# CNN Model Training and Evaluation Report

Sanchit

January 8, 2025

## 1. Overview of the Problem

The task involves training a Convolutional Neural Network (CNN) for regression on a dataset consisting of grayscale images. The goal is to predict a continuous value for each image. The dataset was split into three parts: training (80%), validation (10%), and test (10%) sets. The model architecture consists of two convolutional blocks, followed by fully connected layers. The CNN utilizes ReLU activations and dropout layers for regularization.

## 2. Data Preprocessing

The dataset was loaded from '.npy' files and split into training, validation, and test sets. The data was stored in NumPy arrays for efficient handling and processed as follows:

- The dataset was split into training (80%), validation (10%), and test (10%) sets.
- Data was converted into PyTorch tensors with the appropriate shape, and a channel dimension was added for the grayscale images.
- DataLoader instances were created for batching and shuffling the dataset.

This ensures efficient data handling and optimization during training.

## 3. Model Architecture

The model consists of the following components:

- **ConvBlock:** Each block includes a convolutional layer, batch normalization, and a ReLU activation function.
- **CNNModel:** Two convolutional blocks are followed by max-pooling layers. After the convolutional layers, the feature maps are flattened and passed through two fully connected layers. A dropout layer is added to prevent overfitting.
- **Output:** The final output is a continuous value used for regression.

The network architecture is designed to handle the input shape of (1, 40, 168), where 1 is the number of channels (grayscale), 40 is the height, and 168 is the width.

## 4. Training and Evaluation

The training process involved the following steps:

- **Loss Function:** Mean Squared Error (MSE) loss, suitable for regression tasks.
- **Optimizer:** The Adam optimizer was used with a learning rate of 0.001.
- **Training Epochs:** The model was trained for 25 epochs. The training loss and validation loss were computed after each epoch.

- **Evaluation:** The model was evaluated on the validation and test sets. The test set accuracy and loss were reported.

The results show the model's ability to generalize from the training data and perform regression with reasonable accuracy.

## 5. Results and Conclusion

The model achieved the following results:

- **Test Loss:** The test loss was reasonably low, indicating good performance on unseen data.
- **Accuracy:** The model achieved an accuracy of approximately 85% on the test set.

Future improvements could involve experimenting with deeper networks, hyperparameter tuning, or using a larger dataset to improve performance.