) that even if you find one example of connectionist systematicity, it doesn't really count because connectionism should be systematic "in general" to be taken seriously as a cognitive model. Investigating this learning ability is a prerequisite to developing more cognitively faithful models, such as of infant language acquisition.

References

[1] J.A. Fodor and Z.W. Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28:3–71, 1988.

[2] R.F. Hadley. Systematicity in connectionist language learning. *Mind and Language*, 9(3):247–272, 1994.

[3] F. van der Velde, G. van der Voort van der Kleij, and M. de Kamps. Lack of combinatorial productivity in language processing with simple recurrent networks. *Connection Science*, 16(1):21–46, 2004.

[4] S. Frank. Learn more by training less: systematicity in sentence processing by recurrent networks. *Connection Science*, 18(3):287–302, 2006.

[5] H. Jaeger. Adaptive nonlinear system identification with echo state networks. In *Advances in NIPS, 15*, pp. 593–600, MIT Press, 2003.

[6] I. Farkaš and M.W. Crocker. Recurrent networks and natural language: exploiting self-organization. In *Proc. of 28th Ann. Conf. of Cog. Sci. Soc.*, pp. 1275–1280, 2006.

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

[8] P. Tiňo, I. Farkaš, and J. van Mourik. Dynamics and topographic organization in recursive self-organizing map. *Neural Computation*, 18:2529–2567, 2006.

[9] D. James and R. Miikkulainen. SardNet: a self-organizing feature map for sequences. In *Advances in NIPS, 7,* pp. 577–584. MIT Press, 1995.

[10] B. Hammer, A. Micheli, A. Sperduti and M. Strickert. Recursive self-organizing network models. *Neural Networks*, 17:1061–1085, 2004.