IT3708 Homework 4:
Evolving Artificial Neural Networks (EANNs)

Purpose: Learn how to combine evolutionary algorithms and artificial neural networks for supervised learning tasks.

1 Assignment

You will now use your evolutionary algorithm (EA) from earlier assignments to evolve weight vectors (i.e., sets of weights) for artificial neural networks (ANNs) whose topologies are designed to solve the same supervised learning tasks as in assignment # 3.

In this exercise, each individual in the EA population will be a complete set of weights for the entire ANN, where you decide the topology of the ANN ahead of time, based on the most successful topologies that you found in exercise 3.

To test the fitness of a weight set, simply load each weight into the ANN and run the ANN on **the complete data set** for a given problem. There is no longer a need to separate the data into training and test sets; now, all of the data is for (fitness) testing.

For each data case, simply record the deviation between the ANN's output and the desired output, and sum the squares (or absolute values) of these deviations into a total error term. Fitness will then be inversely proportional to this total error.

This should entail only very minor extensions to your EA, since encoding connection weights as binary numbers and then scaling them to **real** numbers in a range such as (-5, 5) is a simple operation. You may also want to add a restriction to crossover such that crossover points can only be chosen at the boundaries between the input-weight groups for each neuron. This is a common strategy in EANN usage.

No significant changes should be needed for your code from exercise 3. Now you are simply turning off backpropagation, and now you are only doing one pass of the data set through a network.

Remember to include weights for your dummy links, i.e., those that code the thresholds for your neurons, if you used such links during backpropagation.

The key empirical aspect of this exercise is to compare the performance of the EANN to the ANN with backpropagation. To do so, you will need a common performance metric: the number of passes through the data set, P. For example, in the EANN, if the population size is 50 and you run for 300 generations, then $P = 15000 = 50 \times 300$, since each individual will require one pass of the data set. Similarly, a backpropagating ANN will require one data-set pass per epoch, so P = # epochs. Here, we ignore the fact that the training set is not the complete data set, but to be more accurate, you should include that fraction in the calculation: P = EF, where E = # epochs, and F = the fraction of the data set used for training.

For **each** of the supervised-learning problems that you tackled in assignment 3, apply your EANN to it and attempt to find solutions with a similar total error as you attained in exercise 3. If used alone, the test error from backpropagation must be scaled, since exercise 3 uses only a fraction, (1-F) of the data set for testing. Hence, you should run your EANN until its error, E_{evol}, approaches:

$$\frac{E_{bp}}{1-F} \tag{1}$$

where E_{bp} is the test error found with backpropagation, and 1-F is the test-set fraction. Alternatively, you can compare E_{evol} to $E_{bp}+E_{train}$, where the latter addend is the final **training** error from backpropagation. It is difficult to say which comparison is most fair. Just choose one: either $E_{bp}+E_{train}$ or $\frac{E_{bp}}{1-F}$ and call it your *target error*.

Record the EANN generation,G, at which E_{evol} first becomes $\leq$ the target error. Then compare GM, where M is the population size, to EF (see above), to determine whether evolution or backpropagation was most effective on that particular data set. Do this for each of your data sets from exercise 3.

If you can never get E_{evol} below the target error, you are free to go back and run backpropagation for FEWER epochs in an attempt to increase the target error. Of course, if you overtrained with backpropagation, then this could have the opposite effect! In either case, if you do go back and try to **weaken** your results from exercise 3, please indicate this in your report.

2 Deliverables

1. A overview description of your code. Explain the code changes needed to integrate the two systems, particularly the genotype changes needed to handle weight vectors.
2. Provide fitness plots (x-axis: generation, y-axis: best and average fitness) for each of your runs and a brief description of these plots.
3. Provide the efficiency comparisons for each data set.
4. Include any other interesting observations that you have about EANNs in general or your experiments in particular.