University of Wollongong

School of Computing and Information Technology

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

Assignment 2 **(Due: 5:30PM, Wednesday 1st May)** **10 marks**

--- Part 1 (Support Vector Machines, 5 marks) ---

Aim:

This assignment is intended to provide basic experience in solving classification and regression problems with Support Vector Machines (SVMs). After having completed this assignment you should know how to train and test a classifier/regressor by SVM package LIBSVM (https://www.csie.ntu.edu.tw/~cjlin/libsvm/)

Assignment Specification:

1. Download "The practical guide" from https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf and read it carefully. You are strongly encouraged to try the examples given in this guide to obtain better understanding of SVM classification and regression. You need to download the LIBSVM library from https://www.csie.ntu.edu.tw/~cjlin/libsvm/ and read README to know how to use it;
2. Three data sets, 1-SpiralData1.txt, data2.txt and data3.txt, are provided in this assignment. They are just the datasets that you have become familiar with in assignment one. For 1-SpiralData1.txt, randomly partition all the 192 data into 60% as training data and 40% as test data. For data2.txt and data3.txt, just use the training and test data predefined in the head of the files.
3. Using the knowledge you learn from "The practical guide," for each of the three datasets, train an SVM classifier or regressor with the training data and test it on the test data. Try your best to achieve the highest accuracy on the test data by carefully tuning the parameters of SVMs with training data.
4. Write a report on this part. **Address the following questions separately with a clear heading**.
    a. An introduction of this part of assignment, datasets, and how the training and test data are defined;
    b. Indicate which kind of SVM (classifier or regressor; linear or nonlinear) is chosen for each dataset and explain your choice;
    c. How you tune the parameters of SVMs with training data (for example, the number of parameters, the range you choose for each parameter, the number of grids you use, or any tools you use, etc.);
    d. The achieved classification or regression accuracy on the training and test data for each of the three datasets;
    e. Detailed discussion and analysis of what you have observed and experienced. Compare the training/test of SVM with the training/test of neural networks in assignment one.
    f. Attach a printout of the commands that you use to train and test your SVM classifier and regressor with LIBSVM. Group these commands for each of the three data sets clearly.

--- Part 2 (Self Organizing Map, 5 marks) ---

Aim: This assignment is intended to provide basic experience in implementing self-organizing map (SOM). After having completed this assignment you should know how to realize an SOM network, understand its training process, and interpret the learned weights.

Assignment Specification:

1. A subset of the MNIST data set (http://yann.lecun.com/exdb/mnist/) is provided with this assignment in "SOM_MNIST_data.txt". It consists of 5,000 examples, each of which corresponds to one column of this text file. Each example has been reshaped from a 28 by 28 gray-level image into a 784-

dimensional feature vector. You will be able to view the original images by reshaping each feature vector back and display the 28 by 28 matrix with appropriate image-processing software.

2. Read the lecture notes and other resources (for example, Chapter 9 of [1] below) to review SOM. Basically, given a dataset, SOM aims to learn a set of prototypes of the data and spatially arrange the prototypes in a way that is indicative of the data distribution in the original input space. Implement an SOM neural network and train its weights with the provided dataset. The default size of the 2D lattice is 10 by 10. You can use a reasonably larger or smaller size according to the computational resource available to you. An example code written in Matlab is provided **for your reference**. Note that you are required to implement SOM in C++ by yourself and are **NOT** allowed to use this Matlab code for this assignment.
   [1] Neural Networks and Learning Machines (3rd Edition), Simon Haykin, Pearson, November 2008.
3. Write a report on this part. **Address the following questions separately with a clear heading**.
   1) A brief introduction on the MNIST data set (read the above link) and the examples provided in this assignment;
   2) An introduction of the steps of training an SOM neural network. Particularly, describe the two phases (ordering and convergence) of the training process and how to set the learning parameters in the two phases;
   3) The change of the weights between two consecutive epochs is indicative of the convergence of the training process. To characterize this change, for each weight vector compute the Euclidean distance between its values in the t-th and (t+1)-th iterations, and then use the sum of all the Euclidean distances as a criterion. Plot the value of this criterion with respect to the number of epochs and describe its evolution;
   4) Plot the learned weight vectors of the 2D lattices corresponding to the following **three** stages. The first one is at the initialization stage and the third one is at the convergence (or stable) stage, while the second one is in between. Each weight vector shall be plotted as a 28 by 28 image. Example figures of the first and last stages are provided in next page for your reference. Note that your figures are not necessarily same as the examples.
   5) You are encouraged to investigate various settings to train this SOM neural network, including the number of training examples, the size of 2D lattice, the learning rate, the size of neighborhood, and the number of epochs, etc. Provide detailed discussion and analysis of what you have observed and experienced.

Submit:

Submit your program on UNIX via the submit command **before the deadline** and hand in your report with a cover page **before the deadline**.

For part 1: No need to submit any code (since you will attach a printout of the commands that you use)

For part 2: Before submitting your code check the format to ensure the format and newlines appear correct on UNIX. (Marks will be deducted for untidy or incorrectly formatted work.) To avoid formatting problems avoid using tabs and use 4 spaces instead of tab to indent you code. Make sure your file is named: **som.cpp**.

Put both part 1 and part 2 into a single report. Do not write two separated reports.

Submit using the submit facility on UNIX ie:

$ submit -u login -c CSCI964 -a 2 som.cpp

where 'login' is your UNIX login ID.

We will attempt to run your program on banshee. If problems are encountered running your program, you may be required to demonstrate your program to the coordinator at a prearranged time. If a request for a demonstration is made and no demonstration is done, a penalty of 2 marks (minimum) will be applied. **Marks will be awarded for a comprehensive report, correct program design, implementation, style and performance.** Any request for an extension of the submission deadline must be made by applying for academic consideration 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 one week late.

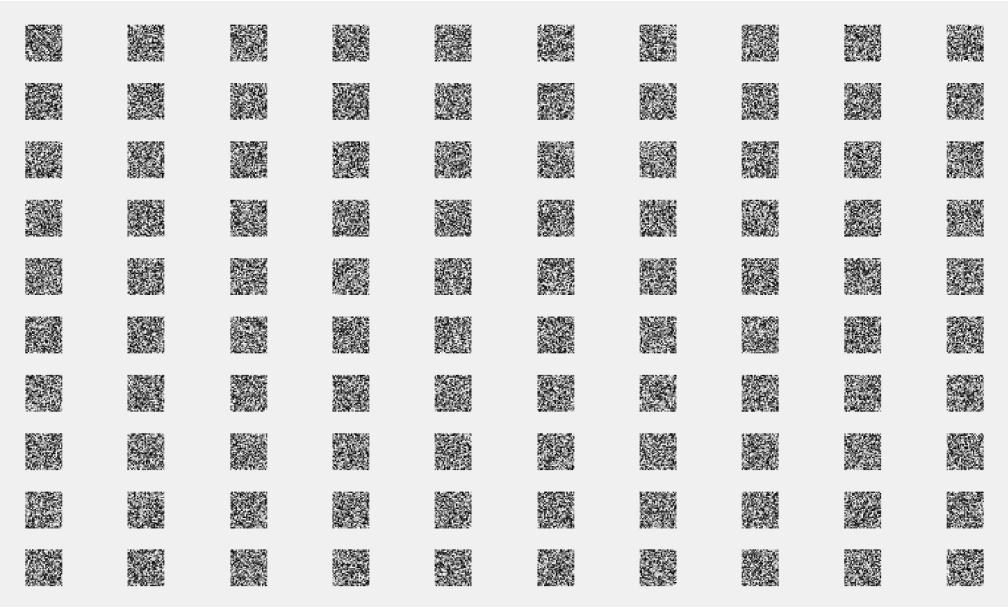

Fig. 1 Visualization of the **initial** weight vectors

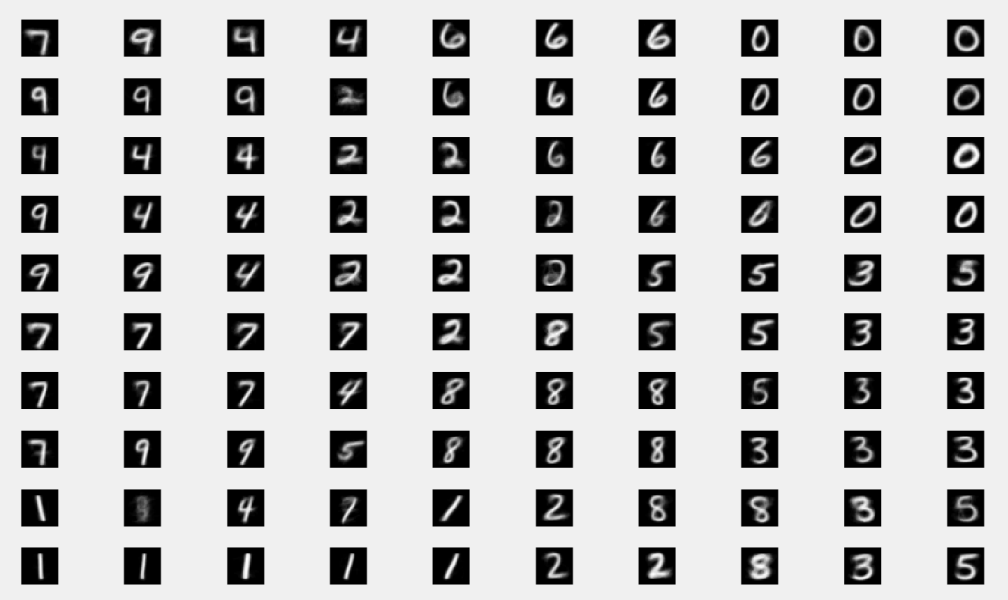

Fig. 2 Visualization of the **evolved** weight vectors