# Baseline Convolutional Neural Network for Digit-Sum Regression

Rishabh Bhattacharya 2023121011

## Introduction

This report describes a baseline model that predicts the sum of digits in an image by means of a simple convolutional neural network (CNN). The task is to read input images and output a single integer that corresponds to the sum of all digits present. The following sections outline how data is loaded and normalized, how the architecture is designed, and how training progress is monitored.

## Data Preparation

Data and labels are loaded from `.npy` files, concatenating multiple sources into a single dataset. Each image is reshaped from $(H, W)$ to the 4D format $(N, 1, H, W)$. Pixel values are then normalized into the $[0, 1]$ range by division by 255. The complete dataset is split into training and validation sets, with training corresponding to 90% and validation to 10% of the samples. Both subsets are wrapped in `TensorDataset` and batched using `DataLoader`.

## CNN Model Architecture

The model is defined with:

- Two convolutional layers, each followed by ReLU and max-pooling, reducing the spatial dimensions progressively.
- A flattening operation after the final pooling. The resulting feature vector is passed to a fully connected layer of moderate dimension (e.g., 128 units), followed by dropout at a probability of 0.2.
- A final linear layer that outputs a single real number, interpreted as the regression value for the sum of digits.

All hidden layers apply the ReLU activation function. This configuration is suitable for inputs around $40 \times 168$ after initial preprocessing steps, though the core ideas can be generalized to other dimensions.

## Training Procedure

A mean squared error (MSE) loss function is used for regression, optimized with Adam at a learning rate of approximately $10^{-3}$. Training occurs over a specified number of epochs (for example, 30). Each epoch:

1. Iterates through mini-batches of training data, computes the forward pass, calculates MSE loss, and updates parameters via backpropagation.
2. Accumulates and records the training loss across batches.
3. Evaluates on the validation set to track the validation loss, and updates the best model checkpoint whenever a new lowest validation loss appears.

## Training Analysis

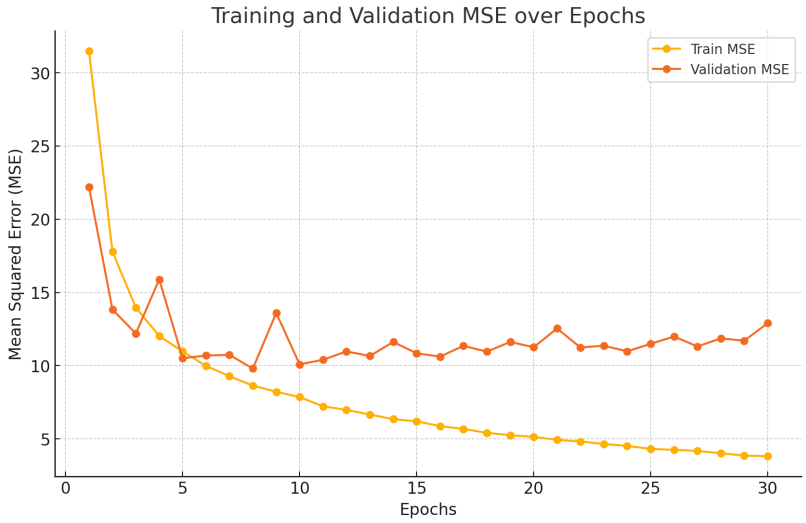


Figure 1: Epoch-wise training (yellow) and validation (orange) MSE for the baseline CNN model. While the training MSE steadily decreases, the validation MSE plateaus and fluctuates, indicating limited improvement on unseen data and possible overfitting.

## Evaluation

After the final epoch, the best model parameters (associated with the lowest validation loss) are loaded. This model is evaluated on the entire validation set, yielding:

- A final MSE measurement.
- An integer accuracy metric.

This accuracy metric is the proportion of images whose abs difference is 0.5 from the ground truth.