

HIGH IMPACT SKILLS DEVELOPMENT PROGRAM FOR GILGIT BALTISTAN

CNN Model for SVHN Dataset

`Name: Noor ul wara

Email: [noorulwara@gmail.com](mailto:noorulwara@gmail.com)

GitHub Account: https://github.com/noorulwara

## Overview

This project aims to develop a Convolutional Neural Network (CNN) model to classify images of digits using the Street View House Numbers (SVHN) dataset. The dataset presents several real-world challenges, including variations in lighting, scale, and digit orientation. Despite these complexities, the goal is to build a robust model capable of accurately recognizing and classifying digits in diverse conditions.

The outcomes of this project are highly applicable in various fields, such as automatic number plate recognition, postal address scanning, and digit classification in mobile applications. Participants will gain practical experience in computer vision by implementing and refining a CNN model to address real-world digit classification challenges.

## Dataset

The project utilizes the Street View House Numbers (SVHN) dataset, which comprises more than 600,000 labeled digit images. Unlike synthetic datasets, SVHN captures digits in natural scenes, often with noisy backgrounds and distortions. This makes it a valuable resource for developing models that can handle real-world image data. The dataset is available for download from SVHN Dataset.

The dataset is stored in .mat file format, which can be loaded using the scipy.io.loadmat function. It contains images and corresponding labels for digits 0–9, with considerable variations in context and appearance.

## Preprocessing

Effective preprocessing is crucial for achieving good performance on the SVHN dataset. Key preprocessing steps include:

1. **Loading the Dataset:** The dataset is loaded from .mat files using helper functions designed for easy access and manipulation.
2. **Reshaping Input Data:** The input data is reshaped to fit the expected input format for the CNN. Each image is converted into a suitable format for processing by the neural network.
3. **Label Adjustments:** The dataset labels are adjusted so that the digit ‘0’ is properly represented, as it is often misinterpreted or mislabeled.
4. **One-Hot Encoding:** The labels are one-hot encoded to prepare them for the categorical cross-entropy loss function, which is commonly used in multi-class classification tasks.
5. **Normalization:** The pixel values of the images, originally in the range of 0–255, are normalized to a range of 0–1 to standardize input data and improve model training

performance.

Additionally, sample images from the training set are displayed to provide insight into the dataset's diversity and complexity. These samples illustrate the wide range of conditions in which digits appear, reinforcing the importance of building a robust model.

## Model Architecture

The CNN model was constructed using TensorFlow/Keras. The architecture was designed to effectively capture the spatial hierarchies of the digit images and consists of the following key components:

Convolutional Layers: Three convolutional layers with ReLU activation functions and max-pooling operations, designed to progressively reduce the dimensionality of the input data while preserving critical features.

Flattening Layer: Converts the 2D feature maps from the convolutional layers into a 1D vector, which can be processed by the dense layers.

Fully Connected (Dense) Layers: Two dense layers, including the final output layer, which uses softmax activation for multi-class classification, providing probability distributions across the 10 digit classes.

## Training

The model was compiled using the Adam optimizer and the categorical cross-entropy loss function, which is appropriate for multi-class classification tasks. The training process was conducted over 10 epochs with a batch size of 64. During training, both accuracy and loss metrics were monitored on the training and validation datasets to ensure the model was learning effectively.

## Evaluation

After training, the model was evaluated on a separate test set. The model achieved a test accuracy of approximately 87.65%, indicating a high level of generalization to unseen data. This performance is promising given the complexities of the SVHN dataset, which includes significant variability in digit appearance due to factors such as lighting, distortion, and background noise.

## Visualization

The performance of the model during training is visualized through plots of training and validation accuracy, as well as loss over each epoch. These plots help to identify trends in model learning, such as convergence and potential overfitting or underfitting. The visual analysis supports further tuning and optimization of the model.

## Conclusion

The CNN model developed in this project successfully tackled the challenge of real-world digit classification using the SVHN dataset, achieving a strong test accuracy of 87.65%. This project demonstrated the importance of preprocessing, model design, and evaluation in building effective computer vision systems. Future improvements may include experimenting with deeper architectures or leveraging transfer learning using pre-trained models to further enhance performance.