Appears in International Conference on Neural Networks, ICNN 96, Washington DC, pp. 1853-1858, June 2-6, 1996.

Can Recurrent Neural Networks Learn Natural Language Grammars?

Steve Lawrence[*], C. Lee Giles, Sandiway Fong[†]
lawrence@elec.uq.edu.au, {giles,sandiway}@research.nj.nec.com

NEC Research Institute, 4 Independence Way, Princeton, NJ 08540

ABSTRACT

Recurrent neural networks are complex parametric dynamic systems that can exhibit a wide range of different behavior. We consider the task of grammatical inference with recurrent neural networks. Specifically, we consider the task of classifying natural language sentences as grammatical or ungrammatical - can a recurrent neural network be made to exhibit the same kind of discriminatory power which is provided by the Principles and Parameters linguistic framework, or Government and Binding theory? We attempt to train a network, without the bifurcation into learned vs. innate components assumed by Chomsky, to produce the same judgments as native speakers on sharply grammatical/ungrammatical data. We consider how a recurrent neural network could possess linguistic capability, and investigate the properties of Elman, Narendra & Parthasarathy (N&P) and Williams & Zipser (W&Z) recurrent networks, and Frasconi-Gori-Soda (FGS) locally recurrent networks in this setting. We show that both Elman and W&Z recurrent neural networks are able to learn an appropriate grammar.

1. Motivation

1.1. Representational Power of Recurrent Neural Networks

Natural language has traditionally been handled using symbolic computation and recursive processes. The most successful stochastic language models have been based on finite-state descriptions such as n-grams or hidden Markov models. However, finite-state models cannot represent hierarchical structures as found in natural language[1] [20]. In the past few years several recurrent neural network architectures have emerged which have been used for grammatical inference [4] [11]. Recurrent networks have been used for several smaller natural language problems, eg. papers using the Elman network for natural language tasks include: [7] [13]. Neural network models have been shown to be able to account for a variety of phenomena in phonology [10] [27], morphology [12] [24] and role assignment [18] [26]. Induction of simpler grammars has been addressed often - eg. [28] [11] on learning Tomita languages. Our task differs from these in that the grammar is considerably more complex.

In the Chomsky hierarchy of phrase structured grammars, the simplest grammar and its associated automata are regular grammars and finite state automata (FSA). However, it has been firmly established [1] that the syntactic structures of natural language cannot be parsimoniously described by regular languages. Certain phenomena (eg. center embedding) are more compactly described by context-free grammars which are recognized by push-down automata, while others (eg. crossed-serial dependencies and agreement) are better described by context-sensitive grammars which are recognized by linear bounded automata [22].

[*] http://www.neci.nj.nec.com/homepages/lawrence, http://www.elec.uq.edu.au/~lawrence

[†] Steve Lawrence is also with Electrical and Computer Engineering, University of Queensland, St. Lucia Qld 4072, Australia. Lee Giles is also with the Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742.

[1] The inside-outside re-estimation algorithm is an extension of hidden Markov models intended to be useful for learning hierarchical systems. The algorithm is currently only practical for relatively small grammars [20].

It has been shown that recurrent networks have the representational power required for hierarchical solutions [8], and that they are Turing equivalent [25]. The recurrent neural networks investigated in this paper constitute complex, dynamical systems. Crutchfield and Young [5] have studied the computational complexity of dynamical systems reaching the onset of chaos via period-doubling. They have shown that these systems are not regular, but are finitely described by indexed context-free grammars. Several modern computational linguistic grammatical theories fall in this class [15] [23].

1.2. Language and Its Acquisition

One of the most important questions for the study of human language is: How do people manage to learn such a complex rule system? A system so complex that linguists to date have been unable to construct a formal description [3]. Let us consider for a moment the kind of knowledge about language that native speakers often take for granted. Any native speaker knows that the adjective *happy* requires a complementizer *for* when it is followed by a infinitival sentential complement that contains an overt subject, but that the verb *believe* cannot take *for*. Furthermore, *happy* may be followed by a clause with a missing (or understood) subject, but *believe* cannot:[2]

*I am happy John to be here	I believe John to be here
I am happy for John to be here	*I believe for John to be here
I am happy to be here	*I believe to be here

Grammaticality judgments of this sort are sometimes subtle but, nevertheless, unarguably form part of the native speaker's language competence.

The language faculty has impressive discriminatory power, in the sense that a single word, as seen in the example above, can result in sharp differences in acceptability or interpretation. Given this and many other examples across various languages, some linguists (chiefly Chomsky [2]) have hypothesized that such knowledge is only partially acquired: the lack of variation across speakers, and indeed, languages for certain classes of data suggests that there exists a fixed or innate component of the language system modulated by minor language-specific parametric variations, e.g. surface word order. This framework is known as Government-and-Binding (GB) theory, or more recently as the Principles-and-Parameters framework. In this paper, we investigate whether a neural network can be made to exhibit the same kind of discriminatory power on the sort of data GB-linguists have examined. More precisely, our goal is to train a neural network from scratch, ie. without the bifurcation into learned vs. innate components assumed by Chomsky, to produce the same judgments as native speakers on the grammatical/ungrammatical pairs of the sort discussed above.

2. Data

Our data consists of 552 English positive and negative examples taken from an introductory GB-linguistics textbook by Lasnik and Uriagereka [16]. Most of these examples are organized into minimal pairs like the example *I am happy for John to be here/*I am happy John to be here* that we have seen earlier. We note here that the minimal nature of the changes involved suggests that our dataset may represent an especially difficult task. Due to the small sample size, the raw data was first converted (using an existing parser) into the major syntactic categories assumed under GB-theory.

The part-of-speech tagging represents the sole grammatical information supplied to the neural network about particular sentences in addition to the grammaticality status. A small but important refinement that was implemented was to include subcategorization information for the major predicates, namely nouns, verbs, adjectives and prepositions. For example, an intransitive verb such as *sleep* would be placed into a different class from the obligatorily transitive verb *hit*. Similarly, verbs that take sentential complements or double objects such as *seem*, *give* or *persuade* would be representative of other classes.[3] Flushing out the

[2] As is conventional, we use the asterisk to mark examples of ungrammaticality.

[3] Following classical GB theory, these classes are synthesized from the theta-grids of individual predicates via the Canonical Structural Realization (CSR) mechanism of Pesetsky [21].

subcategorization requirements along these lines for lexical items in the training set resulted in 9 classes for verbs, 4 for nouns and adjectives, and 2 for prepositions. As an example of the conversion, the sentence "I am eager for John to be here" is converted into the symbol sequence "n4 v2 a2 c n4 p1 v2 adv" where "n4" is noun sub-class 4, etc. We note here that tagging was done in a completely context-free manner. Obviously, a word, e.g. *to*, may be part of more than one part-of-speech.

3. Network Models

We tested the following models. We expect the FGS architecture to be unable to perform the task and include it primarily as a control case.

1. *Frasconi-Gori-Soda locally recurrent networks [9].* A multi-layer perception augmented with local feedback around each hidden node. We have used the local-output version where the output of a node, $y(t) = f(wy(t-1) + \sum_{i=0}^{n} w_i x_i)$.
2. *Narendra and Parthasarathy [19].* A recurrent network with feedback connections from each output node to all hidden nodes.
3. *Elman [8].* A recurrent network with feedback from each hidden node to all hidden nodes.
4. *Williams and Zipser [29].* A recurrent network where all nodes are connected to all other nodes.

For input to the neural networks, the data was encoded into a fixed length window made up of segments containing eight separate inputs, corresponding to the classifications noun, verb, adjective, etc. Sub-categories of the classes were linearly encoded into each input in a manner demonstrated by the specific values for the noun input: Not a noun = 0, noun class 1 = 0.5, noun class 2 = 0.667, noun class 3 = 0.833, noun class 4 = 1. The linear order was defined using our judgment of the similarity between the various sub-categories. Two outputs were used in the neural networks, corresponding to grammatical and ungrammatical classifications. A confidence criteria was used: $y_{max} \times (y_{max} - y_{min})$[4]. The data was input to the neural networks with a window which is passed over the sentence in temporal order from the start to the end. The size of the window was variable from one word to the length of the longest sentence.

4. Gradient Descent

We have used backpropagation-through-time [30] to train the globally-recurrent networks and the gradient descent algorithm described by the authors for the FGS network. The error surface of a multi-layer network is generally non-convex, non-quadratic, and often has large dimensionality. We found the standard gradient descent algorithms to be impractical for our problem. We investigated the techniques described below for improving convergence. Although these techniques are heuristic and reduce the elegance of the solution, they are motivated by analyzing the operation of the algorithm and are applicable to many other problems. Due to the dependence on the initial parameters, we have attempted to make the results as significant as possible by performing multiple simulations with different initial weights and training set/test set combinations. However, due to the complexity of the task, we could not perform as many simulations as desired. The standard deviation of the NMSE values is included to help assess the significance of the results. Table 1 shows some results for using and not using the techniques listed below. Except where noted, these results are for Elman networks using: two word inputs (i.e. a sliding window of the current and previous word), 10 hidden nodes, the quadratic cost function, the logistic activation function, sigmoid output activations, one hidden layer, the learning rate schedule shown below, an initial learning rate of 0.2, the weight initialization strategy discussed below, and one million stochastic updates.

1. Detection of Significant Error Increases. If the NMSE increases significantly during training then network weights are restored from a previous epoch and are perturbed to prevent updating to the same point. We have found this technique to increase robustness of the algorithm when using learning rates large enough to help avoid problems due to local minima and "flat spots" on the error surface, particularly in the case of the Williams & Zipser network.

[4] For an output range of 0 to 1 and softmax outputs.

Standard	NMSE	Std. Dev.	Variation	NMSE	Std. Dev.
Update: batch	0.931	0.0036	**Update: stochastic**	0.366	0.035
Learning rate: constant	0.742	0.154	**Learning rate: schedule**	0.394	0.035
Activation: logistic	0.387	0.023	Activation: tanh	0.405	0.14

Table: 1. Comparisons of using and not using various convergence techniques. All other parameters are constant in each case: Elman networks using: two word inputs (i.e. a sliding window of the current and previous word), 10 hidden nodes, the quadratic cost function, the logistic activation function, sigmoid output activations, one hidden layer, the learning rate schedule shown below, an initial learning rate of 0.2, the weight initialization strategy discussed below, and one million stochastic updates. Each NMSE result represents the average of four simulations. The standard deviation value given is the standard deviation of the four individual results.

2. Target outputs. Targets outputs were 0.1 and 0.9 using the logistic activation function and -0.8 and 0.8 using the *tanh* activation function. This helps avoid saturating the sigmoid function. If targets were set to the asymptotes of the sigmoid this would tend to: a) drive the weights to infinity, b) cause outlier data to produce very large gradients due to the large weights, and c) produce binary outputs even when incorrect - leading to decreased reliability of the confidence measure.

3. Stochastic updating. In stochastic update parameters are updated after each pattern presentation, whereas in batch update gradients are accumulated over the complete training set. Batch update attempts to follow the true gradient, whereas stochastic is similar to adding noise to the true gradient. In the results reported, the training times were equalized by reducing the number of updates for the batch case. Batch update often converges quicker using a higher learning rate than the optimal rate used for stochastic update[5], hence we investigated altering the learning rate for the batch case. We were unable to obtain significant convergence as shown in table 1.

4. Learning rate schedule. Relatively high learning rates are typically used in order to help avoid slow convergence and local minima. However, a constant learning rate results in significant parameter and performance fluctuation during the entire training cycle such that the performance of the network can alter significantly from the beginning to the end of the final epoch. Moody and Darkin have proposed "search then converge" learning rate schedules of the form [6]: $\eta(t) = \eta_0 / (1 + t/\tau)$ where $\eta(t)$ is the learning rate at time t, η_0 is the initial learning rate, and τ is a constant.

We have found the use of learning rate schedules to improve performance considerably as shown in table 1.

5. Sigmoid functions. Symmetric sigmoid functions (eg. tanh) often improve convergence over the standard logistic function. For our particular problem we found the difference was minor as shown in table 1.

5. Results

Our results are based on multiple training/test set partitions and multiple random seeds. We have also used a set of Japanese control data. Japanese is at the opposite end of the language spectrum when compared with English, and we expect a model trained on the English data to perform poorly on the Japanese data. Indeed, all models do perform poorly on the Japanese data.

Five simulations were performed for each architecture. Table 2 summarizes the results obtained with the various networks. In order to make the number of weights in each architecture approximately equal we have used only single word inputs for the W&Z model but two word inputs for the others. This reduction in dimensionality for the W&Z network improved performance. More details can be found in [17].

Our goal was to train a network using only a small temporal input window. Initially, we were unable to do this. With the addition of the techniques described earlier we were able to train Elman networks with sequences of the last two words as input to give 100% correct classification on the training data. Generalization on the test data resulted in 74% correct classification on average. This is better than the

[5] Stochastic update does not generally tolerate as high a learning rate as batch update due to the noise injected by the stochastic nature of the updates.

performance obtained using any of the other networks, however it is still quite low. The data is quite sparse and we expect increased generalization performance as the amount of data increases, as well as increased difficulty in training. Additionally, the dataset has been hand-designed by GB linguists to cover a range of grammatical structures and it is likely that the separation into the training and test sets creates a test set containing may grammatical structures that are not covered in the training set. The Williams & Zipser network also performed reasonably well with 71% correct classification of the test set.

TRAIN	Classification	Std. dev.	Confidence
Elman	**99.6%**	0.84	78.8%
FGS	67.1%	1.22	17.6%
N&P	75.2%	1.41	32.2%
W&Z	**91.7%**	2.26	63.2%

ENGLISH TEST	Classification	Std. dev.	Confidence
Elman	**74.2%**	3.82	75.4%
FGS	59.0%	1.52	18.8%
N&P	60.6%	0.97	26.9%
W&Z	**71.3%**	0.75	65.1%

Table: 2. Results of the network architecture comparison. The classification values reported are an average of five individual simulations, the standard deviation value is the standard deviation of the five individual results, and the confidence is the average confidence of the networks.

6. Conclusions

We investigated the use of FGS, N&P, Elman and W&Z networks for classifying natural language sentences as grammatical or ungrammatical, thereby exhibiting the same kind of discriminatory power provided by the Principles and Parameters linguistic framework, or Government-and-Binding theory. From best to worst performance, the architectures are: Elman, W&Z, N&P and FGS. Theoretically, the W&Z network is the most powerful in terms of representational ability, yet the Elman network provides better performance. Investigation shows that this is due to the more complex error surface of the W&Z architecture.

Are the networks learning the grammar? The hierarchy of architectures with increasing computational power give an insight into whether the increased power is used to model the more complex structures found in the grammar. The FGS network can be considered a control case - it is not capable of representing the kind of structures found in natural language. The fact that the more powerful Elman and W&Z networks do provide increased performance suggests that they are able to find structure in the data which cannot be modeled by the remaining networks. Another comparison of recurrent neural network architectures, that of Giles and Horne [14], compared various networks on randomly generated 6 and 64-state finite memory machines. The locally recurrent and N&P networks proved as good or superior to more powerful networks like the Elman network, indicating that either the task did not require the increased power, or the vanilla backpropagation-through-time learning algorithm was unable to exploit it.

We have shown that both Elman and W&Z recurrent neural networks are able to learn an appropriate grammar for discriminating between the sharply grammatical/ungrammatical pairs used by GB-linguists. However, generalization is limited by the amount of data available, and we expect increased difficulty in training the models as more data is used. We need to continue to address the convergence of the training algorithms, and believe that further improvement is possible by addressing the nature of parameter updating during gradient descent. However, a point must be reached after which improvement with gradient descent based algorithms requires consideration of the nature of the error surface. This is related to the input and output encodings, the ability of parameter updates to modify network behavior without destroying previously learned information, and the method by which the network implements structures such as hierarchical and recursive relations.

References

[1] N.A. Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, IT-2:113–124, 1956.

[2] N.A. Chomsky. *Lectures on Government and Binding*. Foris Publications, 1981.

[3] N.A. Chomsky. *Knowledge of Language: Its Nature, Origin, and Use*. Prager, 1986.

[4] A. Cleeremans, D. Servan-Schreiber, and J.L. McClelland. Finite state automata and simple recurrent networks. *Neural Computation*, 1(3):372–381, 1989.

[5] J. P. Crutchfield and K. Young. Computation at the onset of chaos. In W. Zurek, editor, *Complexity, Entropy and the Physics of Information*. Addison-Wesley, Reading, MA, 1989.

[6] C. Darken and J.E. Moody. Note on learning rate schedules for stochastic optimization. In R.P. Lippmann, J.E. Moody, and D. S. Touretzky, editors, *Advances in Neural Information Processing Systems*, volume 3, pages 832–838. Morgan Kaufmann, San Mateo, CA, 1991.

[7] J.L. Elman. Structured representations and connectionist models. In *6th Annual Proceedings of the Cognitive Science Society*, pages 17–25, 1984.

[8] J.L. Elman. Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, 7(2/3):195–226, 1991.

[9] P. Frasconi, M. Gori, and G. Soda. Local feedback multilayered networks. *Neural Computation*, 4(1):120–130, 1992.

[10] M. Gasser and C. Lee. Networks that learn phonology. Technical report, Computer Science Department, Indiana University, 1990.

[11] C. Lee Giles, C.B. Miller, D. Chen, H.H. Chen, G.Z. Sun, and Y.C. Lee. Learning and extracting finite state automata with second-order recurrent neural networks. *Neural Computation*, 4(3):393–405, 1992.

[12] M. Hare, D. Corina, and G.W. Cottrell. Connectionist perspective on prosodic structure. Technical Report CRL Newsletter Volume 3 Number 2, Centre for Research in Language, University of California, San Diego, 1989.

[13] Catherine L. Harris and J.L. Elman. Representing variable information with simple recurrent networks. In *6th Annual Proceedings of the Cognitive Science Society*, pages 635–642, 1984.

[14] B. G. Horne and C. Lee Giles. An experimental comparison of recurrent neural networks. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 697–704. MIT Press, 1995.

[15] A. K. Joshi. Tree adjoining grammars: how much context-sensitivity is required to provide reasonable structural descriptions? In L. Karttunen D. R. Dowty and A. M. Zwicky, editors, *Natural Language Parsing*. Cambridge University Press, Cambridge, 1985.

[16] H. Lasnik and J. Uriagereka. *A Course in GB Syntax: Lectures on Binding and Empty Categories*. MIT Press, Cambridge, MA, 1988.

[17] Steve Lawrence, C. Lee Giles, and Sandiway Fong. On the applicability of neural network and machine learning methodologies to natural language processing. Technical Report UMIACS-TR-95-64 and CS-TR-3479, Institute for Advanced Computer Studies, University of Maryland, College Park MD 20742, 1995.

[18] R. Miikkulainen and M. Dyer. Encoding input/output representations in connectionist cognitive systems. In D. S. Touretzky, G. E. Hinton, and T. J. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 188–195, Los Altos, CA, 1989. Morgan Kaufmann.

[19] K.S. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1(1):4–27, 1990.

[20] F. Pereira and Y. Schabes. Inside-outside re-estimation from partially bracketed corpora. In *Proceedings of the 30th annual meeting of the ACL*, pages 128–135, Newark, 1992.

[21] D. M. Pesetsky. *Paths and Categories*. PhD thesis, MIT, 1982.

[22] J.B. Pollack. The induction of dynamical recognizers. *Machine Learning*, 7:227–252, 1991.

[23] C. Pollard. *Generalised context-free grammars, head grammars and natural language*. PhD thesis, Department of Linguistics, Stanford University, Palo Alto, CA, 1984.

[24] D.E. Rumelhart and J.L. McClelland. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1, 2. MIT Press, Cambridge, 1986.

[25] H.T. Siegelmann and E.D. Sontag. On the computational power of neural nets. *Journal of Computer and System Sciences*, 50(1):132–150, 1995.

[26] M. F. St. John and J.L. McClelland. Learning and applying contextual constraints in sentence comprehension. *Artificial Intelligence*, 46:5–46, 1990.

[27] D. S. Touretzky. Towards a connectionist phonology: The "many maps" approach to sequence manipulation. In *Proceedings of the 11th Annual Conference of the Cognitive Science Society*, pages 188–195, 1989.

[28] R.L. Watrous and G.M. Kuhn. Induction of finite-state languages using second-order recurrent networks. *Neural Computation*, 4(3):406, 1992.

[29] R.J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989.

[30] R.J. Williams and D. Zipser. Gradient-based learning algorithms for recurrent connectionist networks. In Y. Chauvin and D.E. Rumelhart, editors, *Backpropagation: Theory, Architectures, and Applications*. Erlbaum, Hillsdale, NJ, 1990.