Generalization and Systematicity in Echo State Networks

Stefan L. Frank (sfrank@science.uva.nl)
Institute for Logic, Language and Computation, University of Amsterdam
Plantage Muidergracht 24, 1018 TV Amsterdam, The Netherlands

Michal Čerňanský (cernansky@fiit.stuba.sk)
Institute of Applied Informatics, Slovak University of Technology
Ilkovicova 3, 842 16 Bratislava 4, Slovakia

Abstract

Echo state networks (ESNs) are recurrent neural networks that can be trained efficiently because the weights of recurrent connections remain fixed at random values. Investigations of these networks' ability to generalize in sentence-processing tasks have resulted in mixed outcomes. Here, we argue that ESNs do generalize but that they are not systematic, which we define as the ability to generally outperform Markov models on test sentences that violate the training sentences' grammar. Moreover, we show that systematicity in ESNs can easily be obtained by switching from arbitrary to informative representations of words, suggesting that the information provided by such representations facilitates connectionist systematicity.

Keywords: Recurrent neural networks; Echo state networks; Markov models; Generalization; Systematicity; Sentence processing; Non-symbolic representations.

Introduction

In an influential paper, Fodor and Pylyshyn (1988) argued that neural networks cannot display the systematicity observed in human language and thought, except by directly implementing a Classical symbol system. Consequently, no progress in cognitive science can be expected from connectionist approaches. Twenty years later, this issue is still debated. Here, we investigate systematicity in connectionist sentence processing, taking next-word prediction as the paradigm task.

In the next-word prediction task, a model is given a set of training sentences and one or more test sentences. Using the information in the training data, the model has to predict the next word at each position in the test sentence(s). Since correct prediction is not generally possible, the model is said to perform perfectly if it gives correct next-word *probabilities*.

In this paper, we provide a definition of systematic performance in next-word prediction, and show that a currently popular type of recurrent neural network, the echo state network (ESN), fails to be systematic. Switching from symbolic to non-symbolic representations of words, however, results in ESN systematicity while retaining the network's desirable property of being efficiently trainable.

Generalization and systematicity

Fodor and Pylyshyn (1988) failed to operationalize systematicity in a manner that allows for quantifying a neural network's systematic behavior. As noted by Hadley (1994), a model's systematicity is apparent in the extent to which it generalizes, that is, its ability to sufficiently deal with untrained inputs. This raises two questions. First, when does the network perform 'sufficiently'? A network that performs worse on new sentences than on training sentences does not necessarily fail to generalize completely. Second, how much should new inputs differ from training examples? Clearly, if just *any* indication of generalization would suffice, the issue would already be decided in favor of connectionism.

To answer the first question, we consider a new input to be processed sufficiently if the network outperforms Markov models. The rationale behind this is as follows: If a well-trained but non-generalizing word-prediction system is faced with the new input sentence $w_{t-x},\ldots,w_{t-2},w_{t-1}$, the best it can do is base its prediction for w_t on the most recent n-word sequence (i.e., $w_{t-n},\ldots,w_{t-1}$; with $n \leq x$) that also appeared in the training data, ignoring the earlier words. Such a model is called an nth order Markov model. Since we have defined the system to be non-generalizing, it cannot use an n-word sequence that is too long to have appeared in the training data. Therefore, we consider a network to generalize (to some extent at least) if it generally performs better than *any* Markov model (i.e., for any n) on test sentences.

In real-life applications, we cannot know which n would be best, so its value needs to be fixed or could depend on the occurrence of the test sequence in the training data, turning the model into a Variable Length Markov model (VLMM). Taking the best Markov model as the baseline for sufficient performance obviously results in a much stricter test than would a fixed n or a VLMM baseline. According to our definition of sufficient performance, therefore, earlier claims of connectionist systematicity that were based on comparisons to a 1st order Markov model (Frank, 2006a) or VLMM (Frank, 2006b) are no longer warranted.

As for the second question, we argue[1] that the difference between 'mere' generalization and systematicity parallels the difference between ergodic and non-ergodic sampling of training sentences. In ergodic sampling, the distribution of the sample is guaranteed to converge to the true distribution as sample size grows. Presumably, this property gives the network the opportunity to correctly process test sentences by some sort of interpolation from the sampled training examples. In non-ergodic sampling, on the other hand, the sample will never come to accurately reflect the true distribution. For example, particular sentences may be excluded from the sample on purpose. In that case, the 'training grammar' that generated the training sentences is not identical to the under-

[1] And for those not convinced by our argument, we *define*.

lying 'true grammar' that can generate *all* grammatical sentences. No model can reliably learn the true grammar from a non-ergodic sample. To correctly process a test sentence that could *not* have been generated by the training grammar, the network needs to generalize to items that are markedly different from what it was trained on.

To summarize, we submit that *a network displays systematicity if it generally outperforms Markov models when processing sentences that could not have been generated by the grammar that generated the training examples.*

Echo state networks

Much of recent research into recurrent neural networks (RNNs) has focused on so-called 'reservoir computing'.[2] In this approach to RNN training, the weights of the network's input and recurrent connections remain untrained. The recurrent part of the network serves as a task-independent 'dynamical reservoir', while a non-recurrent 'read-out' network is trained to produce some desired output from the fluctuating patterns of reservoir activations.

One of the most influential reservoir-computing architectures is the echo state network (ESN; Jaeger, 2001). An ESN's read-out network has just a single layer of units, which means that setting the weights of connections to the output units is a simple linear regression task, which can be performed off-line after a single presentation of the training input. This training efficiency is, in fact, one of the main attractions of ESNs.

There have been only few attempts to apply ESNs to sentence processing, and results were mixed. Tong, Bickett, Christiansen, and Cottrell (2007) found ESN performance on the next-word prediction task to be comparable to that of the more traditional simple recurrent network (SRN; Elman, 1990). Contrary to this, Frank (2006a) reported that generalization by ESNs is impoverished compared to SRNs. Likewise, Čerňanský and Tiňo (2007) showed that ESNs cannot generalize above the level of VLMMs, and claim that SRNs can achieve higher performance than ESNs on some tasks.

Possibly, the crucial difference between the experiments by Tong et al. and Frank (2006a, 2006b) lies in ergodic versus non-ergodic sampling. According to our definition above, Frank tested the ESN for systematicity, while Tong et al. merely investigated non-systematic generalization. The results presented in this paper indeed indicate that ESNs can generalize but are not systematic.

Frank (2006a) showed that an ESN can generalize better than an SRN when a *two*-layer read-out network is used. Unfortunately, training such a network requires a slow, iterative algorithm, such as backpropagation, doing away with much of the charm of ESNs. Here, we shall show that ESN systematicity is possible without such a painstaking search for proper connection weights, keeping more in line with the original ESN approach. This is accomplished by following a suggestion by Phillips (1998), who found networks to lack systematicity in a symbol-processing task and remarked that this might be fixed if, somehow, additional information would be provided by prior similarity among the representations of inputs that should be treated similarly. Since systematicity might be trivially obtained if the modeler has complete freedom to choose any desired set of input representations, Phillips rightly argued that the choice of representations should be independently justified, for example by being based on the training data.

Basically, this is the strategy followed here. Using an efficient and largely task-independent method, informative representations of words are extracted from the training data, replacing the ESN's random (and thereby uninformative) representations. The resulting model outperforms the standard ESN and the Markov models when tested for systematicity.

The language

The language used in our experiments (based on Hadley, Rotaru-Varga, Arnold, & Cardei, 2001), has a 26-word vocabulary, comprising 12 nouns, 10 transitive verbs, 2 prepositions, a relative clause marker, and an end-of-sentence marker denoted *[end]*, which is also considered a word. As there are no semantic constraints, the names of words within each syntactic category are irrelevant and only provided to make sentences more readable. As explained below, the difference between female nouns (N_{fem}; e.g., *women*), male nouns (N_{male}; e.g., *men*) and animal nouns (N_{anim}; e.g., *bats*) is important for distinguishing between training and test sentences.

Table 1 shows the grammar that generates the language's sentences. These can contain two types of embedded clauses: subject-relative clauses (SRCs, as in *girls that see boys...*) and object-relative clauses (ORCs, as in *girls that boys see...*). Since SRCs can themselves contain a relative clause, there is no upper bound to sentence length.

Table 1: Probabilistic context-free grammar of the language. Variable *r* denotes grammatical role (subject or object). The probabilities of different productions are equal, except for NP, where they are given in parentheses.

S	→	NP_{subj} V NP_{obj} *[end]*
NP_r	→	N_r (.7) | N_r SRC (.06) | N_r ORC (.09) | N_r PP_r (.15)
SRC	→	*that* V NP_{obj}
ORC	→	*that* N_{subj} V
PP_r	→	*from* NP_r | *with* NP_r
N_r	→	N_{fem} | N_{male} | N_{anim}
N_{fem}	→	*women* | *girls* | *sisters*
N_{male}	→	*men* | *boys* | *brothers*
N_{anim}	→	*bats* | *giraffes* | *elephants* | *dogs* | *cats* | *mice*
V	→	*chase* | *see* | *swing* | *love* | *avoid* | *follow* | *hate* | *hit* | *eat* | *like*

[2] As is illustrated by the recent publication of a *Neural Networks* special issue on this topic (2007, Vol. 20, No. 3).

Training sentences

To test for systematicity, particular sentences were excluded from the training data by setting restrictions on the grammatical roles (indicated by *r* in Table 1) particular nouns can appear in. Training sentences never have a male noun in subject position or a female noun in object position. Animal nouns can occur in either position. This means that, for generating training sentences, the single production rule for nouns (N_r) in Table 1 was actually replaced by two rules in Table 2.

The models were trained five times, each time using a different set of 5 000 randomly generated training sentences with an average length of 5.9 words.

Table 2: Production rules for nouns when generating training sentences, replacing N_r of Table 1.

N_{subj}	$\rightarrow$	$N_{fem} \mid N_{anim}$
N_{obj}	$\rightarrow$	$N_{male} \mid N_{anim}$

Test sentences

The models are tested on two groups of new sentences: generalization-test sentences and systematicity-test sentences. Generalization-test sentences are subject to the restrictions on the nouns' grammatical roles that also apply to training sentences. That is, the production rules for nouns were as in Table 2. As a result, the training data formed an ergodic sample with respect to generalization-test sentences.

According to the true grammar in Table 1, all nouns should be treated equally, that is, wherever a noun can occur, *any* noun can occur. Systematicity-test sentence are grammatical according to this true grammar but since they violate the noun-role restrictions of Table 2, they were not generated by the grammar that generated training sentences. With respect to systematicity-test sentences, therefore, the training data form a non-ergodic sample.

Following Frank (2006b), test sentences have either a SRC or an ORC, that modifies either the first or second noun, making four types of test sentences, labeled SRC1, SRC2, ORC1, and ORC2. When testing for systematicity, the models processed *all* sentence with the structures of those in Table 3. Since there are three different (fe)male nouns and ten verbs (and each of the four types of test sentence has three nouns and two verbs), the number of systematicity-test sentences is $4 \times 3^3 \times 10^2 = 10\,800$. An example of each test sentence type can be found in Figure 1.

Note that each systematicity-test sentence (unlike any training sentence) begins with a male noun, so systematicity-test sentences differ from all training sentences from the very first word. This is not the case for generalization-test sentences, which (like all training sentences) begin with a female noun. As a result, the models do not need to generalize when processing the first word(s) of generalization-test sentences. Up to a point, each generalization-test sentence will have appeared in the training data. Only from this point onwards is generalization required to process the sentences. This is why, in Figure 1, no generalization results are plotted for the first two or three words. In fact, 41 out of 10 800 potential generalization-test sentences appeared in the training data, so these were not used in the generalization test at all.

Table 3: Structure of four types of systematicity-test sentences. Replacing N_{fem} by N_{male} and vice versa turns these into generalization-test sentences.

Type	Sentence structure
SRC1	N_{male} *that* V N_{fem} V N_{fem} *[end]*
SRC2	N_{male} V N_{fem} *that* V N_{fem} *[end]*
ORC1	N_{male} *that* N_{male} V V N_{fem} *[end]*
ORC2	N_{male} V N_{fem} *that* N_{male} V *[end]*

The models

Markov models

Let $N(w_{t-n},\ldots,w_{t-1})$ denote the number of times that the *n*-word test sequence $w_{t-n},\ldots,w_{t-1}$ appears in the training data. According to the *n*th order Markov model, the probability that word *i* directly follows the sequence equals[3]

$$\Pr(i|w_{t-n},\ldots,w_{t-1}) = \frac{N(w_{t-n},\ldots,w_{t-1},i)}{N(w_{t-n},\ldots,w_{t-1})}. \qquad (1)$$

Specific Markov models differ in the value of *n*, that is, in their order. In the simplest case, $n = 0$, so $\Pr(i) = N(i)/N$, where N is the number of words in the training data. This so-called *unigram* model ignores the input all together and always estimates the probability of a word by its relative frequency in in the training data. In the *bigram* model, $n = 1$, so only the current input is taken into account and Equation 1 reduces to $\Pr(i|w_{t-1}) = N(w_{t-1},i)/N(w_{t-1})$.

Note that larger *n* does not need to result in more accurate predictions. It is even possible that the simple unigram model outperforms all higher order Markov models, and in fact it often does in our systematicity tests. Therefore, at each point of each test sentence type, we take the *best* Markov model for all *n* (up to the number of words in the test sentence so far).

Echo state network

The architecture of our ESN is basically the same as that of a three-layer SRN: Words are presented at the input layer, whose activation is propagated to the hidden layer (called 'dynamical reservoir' in an ESN) that also receives its own previous activation state. The output layer receives activation from the hidden layer, and is trained to predict the next input word. As mentioned in the Introduction, the main difference between an ESN and an SRN is that an ESN has fixed,

[3] Markov models often involve smoothing to prevent the occurrence of a zero in the fraction of Equation 1. We also ran our experiments using Laplace smoothing, but found no qualitative differences with the results of the unsmoothed models presented here.

random input and recurrent connections weights, whereas all weights of an SRN are adjusted during training.

Input When word i forms the input to the ESN, it is represented by a vector $\mathbf{w}_i = (w_{i,1}, \ldots, w_{i,26})$, the number of elements of which equals the number of word types in the language.[4] In a standard ESN, the values in these vectors are chosen at random. Here, we compare this approach to one in which word representations are based on the training data. Bullinaria and Levy (2007) compared different techniques for extracting such representations from text corpora and found a surprisingly simple method to result in very good performance on a variety of syntactic and semantic tasks. Our model uses this so-called 'ratios' method, according to which the jth element of the vector representing word i depends on the number of times words i and j occur next to each other in the training data:

$$w_{i,j} = N \times \frac{N(i,j) + N(j,i)}{N(i)N(j)}.$$

The network that uses these word representations will be called ESN+. To make the comparison between ESN and ESN+ as fair as possible, ESN's random representations are obtained by randomly reordering all values $w_{i,j}$ so that word representations in ESN and ESN+ contain the same values. Only in ESN+, however, do they provide information about word co-occurrences in the training sentences.

Dynamical reservoir The ESN's dynamical reservoir (DR) consists of k units that receive input from some external source and from each other. The $k \times k$ matrix $\mathbf{W}_{\text{dr}}$ contains the weights of connections between the DR units. The DR is sparsely connected in that 85% of values in $\mathbf{W}_{\text{dr}}$ are 0. All other values are taken randomly from a uniform distribution centered at 0, after which they are rescaled such that the spectral radius of $\mathbf{W}_{\text{dr}}$ (i.e., its largest eigenvalue) equals 1. Each of the five repetitions of ESN(+) training used another $\mathbf{W}_{\text{dr}}$.

The DR's activation vector at time step t is denoted $\mathbf{a}_{\text{dr}}(t) \in [0,1]^k$. At each time step, the vector representing the current input word i enters the DR, which also receives its own previous activation. The new activation vector is computed by

$$\mathbf{a}_{\text{dr}}(t) = \mathbf{f}(s_{\text{dr}}\mathbf{W}_{\text{dr}}\mathbf{a}_{\text{dr}}(t-1) + s_{\text{in}}\mathbf{w}_i), \qquad (2)$$

where s_{dr} and s_{in} are parameters controlling DR and input scaling respectively, $\mathbf{a}_{\text{dr}}(t-1)$ is the DR state in the previous time step (with $\mathbf{a}_{\text{dr}}(0) = .\mathbf{5}$ at the beginning of each sentence), and $\mathbf{f}$ is the logistic function. Note that the addition in Equation 2 is only possible if $k = 26$. Since we use values of $k > 26$, vector $\mathbf{w}_i$ should be imagined as having $k - 26$ zeros concatenated to it.

[4] A more common but equivalent way to denote this is by collecting the input vectors in an input connection weight matrix $\mathbf{W}_{\text{in}} = (\mathbf{w}_1, \ldots, \mathbf{w}_{26})$ that is multiplied by an input activation vector $\mathbf{a}_{\text{in}} = (a_1, \ldots, a_{26})$ with $a_i = 1$ if i is the current input and $a_i = 0$ otherwise.

Output The DR sends activation to the network's 26 output units which correspond to the language's 26 words. The weights of connections from DR units to outputs are collected in the $26 \times k$ matrix $\mathbf{W}_{\text{out}}$. Also, each output unit receives a bias activation. The output at time step t equals

$$\mathbf{a}_{\text{out}}(t) = \mathbf{W}_{\text{out}}\mathbf{a}_{\text{dr}}(t) + \mathbf{b},$$

where $\mathbf{b}$ is the vector of bias activations. Output vector $\mathbf{a}_{\text{out}}$ is transformed to a probability distribution by setting all its negative values to 0 and rescaling the rest to sum to 1.

Training Optimal output connection weights and bias vector, $\mathbf{W}_{\text{out}}$ and $\mathbf{b}$, are easy to find without any iterative training method.[5] First, we construct a $26 \times (N-1)$ target matrix $\mathbf{U} = (\mathbf{u}(1), \mathbf{u}(2), \ldots, \mathbf{u}(N-1))$, where each $\mathbf{u}(t)$ is a column vector of 0s except for a single 1 for the element corresponding to the input word at $t+1$. That is, the vector $\mathbf{u}(t)$ forms the correct prediction of the input at $t+1$.

Next, the complete training sequence (excluding the last word) is run through the DR, according to Equation 2. The resulting vectors $\mathbf{a}_{\text{dr}}$ are collected in a matrix $\mathbf{A}$ to which a row of 1s is concatenated, resulting in a $(k+1) \times (N-1)$-matrix. The connection weights and bias values are now computed by multiplying $\mathbf{U}$ with $\mathbf{A}$'s pseudoinverse: $\mathbf{W} = \mathbf{U}\mathbf{A}^{-1}$. The last column of $\mathbf{W}$ forms the bias vector $\mathbf{b}$, while the rest of $\mathbf{W}$ equals $\mathbf{W}_{\text{out}}$. If the ESN would process the training sequence again, these $\mathbf{W}_{\text{out}}$ and $\mathbf{b}$ minimize the MSE between network outputs $\mathbf{a}_{\text{out}}(t)$ and corresponding targets $\mathbf{u}(t)$.

Results

Performance on test sentences is rated by computing the cosines between the estimated next-word probabilities (i.e., the model's output vectors) and the true probabilities according to the grammar of Table 1. Values close to 1 indicate good generalization performance, while a cosine of 0 means that the two probability distributions are perpendicular. All results presented below are averaged over the five training repetitions.

ESN parameter setting

Three ESN parameters were manipulated: DR size $k \in \{100, 200, 300\}$, DR scaling $s_{\text{dr}} \in \{.05, .1, .3, .5, .7, .9, .95\}$, and input scaling $s_{\text{in}} \in \{.02, .1, .4, 2\}$. We took the parameter setting that resulted in best average performance on systematicity test sentences, for ESN and ESN+ separately. These values are shown in Table 4. Clearly, ESN+ can perform better than ESN. Note that ESN+ (unlike ESN) performs best at the extreme end of the parameter space that was explored, suggesting that its performance can be further improved by setting the parameters at more extreme values.

Generalization

The top row of Figure 1 plots the results for ESN, ESN+, and the best Markov model, at each word of each of the four types

[5] See Jaeger (2001) for a more comprehensive explanation of the ESN training procedure.

Table 4: Parameter values resulting in highest average performance on systematicity test sentences.

	parameter			average
Model	k	s_{dr}	s_{in}	performance
ESN	200	.1	.4	.834
ESN+	100	.95	2	.923

of generalization test sentences. Overall, ESN(+) performs at least as well as Markov models and can therefore be said to generalize. Only at the 5th word of ORC2 test sentences does ESN (but not ESN+) do worse than the best Markov model. The difference is small but highly significant ($N = 1280, z = 29.5, p \approx 0$ in a Wilcoxon matched-pairs signed-rank test).

Systematicity

As shown in the bottom row of Figure 1, Markov models do quite badly at many points of systematicity test sentences. For example, performance at the first word (a male noun) is very low. This is because male nouns never occurred in sentence-initial position in training sentences. ESNs, however, do not suffer from this problem: They score nearly perfectly at this point in systematicity test sentences. In general, ESN+ does at least as well as Markov models. Especially when Markov models perform badly, ESN+ does much better. Only at the fourth word of SRC1 sentences is ESN+ performance slightly (but significantly: $N = 1328, z = 17.9, p \approx 0$) lower than that of Markov models.

In contrast to ESN+, the standard ESN model often performs much worse than the best Markov model. According to our definition, this means that ESN (unlike ESN+) does not display systematicity.

Conclusion

After training an SRN on the prediction task in symbol-sequence processing, Čerňanský, Makula, and Beňušková (2007) attributed most of successful generalization to the learned representations of input symbols. This finding suggests that training recurrent connections may not be very important for some commonly used data sets, and that ESN generalization can be improved by adjusting the input representations instead of leaving them random. Indeed, this is precisely what we found. Importantly, appropriate representations could be computed efficiently from the training data.

We have shown that an ESN generalizes but is not systematic in sentence processing: It generally performs better than a Markov model on generalization-test sentences, but not on systematicity-test sentences. This finding sheds light on the apparent inconsistency between Tong et al.'s (2007) and Frank's (2006a) conclusions on ESNs' ability to generalize: Unlike Tong et al., Frank tested for systematicity. Moreover, our results show that ESN+ makes systematic connectionist sentence processing possible without the need for backpropagation or any other iterative training algorithm.

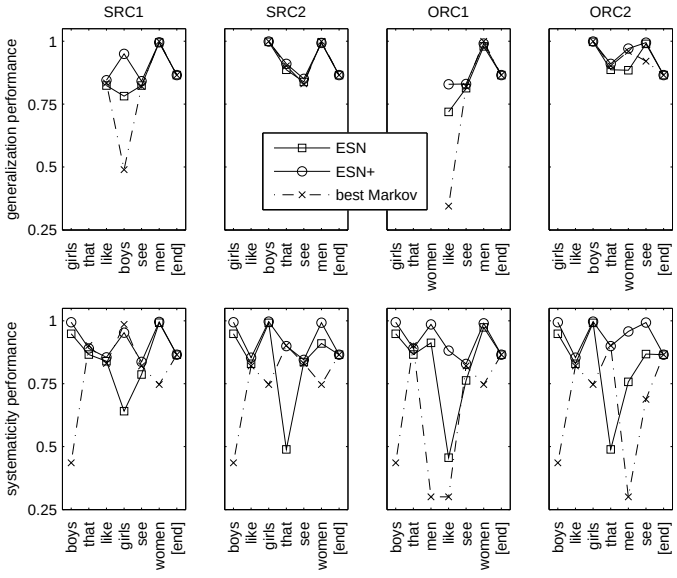

Figure 1: Performance at each point in four types of generalization (top) and systematicity (bottom) test sentences, by ESN, ESN+, and the best Markov model. Results are averaged over all test sentences of a type; those shown on the x-axis are just examples.

Systematicity and representation

Peirce (1903/1985) defines a representation as *symbolic* if its form is related arbitrarily to its meaning. It is clear that Markov models treat words as symbols in this sense: The word forms *i* and *j* (or *girls* and *women*, for that matter) provide no information whatsoever about their meaning or the positions they can take in a sentence. It is often believed that neural networks are non-symbolic because they use distributed vector representations. However, in standard ESNs, words are represented by *random* vectors, which are arbitrary by definition. ESNs therefore represent words symbolically. In contrast, in ESN+ (and trained SRNs) relations among the words' representations reflect relations among the words themselves. More precisely, words belonging to the same grammatical category have similar representations. Consequently, they affect the network's dynamical reservoir similarly. This non-symbolic representational scheme is crucial for the systematicity observed in the ESN+ model.

Tiňo, Čerňanský, and Beňušková (2004) showed that the state space of an ESN's dynamical reservoir (and, more generally, of an RNN with small random weights) shows considerable structural differentiation when processing symbol sequences. Each symbol has an attractor point in the state space, and every time a symbol is presented to the network, its state moves towards that symbol's attractor. Since the previous state was determined mostly by the symbol previously presented (which also moved the state towards its attractor), the DR's current state reflects the history of all previously presented symbols. An ESN explicitly uses this organization.

However, since symbol representations in a standard ESN are random, so are the attractor points. This makes it difficult for the network to generalize over symbols that should be treated similarly (e.g., because they are all nouns).

In ESN+, the situation is different: The attractor points of words from the same grammatical category are closer together than those of words from different categories. This facilitates generalization over words from the same grammatical category. As a result, ESN+ outperforms ESN when faced with systematicity test sentences. This finding illustrates the importance of switching from symbolic to non-symbolic representations. Likewise, Frank, Haselager, and Van Rooij (2007) argue that the use of non-symbolic representations of sentential meaning is vital to the semantic systematicity displayed in their connectionist sentence-comprehension model.

Weak and strong systematicity

In an investigation of systematicity in connectionist models of sentence processing, Hadley (1994) argued that the models that were around at the time did not account for human levels of systematicity because they displayed only 'weak systematicity', as he called it. Hadley defined weak systematicity as the ability to correctly process test sentences that have words occurring only in the same positions they held during training. In contrast, 'strong systematicity' is the ability to correctly process test sentences that have words in positions that differ from those in the training examples. Moreover, the network should also be able to handle test sentences with embedded clauses containing words in untrained positions. According to Hadley, people display strong systematicity, whereas neural networks are only weakly systematic at best.

Hadley's (1994) notion of weak systematicity subsumes our definition of non-systematic generalization (i.e., sufficient processing of test inputs after ergodic sampling). This is because, in a large enough ergodic sample of training sentences, all words will have occurred in all possible positions. Therefore, a non-systematic generalizing model (according to our definition) is weakly systematic (in Hadley's sense).

Our current test for systematicity comes down to testing for Hadley's strong systematicity. The restrictions on the occurrence of particular nouns in particular grammatical roles make sure that all systematicity-test sentences have words occurring in novel positions, both in their main clause and in the embedded clause. Therefore, not only have we shown that ESN+ can behave systematically, we have also met Hadley's challenge of displaying *strong* connectionist systematicity.

Acknowledgments

The research presented here was supported by grant 451-04-043 of the Netherlands Organization for Scientific Research (NWO) and grant APVV-20-030204 of the Slovak Research and Development Agency (APVV).

References

Bullinaria, J. A., & Levy, J. P. (2007). Extracting semantic representations from word co-occurrence statistics: a computational study. *Behavior Research Methods, 39*, 510–526.

Čerňanský, M., Makula, M., & Beňušková, Ľ. (2007). Organization of the state space of a simple recurrent network before and after training on linguistic structures. *Neural Networks, 20*, 236–244.

Čerňanský, M., & Tiňo, P. (2007). Comparison of Echo State Networks with Simple Recurrent Networks and Variable-Length Markov Models on symbolic sequences. In *Proceedings of ICANN 2007*. Berlin: Springer.

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

Fodor, J. A., & Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: a critical analysis. *Cognition, 28*, 3–71.

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

Frank, S. L. (2006b). Strong systematicity in sentence processing by an Echo State Network. In *Proceedings of ICANN 2006*. Berlin: Springer.

Frank, S. L., Haselager, W. F. G., & Van Rooij, I. (2007). *Connectionist semantic systematicity.* (Manuscript submitted for publication)

Hadley, R. F. (1994). Systematicity in connectionist language learning. *Mind & Language, 9*(3), 247–272.

Hadley, R. F., Rotaru-Varga, A., Arnold, D. V., & Cardei, V. C. (2001). Syntactic systematicity arising from semantic predictions in a Hebbian-competitive network. *Connection Science, 13*(1), 73–94.

Jaeger, H. (2001). *The "echo state" approach to analysing and training recurrent neural networks.* GMD report no. 148. GMD — German National Research Institute for Computer Science. http://www.faculty.iu-bremen.de/hjaeger/pubs/EchoStatesTechRep.pdf.

Peirce, C. S. (1903/1985). Logic as semiotics: The theory of signs. In R. E. Innis (Ed.), *Semiotics: An introductory anthology.* Bloomington, IN: Indiana University Press.

Phillips, S. (1998). Are feedforward and recurrent networks systematic? Analysis and implications for a connectionist cognitive architecture. *Connection Science, 10*, 137–160.

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

Tong, M. H., Bickett, A. D., Christiansen, E. M., & Cottrell, G. W. (2007). Learning grammatical structure with Echo State Networks. *Neural Networks, 20*, 424–432.