# **Typographic Conventions**

Functions (script) names starting with Capital letters correspond to “GUI functions”

Functions (script) names starting with small letters correspond to “core mathematical functions” used to develop the network

All variable names start with capital letters

All structure names start with small letters

# **Functions used**

## **cnnsetup.m**

### **Syntax**

cnn = cnnsetup(cnn, Train\_X, Train\_Y)

### **Input Arguments:**

Train\_x: Training inputs (Dimensions: Number of entries Number of features)

Train\_y: Training targets (Dimensions: Number of entries Number of classes)

cnn: A structure which contains information about number of Convolution layers and number of sub-sampling layers. Fields of structure contain information about size of convolution kernel, sub sampling (pooling) scale and number of filters. For example, to initialize a cnn structure for input layer, 2 convolution layers and 2 pooling layers, following syntax should be followed

cnn.layers = {

struct('type', 'i') %input layer

struct('type', 'c', 'outputmaps', 20, 'kernelsize', 5) %convolution layer

struct('type', 's', 'scale', 2) %sub sampling layer

struct('type', 'c', 'outputmaps', 20, 'kernelsize', 5) %convolution layer

struct('type', 's', 'scale', 2) %subsampling layer

};

### **Output Arguments:**

cnn: A CNN network initialized with specified parameters

## **cnntrain.m**

### **Syntax**

cnn = cnntrain(cnn, Train\_X, Train\_Y, opts)

### **Input Arguments**

**Train\_X:** Testing inputs (Dimensions: Number of Images Image Height Image Width)

**Train\_Y:** Testing targets (Dimensions: Number of Images Number of classes)

**cnn:** Initialized CNN

**opts:** option structure which contains following fields

Learning Parameter (alpha)

Batch size

Number of Training epochs

### **Output Arguments:**

**cnn:** Trained CNN

## **cnntest.m**

### **Syntax**

[Er, Bad,Predicted,Actual] = cnntest(cnn, X,Y)

### **Input Arguments**

**X:** Inputs (Dimensions: Number of Images Image Height Image Width)

**Y:** Targets (Dimensions: Number of Images Number of classes)

**cnn:** Trained CNN

### **Output Arguments:**

**Actual:** Actual Labels

**Expected:** Labels predicted by network

**Er:** Error

**Bad:** Indices of misclassified images

## **Cnnff.m**

Feed forward pass for the network

### **Sytntax**

[cnn,Out] = cnnff(cnn, Batch\_X)

### **Input** **arguments**

**Batch\_X:** A batch of images

**cnn:** CNN Structure

### **Output**

**cnn:** CNN with weights adjusted in feed forward pass

**Out:** Output of a feedforward pass

## **Cnnbp.m**

Back propagation pass for fine tuning learned weights of feed forward pass

### **Sytntax**

cnn = cnnbp(cnn, Batch\_Y,Out)

### Input arguments

**Batch\_Y:** A batch of Image labels which will be used to fine tune learned weights

**cnn:** CNN Structure

**Out:** Output of a feedforward pass which will be used to calculate errors by comparing it to Batch\_Y

### **Output**

**cnn:** CNN with weights adjusted by back propagation of errors

# **Data format**

Let’s say we have total N =1000 images and 10 classes. Let’s say image dimensions are

Images should be stored in 3D array with dimensions

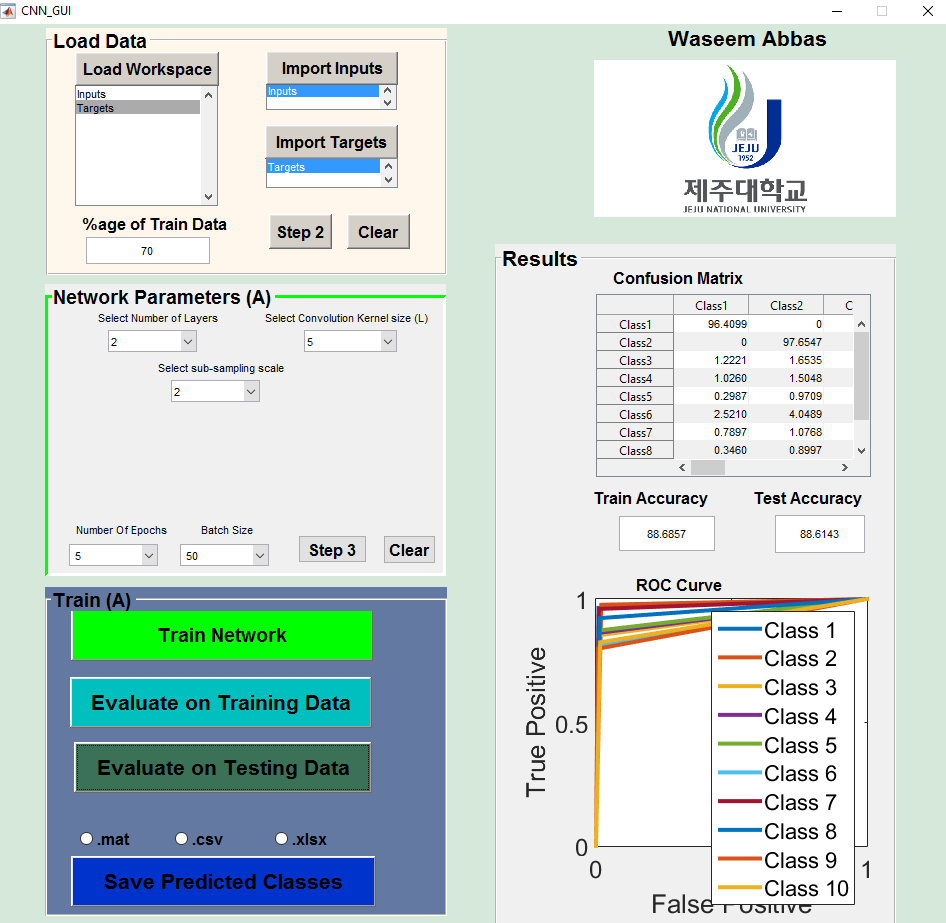
Target data dimensions are

### Important Note

Works only on images

# **GUI**

Load Inputs and Targets which should be in the format specified above and then run ‘ELM\_DeepNet\_GUI.m’



**4**

**3**

**2**

**1**

There are 4 parts of the GUI, Load Data, Network Parameters, Train and Results

## **Load Data**

This panel is used to load data from workspace. To import inputs, first click “load workspace” and then select the inputs. Then click on ‘import inputs’ button. Similarly, select targets and then click ‘Import Targets’. Once inputs and targets are imported, click on ‘step 2’. It will activate the next panel

## **Parameters**

This panel is used to set network parameters. Use following steps to properly initialize network parameters

1. Select number of layers (It will set number of convolution and pooling layers)
2. Let’s say you have selected 4 layers, you will see 4 popup menus created below. Each correspond to a convolution layer. Enter number of features (filters) you want to learn in each layer
3. Select convolution kernel size (It is fixed for all convolution layers)
4. Select pooling (sub-sampling) scale (It is fixed for all pooling layers)
5. Select number of epochs, and Batch size
6. Once you have completed step a-e, click on ‘step 3’. Network parameters will be saved and next panel will be activated

## **Train**

This panel is used to train and evaluate the network. First, hit ‘Train Network’ button. When network is trained, it can be evaluated on train data and test data. The results can also be saved.

## **Results**

In this panel, confusion matrix and ROC curve can be observed