**Deep Artificial Neural Network Optimization - Project Report**

Course: Computational Intelligence for Optimization

Authors: Joris Bertens, Antonio Curado, Lukas Früchtnicht, Alexandra Baur

1. **Introduction**
   1. **Problem description**

The problem to solve in this project is a type of continuous optimization problem, where the fitness evaluation of the possible solutions are evaluated before their validation. This validation measures whether the accuracy of the absolute difference between training and validation fitness doesn´t surpass a defined threshold.

The purpose for the artificial neural network is to recognize handwritten digits.

* 1. **Description of provided artificial neural network and project restrictions**

The handwritten digits to be recognized by the artificial neural network are represented by 8x8 raster images, where the color (gray scale) is defined by an integer value [0, 16]. To provide this information to the network, these images are flattened and provided to the 64 input neurons of the network. Therefore the possible solutions will be represented by an array of 64.000 floating points.

Furthermore, the network has two hidden layers that consist of 10 neurons each and another 10 output neurons (total number of 64.000 connections).

The restrictions on the optimization of the neural network are to not change the architecture of the network nor the training data, creating more than 5000 offsprings per run (50 per generation), to not change the proportions of the test and validation data and not to change the validation threshold nor the configuration of the initial weights.

* 1. **Objective of the project**

The objective is the optimization of the weights of the 64.000 connection in the given artificial neural network using optimization algorithms discussed during the semester. In the end, the artificial neural network should be able to recognize unseen handwritten digits.

This goal should be achieved by benchmarking different algorithms and parameters (especially variations of the Genetic Algorithm).

1. **Exploration of Parameters of the given Algorithms**

The first step in order to optimize the network, is to explore and test different parameters on the existing algorithms.

* 1. **Hill Climbing**
  2. **Simulated Annealing**
  3. **Genetic Algorithm**

1. **Implemented Methods**

To be able to improve the Genetic Algorithms, the following different methods for all the steps of the algorithm were implemented. The methods used were discussed during the lectures of this semester or found in the indicated papers during research.

* 1. **Selection**
     1. **Tournament selection**

The tournament selection was already given for the Genetic algorithm. It chooses randomly individuals from the population and fills a tournament pool of defined size (pressure). Of this pool it chooses the individual with the highest fitness.

* + 1. **Boltzmann Selection**

The Boltzmann selection is similar to the tournament selection, but increases the selection pressure over the generations. With this procedure there is a bigger variety at the beginning.

* + 1. **Roulette-Wheel-selection**

The roulette wheel selection is a random selection where the probability for the individual of being chosen is represented by its fitness proportional to the overall sum of fitnesses of all individuals.

* + 1. **Rank selection**

The rank selection works like the roulette wheel selection, but the probability of being chosen is represented by the rank of the individual. The distance of the fitnesses of the individuals are not taken into consideration.

* + 1. **Random selection**

With random selection the individuals chosen to be parents are randomly selected from the previous population.

* + 1. **Best selection**

The best selection picks the best individuals from the previous population to be parents.

* 1. **Crossover**
     1. **One-point-crossover**

The one point crossover cuts the parents at a random position and shuffles the parts to create new offsprings.

* + 1. **Two-point-crossover**

Two point crossover cuts the parents at two random position and shuffles the parts to create new offsprings.

* + 1. **Uniform-swap**
    2. **Arithmetic crossover**
    3. **Random crossover**
  1. **Mutation**
     1. **Ball mutation**
     2. **Ball mutation with boundaries**
     3. **Random member mutation**
     4. **Swap mutation**
  2. **Others**
     1. **Elitism**
     2. **Two population genetic algorithm**
     3. **Mating pool**
     4. **Worst removal**
     5. **Measuring genotype/phenotype**
     6. **Particle Swarm Optimization**
     7. **Decreasing Mutation rate**

1. **Analysis of Combinations**
2. **Conclusion**
3. **References**