An Estimation of Distribution-like Algorithm based on a Denoising Autoencoder

Alex, Sid, and Chrisantha

School of Electronic Engineering and Computer Science
Queen Mary, University of London
{a.churchill,ssgd,c.t.fernando}@qmul.ac.uk

Abstract. In this paper we present a novel neural-based optimisation algorithm. The algorithm follows the traditional generate-update methodology of Estimation of Distribution algorithms, using a denoising autoencoder to learn the structure of promising solutions within its hidden layer, with the output neurons defining a probability distribution that is sampled from to produce new solutions. The algorithm is shown to outperform a canonical Genetic Algorithm on several combinatorial problems, including the multi dimensional 0/1 knapsack problem, MAXSAT and the Hierarchical If and Only If. Analysis shows that the neural network is able to learn interesting structural features of the search space, while the sampling method employed supports continued exploration, enabling optimal solutions to be found on NP-hard problems.

1 Introduction

Estimation of Distribution Algorithms (EDAs) are a growing field in Evolutionary Computation, which attempt to build a statistical model of sections of a search space in order to uncover underlying structure and guide search in an efficient manner [1]. At the heart of an EDA lies a model-building algorithm. Examples include Bayesian Networks [2], Markov Networks [**?**] and K-Means clustering [**?**]. In this paper we introduce a novel neural-based method for modelling, a denoising autoencoder. An auto encoder is a feed forward neural network, consisting of at least one hidden layer, which is trained to reproduce its inputs from its outputs. Over the course of training the hidden layer learns a representation of the data, which has be used for reconstructing missing data [**?**] and dimensionality reduction [**?**]. The algorithm introduced in this paper trains a single autoencoder with promising solutions from a population, from which structure of the search space is learnt by the representation in the hidden layer. This differs from traditional EDAs such as PBIL [**?**] or ECGA [**?**] as an explicit statistical model is not produced. However, the learnt structure can be leveraged to produce new solutions by inputting an existing or randomly generated solution into the network and sampling from the output neurons using a binomial distribution. Results presented in Section 6 show that the autoencoder method is able to outperform a canonical Genetic Algorithm on a range of combinatorial and hierarchical problems.

2 Background

EDAs (also known as Probabilistic-Model-Building Genetic Algorithms) are population based optimisers, that typically replace genetic operators with a statistical model. The rationale behind the model building and related linkage learning approach is that dependencies between variables can be captured and preserved, which can be easily lost in standard Evolutionary Search, and new solutions generated around promising structures. Early work on EDAs concentrated on methods that explicitly modelled the probabilities of independent alleles occurring in a population of genotypes. These include the compact Genetic Algorithm [**?**], PBIL [**?**] and Univariate Marginal Probability methods[**?**]. Improved success was found by modelling multivariate dependencies using clustering algorithms (ECGA) [**?**], Bayesian Networks [2], Markov Networks [**?**] and tree structures [**?**], among others. The use of the autoencoder model in this paper is motivated by its potential to learn high-dimensional dependencies in data, while still maintaining a low computational cost in terms of training time. Recently there has been interest in neural-based methods in an EDA context for multi-objective optimisation. A Growing Neural Gas (GNG) was used as a model in [**?**], employing a competitive Hebbian learning rule to cluster data without having to pre specify the number of groups. A shallow Restricted Boltzmann Machine (RBM) was used to model high dimensional data in [**?**]. An autoencoder is another neural-based method for unsupervised learning of features that has potential to model solution structure and has hitherto not been applied to combinatorial or continuous optimisation.

A second motivation for this approach is to investigate methods in which Evolutionary Algorithms can be implemented in neural structures. The *Neural Replicator Hypothesis* [**?**] proposes that evolutionary processes could operate in the brain at ontogenetic timescales. In [3] a network of spiking neurons combined with Hebbian learning, which enabled linkages to be found between features, was applied to combinatorial problems and solved the 128-bit HIFF problem.

3 Methods

3.1 Denoising Autoencoder

Sid's stuff here.

3.2 Optimisation Algorithm

The pipeline of DAGA is similar to other EDAs and incorporates techniques from HBOA [2]. A population of solutions is maintained, P_t, updated at each iteration, t. An initial population of solutions, P_0, is drawn from a uniform distribution. These solutions are evaluated and the fittest unique x% are selected (i.e. truncation selection) to be in the training set, $\hat{P}_t$. The Denoising Autoencoder, D_t, is then trained with $\hat{P}_t$ for e epochs. Following training, a new set of solutions, S_t, is selected from P using tournament selection (with replacement). Each

member of S_t is inputted to D_t, and the output vector, y, is sampled from using a binomial distribution, to produce a new solution. This solution is included in P_{t+1} if it is better than its closest neighbour according to Restricted Tournament Selection (RTR) [2]. Pseudocode for DAGA is presented in Algorithm 1.

4 Experiments

Experiments intro blurb.

4.1 Multi-dimensional Knapsack Problem

Here we wish to choose of subset of N items to maximise the total value of items, z, while satisfying m constraints. We wish to maximise:

$z = \sum_{j=1}^{N} v_j x_j$, subject to $\sum_{j=1}^{N} w_{ij} x_i \leq c_i, i = 1, ..., m$

$x_i \in \{0, 1\}, j = 1, ..., N$.

In the results below, two problems are tackled. The first is the Weing8 instance [**?**], which has 105 items and two constraints (optimal solution is 602,319), and the second is a randomly generated instance with 500 items and one constraint (optimal solution is 104,000). Both instances are available in the supplementary material. If any constraint is violated, the total value of chosen items is multiplied by -1.

4.2 Hierarchical If and only If

4.3 Royal Road

The Royal Road function as defined by Mitchell et al. [**?**], divides a bit string into a number of equally sized, non-overlapping partitions. If all of the bits in the partition are correct (e.g. all 1s), then the partition's fitness contribution is added to the overall fitness. The existence of these "Building blocks" was meant to act as a "royal road" for GAs compared to hill-climbers but bad alleles hitch-hiking to good partial solutions slows down convergence speed. The problem is defined as,

$$s(i) = sumofbits...$$

4.4 MAXSAT

5 Results

DAGA is compared to a canonical generational Genetic Algorithm (GA) on the problems described in Section 4. The GA employs tournament selection to select parents, two point crossover for recombination (with probability p_c) and a probability, p_m, of a bit flip mutation at each allele. For each experiment a

large parameter sweep was performed on both DAGA and the GA and the best configurations were chosen for comparison. Figure **??** presents graphs showing the mean fitness (averaged over ten trials) of the best solution in the population at each iteration for the two algorithms on the 5 problems. Details of the best solutions found are given in table **??**.

6 Discussion

7 Conclusion

Acknowledgments. The work is funded by the FQEB Templeton grant "Bayes and Darwin", and the FP-7 FET OPEN Grant INSIGHT.

References

1. Pelikan, M., Sastry, K., Cantú-Paz, E.: Scalable optimization via probabilistic modeling. Springer (2006)
2. Pelikan, M.: Hierarchical Bayesian optimization algorithm. Springer (2005)
3. Fernando, C., Goldstein, R., Szathmáry, E.: The neuronal replicator hypothesis. Neural Computation **22**(11) (2010) 2809–2857