University of Wollongong

School of Computing and Information Technology

CSCI464/964 **Computational Intelligence** **Autumn 2019**

Assignment **(Due at 3:30pm, 27 March 2019, Wednesday)** **10 marks**

Aim:

This assignment is intended to provide basic experience in writing neural network applications and conducting classification experiments with MLPs. After having completed this assignment you should know how to implement a back propagation multi-layer neural network which can be used for a variety of classification tasks.

Preliminaries:

Read through the lecture notes on MLP neural networks, paying particular attention to the feed forward classification algorithm and the back propagation learning algorithm. To assist with this assignment **a 3-layer (input layer, one hidden layer, and output layer)** neural network trained with the back propagation algorithm written in C++ is provided in the file mlp.cpp. Please study this program in context with the lecture notes so that you thoroughly understand its operation. A number of training data files are also provided for you to experiment with. Make note of how the MLP's parameters are read from the data file. Before commencing any coding, compile and run the given MLP with the data to confirm that it works. You should notice that the MLP is able to converge on some of the data sets with relatively low error rates but on other datasets this 3-layer MLP performs poorly.

Assignment Specification:

Your main task in this assignment is to implement selectable ordering of the training data and make the net more powerful by enabling the MLP to be configured as a **3-, 4- or 5-layer MLP** with a specified number of neurons in each layer and to use the MLP to classify all the given data. You are to also provide a test function so that the MLP can learn training data and be tested with different test data. For this assignment you can write and compile your code on PC or UNIX platforms. To complete this assignment, it is recommended you follow the steps below.

Step 1: (3 marks) To improve MLP training, implement an option for providing selectable ordering of the training data. The "Ordering" parameter should be added to the definition header of the training data files after ObjErr. e.g.:

```
. . .
Mtm1: 1.2
Mtm2: 0.4
ObjErr: 0.005
Ordering: 1
. . .
```

"**Ordering**" determines which training pattern is selected each training iteration (epoch). The options are:

0 Fixed	Always uses the same given order of training patterns.
1 Random	Make completely different (random) order at each epoch (i.e. new permutation).
2 Random Swap	Random permutation at start. After each epoch, two patterns are randomly selected and exchanged.

Tip: Try using an index array to rearrange the selection order.

Step 2: (3 marks) Implement **4- and 5-layer neural networks (i.e., have 2 and 3 hidden layers, respectively)** trained with the back propagation algorithm by modifying the code in mlp.cpp. To do this rename the `TrainNet()` function to **`TrainNet3()`** and make modified copies of this function (named: **`TrainNet4()`** and **`TrainNet5()`**) with **4- and 5-layers** respectively. Then incorporate a switch statement into your code so that the appropriate `TrainNet()` function is invoked according to the data specs. Test your completed code on the provided data and ensure that the MLP configuration in the data files (i.e., the number of layers, number of neurons, etc.) is being complied with.

Step 3: Write a `TestNet()` function that tests the MLP's performance by using the MLP trained with the training data to classify the test data in the data files and report the error rate. A typical run of you MLP should look like:

```
NetArch: IP:8 H1:5 OP:1
Params:  LrnRate: 0.6  Mtm1: 1.2  Mtm2: 0.4

Training mlp for 1000 iterations:
   #       MinErr      AveErr      MaxErr     %Correct
   1:     0.000006    0.077862    0.713725    19.0373
   2:     0.000000    0.072893    0.673643    17.1607
   3:     0.000018    0.072357    0.670814    16.9976
   4:     0.000002    0.071879    0.669441    16.9976
   5:     0.000012    0.071394    0.668451    16.8072
   6:     0.000005    0.071004    0.667836    17.0247
   7:     0.000003    0.070734    0.667509    17.2151
   8:     0.000047    0.070535    0.667491    17.4055
   . . . .
1000:     0.000126    0.001256    0.008060     5.1607

Testing mlp:
           MinErr      AveErr      MaxErr     %Correct
          0.001126    0.008256    0.015060    10.1607

End of program.
```

Step 4: (4 Marks) Your task here is to devise various back-propagation neural network of minimum size (in terms of the number of neurons in the MLP) that can correctly classify the Two Spiral Problem (data1.txt) and the other data sets associated with Problems 2 and 3 (see below). Note: the data for problems 2 and 3 may need extra work, like normalizing the data, dividing it into training and test data and adding the MLP header information. You should experiment with various MLPs with variations in the numbers of layers, number of neurons in each layer, parameters (e.g., learning rate and momentum) and the number of training iterations. If you are unable to classify all the data correctly, then your final MLP configuration should be a best case compromise between size and performance. For each data set you **must** write a report comprised of the following information **with the given headings**:

1) **About the problem**
   A brief description of the problem.
2) **On the various MLPs experimented**
   A report of the various MLPs (at least 3 MLPs) that you experimented with including any parameter changes you made and the results that were achieved. Try running each MLP a few times to determine if the problem has local minimums. If so state this in the report and see if this can be avoided by increasing the momentum. You should also indicate the approximate architecture that you think is most suited for the problem and how long and the number of epochs it takes to learn the data. (e.g., "… the best MLP for the problem appears to be a network with a large number of hidden layers with few nodes in each layer…"). Use graphs if you think this is helpful in understanding the performance differences of the various MLPs.
3) **On the final MLP's architecture**
   Report on the final MLP that you consider is either the one that classifies all the data correctly or is a best case compromise between size and performance. Provide a description of the final MLP architecture together with a drawing of the MLP.
4) **On the final MLP's performance**
   Write a brief summary of the final MLP's performance. Also, provide a graph showing the epochs vs error rate. (Note: for binary classification problems, the error rate should indicate the percentage of patterns correctly classified on the training and test data.) Was the final MLP able to learn all the training data and classify the test data correctly? If not, why not? Are there local minimums with this MLP?

Submit:

Neatly print your reports and code (i.e., first the reports and then the C++ code) on A4 pages with an appropriate cover sheet and hand it in during the lecture **on the 27th of March 2019**. Make sure your reports and code are correctly formatted and titled. (Marks will be deducted for untidy or incorrectly formatted work.) To avoid formatting problems with your code, use 2 or 3 spaces instead of tabs to indent your code and use courier font.

Also, **before 3:30pm on the 27th of March 2019**, submit your source code in the file mlp.cpp via the submit facility on UNIX:

```
$ submit -u login -c CSCI964 -a 1 mlp.cpp
```

where 'login' is your UNIX login ID.

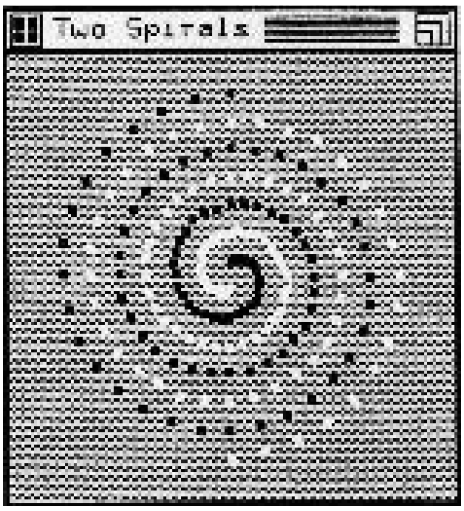
Note: Failure of your code to compile may attract zero marks. Code or reports considered to be the same due to copying will attract zero marks. To receive full marks for your assignment your program must run on the unix machines at uow. You may also be requested to demonstrate your program if the testing of your program proves to be ineffective. Marks will be awarded for correct design, implementation and style. An extension of time for the

assignment submission or demonstration may be granted in certain circumstances. Any request for an extension of the submission deadline or demonstration time limit must be made to the Subject Coordinator before the submission deadline. Supporting documentation must accompany the request for any extension. Late assignment submissions without granted extension will be marked but the mark awarded will be reduced by 1 mark for each day late. Assignments will not be accepted if more than five days late.

Description of Data

Problem 1 – Two Spiral Problem (1-SpiralData1.txt)

The two-spiral **problem** can be described as a two class problem where each class is an inter twined spiral on a two dimensional plane. This is a classical example of a non-linear data problem. With this problem it is impossible to separate two spirals coiling around each other with a linear classification method so this problem requires a multiplayer neural network to solve. The spiral data considered here is comprised of 194 training pattern and 194 test patterns. The output classes are represented as 0 and 1. To classify the data any output from the net less than or equal to 0.5 can be considered as belonging to class 0 else the output can be considered to belong to class 1.



Files:

1-SpiralData.txt Spiral point data

Note: The spiral data file also has the mlp header info needed to run the data on an mlp. The training and test data are the same data. You should adjust the mlp parameters to achieve the best learning result.

Problem 2 – Abalone Age Problem (data2.txt)

This problem is about predicting the age of abalone from physical measurements. The age of abalone can be determined by cutting the shell through the cone, staining it, and counting the number of rings through a microscope. As this is a boring and time-consuming task it would be better if other measurements can be taken and used to predict the age. The following attributes have been taken from abalone specimens to form the training and test data.

Name	Data Type	Meas.	Description
Sex	nominal		M=0, F=1, and I=2 (infant)
Length	continuous	mm	Longest shell measurement
Diameter	continuous	mm	perpendicular to length
Height	continuous	mm	with meat in shell
Whole weight	continuous	grams	whole abalone
Shucked weight	continuous	grams	weight of meat
Viscera weight	continuous	grams	gut weight (after bleeding)
Shell weight	continuous	grams	after being dried
Rings	integer		+1.5 gives the age in years

Note: the Rings attribute has been normalized between 0..1 to suit mlp learning and is the output the mlp will try to learn. Thus our data here is comprised of 8 inputs and one output

Problem 3 – SPECT Heart Diagnosis Problem (data3.txt)

This dataset describes diagnosing of cardiac Single Proton Emission Computed Tomography (SPECT) images. Each of the patients is classified into two categories: normal = 0 and abnormal = 1. The database of the SPECT image sets (from patients) was processed to extract 44 features that summarize the original SPECT images. The learning task is to learn to predict the diagnosis from the 44 image features.

--- END ---