Exponential Generalizations from a Polynomial Number of Examples in a Combinatorial Domain

Steven Phillips† and Janet Wiles‡

Departments of Computer Science†‡ and Psychology‡
The University of Queensland, QLD 4072 Australia.

Abstract

Combinatorial domains are of interest because they allow a large repertoire of behaviours to be described from a few relatively simple rules. However, the problem that combinatorial domains pose for machines that learn from examples, such as Connectionist networks, is: how can target behaviour be adequately learnt without a corresponding explosion in training examples? We show, both theoretically (by using an existing corollary [9]) and empirically, that a feedforward network can generalize to an order of examples greater than that on which it was trained. Specifically, for the encoding of N-tuples, where the example space grows exponentially with N, only a polynomial number of training examples was required to achieve a fixed degree of accuracy over the entire domain.

Introduction

Acquiring appropriate behaviour over combinatorial domains (such as language) is seen as particularly difficult for machines that learn by examples, yet children acquire their first language with no formal instruction (effectively, by example). The difficulty such domains pose for Connectionist networks is in obtaining a *good* coverage of the space (i.e. systematic behaviour) from a *reasonable* (no more than polynomial) number of examples. It is this issue regarding the necessary acquisition of systematic behaviour that is seen as a weakness in the Connectionist approach to modeling cognition [5]. In this paper, we demonstrate a network that generalizes over an exponential space of examples from only a polynomial number of examples.

In section 1, the combinatorial learning domain (auto-association of N-tuples) is described and previous work on this problem is reviewed. Here, the problem is restated in terms of the probably approximately correct (PAC) learning framework [11]. In section 2, an existing result due to Shawe-Taylor and Anthony [9] is applied to this problem to show that in fact generalization to within a fixed degree of accuracy over this exponential space of patterns requires at most a polynomial number of training examples. This bound however is quite high and in section 3, we show empirically that far fewer training examples are needed. We also show that these examples can be learnt in reasonable time. A discussion of these results is given in section 4.

1 Auto-associating N-tuples

One of the simplest combinatorial domains is the auto-association of N-tuples, where say, an object represented as N instantiated variables, must be encoded and later recovered. The set of N-tuples S is defined (using Z-notation [10]) as: $S = S_1 \times \ldots \times S_N = \{x_1 : S_1; \ldots; x_N : S_N \bullet (x_1, \ldots, x_N)\}$, where S_i is the set of possible values at position i of the tuple. We will consider the case where $S_1 = \ldots = S_n$. The auto-association of N-tuples is simply the identity function that maps every element in S to itself.

Brousse and Smolensky [3] studied the auto-association of N-tuples (strings of length N) by a feedforward network. They showed that from a fixed number of training examples, the number of generalizations and virtual memories (patterns learnt within five additional learning trials) grew exponentially with the order of the tuple (see [3]; Figures 5 & 7, respectively). However, this growth did not match the growth in the total number of patterns. In fact, a plot of tuple order versus percentage of pattern space for generalizations, and generalizations plus virtual memories (reconstructed from [3], Figures 5 & 7), shows, in both cases, exponential decreases with respect to tuple order (Figure 1).

It is precisely this lack of generalization over such structured domains (i.e. lack of systematic behaviour) that has lead to the strong criticism of the

Connectionist approach to cognitive modeling [5]. If we regard systematic behaviour as almost equal capability over a particular domain, then we can restate the problem in terms of PAC learning by asking two questions. First, how many training examples are required to maintain a fixed degree of accuracy over the entire space? (In other words, what is the *sample complexity*?) Second, how long will it take to learn these examples? (In other words, what is the *time complexity*?) In the next section, we give a theoretical upper bound on the sample complexity for this problem.

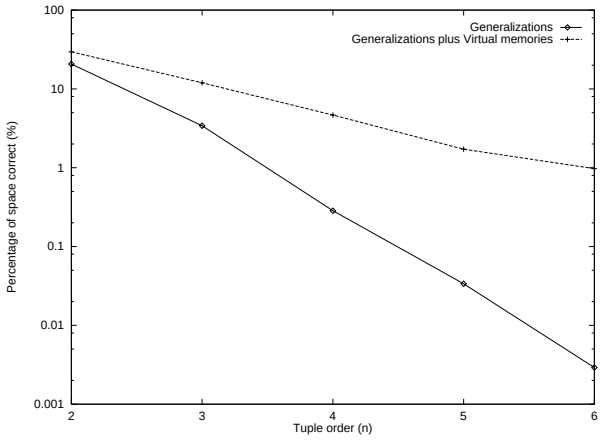

Figure 1: A log-linear plot of tuple order (N) versus the number of generalized and generalized plus virtual memories. The linear relationship indicates an exponential decrease in percentage of space correct. Reconstructed from Brousse & Smolensky, 1989; Figures 5 & 7.

2 Sample complexity

Before sample complexity can be calculated we must describe the network that can represent the target function. The auto-association of N-tuples can be represented by a sequence of N $k-l-k$ encoders in parallel, where all $N \times k$ input units are completely connected to all $N \times l$ hidden units that are completely connected to all $N \times k$ output units. The sample complexity result of Baum and Haussler [2] is not applicable in this case as multiple output units are involved. However, there is a corollary by Shawe-Taylor and Anthony [9] that, interestingly enough, provides an upper bound on the number of training examples independent of the number of output units. It applies to feedforward networks of threshold units with one hidden layer. Restated, it says:

> Given an accuracy parameter ϵ and a confidence parameter δ, for a feedforward network with W variable weights and n computational nodes, with probability greater than $1-\delta$ the network will give correct output with probability greater than $1-\epsilon$ on inputs drawn according to some distribution, provided it correctly computes the function on a sample (drawn from the same distribution) of size at least
>
> $$m_0 = m_0(\epsilon, \delta) = \frac{1}{\epsilon(1-\sqrt{\epsilon})} \left[ln\left(\frac{1.3}{\delta}\right) + 4(W+n) log_2(en) ln\left(\frac{6}{\epsilon}\right) \right]$$
>
> (Shawe-Taylor and Anthony, 1991: p116)

where W is the number of variable weights into all hidden units plus a single output unit, and n is the number of hidden units. In our network of N $k-l-k$ encoders, $n = N \times l$, and $W = klN^2 + 2lN + 1$.

If we fix the accuracy and confidence parameters, and the number of possible values at each tuple position (k), then we can determine an upper bound on sample complexity in terms of N, the tuple order. Given that the number of input and output units is $N \times k$ (i.e. one input and output unit for each instantiation for each variable), and the number of hidden units[1] is $N \times \lceil log_2 k \rceil$, then the upper bound on the maximum number of training examples required is

$$m_0 = O\left(\frac{N^2 k \, log_2 k}{\epsilon} log_2 \left(\frac{N \, log_2 k}{\epsilon} \right) \right) = O(N^2 log_2 N)$$

Thus, the network only requires a polynomial number of training examples (to within a log factor). Provided the network can *load* (acquire the correct behaviour on) these examples it will, with high probably, correctly generalize to an exponential number of future examples.

3 Empirical results

Given few assumptions regarding the function and probability distribution, the upper bound in the previous section is necessarily quite high. The crossover point where the size of the space of generalized examples begins to exceed the number of training examples occurs with training sets of size on the order of tens of thousands of examples. The result also assumes a network that can load the training examples in reasonable time.

[1] The minimum number of threshold units required by a single encoder of k items is $\lceil log_2 k \rceil$.

In this section we examine the sample complexity and time complexity of a feedforward network of sigmoidal units on this task. Although the result in the previous section applied to networks of threshold units, we hypothesize that sample complexity for a feedforward network of sigmoidal units on the auto-association of N-tuples is also polynomial.

Method For this study we examined the number of training examples required to obtain at least 95% accuracy ($\epsilon < 0.05$) with at least 99% confidence ($\delta < 0.01$) on future examples for tuples of order $N = 2$ to $N = 10$ (in increments of 2). The number of possible values at each tuple position was set at ten ($|S_i| = k = 10$).

Each network was trained using the standard back-propagation algorithm [8] on pattern sets ranging from 10 to 1500 patterns randomly generated from a uniform distribution. Training continued until all output units were on the right side of 0.5 for all training patterns. Testing was done on a further 1000 randomly generated examples from the same distribution.

For each value of N and each training set size, five trials were conducted. The number of test patterns for which the output was on the right side of 0.5 for all output units was recorded, from which the 99% confidence intervals could be calculated. For a particular N, training set size was increased until we were at least 99% confident that the network would be at least 95% correct on future examples. The training set size which met this criterion was recorded.

In this simulation we used sigmoidal units (rather than thresholds) as it reduces the number of necessary hidden units[2] to $2N$. We also used a block encoding for the representation of patterns at the output layer (where half of each group of k units are *on* consecutively, with wraparound, and the rest *off*) instead of the more common local encoding (one unit *on*, the rest *off*). This representation was chosen as it dramatically improves the learning time of the $k - 2 - k$ encoder problem [1], which has previously been shown to be particularly difficult for backpropagation-style networks [7]. It does not, however, change the logical nature of the task.

Result A log-log plot of tuple order versus training set size shows the training set grows much slower than the total space of patterns (Figure 2). The linear relationship indicates that growth in training set size was a low-order polynomial whereas the growth in total patterns is exponential.

A log-log plot of tuple order versus training time (total number of weight updates required to load all training patterns) shows time increasing as a polynomial of tuple order (Figure 3).

[2] Two hidden units being the most required for a single encoder of sigmoidal units [6].

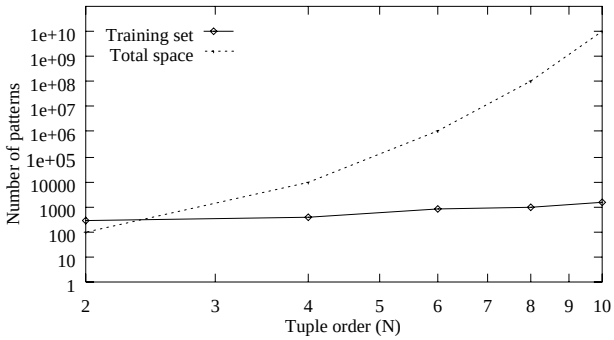

Figure 2: A log-log plot of tuple order (N) versus number of training patterns and total patterns. The linear relationship indicates sample complexity as a low-order polynomial.

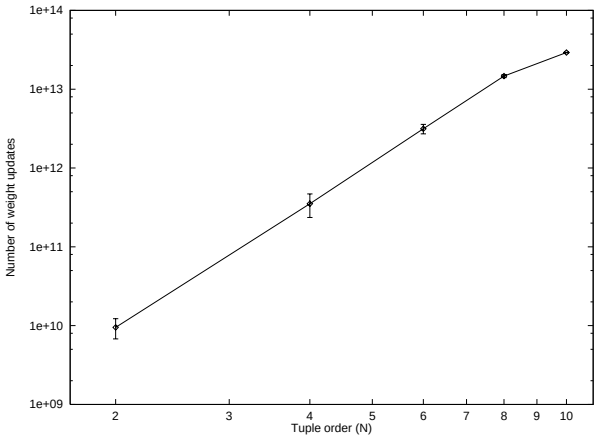

Figure 3: A log-log plot of tuple order (N) versus mean number of total weight updates. The linear relationship indicates time complexity as a polynomial of N. The error bars indicate one standard deviation.

4 Discussion

Both theoretical and empirical results showed the feedforward network generalizing to an exponential number of examples from a polynomial number of training examples. Provided the network can load the training examples, it will generalize (in most cases) to an exponential number of future examples. The empirical results also demonstrated that the network could learn the training examples in reasonable time.

A similar treatment can be applied to the situation where the tuple order is fixed (say, 2), and the number of instantiations (k) is varied. By applying the second

equation it can be seen that the sample complexity is no more than $O\left(k\left(log_2 k\right)^2\right)$. Empirically, we expect this to be very close to linear.

This result, apart from demonstrating a generalization property of feedforward networks, also has an implication for the Connectionist cognitive modeler. A major criticism of Connectionist models of cognitive behaviour is that they lack the *necessary* capacity for systematic behaviour. The problem that systematicity poses for Connectionists is

> ...not to show that systematic cognitive capacities are *possible* given the assumptions of a Connectionist architecture, but to explain how systematicity could be *necessary* – how it could be a *law* that cognitive capacities are systematic – given those assumptions.
> (Fodor and McLaughlin, 1990: p202; emphasis in the original)

The results reported here suggest systematicity not as a consequence of a pre-defined architectural bias, but as a consequence of the systematic environment over which the network optimizes its behaviour. The network will learn to be as systematic, or unsystematic, as the environment dictates. This then poses the question: is the systematic nature of cognition also a consequence of a systematic environment? If the answer is yes, then this paper shows how Connectionism can demonstrate the acquisition of systematic behaviour without the heavy reliance on innate architectural biases. However, if the answer is no, that is, people are systematic almost independent of their experience, then Connectionist models will require stronger architectural biases than previously used. What role the environment plays in the acquisition of systematic behaviour, we leave as an open question.

Acknowledgements

The authors would like to thank Simon Dennis, Peter Bartlett and Paul Smolensky for helpful discussions, and Paul Bakker for suggested improvements. This research was supported by a UQPRA to the first author and an ARC grant to the second author.

References

[1] P Bakker, S Phillips, and J Wiles. The N-2-N encoder: A matter of representation. In S Gielen and B Kappen, editors, *ICANN'93: Proceedings of the International Conference on Artificial Neural Networks*, pages 554–557, London, September 1993. Springer-Verlag.

[2] E B Baum and D Haussler. What size net gives valid generalization? *Neural Computation*, 1:151–160, January 1989.

[3] O J Brousse and P Smolensky. Virtual memories and massive generalization in connectionist combinatorial learning. In *Proceedings of the 11th Annual Conference of the Cognitive Science Society*, pages 380–387, Hillsdale, NJ, 1989. Lawrence Erlbaum.

[4] J A Fodor and B P McLaughlin. Connectionism and the problem of systematicity: Why Smolensky's solution doesn't work. *Cognition*, 35:183–204, 1990.

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

[6] L Kruglyak. How to solve the N bit encoder problem with just two hidden units. *Neural Computation*, 2:399–401, 1990.

[7] R Lister. Visualizing weight dynamics in the N-2-N encoder. In *Proceedings of the IEEE International Conference on Neural Networks*, Piscataway, NJ, 1993. IEEE Service Center.

[8] D E Rumelhart, G E Hinton, and R J Williams. *Learning Internal Representations by Error Propagation*, volume 1 of *Computational models of cognition and perception*, chapter 8, pages 319–362. MIT Press, Cambridge, MA, 1986.

[9] J Shawe-Taylor and M Anthony. Sample sizes for multiple-output threshold networks. *Network, Computation in Neural Systems*, 2(1):107–117, February 1991.

[10] J M Spivey. *The Z Notation: A reference manual.* Computer Science. Prentice Hall, New York, NY, 1989.

[11] L G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.