# Applied Machine Learning T2 Lab Evaluation

Date: 9th April 2016

## Digit Recognition using Artificial Neural Networks

The objective of this problem is to perform digit recognition given an image of a handwritten digit using Artificial Neural Networks (ANN). This problem has 2 parts: Perform the digit recognition using a baseline neural network and then modify the network by tweaking its hyper parameters and report the results in each of the cases.

You are provided with a subset of the MNIST dataset (which is a large database of handwritten digits commonly used for training various image processing systems) for this task.

The broad steps to be followed are as follows:

- Normalize the images to a standard size
- Implement the ANN
- Build and train the network.
- Test the baseline network developed above and report the accuracy
- Tune the network by tweaking hyper parameters using the validation data, test and report accuracies for each of the cases.

## Part 1: Perform digit recognition using a baseline ANN

### Phase 1 – Image Normalization

Normalize the images using an image processing library to a common format. Initially choose the format to be 16 x 8, later you can try 16 x 16, 28 x 28.

Checkpoint #1: Normalized JPEG (double click the image and observe)

Phase 2 – Implement the ANN

1. You are given the skeletal code base for the ANN. Complete the implementation of different functions without making changes to the skeletal code base provided.
2. Form the features such that each pixel is a feature. Note that the pixel is a vector of RGB. Thus if you have 16 x 8 image, you will form 128 x 3 = 384 input units or features. Normalize the input before feeding it into the network.
3. Let the output vector emitted by the network be a one hot vector of dimension 10.
   Note: A one hot vector is a binary vector where only a single bit is set to 1 whereas all other bits are set to 0.
   One_hot_vec(1) = 0100000000
   One_hot_vec(8) = 0000000010

4. Partition the data into training, validation and test datasets in the ratio 70%, 15%, 15% respectively.
5. Build 2 networks – one with 4 hidden units and another with 8 hidden units.
6. Train the neural network using backpropagation. Use cross-entropy as the cost function.
7. Test the network on the test dataset.

Checkpoint #2: Neural Network implementation, datasets, test results

## Part 2: Tune the ANN by tweaking its hyper-parameters and report results for each case

Regularization is a way of controlling the capacity of Neural Networks to prevent overfitting. Regularization is generally performed on the validation dataset during training. After training and validation, the network is tested on the test dataset and accuracy is reported.

The following tasks are related to tweaking the model parameters (during regularization) and the hyper parameters (learning rate) to see how the model behaves in each scenario:

1. Test the ANN for learning rates of 0.005, 0.02 and note the accuracies observed. *
2. Tune the ANN by performing L2 regularization and note the accuracies observed for different values of regularization coefficient ‘λ’.

Note:

- The task marked with * is mandatory to be completed by the end of the lab duration.
- Accuracies are reported w.r.t ANNs of hidden layer sizes 4 and 8.

Checkpoint #3: Results achieved by tuning the network.

Deliverables:

1. By the end of the lab, you are expected to have demonstrated the baseline ANN and the mandatory task of Part 2 working for the image size 16x8.
2. A zip named ‘team_id_T2.zip’ with source code of all your py modules and pickle files of the networks developed.
3. A Facebook post by Sunday 9:30pm comparing the baseline network results with the tuned network (all scenarios tested by you) and your inferences on the same.

Best wishes from your faculty and seniors ☺