

**NATIONAL UNIVERSITY SCIENCE AND TECHNOLOGY (NUST)**

(**High Impact Skills Development Program for Gilgit Baltistan**)

**Project Title: SVHN Dataset Classification Using CNN**

A Project Report

**Name: TUFAIL ASHRAF**

**Email Address:** [tufaildev021@gmail.com](mailto:tufaildev021@gmail.com)

**GitHub:** <https://github.com/tufailashraf/DSAI>

**Roll No.** GIL-DSAI-030

# Objective:

The goal of this project is to build and train a Convolutional Neural Network (CNN) to classify digits from the Street View House Numbers (SVHN) dataset.

# Project Structure:

## 1. Imports and Setup:

The necessary libraries and modules are imported, including:  
- TensorFlow and Keras for building the CNN model.  
- NumPy for numerical operations.  
- Matplotlib for visualizing data.  
- Scikit-learn for metrics and dataset splitting.  
- SciPy for loading the .mat files containing the SVHN dataset.

## 2. Dataset Acquisition:

A helper function `download\_svhn` is defined to download the dataset files if not already present locally:  
- The training dataset is downloaded from: http://ufldl.stanford.edu/housenumbers/train\_32x32.mat  
- The test dataset is downloaded from: http://ufldl.stanford.edu/housenumbers/test\_32x32.mat

## 3. Loading the Dataset:

The .mat files for both training and testing data are loaded using `loadmat` from the `scipy.io` module. The dataset contains images and corresponding labels of digits found in street view house numbers.

## 4. Data Preparation:

- Image Data (X\_train, X\_test): The images are extracted from the loaded .mat files.  
- Labels (y\_train, y\_test): The labels are also extracted and will be one-hot encoded to serve as the target output for the CNN model.

## 5. Model Architecture:

The CNN model is structured using the Keras Sequential API with the following layers:  
- Convolutional Layers (Conv2D): These layers apply filters to the input images to extract important features.  
- MaxPooling Layers (MaxPooling2D): These layers downsample the images, reducing the dimensionality and focusing on important features.  
- Flatten Layer: Converts the 2D output of the convolutional layers into a 1D array.  
- Dense Layers: Fully connected layers for classification.  
- Dropout Layer: Used to prevent overfitting by randomly deactivating neurons during training.

## 6. Training the Model:

The dataset is split into training and validation sets, and the model is trained using the Adam optimizer and categorical cross-entropy as the loss function.

## 7. Evaluation and Visualization:

The model is evaluated on the test set, and performance is measured using accuracy and confusion matrix visualization to display the model's ability to correctly classify each digit.

# Next Steps:

- Evaluate the model's performance using additional metrics such as precision, recall, and F1-score.  
- Further tuning of hyperparameters like learning rate, batch size, and number of epochs to improve performance.  
- Consider data augmentation techniques to improve model generalization

**THE END**