GitHub Link: https://github.com/mhdatheek136/age_classifier_project

Summery

I didn't really simplify anything, and I kept the original image resolutions without resizing. It was a bit challenging to work in PyTorch since I only had experience with TensorFlow before. I aimed to make the model complex by adding multiple convolutional and fully connected layers, and I experimented with different learning rates. However, I struggled to achieve good accuracy. While the loss was consistently decreasing, the accuracy didn't improve much, which I believe is mainly due to the lack of data and the subtle changes in facial wrinkles across different age categories could not be captured by the model.

a. Model Architecture

I used a CNN with fully connected layers at the end because it's a standard approach for image training. My model has three convolutional layers with max pooling and ReLU activation, followed by three fully connected layers, ending with a softmax layer. I also tried a simpler version with two convolutional and two fully connected layers, but there was no improvement.

b. Classification Accuracy

The accuracy achieved was around 1.429. I used the Adam optimizer with learning rates of 0.001 and 0.01, but both produced similar results.

c. What Worked Well

Honestly, I can't pinpoint anything that worked well. The accuracy was just a bit above random guessing (0.1).

d. What Did Not Work Well

Due to memory limitations, I had to split the data into two batches of size 35. Despite trying various hyperparameter changes, the accuracy didn't budge from around 1.429.

e. Improving Results

A major improvement would come from collecting more data. Additionally, enhancing the images to better highlight wrinkles and other facial features that change slightly with age could help.

f. Using the Dataset for Age Transformation

If I were to train a model to make a person look older or younger, I'd definitely focus on gathering more diverse data with various backgrounds. Assuming I had collected as much data as possible, I think using a GAN could be an effective approach for this task.