Module Title: Deep Learning
Module Code: CS4S772

Student Name: Salma Javid
Enrollment ID: 30107961

Predicting Age from Image — A Deep Learning Approach

[Web Deployment with Anvil's Interface]

I. Introduction

Problem Statement: Develop a deep learning model that predicts a person's age from an RGB image, and deploy it using Anvil's web interface.

II. Dataset Description: Insights & Revelations

Inspecting the Image Folder: The dataset schema is images of Indian cinema actors and actresses belonging to Bollywood, Tollywood, etc. It includes the vintage types and the current ones as well.
Total count of images: 19,906
Color Diversity in images: Grayscale and color
Dimensions: Range from (as small as) 8 x 11 to (as large as) 543 x 616
Additional Information: Contains a few duplicates.
[eg: ID: 19605 and 20597 are similar]
Classes (refers to the age group): 3 — Young, Middle, Old.

III. Checking the CSV File

Count of rows/records: 19,907 (including headers)
Count of Columns: 02 [ID, Class]
Unique Classes: 'MIDDLE' 'YOUNG' 'OLD'
Note: There is no certain order of the IDs and Class.

Data Segmentation — Count of Class feature
Middle: 10804 | Young: 6706 | Old: 2396

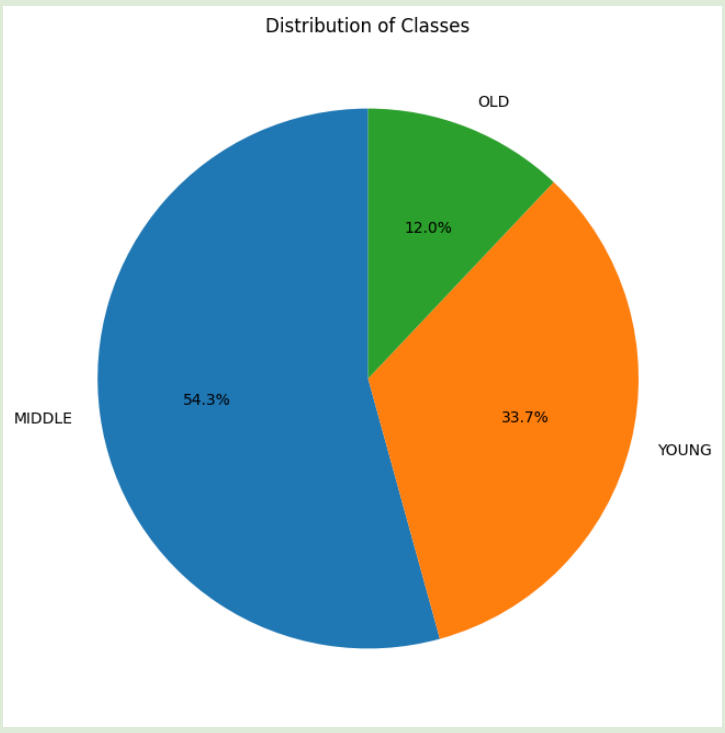
Testing Two Sorting Scenarios — Observation:

1. Sort the IDs in ascending order (checking if it arranges the age in ascending order too).
   Result: No certain order found in age. It is still a mix of young, middle, and old age class.
2. Sort the age (Class) in alphabetical order.
   Result: No particular order in ID listings found.

Conclusion: Either the data is manipulated or we are provided only selected records from a large database. On the bright side, the labels corresponding to images are correctly classed. ***Phew!***

IV. Data Visualization

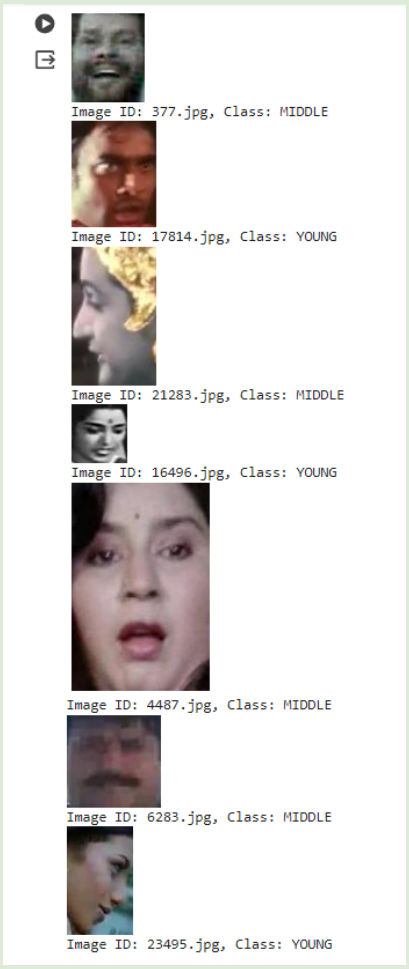
Distribution of the Class feature: Let's plot a Pie chart displaying all classes (Age Groups): Young, Middle and Old.



A Brief Explanation of the Pie Chart:
The Middle age group (or class) represented in **Blue** is the largest group with approximately 54.3% of the dataset. The Young group represented in **Orange** color makes up around 33.7% of the dataset. While the oldies (represented in **Green**) are the smallest set, accounting for only 12% of the dataset.

Displaying Images with ID & Class

BEFORE PREPROCESSING



Result: Each image is of a different size and scale.

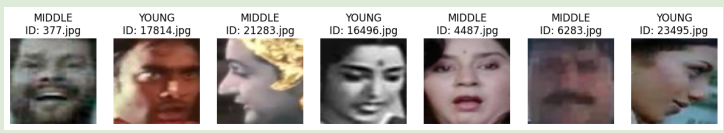
V. Data Preprocessing & Augmentation

Resizing of all images is done to 64 x 64 — a consistent size ensures uniform input and small size gets **results quickly**.

Rescaling: Normalization or Standardization of pixel values to a standard scale range between 0 and 1 is applied to the dataset to ensure a **consistent scale**.

Data Augmentation: Using ImageDataGenerator for augmentation techniques like image rotation, horizontal and vertical shifts, shearing, zooming, flipping, etc. to generate additional training samples, **improving the model's robustness and generalization.**

AFTER PREPROCESSING



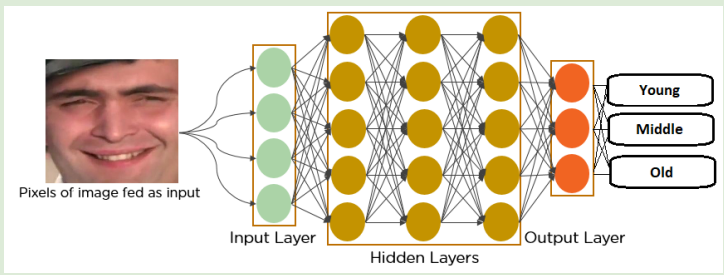
Result: Now all images are of the same size and scale.

VI. Data Splitting — Model Training & Testing

I'm splitting the dataset into a **70:30** Ratio.
70% training and 30% testing set
Number of training images: 13934
Number of testing images: 5972

VII. Convolutional Neural Network (CNN)

Rationale: I am employing Convolutional Neural Networks [CNNs] as it excels at learning hierarchical and spatial representations from image data, making it well-suited for age prediction tasks. Their capacity to capture facial features, hierarchies, and global context, coupled with translation invariance and data efficiency, positions CNNs as powerful tools for accurately predicting age from diverse image datasets.



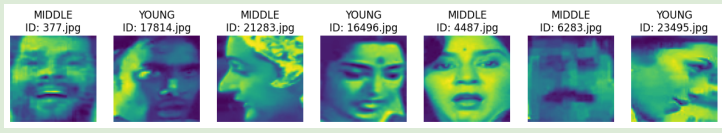
Training The Model with 10 Epochs

Results:
Training Set Accuracy: 59.42%
Testing Set Accuracy: 60.70%

Converting all Images to GrayScale

(*The intention is to increase accuracy*)

- Grayscale images need fewer computational resources
- They potentially speed up the training times
- Take less memory when compared to colored images



Model Trained on CNN — Results after 10 Epochs:
Training Set Accuracy: 57.15%
Testing Set Accuracy: 62.06%
Result: Almost the same as RGB images

Graph Displaying Model Performance

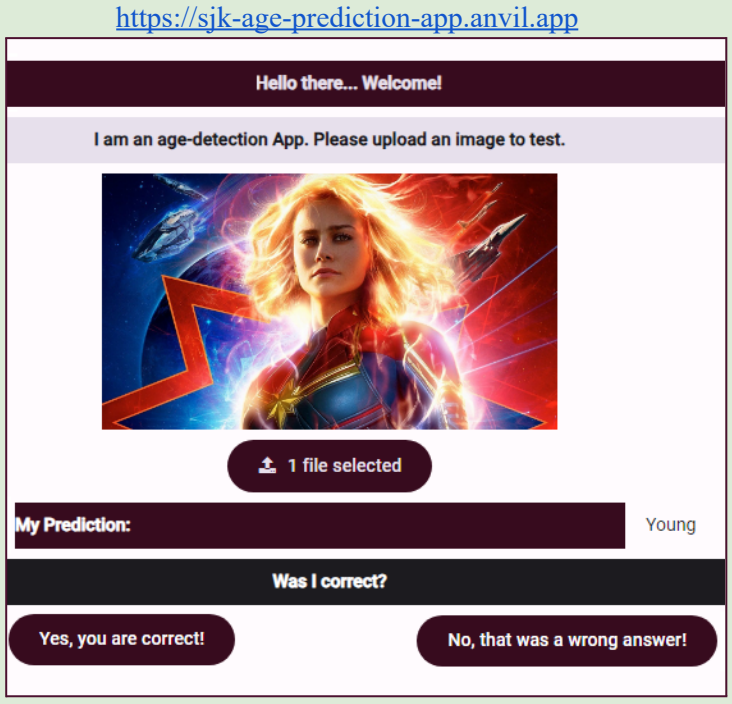
Visualizing training loss, testing loss, training accuracy, and testing accuracy results through a line graph

VIII. Anvil Web Deployment Process

Age Prediction App GUI Description: It begins with a welcome message. The second section introduces itself as an Age Detection App, requesting the user to upload an image using the 'file loader button' displayed below. Once the user uploads an image, it is displayed and an age prediction is made below it. The App also asks the user's input if it was correct in its prediction **(or not!)**.

SJK Age Prediction App

https://sjk-age-prediction-app.anvil.app



Future GUI Plans: To make an app which not only predicts the age of a person but also additional characteristics such as gender, skin colour, emotions, expressions, region, category, etc.

Challenges & Conclusion: The accuracy score of my model after 20 epochs is around 59-60%. This is probably leading the Anvil App into errors — The prediction is not always correct. Potential factors contributing to misclassifications could include variations in facial features, lighting conditions, and other complexities. Enhancements in dataset quality, model architecture, and further fine-tuning could be explored to boost accuracy.