CS 4641 Assignment 2: Randomized Search

Student: Gen Mark Tanno
Instructor: Dr. Byron Boots

***Abstract* - The project aims to explore random search algorithms and their applications in different problem domains. The first part of the project was to implement three local random search algorithms: randomized hill climbing, simulated annealing, and a genetic algorithm. These algorithms were then used instead of backpropagation for a neural network to find good weights. The dataset that was used for this part was the pulsar star dataset that was taken from Kaggle. The dataset contains 8 attributes including mean, standard deviation, and skewness of signal-to-noise ratio readings. The dataset also contains a classification of whether the star is pulsar or not. The second part of the project was to apply the three search techniques to two optimization problems in which the continuous peaks problem and the travelling salesman problem (TSP) were chosen.**

RANDOMIZED HILL CLIMBING

Randomized hill climbing is an algorithm built on top of the hill climbing algorithm that is an iterative algorithm that attempts to find a better solution by making incremental changes. A MATLAB implementation was used to gather data which were then plotted in EXCEL. These were also used for the simulated annealing and the genetic algorithm Different numbers of iterations were used to see the effect of increasing the number of iterations on the training and testing errors. Like the Supervised Learning assignment, the data set were split into 70% training set and 30% testing set.

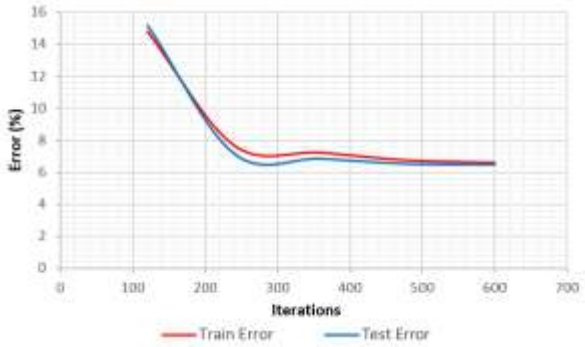

Figure 1: Training and testing errors based on the number of iterations

Figure 1 shows that the training and the testing sets behaved similarly. The error decreases as the number of iterations increases. Moreover, the datasets start to converge, and the error does not significantly change in larger numbers of iterations. The running time at 600 iterations was 37.81 seconds and it was found that the lowest training error of 6.59% and lowest testing error of 6.5% was at 600 iterations.

Based on the neural networks' performance on the pulsar stars dataset in the previous homework, testing and training errors of 3.5% were achieved. This is an evidence of a trade off between backpropagation that overfits data and randomized hill climbing that requires a lot of iterations and running time to achieve good accuracy.

SIMULATED ANNEALING

Simulated annealing is a method that models the physical process of heating a material and then slowly lowering the temperature to decrease defects. The algorithm can approximate the global optimum of a function and is used for unconstrained and bound-constrained optimization problems. Like the previous one, the training and testing errors were obtained, but this in different max function evaluations.

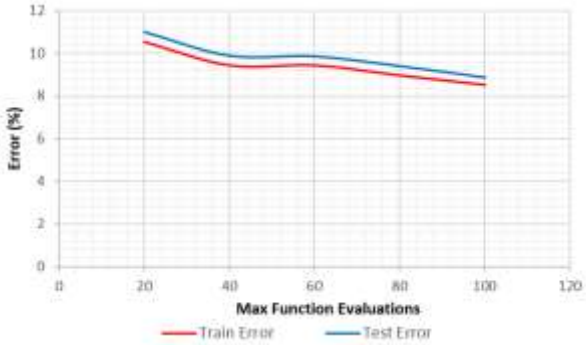

Figure 2: Training and testing errors based on the max function evaluations

Figure 2 shows that the training and testing error values are close. But unlike the graph for randomized hill climbing, the testing error is a bit higher than training error which shows less bias in this case. Further investigation was made by modifying some parameters. First, the behaviors as a result of having initial temperatures of 5, 10, and 15 were observed.

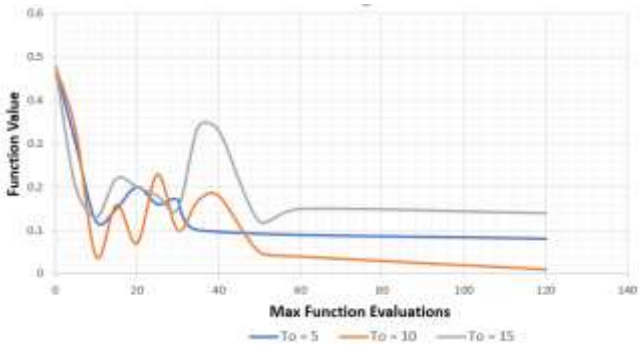

Figure 3: Max function evaluations versus function value

Figure 2 shows that the data converge to different values given different initial temperatures. Furthermore, An initial temperature value of 10 seems to be the optimal among the three as the function value reached is minimum which is

desired. Next, the effect of annealing schedule on the optimal fitness values and computing time was tested.

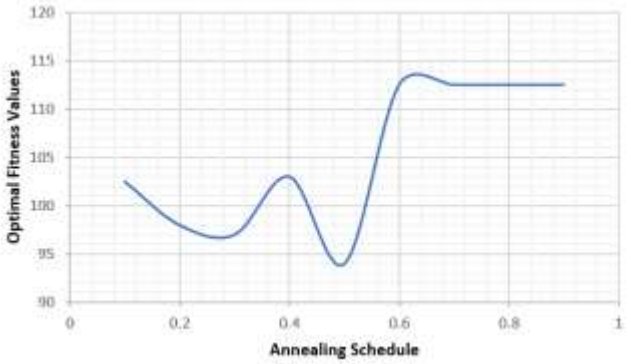

Figure 4: Optimal fitness values for varying annealing schedule for simulated annealing

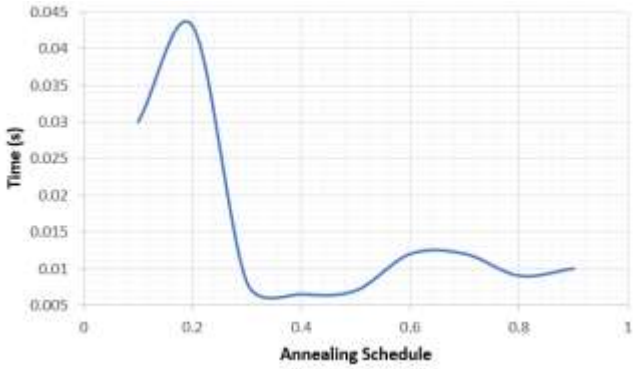

Figure 5: Computing time for varying annealing schedule for simulated annealing

From Figure 4, the optimal fitness value peaks when annealing schedule is between 0.6 and 0.7. This is a very plausible result as the higher the annealing schedule is, the search space is explored completely. Optimal fitness values were shown low in low annealing schedules because solutions have tendencies to get stuck in local optima. As for the running time, it took the longest when annealing schedule was around 0,2 and relatively faster in higher annealing schedules.

GENETIC ALGORITHM

Genetic algorithm is an algorithm based on natural selection. It modifies a population of individual solutions where each step, the algorithm selects individuals at random from the current population to be parents and uses them to produce the children of the next generation. Similar to the previous algorithms, the training and testing errors were deduced, with a slight change with generations as the variable in the x axis.

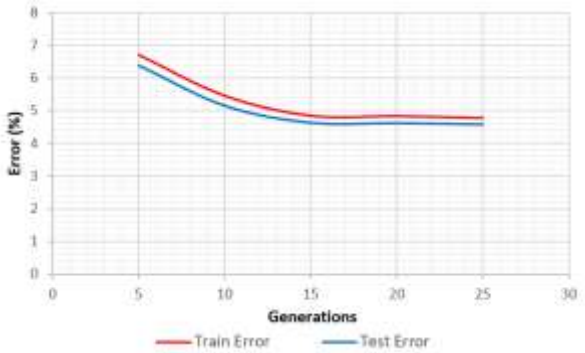

Figure 6: Training and testing errors based on the number of generations

Figure 6 shows that like the previous algorithms, the testing and training errors have almost equivalent values, and that the error is low. Unlike simulated annealing however, the training error is higher than the testing error. Thus, there is higher bias using this algorithm compared to simulated annealing. Nonetheless, genetic algorithm provides promising results when used with a neural network.

There are three hyperparameters that can be tuned for the genetic algorithm: population size, amount to mate per iteration, and amount to mutate per generation. For this assignment, I chose to vary population size with having the amount to mate and amount to mutate be half the population size, i.e. amount to mate is equal to the amount to mutate.

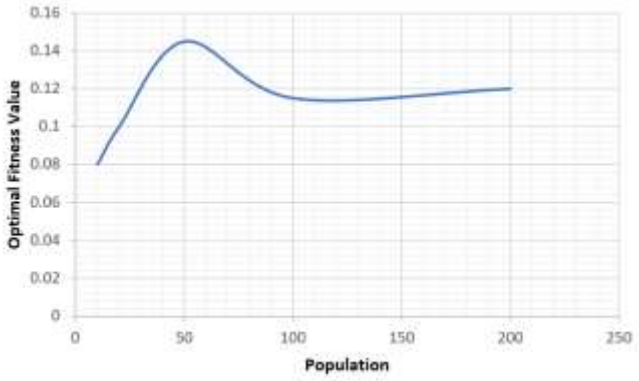

Figure 7: Optimal fitness values for varying population size with equal amounts of mating and mutating

Figure 7 shows that the max fitness value is obtained when population size is about 50, but the fitness value does not significantly drop after a population size of 50. This supports Gotshall et al.'s finding that the greater the population size the greater the chance that the initial state of the population will contain a chromosome representing the optimal solution.

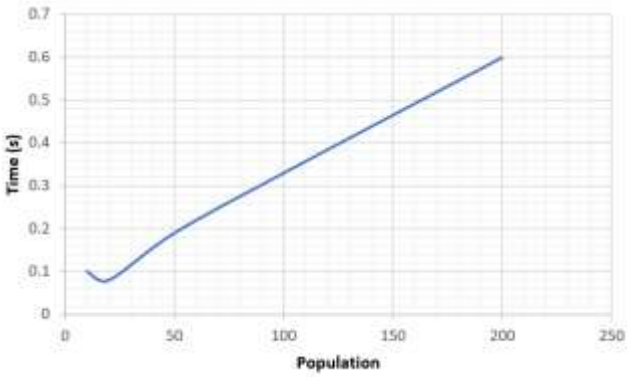

Figure 8: Computing time for varying population size for a genetic algorithm

Figure 8 shows the computing time increases as the population is increased. Moreover, the relationship between the population size and the time is close to linear.

For this case, we might want a population size of 50 for equal mutation and mating because it produces maximum fitness value and low computing time. However, it does not tell us how to find the optimal population size in other cases. The optimal population size may be dependent on the dimensionality as stated by Gotshall et al.

TRAVELING SALESMAN PROBLEM

The aim of the Traveling Salesman Problem is to find the shortest possible route that visits every city exactly once and returns to the starting point given a set of cities and distance between every pair of cities.

The problem ran in ABAGAIL with the three algorithms and data were recorded as shown below.

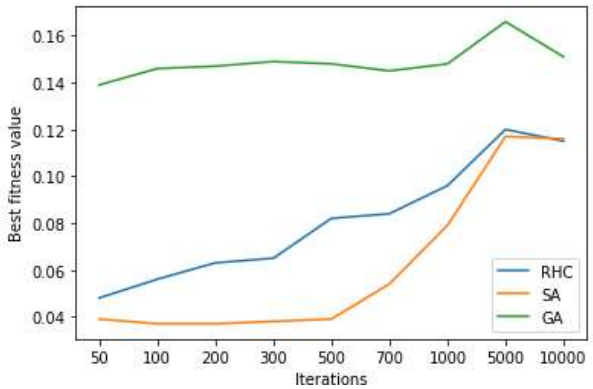

Figure 9: Fitness value for varying numbers of iterations

Figure 9 shows iterations of up to 10000, and the genetic algorithm performed the best all at any given iteration. This is because the genetic algorithm performs well on problems with frequently changing heuristics. The Travelling Salesman Problem is also an exploration problem which the genetic algorithm is well capable to performing.

CONTINUOUS PEAKS PROBLEM

The Continuous Peaks Problem aims to find the global optimum based on local optima. The algorithms ran in ABAGAIL again the results are show below.

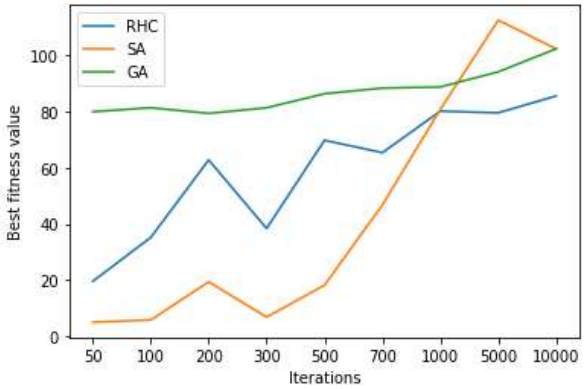

Figure 10: Fitness value for varying numbers of iterations

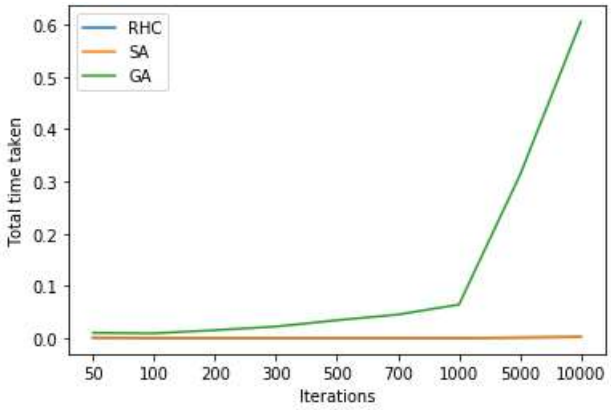

Figure 11: Computing time for varying numbers of iterations

Figure 10 shows that simulated annealing and genetic algorithm performed best at different iterations. However, as seen in Figure 11, simulated annealing did better overall because simulated annealing does not take too much computing time.

REFERENCES

https://www.geeksforgeeks.org/travelling-salesman-problem-set-1/

https://www.mathworks.com/help/gads/what-is-simulated-annealing.html

https://www.mathworks.com/help/gads/what-is-the-genetic-algorithm.html

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.105.2431&rep=rep1&type=pdf