Bio-Inspired Computation: Coursework 2
Evolving a Neural Network to Classify Neuromuscular signals

You are going to evolve neural networks that will predict, from neuromuscular signals, whether the subject is walking, standing or running. See http://www.macs.hw.ac.uk/~dwcorne/Teaching/biccw2slides.pptx, which provides information about these data (covered in DWC’s lecture 7). In the slides, you can see how the data have been processed into a training set for a neural network. So that not too much processing time is needed, the datasets you pick up only contain about 2% of what I have. The data we have are from Pamela Hardaker at de Montfort University, who is collecting and working on this type of data for her PhD.

What you are asked to do:
1. Pick up my evolving neural network code, and other materials, from here: http://www.macs.hw.ac.uk/~dwcorne/Teaching/biccw2/
2. Write the C code for the COURSEWORK_CROSSOVER and COURSEWORK_MUTATION functions inside the evolvenn.c file.
3. Perform a number of experiments.
4. Write a 3-side report.

Again, in more detail

1. Materials
‘evolvenn.c’ is a C program. You can compile and run it on a unix of linux machine (or, e.g., by logging into a linux machine via cygwin or similar on Windows). Compile it like this:

```
[command-prompt] gcc evolvenn.c -o eann -lm
```

... that will mean ‘eann’ is the executable that you run. You run it like this:

```
[command-prompt] ./eann input_file
```

where `input_file` contains various parameters. Have a look at the two examples provided ‘inputs1.txt’ and ‘inputs2.txt’. In your experiments, you will probably change only three lines: the seed (you need a different random seed for each experiment), the alpha value, and the output file (which gives you the weights of the best network found in that experiment). When you run the program, it gives you updates every time it finds a new best network, and/or just reminds you of the best result so far every 1000 evaluations. An example line is:

```
best at 39050 evals: training: 72.5% - test 68.21% - fitness 72.22
```

The NN is being trained on the training set only. This line shows that on the training set it achieves 72.5% accuracy, in the sense that it correctly classifies 72.5% of cases. Remember that there are 3 outputs, so if the target output is “1 0 0” this means class “standing”, if it is “ 0 1 0 “, this means class “walking”, and “ 0 0 1“ means running. The accuracy figure is based on the highest value, so if the network’s output is “0.3 0.41 0.1”, this is taken to mean a prediction of “walking”. The program also works out the network’s mean-squared error based on the target outputs. You see that given in the ‘new best’ output. The actual fitness function – which the EA is trying to maximise, is accuracy minus mse. Hence, in the above line, you’ll notice the fitness value is 72.32, so the mse in this case must have been 0.28.

The simple shell script ‘simple_batch’ shows you how you can run several experiments, and keep the screen output in a results file. You run it by just typing ‘simple_batch’ at the command prompt.

2. Write code
The EANN code builds a neural network with 9 inputs, 12 hidden units, and 3 outputs. It evolves the 159 weights for
this network (including the bias weights) using a steady state, binary tournament selection EA. In the file you pick up from me, it uses simple one-point crossover, and for mutation it just chooses a single gene at random and changes it to a new value between -1 and 1. You need to implement two types of Gaussian mutation, and two types of crossover, and experiment with them. Your implementation will be within the COURSEWORK_CROSSOVER and COURSEWORK_MUTATE functions in the code. My comments in and around those functions, and their example implementations, should be all you need to get going, plus the last few of my Operators slides:

http://www.macs.hw.ac.uk/~dwcorne/Teaching/bic010_3am.ppt

For mutation: very simple: just implement the following operator:

Choose three genes uniformly at random (in the code, there are ‘numgenes’ genes), and add a (different) Gaussian perturbation (mean 0 and standard deviation 1) to each of those genes.

This is very simple because I provide ‘grand()’ to give you the necessary Gaussian perturbation.

For Crossover, I want you to experiment with line crossover, box crossover, and also extended line crossover (with alpha at 0.1 and at 0.25), and extended box crossover (with alpha at 0.1 and at 0.25). So, six different crossover operators in all. I want you to compare the results from these 6 different operators. You will probably need to do two different versions of the programme – one for box and one for line crossover – but other variations should be able to be handled simply by changing the alpha value in the input files.

3. Experiments:
A single experiment means running the program at least 5 times with different random seeds (first parameter in the input file). Minimally, I expect you to do six experiments, and record the results. That’s one experiment each for each of these six different crossover operators: line, extended line (alpha 0.1), extended line (alpha 0.25), box, extended box (alpha 0.1), extended box (alpha 0.25).

Additional experiments, for Level 11 students (MSc, & final year MEng) only
Choose any specific crossover (e.g. extended box with alpha at 0.1), and investigate how well this method can perform in each of the following scenarios: scenario1: ‘walking’ vs ‘not walking’; scenario2: ‘standing’ vs ‘not standing’. To investigate each of those scenarios, you will need to modify the training and test data appropriately, and the inputs file.

4. Report:
Your 3-page report (3 sides) should contain:

- Your COURSEWORK_MUTATION and COURSEWORK_CROSSOVER code.
- A basic table of results for the six experiments.
- A discussion of the results (maximum 400 words) which tries to provide a justified recommendation of the best crossover method to use for this problem, and makes any other observations you find appropriate.

In addition, for Level 11 students:

- A brief explanation of how you modified the data and input files for the extra scenarios.
- An additional table showing the results of the extra experiments.
- A discussion (minimum 100 words) of the results of the extra experiments.

I deliberately do not specify more details (e.g. what ‘results’ to tabulate, etc...), because I want to see what you think is appropriate. You can also include additional experiments, graphs, and so on, as long as the report is max 3 pages.

Handin: email your report to me at dwcorne@gmail.com with subject line “BIC NN” by 23:59pm Saturday 15th November 2014.

Marking scheme:
This is worth 20% of the module. Of those 20%, your code will get 5, your table and/or other display of results will get 5, and your discussion will get 10.