Multi-Population Genetic Algorithm for the Optimization of Computer Image Recognition Training

Holden Jones
James A. Bilitski, Ph.D.
Department of Computer Science, University of Pittsburgh at Johnstown

1. Abstract

Machine learning algorithms face the challenge of being accurate as well as being computationally intensive. This research addresses these issues by applying optimizations for training an artificial neural network in an image recognition system by using a multi-population genetic algorithm. This study used Google's TensorFlow -- an open source software library that provides a framework for neural networks. Experimentation used a multiple-population genetic algorithm to optimize the parameters for the training of a basic image recognition network. Traditional genetic algorithms utilize a single population of individuals.[1] The use of a multi-population genetic algorithm allows for the speed and accuracy of the network, quantified as a fitness value, to be optimized over potentially tens of thousands of generations of separate populations of individuals. These multiple populations allow for separate evolutionary divergence—much like how geographic dividers caused divergent populations in the evolution of actual life. The use of a small amount of cross-population immigration facilitates the sharing of beneficial genetic data across the entire set. The results show that parameters of the neural network were noticeably optimized.

2. Introduction

In artificial intelligence, the creation of new neural networks is often a problem of guesswork. Without extensive manual testing, it is incredibly difficult to figure out the exact number of layers, number of neurons, and other parameters to use in the setup of the average system. These small tweaks and changes can take a very long time to optimize properly without automation.

As the importance of neural networks in our everyday lives grows, from Spotify's song-recommending artificial brain[2] to Facebook's ability to accurately recognize faces in pictures[3], the efficiency of creating these networks becomes an increasingly important task to automate. The more automatic the creation of a neural network can be, the easier it is for more developers to be able to use them in a variety of important applications.

3. Methodology

In this study, Google TensorFlow GPU-accelerated optical character recognition neural network was created, using the MNIST Handwritten Digit Database [4] as input data to train on. Using this collection of over 55,000 images of single numeric digits, a four-layer neural network was created, with the first three layers having a non-set number of neurons, and the final layer having ten output neurons (For the digits 0 through 9, as shown in Figure 1).

Figure 1: A collection of handwritten digits from the MNIST database. [5]

To optimize the number of neurons for these flexible layers, a Multi-Population Evolutionary Algorithm was created, with the three numbers of neurons used as the conditional parameters, and the resulting time spent training, alongside the final accuracy percentage, being used as a combined fitness value, the measure of the success of the optimization, through the formula:

Fitness = Training Time * (1—Accuracy)

The algorithm was split into three separate populations, labelled as Blue, Red, and Yellow, each comprised of six genetic individuals. Each of these populations was evolved over 150 generations, with a 1% chance of genetic mutation per individual per generation and a 50% chance of a single-value swap genetic crossover between fitness-weighted parent choices.

To best benefit from the multi-population evolutionary algorithm, a mandatory immigration operation was performed every 20 generations, causing the top member in each population to be rotated to the next in the cycle.

A traditional single-population evolutionary algorithm with 18 members was run as a baseline from which to compare the multi-population network.

4. Results & Conclusions

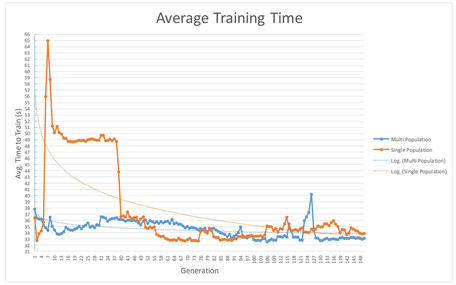

Figure 2 : Average Training Time per Generation

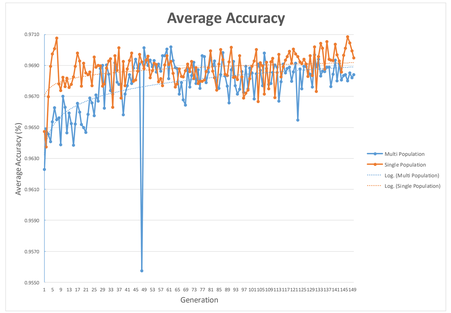

Figure 3 : Average Accuracy per Generation

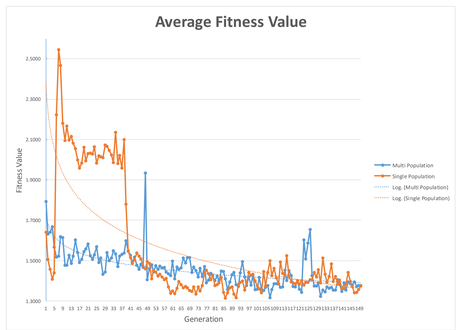

Figure 4: Average Fitness Value per Generation

During training, both algorithms took slightly over 24 hours to complete their 150 generations, and afterwards resulted in 8100 data points from which to observe the optimization trends. Both algorithms achieved a general trend towards lower training time and higher accuracy, resulting in a beneficial optimization of the randomly-initialized neural network without outside input.

From start to finish, the Multi-Population algorithm saw a nine-second decrease in maximum recorded training time, and, as shown in Figure 2, a four-second decrease in average training time. While the results for training accuracy were not quite as pronounced, with a 0.54% overall increase in accuracy over the evolutionary period for the Multi-Population algorithm shown in Figure 3, an overall trend towards increased accuracy can be seen in the graphed data.

Combined, the Average Fitness Value graph shown in Figure 4 portrays one of the benefits of the multi-population algorithm over the single-population algorithm: as three separate populations are generated and evolve separately, negative initial genetic data is able to be improved far more easily with the immigration operator every 20 generations. If a single population fixates on negative genetic data, immigrants from other populations end up allowing beneficial genes to be re-introduced into the genetic pool. While the Single-Population algorithm fixates on long training times for nearly 40 generations, the Multi-Population algorithm breeds its defects out more quickly, resulting in a beneficially low fitness value.

The algorithm was able to produce a statistically relevant result, optimizing the parameters of the basic image recognition network to a recognizable extent without any human interaction. Further improvements to the methodology may improve performance. One such improvement would be to independently process the populations on individual processors where the data is shared every 20 generations.

Citations

[1] Holland, J. H. (1975). Adaptation in natural and artificial systems. Ann Arbor, MI: University of Michigan Press.

[2] Dieleman, Sander. "Recommending Music on Spotify with Deep Learning." *Sander Dieleman*. Sander Dieleman, 5 Aug. 2014. Web.

[3] Taigman, Yaniv, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. "DeepFace: Closing the Gap to Human-Level Performance in Face Verification." *Facebook Research*. Facebook, 24 June 2014. Web.

[4] LeCun, Yann, Corinna Cortes, and Christopher J.C. Burges. "The MNIST Database." *MNIST Handwritten Digit Database, Yann LeCun, Corinna Cortes and Chris Burges*. New York University, Google Labs, n.d. Web. 06 Apr. 2017.

[5] TensorLayer Contributors. "MNIST Digits" *Tutorial—TensorLayer 1.4.2 Documentation,* TensorLayer. 2016.