Weekly Report

Week 5

Nguyen Huu Huy Thinh

2025-03-03

Abstract

This report summarizes the development of a custom Convolutional Neural Network (CNN) model designed for plastering work. The report details the dataset collection process, model architecture, training procedure, and evaluation metrics. Finally, the report presents the model's performance and discusses potential improvements for future iterations.

1 Dataset Overview

- **Dataset Overview**: The dataset contains 5500 images, with 500 original images (Figure 1) and 5000 augmented images (Figure 2). Out of these, 5400 images (original and augmented) are used for training, and 100 original images (20% of 500 original images) are used for testing.

Figure 1: Original Images

- **Augmentation methods:**
  - Flip (p=0.3): Randomly flipping images horizontally
  - Rotation (angle range: ±20 degrees): Rotating images by a random angle
  - Color Jitter (brightness=0.3, contrast=0.3): Adjusting the brightness, contrast, and saturation of images
  - Random Affine (degree of rotation: 10, translation=0.1): Applying random affine transformations, such as scaling, translation, and shearing
  - Random Perspective (distortion scale=0.2, p=1): Applying random perspective transformations to simulate various viewing angles

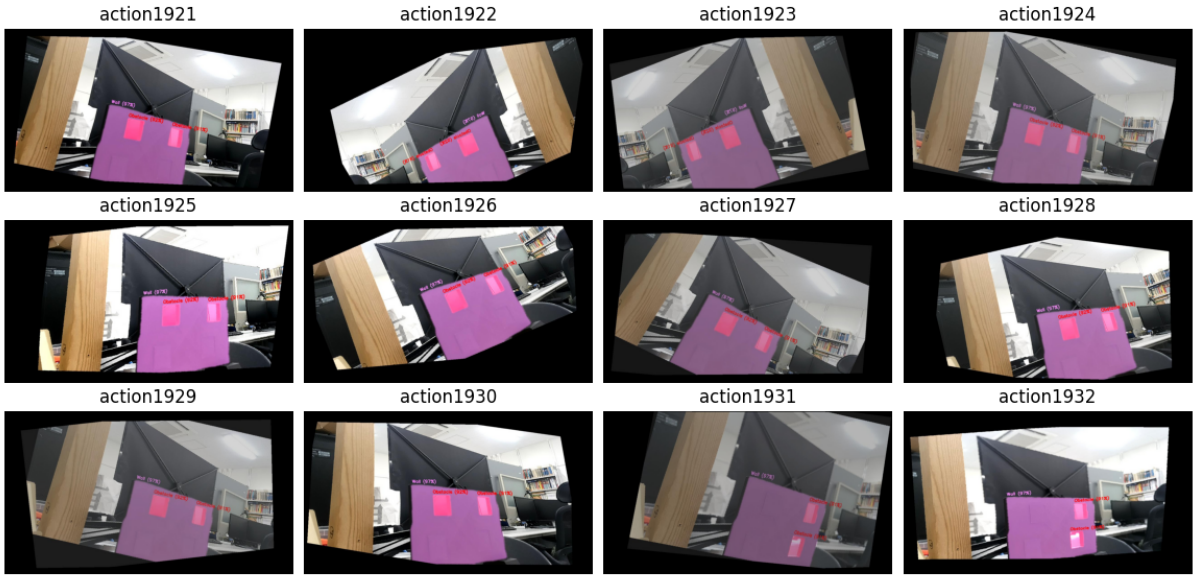

Figure 2: Augmentation Images

- **Data Components**: Each data point, referred to as an action, consists of two main elements:
  - **Image**: A 320 x 180 pixel image captured during the plastering task like Figure 1 and 2.
  - **Information (Figure 3)**: Stored as a JSON file, it includes:
    - **Position of the Wall (Figure 4)**: The relative position of the wall to the robot arm.
    - **Sequence of Moves**: The robot arm's movement sequence to be predicted consists of 14 steps (time steps), with each step having 4 angles, one for each joint of the robot arm. If the sequence is shorter than 14 steps, I will pad the remaining steps with the values [-1, -1, -1, -1]. Additionally, a pad mask will be used to identify the padded regions, ensuring that these padded values do not affect the training process. The pad mask will indicate which parts of the sequence are valid and which are padding, allowing the model to ignore the padding during training.

For the custom model, the input will consist of both the image and the position data of the wall. The model will use the image to analyze the plastering task and the wall position to predict the robot arm's sequence of moves.

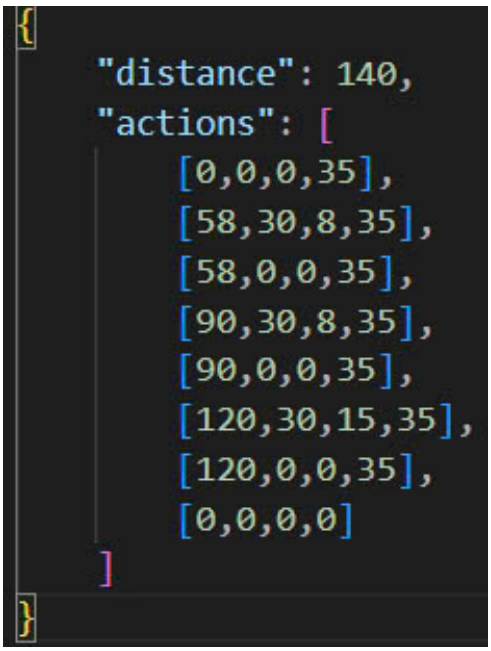

Figure 3: Json Files Information

In this dataset, only the wall facing the robot arm is considered, as shown in Figure 4 below. However, the dataset can be easily expanded to include the full coordinate positions of the wall relative to the robot arm, allowing for more comprehensive analysis and modeling.

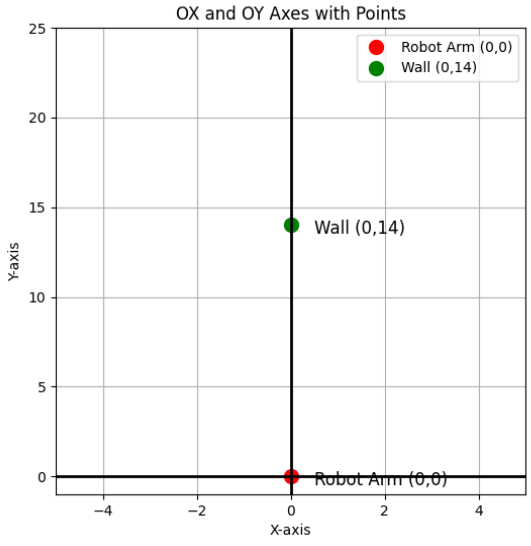

Figure 4: Position of the Wall related to the robot arm

2 Model Architecture

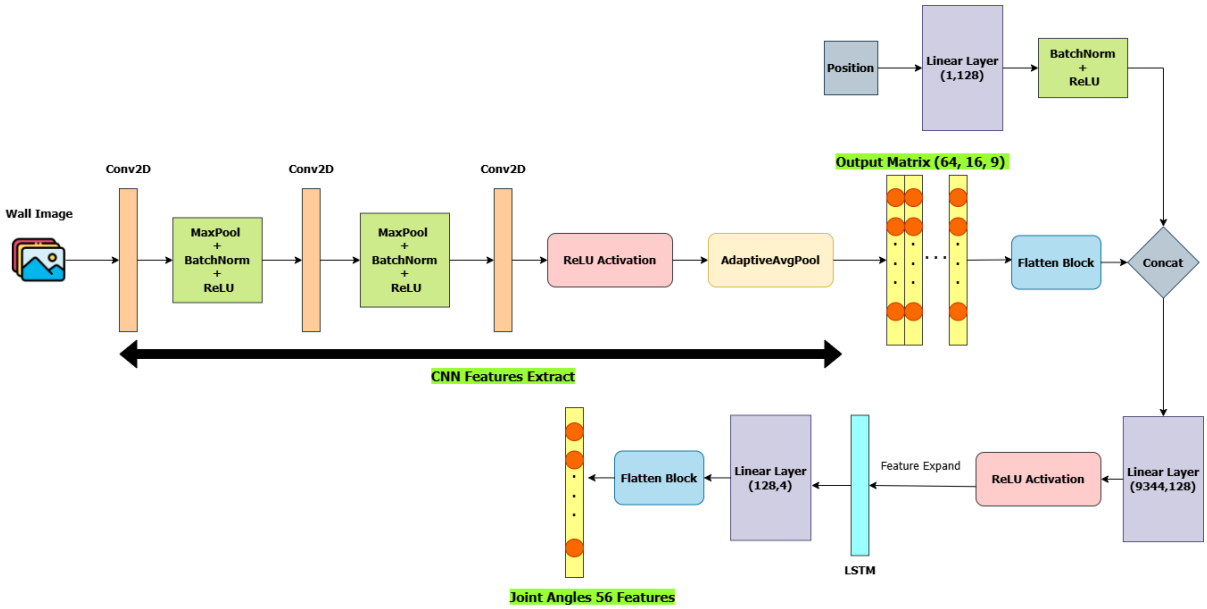

Figure 5: Model Architecture

2.1 Input Components

- Image Input: RGB images of shape (batch_size, 3, 320, 180).
- Position Input: A single scalar value per sample (batch_size, 1).

2.2 Features Extraction

- Convolutional Neural Network (CNN):
  - Three convolutional layers extract spatial features from the segmented image.
  - Each layer uses Batch Normalization and ReLU activation.
  - The final output is reduced to a fixed-size representation via Adaptive Average Pooling, resulting in a feature map of shape (64, 16, 9).
- Distance Feature Encoder:
  - A fully connected (FC) layer processes the distance input, expanding it to a 128-dimensional representation.
- Feature Projection:
  - The extracted CNN features ($64 \times 16 \times 9 = 9216$) and distance features (128) are concatenated, forming a 9344-dimensional input

  - This is passed through another fully connected layer with ReLU activation, mapping it to a 128-dimensional hidden state.

2.3 Sequence Modeling with LSTM

- The extracted feature vector is expanded to match the sequence length (max_seq_len = 14) by repeating it across the time dimension.
- A 2-layer LSTM processes this sequence, capturing temporal dependencies in the robot's movement.

2.4 Output Layer

- The LSTM's hidden states are passed through a fully connected layer, predicting an action vector (action_dim = 4) for each timestep.
- The output has the shape (batch_size, max_seq_len, action_dim)
- Finally, it is flattened into (batch_size, action_dim $\times$ max_seq_len = 56) for easier processing.

3 Training and Evaluation

3.1 Training Phase

For training, the model was set up with the following parameters:

- Epochs: 20
- Batch size: 32
- Learning Rate: 1e-4
- Weight Decay: 1e-4
- Optimizer: AdamW
- Loss Function: L1 Loss

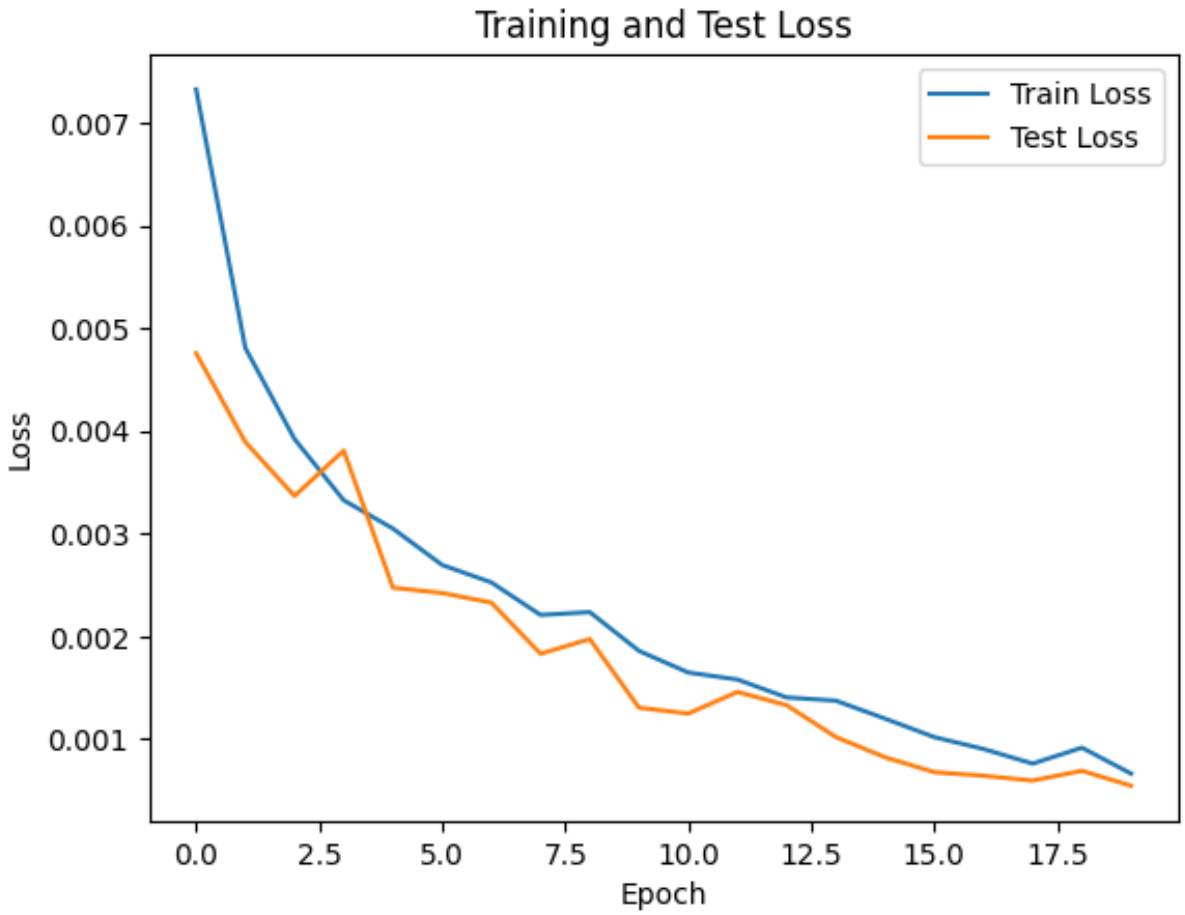

Figure 6: Loss Diagram

The Training and Test Loss diagram (Figure 6) shows how the model learned to predict joint angles for the robot arm over 20 epochs. The x-axis is the number of epochs (0 to 20), and the y-axis is the loss (error), which goes from 0 to 0.07. The blue line is the train loss (error on training data), and the orange line is the test loss (error on new data). Both lines start high (around 0.06–0.07) and drop to about 0.01 by epoch 20, which means the model got better at predicting angles—the errors got smaller as it trained. The train and test loss are close, so the model works well on new data, not just the training data.

3.2 Inference Phase

This diagram shows how a robot arm uses OAK-1 camera to see its workspace, including walls and objects. The camera takes a picture, and a tool called YOLO finds the objects in it. The picture and position information go to a custom CNN Model, which decides how the robot arm should move. The model makes a list of steps, and the robot arm follows these steps to plaster the wall.

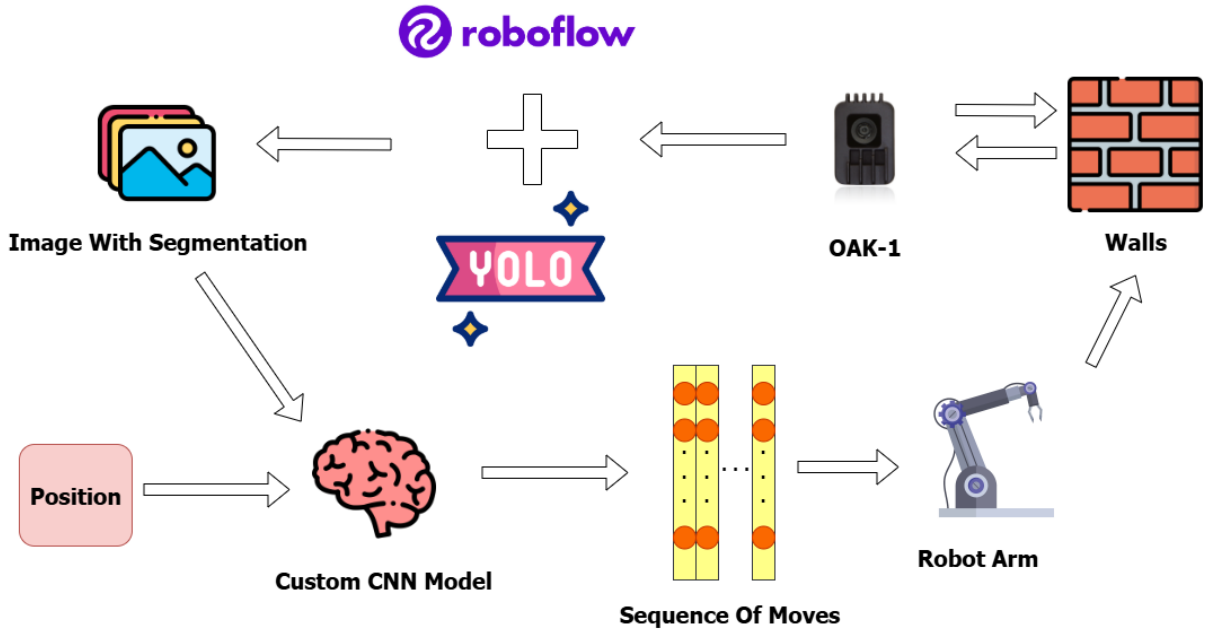

Figure 7: Inference Diagram

Video Link:

- Full Wall: https://drive.google.com/video1
- Part of Wall: https://drive.google.com/video2

3.3 Evaluation Phase

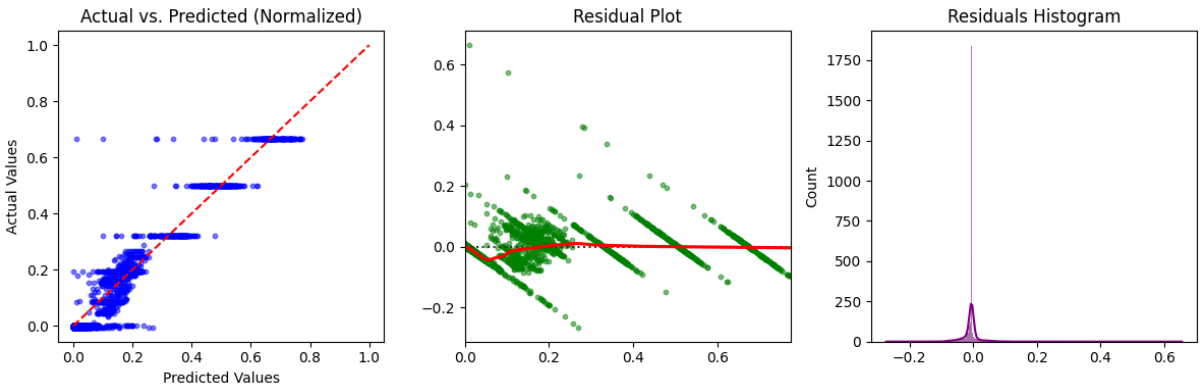

Figure 8: Evaluation Diagram

- R^2 Value: 0.9575

  This suggests that the model explains 95.75% of the variance in the data, confirming its strong predictive power.

- Actual vs. Predicted Plot (Normalized)

- Each dot represents one joint angle out of the 5600 total angles (100 samples $\times$ 14 steps $\times$ 4 angles)
- The x-axis is the predicted angle (what the model guessed), and the y-axis is the real angle (the correct value)
- There's a red dashed line from (0,0) to (1,1). If a dot is on this line, it means the prediction was perfect (predicted angle = real angle)
- The points are mostly clustered near the diagonal line, which is a good sign that the NN's predictions are accurate. But at the low end (near 0) and high end (near 1), some dots are farther from the line. This means model sometimes makes mistakes when predicting very small or very big values.

- Residual Plot:
  - Residuals are the differences between the real values and the predicted values.
  - The x-axis shows the values the model predicted. Each green dot represents one of the 5600 joint angles.
  - The y-axis shows the residuals, which are the errors: Real Angle - Predicted Angle. If the residual is 0, it means the prediction was perfect (real = predicted). If the residual is positive, it means the real angle was bigger than the predicted angle. If the residual is negative, it means the real angle was smaller than the predicted angle.
  - The red line is a smooth curve that shows the average trend of the residuals as the predicted values change. Ideally, this line should be flat and close to 0, meaning the errors don't change in a pattern—they're random.
  - The sloping red line shows that the CNN model's errors follow a pattern. It under-predicts for smaller angles (residuals are positive near x = 0) and over-predicts for bigger angles (residuals are negative near x = 0.4–0.6). This means the model is making consistent mistakes depending on the size of the predicted angle. For the robot arm, this might mean it moves too little when it should make a small movement, and too much when it should make a bigger movement.
  - Outliner: Outliers are residuals that are much bigger (or smaller) than most of the others, meaning the model's prediction was way off for those joint angles.

The dots outside the red dashed lines show that for some joint angles, the model makes really big mistakes. This might cause the robot arm to move to the wrong position in those cases.

- Residuals Histogram:
  - A residuals histogram is a plot that shows the distribution of the model's errors
  - The x-axis shows the residuals, which are the errors: Real Angle - Predicted Angle.
  - The y-axis shows how many residuals fall into each range on the x-axis.
  - The purple bars show how many residuals (errors) fall into different ranges. Each bar covers a small range of residuals (like -0.2 to -0.15, -0.15 to -0.1, etc.). The purple line is a smooth curve that shows the overall shape of the residuals' distribution. If the line is high, it means there are a lot of residuals in that range.
  - There's a big peak near 0, which means a lot of predictions have very small errors (close to 0). But there's a long tail going out to 0.6 on the positive side, meaning there are quite a few predictions where the model under-predicted (real angles were bigger than predicted).

4 Conclusion

The model achieved high accuracy but struggled with outliers. The next step is to improve its robustness, focusing on handling these outliers more effectively.

References