Tree-based Analysis of Simple Recurrent Network Learning

Ivelin Stoianov
Dept. Alfa-Informatica, Faculty of Arts, Groningen University, POBox 716, 9700 AS Groningen,
The Netherlands, Email:stoianov@let.rug.nl

1 Simple recurrent networks for natural language phonotactics analysis.

In searching for a connectionist paradigm capable of natural language processing, many researchers have explored the Simple Recurrent Network (SRN) such as Elman(1990), Cleermance(1993), Reilly(1995) and Lawrence(1996). SRNs have a context layer that keeps track of the past hidden neuron activations and enables them to deal with sequential data. The events in Natural Language span time so SRNs are needed to deal with them.

Among the various levels of language processing, a phonological level can be distinguished. The Phonology deals with phonemes or graphemes – the latter in the case when one works with orthographic word representations. The principles governing the combinations of these symbols is called phonotactics (Laver'1994). It is a good starting point for connectionist language analysis because there are not too many basic entities. The number of the symbols varies between 26 (for the Latin graphemes) and 50 [*](for the phonemes).

Recently, some experiments considering phonotactics modelling with SRNs have been carried out by Stoianov(1997), Rodd(1997). The neural network in Stoianov(1997) was trained to study the phonotactics of a large Dutch word corpus. This problem was implemented as an SRN learning task – to predict the symbol following the left context given to the input layer so far. Words were applied to the network, symbol by symbol, which in turn were encoded orthogonally, that is, one node standing for one symbol (Fig.1). An extra symbol ('#') was used as a delimiter. After the training, the network responded to the input with different neuron activations at the output layer. The more active a given output neuron is, the higher the probability is that it is a successor. The authors used a so-called *optimal threshold method* for establishing the threshold which determines the possible successors. This method was based on examining the network response to a test corpus of words belonging to the trained language and a random corpus, built up from random strings. Two error functions dependent on a threshold were computed, for the test and the random corpora, respectively. The threshold at which both errors had minimal value was selected as an optimal threshold. Using this approach, an SRN, trained to the phonotactics of a Dutch monosyllabic corpus containing 4500 words, was reported to distinguish words from non-words with 7% error. Since the phonotactics of a given language is represented by the constraints allowing a given sequence to be a word or not, and the SRN managed to distinguish words from random strings with tolerable error, the authors claim that SRNs are able to learn the phonotactics of Dutch language.

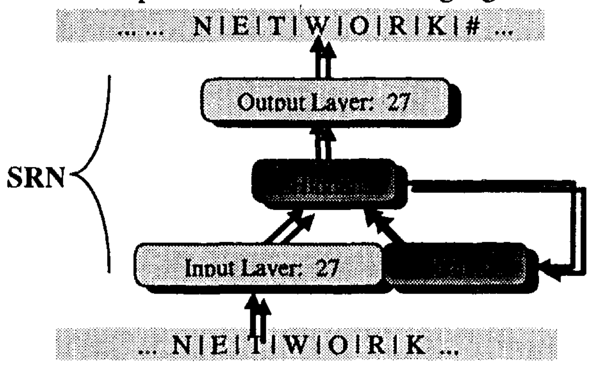

Fig.1. SRN and mechanism of sequence processing. A character is provided to the input and the next one is used for training. In turn, it has to be predicted during the test phase.

In the present report, alternative evaluation procedures are proposed. The network evaluation methods introduced are based on examining the network response to each left context, available in the training corpus. An effective way to represent and use the complete set of context strings is a tree-based data structure. Therefore, these methods are termed *tree-based analysis.* Two possible approaches are proposed for measuring the SRN response accuracy to each left context. The first uses the idea mentioned above of searching a threshold that distinguishes permitted successors from impossible ones. An error as a function of the

* for Dutch, and up to at most 100 in other languages.

threshold is computed. Its minimum value corresponds to the SRN learning error rate. The second approach computes the local proximity between the network response and a vector containing the empirical symbol probabilities that a given symbol would follow the current left context. Two measures are used: $\mathbf{L_2}$ norm and normalised vector multiplication. The mean of these local proximities measures how close the network responses are to the desired responses.

2 Tree-based corpus representation.

There are diverse methods to represent a given set of words (corpus). Lists is the simplest, but they are not optimal with regard to the memory complexity and the time complexity of the operations working with the data. A more effective method is the tree-based representation. Each node in this tree has a maximum of 26 possible children (successors), if we work with orthographic word representations. The root is empty, it does not represent a symbol. It is the beginning of a word. The leaves do not have successors and they always represent the end of a word. A word can end somewhere between the root and the leaves as well. This manner of corpus representation, termed *trie*, is one of the most compact representations and is very effective for different operations with words from the corpus.

In addition to the symbol at each node, we can keep additional information, for example the frequency of a word, if this node is the end of a word. Another useful piece of information is the frequency of each node C, that is, the frequency of each left context. It is computed recursively as a sum of the frequencies of all successors and the frequency of the word ending at this node, provided that such a word exists. These frequencies give us an instant evaluation of the empirical distribution for each successor. In order to compute the successors' empirical distribution vector $\mathbf{T}^c(.)$, we have to normalise the successors' frequencies with respect to their sum.

3 Tree-based evaluation of SRN learning.

During the training of a word, only one output neuron is forced to be active in response to the context presented so far. But usually, in the entire corpus there are several successors following a given context. Therefore, the training should result in output neurons, reproducing the successors' probability distribution. Following this reasoning, we can derive a test procedure that verifies whether the SRN output activations correspond to these local distributions. Another approach related to the practical implementation of a trained SRN is to search for a cue, giving an answer to the question whether given symbol can follow the context provided to the input layer so far. As in the *optimal threshold method* we can search for a threshold that distinguishes these neurons.

The tree-based learning examination methods are recursive procedures that process each tree node, performing an *in-order* (or *depth-first*) tree traversal. This kind of traversal algorithms start from the root and process each sub-tree completely. At each node, a comparison between the SRNs reaction to the input, and the empirical characters distribution is made. Apart from this evaluation, the SRN state, that is, the context layer, has to be kept before moving to one of the sub-trees, in order for it to be reused after traversing this sub-tree.

On the basis of above ideas, two methods for network evaluation are performed at each tree node C. The first one computes an error function $\mathbf{f}^c(\mathbf{t})$ dependent on a threshold $\mathbf{t}$. This function gives the error rate for each threshold $\mathbf{t}$, that is, the ratio of erroneous predictions given $\mathbf{t}$. The values of $\mathbf{f}^c(\mathbf{t})$ are high for close to zero and close to one thresholds, since almost all neurons would permit the correspondent symbols to be successors in the first case, and would not allow any successor in the second case. The minimum will occur somewhere in the middle, where only a few neurons would have an activation higher than this threshold. The training adjusts the weights of the network so that only neurons corresponding to actual successors are active. The SRN evaluation is based on the mean $\mathbf{F(t)}$ of these local error functions (Fig.2a).

The second evaluation method computes the proximity $\mathbf{D}^c = |\mathbf{N}^c(.)\ ,\mathbf{T}^c(.)|$ between the network response $\mathbf{N}^c(.)$ and the local empirical distributions vector $\mathbf{T}^c(.)$ at each tree node. The final evaluation of the SRN training is the mean $\mathbf{D}$ of $\mathbf{D}^c$ for all tree nodes. Two measures are used to compute $\mathbf{D}^c$. The first one is $\mathbf{L_2}$ norm (1):

$$(1)\ |\mathbf{N}^C(.)\ ,\mathbf{T}^C(.)|_{L2} = [\mathbf{M}^{-1}\Sigma_{x=1..M}\,(\mathbf{N}^C(x)-\mathbf{T}^C(x))^2\,]^{1/2}$$

The second is a vector multiplication, normalised with respect to the vector's length (cosine) (2):

$$(2)\ |N^C(.), T^C(.)|_v = (|N^C(.)|\ |T^C(.)|)^{-1} \Sigma_{x=1..M} (N^C(x) T^C(x))$$

where M is the vector size, that is, the number of possible successors (e.g. 27) (see Fig. 2b).

4 Results.

Well-trained SRNs were examined with both the *optimal threshold method* and the *tree-based approaches*. A network with 30 hidden neurons predicted about 11% of the characters erroneously. The same network had mean L_2 distance 0.056 and mean vector-multiplication proximity 0.851. At the same time, the *optimal threshold method* rated the learning at 7% error. Not surprisingly, the tree-based evaluations methods gave higher error rate – they do not examine the SRN response to non-existent left contexts, which in turn are used in the *optimal threshold method.*

Discussion and conclusions.

Alternative evaluation methods for SRN learning are proposed. They examine the network response only to the training input data, which in turn is represented in a tree-based structure. In contrast, previous methods examined trained SRNs with test and random corpora. Both methods give a good idea about the learning attained. Methods used previously estimate the SRN recognition capabilities, while the methods presented here evaluate how close the network response is to the desired response – but for familiar input sequences. The desired response is considered to be the successors' empirical probability distribution. Hence, one of the methods proposed compares the local empirical probabilities to the network response. The other approach searches for a threshold that minimises the prediction error function. The proposed methods have been employed in the evaluation of phonotactics learning, but they can be used in various other tasks as well, wherever the data can be organised hierarchically. I hope, that the proposed analysis will contribute to our understanding of learning carried out in SRNs.

References.

Cleeremans, Axel (1993). *Mechanisms of Implicit Learning*.MIT Press.

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

Elman, J.L., et al. (1996). *Rethinking Innates.* A Bradford Book, The Mit Press.

Haykin, Simon. (1994). *Neural Networks*, Macmillan College Publisher.

Laver,John.(1994).*Principles of phonetics*,Cambr.Un.Pr.

Lawrence, S., et al.(1996).NL Gramatical Inference A Comparison of RNN and ML Methods. *Connectionist, statistical and symbolic approaches to learning for NLP*, Springer-Verlag,pp.33-47

Nerbonne, John, et al (1996). Phonetic Distance between Dutch Dialects. In G.Dureux, W.Daelle-mans & S.Gillis(eds) *Proc.of CLIN, pp.*185-202

Reilly, Ronan G.(1995).Sandy Ideas and Coloured Days: Some Computational Implications of Embodiment. *Art. Intellig. Review*,9: 305-322.,Kluver Ac. Publ.,NL.

Rodd, Jenifer. (1997). Recurrent Neural-Network Learning of Phonological Regula-rities in Turkish, *ACL'97 Workshop: Computational Natural language learning*, pp. 97-106 .

Stoianov, I.P., John Nerbonne and Huub Bouma (1997). Modelling the phonotactic structure of natural language words with Simple Recurrent Networks, *Proc. of 7-th CLIN'97* (in press)

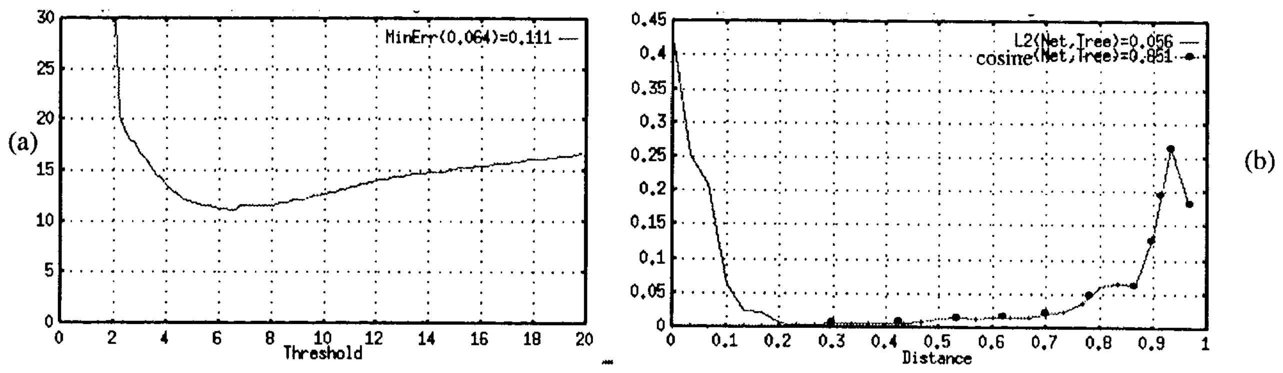

Fig.2. SRN evaluation by: (a.) minimising the error function **F(t)**. (b.) measuring the SRN matching to the empirical successor distributions. The distributions of L_2 distance and cosine are given (see the text).