Report

February 17, 2025

1 Homework 2 Report

1.1 Code Explanation

1. In my solution, the facial keypoint detection is implemented using a custom PyTorch dataset class that applies Albumentations for simultaneous image and keypoint augmentations, ensuring consistent geometric transformations and normalization (dividing keypoints by 96 to scale into [0, 1]). A CNN architecture featuring convolutional layers with batch normalization, dropout for regularization, and a sigmoid activation at the output was chosen to constrain predictions to the normalized range. Specific design choices included normalizing both images and keypoints, handling missing keypoints with a custom masked MSE loss, and using augmentations that, while enhancing generalization, introduced challenges in training stability and required careful tuning to avoid extreme loss values.
2. For external sources that were used was ChatGPT for any errors that I was not able to debug. I also heavily used ChatGPT for debugging and understanding the Albumentations library.

1.2 Discussion

1. Print out the shape, dytpe, min, and max of the arrays and explain them in your report.
   - Images: (7049, 1, 96, 96) int64 0 255
   - Keypoints: (7049, 30) float32 0.686592 95.935646
   - The images values means that there are 7049 grayscale images because of the 1 channel. They each have a dimension of 96x96 pixels and their pixel values range for 0 to 255. The keypoints arrays is 30 values corresponding to 15 (x,y) keypoint pairs. The minimum and maximum values for the keypoints are 0.69 and 95.94 which in line with the image dimensions. the keupoints should be in between the boundaries of the image.
2. Compare your original and improved models in terms of test performance and overfitting.
   - The original model had a average loss of 0.0014 and its test performance was 2.80 for the rmse. This means that the original model was doing quite well in that it was only off by 3% pixel wise. However, due to the complexity increase and also the augmentations that were added to the "improved" model, its loss became a little high at 0.0301 with a test performance of 5.18. Due to the normalization, there is big trade-off between generalization and low training error. This model also might be underfitting due to not being trained long enough since the first model might have overfitted the training data, showing low errors that are not generalizable to the test data.