# CSE 575 Statistical Machine Learning

# **Project 3 - Classification Using Neural Networks and Deep Learning**

# VAMSI KRISHNA KANAGALA ASU ID: 1218608781

**Introduction:**

In this project we have implemented classification using Neural Networks and Deep Learning. For this we have used SVHN dataset which is a real-world image dataset that is built from house numbers in Google Street View images. These images are cropped such that they contains only digits, and it makes this dataset similar to MNIST dataset which consists of handwritten digits. The given dataset is comprised of 10 classes. The main objective of this project is to use Neural Network and apply Deep learning principles to classify the data into its corresponding classes. Among the given dataset, there are 73257 training samples and 26032 testing samples. The resolution of each image in both training and testing samples is 32x32 and it consists of 3 RGB channels.

Before learning the model, the data is normalized to 0-1 values and the converted the labels to one-hot vectors.

**CNN Architecture:**

**Graphical user interface, text, chat or text message

Description automatically generated**

By following the CNN architecture mentioned in the canvas page, implemented 8 hidden layers and their parameters and corresponding outputs as follows:

**Hidden Layer 1 – Convolutional Layer**

Parameters – Output Feature Maps - 64, kernel\_size = (5,5), padding = 'same',   
strides = (1, 1),  Activation function = 'relu',  input shape = (32,32,3)

**Hidden Layer 2 – Max Pooling Layer**

Parameters – pool size = (2,2), strides = (2,2),  padding = 'same'

**Hidden Layer 3 – Convolutional Layer**

Parameters – Output Feature Maps - 64, kernel\_size = (5,5), padding = 'same',   
strides = (1, 1),  Activation function = 'relu',  input shape = (32,32,3)

**Hidden Layer 4 – Max Pooling Layer**

Parameters – pool size = (2,2), strides = (2,2),  padding = 'same'

**Hidden Layer 5 – Fully Connected Layer**

Parameters – Output Feature Maps - 128, kernel\_size = (5,5), padding = 'same',   
strides = (1, 1),  Activation function = 'relu',  input shape = (32,32,3)

**Hidden Layer 6 – Fully Connected Layer**

Nodes – 3072, Activation function = 'relu'

**Hidden Layer 7 – Fully Connected Layer**

Nodes – 2048, Activation function = 'relu'

**Hidden Layer 8 – Fully Connected Layer**

Nodes – 10, Activation function = 'SoftMax'

**Models Parameters**

Model is trained using the following parameters

Optimizer – Stochastic Gradient Descent

Learning Rate – 0.01

Number of Epochs – 20

Batch Size – 120

Our validation set is same as that of Training set

**Observation & Results**

Using the above-mentioned parameters, the training and testing accuracies are obtained as follows

On Training set,

Loss/Error - 0.0124

Accuracy - 0.9972

On Testing set,

Loss/Error - 0.5686

Accuracy - 0.9085

**Plots**

**a) Accuracy v/s Number of Epochs**

**A picture containing shape

Description automatically generated**

When the number of epochs are increased, both Training and Test accuracy increases and after a certain number of epochs the Training accuracy will remain same which is a case of Over-fitting.

**b) Loss v/s Number of Epochs**

**Chart

Description automatically generated with medium confidence**

When the number of epochs are increased, both Training and Test loss decreases and after a certain number of epochs the Training loss is less than Testing loss.