Generalized Compressed Network Search

Rupesh Kumar Srivastava
IDSIA
USI-SUPSI
Manno-Lugano, CH
rupesh@idsia.ch

Jürgen Schmidhuber
IDSIA
USI-SUPSI
Manno-Lugano, CH
juergen@idsia.ch

Faustino Gomez
IDSIA
USI-SUPSI
Manno-Lugano, CH
tino@idsia.ch

ABSTRACT

This paper presents initial results of Generalized Compressed Network Search (GCNS), a method for automatically identifying the important frequencies for neural networks encoded as a set of Fourier-type coefficients (i.e. "compressed" networks [2]). GCNS achieves better compression than our previous approach, and promises better generalization capabilities. Results for a high-dimensional Octopus arm control problem show that a high fitness 3680-weight network can be encoded using less than 10 coefficients, using the frequencies identified by GCNS.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning—*Connectionism and neural nets*

General Terms

Algorithms

Keywords

Neuroevolution, Complexity, Indirect encodings, Recurrent neural networks

1. INTRODUCTION

In previous work [2] we showed that encoding neural network weight matrices indirectly as a set of Fourier-type coefficients can reduce search space dimensionality and discover more 'regular' networks which are simpler in the Kolmogorov sense (the program required to encode them is much shorter). Such networks are expected to have better generalization capabilities [3].

However, up to now, this "compressed" network search has been restricted to band-limited networks where the genome includes all frequencies up to a specified limit frequency. This means that more genes must be searched than may be necessary because only a few, select frequencies may be needed to represent a good network. In this work, we implement a more general approach which automatically determines the subset of frequencies and their energy using a genetic algorithm with variable size chromosomes, where each gene specifies a frequency number as well as value. Taking inspiration from the messy genetic algorithms [1], cut and splice operators are used instead of crossover. By resolving the overspecification and underspecification problems arising from this less restrictive encoding, we are able to find genomes which represent high fitness networks using very few frequencies. Initial results are very encouraging: GCNS consistently identifies isolated frequencies which appear to contribute significantly to high fitness in the high-dimensional Octopus Arm task [4].

index	4	78	0	12	7	12	45	0	97	5
value	5.65	−2.32	6.52	−12.1	2.10	3.46	−5	−7.38	−3.98	1.87

Figure 1: GCNS coefficient genome.

2. DCT NETWORK REPRESENTATION

We consider fully connected recurrent neural networks (FRNNs) with i inputs and single layer of n neurons where some of the neurons are treated as output neurons. The FRNN consists of three weight matrices: an $n \times i$ input matrix, I, an $n \times n$ recurrent matrix, R, and a bias vector t of length n. These three matrices are combined into one $n \times (n + i + 1)$ matrix, and encoded indirectly as a set of Discrete Cosine Transform (DCT) coefficients. To construct a network, the coefficients in a genome are placed in their appropriate position in a coefficient matrix (the same size as the weight matrix), and then the inverse DCT is applied to generate the weight matrix.

In previous work [2], the approach taken was to search *all frequencies*, from the lowest, c_1, up to a limit frequency c_ℓ, specified by the user, where $\ell < N$, and N is the total number of weights in the network. In the present work, the coefficients are no longer restricted to a band-limited spectrum; any frequency can appear in the encoding.

3. GENERALIZED CNS

Generalized Compressed Network Search (GCNS) attempts to simultaneously find the number of coefficients required to represent a high fitness network, their indices, and their values. Variable size chromosomes are used where each gene has two elements: the frequency index and the value (see Figure 1). The coefficient index determines the position of the coefficient in the coefficient matrix.

The overspecification problem (some genes can have multiple copies in the genome) is handled as in messy genetic algorithms [1]. If a coefficient index appears multiple times in a genome, only its first value gets expressed in the phenotype. This results in an *intra-chromosomal dominance operator*. The problem of underspecification (some of the frequencies do not appear in a particular genome) elegantly resolves itself due to the nature of the encoding: if a partic-

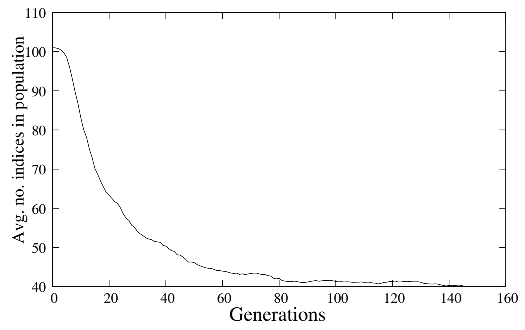

Figure 2: Number of unique coefficient indices (DCT frequencies) in the GCNS population (average of 30 runs).

ular coefficient number does not appear in the genome, it is muted in the phenotype i.e. its value is taken to be zero.

GCNS uses 'cut' and 'splice' operators as in messy genetic algorithms. A 'cut' divides the genome into two parts, each becoming a valid genome. The probability of a cut is $p_c*(l-1)$ where l is the length of the genome and p_c is a parameter. With probability p_s, the 'splice' operator joins two genomes into a single, valid chromosome. A mutation operator perturbs the coefficient indices and values with probability p_{m_i} and p_{m_v} respectively, by simply adding Gaussian noise.

GCNS starts with an initial parent population of size *popsize* with genomes of variable lengths containing frequency indices and values randomly chosen in a given range. At each generation, the cut and splice operators are applied in groups of 2 to randomly chosen members from the parent population (without replacement), and then mutation is applied to all the children. The best *popsize* members from the combined parent and child populations are chosen as the parents for the next generations. The algorithm terminates after the specified number of generations.

4. RESULTS: OCTOPUS ARM CONTROL

The octopus arm consists of p compartments floating in a 2D water environment [4]. Each compartment has a constant volume and contains three controllable muscles (dorsal, transverse and ventral). The state of a compartment is described by the x, y-coordinates of two of its corners plus their corresponding x and y velocities. Together with the arm base rotation, the arm has $8p + 2$ state variables and $3p + 2$ control variables. The goal of the task is to reach a goal position with the tip of the arm, starting from different initial positions, by contracting the appropriate muscles at each 1s step of simulated time. The standard setup uses 3 initial positions; here, only one initial position is used for training (the arm starts hanging straight down), since it turns out that successful networks tend to generalize to the other two initial positions. The fitness function is given by $(1 - (t*d)/(T*D))$ where t is the number of time steps taken to reach the goal, d is final arm tip distance to the goal, T is the maximum number of time steps in a trial, and D is the initial arm tip distance to the goal.

GCNS was run 30 times with $popsize = 100$, $ngen = 150$, $p_c = 0.2$, $p_s = 0.8$, $p_{m_i} = 0.1$ and $p_{m_v} = 0.8$. The initial coefficient indices were chosen at random from $[1, 100]$, and their values from $[-30, 30]$. The mean best fitness over 30 runs was 0.95 while the average number of expressed genes (i.e. non-dominated) in the best genomes was 9.8, one-third the number required in [2] to achieve similar fitness.

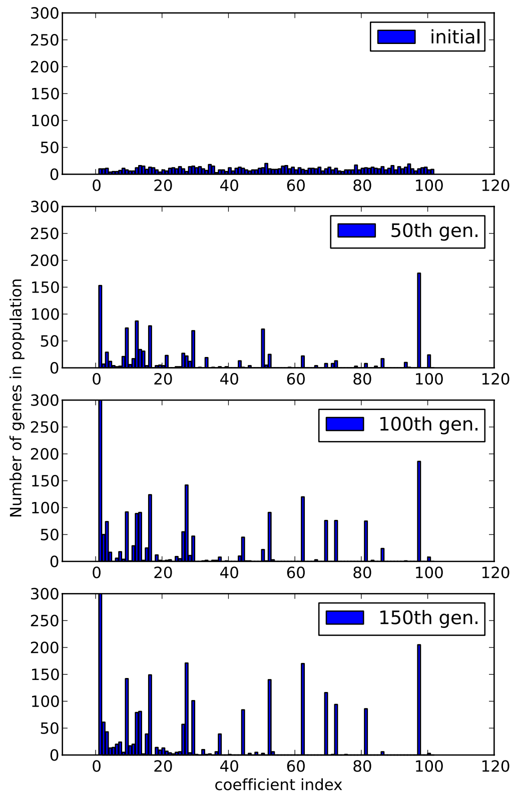

Figure 3: Better coefficient indices required to represent the network are identified by GCNS as the search progresses.

Figure 2 shows how the frequency content in the population declines over the course of evolution as the search converges to just a few frequencies (see Figure 3 for the behavior of a typical run). Interestingly, we found that in addition to the fundamental frequency (index 1, which we expected), almost all of the most fit networks contained either index 84 or 97, with large values. The 2D cosine functions represented by these indices seem to capture a basic regularity inherent in the task, given the network architecture used.

5. ACKNOWLEDGMENTS

This research was funded by the following EU projects: IM-CLeVeR (FP7-ICT-IP-231722) and WAY (FP7-ICT-288551).

6. REFERENCES

[1] D. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis, and first results. *Complex systems*, 3(5):493–530, 1989.

[2] J. Koutník, F. Gomez, and J. Schmidhuber. Evolving neural networks in compressed weight space. In *Proc. of the 12th annual conference on Genetic and evolutionary computation*, pages 619–626. ACM, 2010.

[3] J. Schmidhuber. Discovering neural nets with low kolmogorov complexity and high generalization capability. *Neural Networks*, 10(5):857–873, 1997.

[4] Y. Yekutieli, R. Sagiv-Zohar, R. Aharonov, Y. Engel, B. Hochner, and T. Flash. Dynamic model of the octopus arm. I. Biomechanics of the octopus reaching movement. *Journal of neurophysiology*, 94(2):1443–1458, 2005.