COMP 6721 Applied Artificial Intelligence (Fall 2023)

Project: AI-Ducation Analytics

Part I-III

Team Name: AK_7

Team members' details:

Name	Student ID	Specialization
Gowtham Nalluri	40262135	Data Specialist
Protim Ghosh	40185075	Training Specialist
Md Sakib Ullah Sourav	40264066	Evaluation Specialist

Github link of the project:

https://github.com/Sakibsourav019/AI-Ducation-Analytics-COMP-6721---AK_7-

Contents

Introduction

In this project, we will classify four emotion classes named angry, bored, focused, and neutral. This emotion classification is based on the convolutional neural network (CNN) that can be used to monitor classroom feedback of students and other tasks. In the next parts of the report we described the processes that we adopted one by one that is required as instructed in the project description.

1. Dataset

We used two datasets for this project. The first primary dataset [1] we adopted for the purpose of accomplishing this project is taken from Kaggle. The original dataset is composed of images including two sections: training and validation. The entire dataset has seven classes, namely, angry, disgust, fear, happy, neutral, sad and surprise.

Another dataset [2] we used to create our custom dataset which is also taken from Kaggle.This dataset also has seven classes but different than in [1], namely, anger, contempt, disgust, fear, happy, sadness and surprise. But this dataset does not have any subparts like the first one. It only corresponds to 981 images in total for all seven classes.

As both of the datasets have limitations for certain image classes of our interest and to make our model more efficient, we plan to feed and train it with more diverse data. This is why we used a Python script to scrape images of our desired classes from the internet.

1.1. Overview of the datasets

The first dataset [1] is quite imbalanced when it comes to the number of images per class. From Figure 1 below, we can see that the class “disgust” has a very low number of images in comparison to other classes while “happy” has the highest number of images.

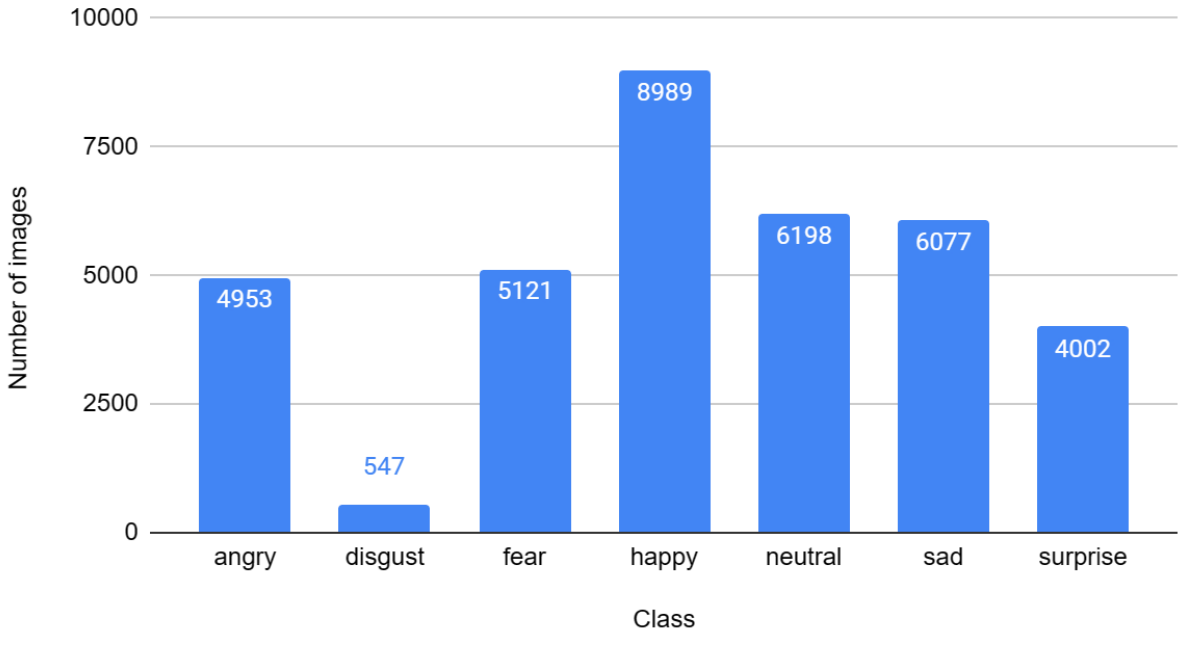

Figure 1: Number of Images vs Class in Dataset 1 [1].

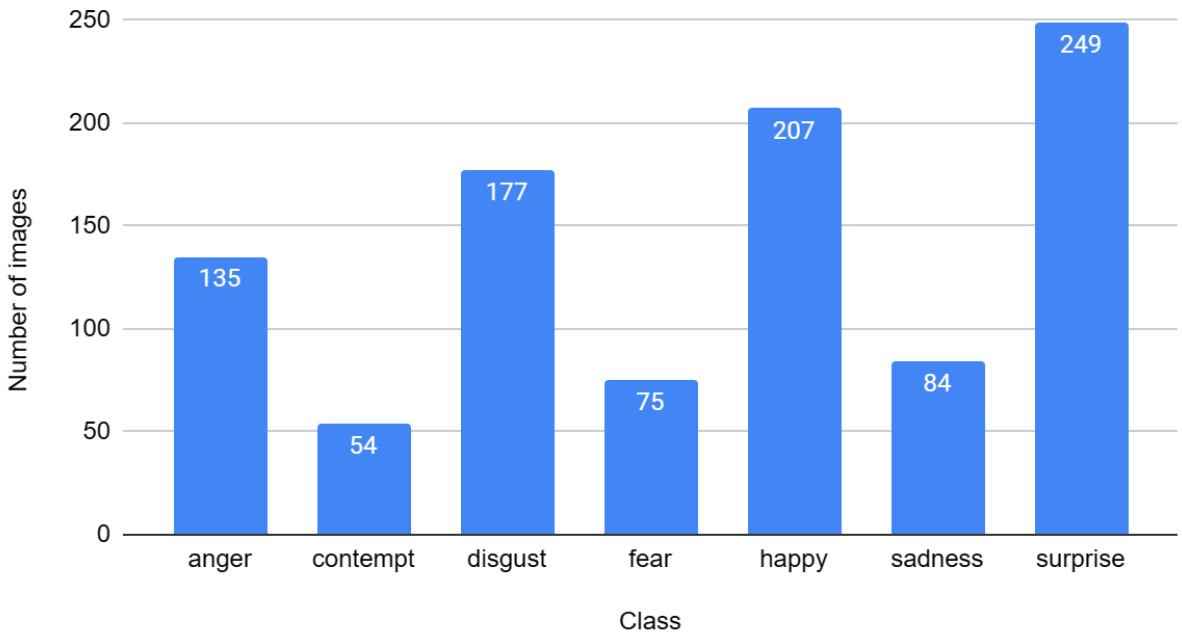

Figure 2: Number of Images vs Class in Dataset 2.

From the dataset 2, it can also be seen that the same trend as the dataset 1 is consistent here. Overall, we can conclude that both the datasets are quite imbalanced.

As mentioned above, we scraped images of all four classes of our interest from the internet using a script that uses Bing search engine to collect images. Thus we collected 39 “bored” images, 33 “neutral” images, 36 “engaged” images and 48 “angry” images as shown in Figure 3.

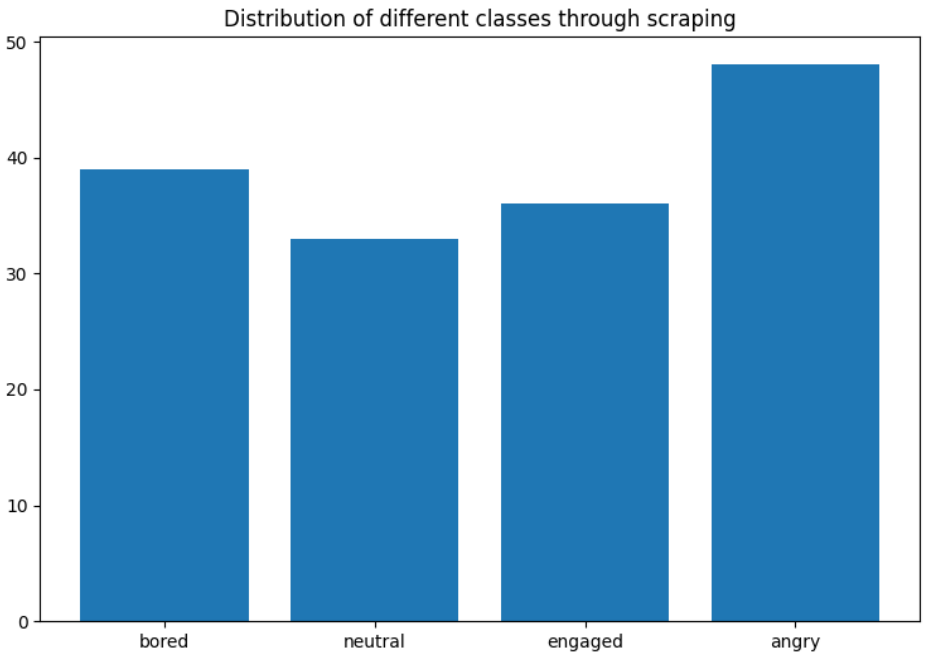

Figure 3: Number of Images vs Class through web scraping.

1.2 Justification for the Dataset Choice

While searching for the datasets, we came across various sources and options. Yet, the type of image data suitable for our task was not adequate. Moreover, some of the data classes we needed for this project have scarcity in the existing available datasets. Hence, we decided to scrape some images and merge them with the datasets so that we could construct our desired classes by combining those. All the data we adopted have frontal face shots with no visible scenic background which makes the datasets well-equipped for our desired goal.

1.3 Provenance information

Below in table 1, we list the informations about the datasets we used in this project-

Table 1: Overview of the two primary datasets that we adopted in this work

	Dataset 1	**Dataset 2**	**Web Scraping**
Name	CKPLUS	Face expression recognition dataset	Bing Search Engine Script
Number of images	35887 (Training - 28821, Validation - 7066)	981	156
Data classes	angry, disgust, fear, happy, neutral, sad, surprise.	anger, contempt, disgust, fear, happy, sadness and surprise	Neutral, engaged, bored, angry
Nature of images	Frontal face shots	Frontal face shots	Frontal face shots
Authors	Unknown	Unknown	Unknown
License	CC0: Public Domain	Unknown	NA
Links	https://www.kaggle.com/datasets/shawon10/ckplus/data	https://www.kaggle.com/datasets/jonathanoheix/face-expression-recognition-dataset	NA

In Table 2, we will get the overview of our custom-developed dataset for this project out of the two primary datasets mentioned above.

Table 2: Overview of the dataset we developed for this project

Number of images	**Image sources**	**Image classes**
Training - 1600 Testing - 600	1. Training - dataset 1 [1] 2. Validation - dataset 2 [2] 3. Web Images through scripts	Neutral, engaged/focused, bored/tired, angry/irritated

1.4 Changes made for Bias Analysis

For bias analysis in Part 3, we divided the dataset into five (5) sub-datasets, based on gender (male, female) and age (young, middle-aged and senior). In each sub-dataset, we kept the four (4) classes of our interest in a balanced manner so that each class had a similar number of images.

2. Data Cleaning

All the images in the datasets [1,2] are in grayscale color and in 224 x 224 size.To comply, we have resized the scraped images into 224 x 224 too.

Below, we can see a demonstration of an “angry” image that we scraped and resized the original rgb image into a 224 x 224 grayscale image.

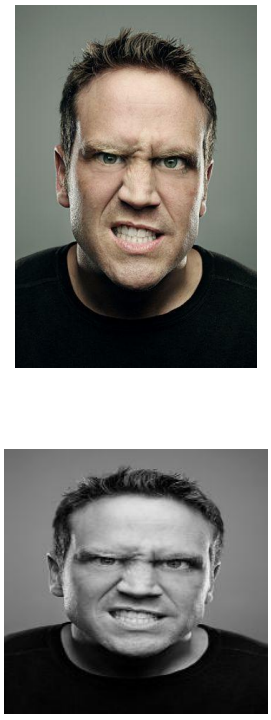

Figure 4. Before and after image resize operation.

Apart from that, the datasets we used as primary sources have more images in total than we need. We operated an in-depth manual eye-skimmed data cleaning process so that we could include most deserving images to the corresponding classes. Hence, we believe the model can get good training data and the model can perform better in the next parts of this project.

In part III, we confirmed all the images are in same resolution and same size that we did in previous part of the project.

3. Labeling

As we mentioned earlier, two datasets [1, 2] have been utilized to define the data classes we need. However, among the four data classes of our interests, two of them named “neutral” and “angry” are common in every dataset. So, we picked the best matched images for our model. But the other two classes “bored/tired” and engaged/focused” were unusual.

Hence, to label these two classes we skimmed all the images in the dataset and looked for other sources in the internet beyond to connect images that fall in these classes. Finally, we came up with a solution. As the solution, we firstly took help from the python script to download internet images using Bing Image Search through a series of written prompts. And secondly, we skim through the images from the dataset [1] to see which images are best fit to “bored/tired” and "engaged/focused” classes according to the criterias and features of expression mentioned in the project guideline. Thus we labeled the images to these two classes.

For part III, the two attributes for the bias analysis have been labeled with our highest concentration. All the members of this project worked together to confirm the reliability of the correct labeling. First, each member independently choose the images per attribute and then the other members confirm it. This way the bias attributes of the five sub-datasets has been confirmed.

4. Dataset Visualization

We used the python function matplotlib and seaborn to show the image data that we constructed for our task.

4.1 Class Distribution

We plotted the class distribution of our training and testing image data below in Figures 5 and 6. The number of training "bored" images is 439, the number of training "neutral" images is 433, the number of training "engaged" images is 436, and the number of training "angry" images is 448.

In the same way, the number of tests "bored", "neutral", "engaged" and "angry" images each are 150.

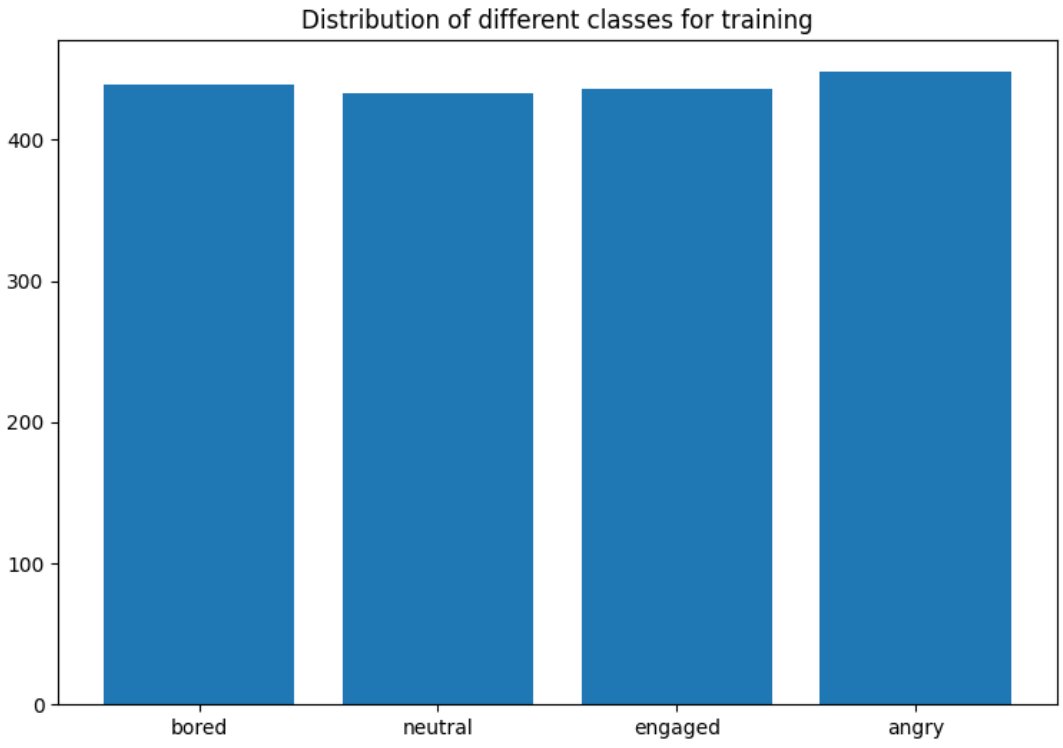

Figure 5. The training data images of four classes.

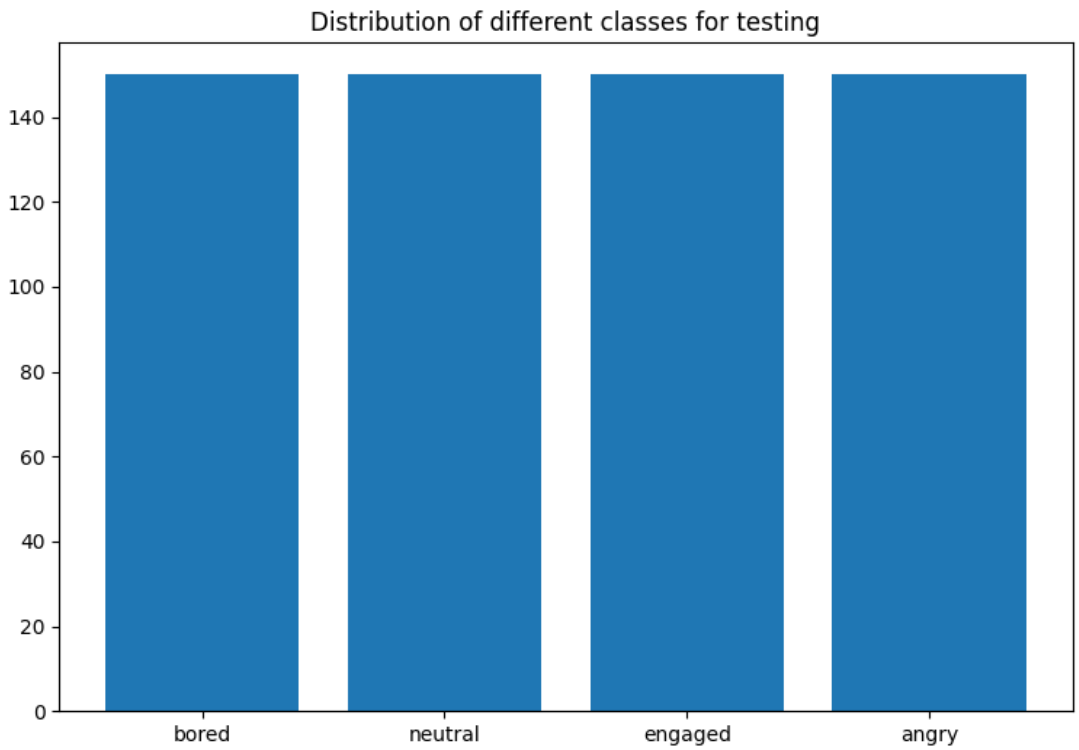

Figure 6. The testing data images of four classes.

For part III, the subsets has balanced distribution. Figure 8 shows the data images of four classes for Female attributes.

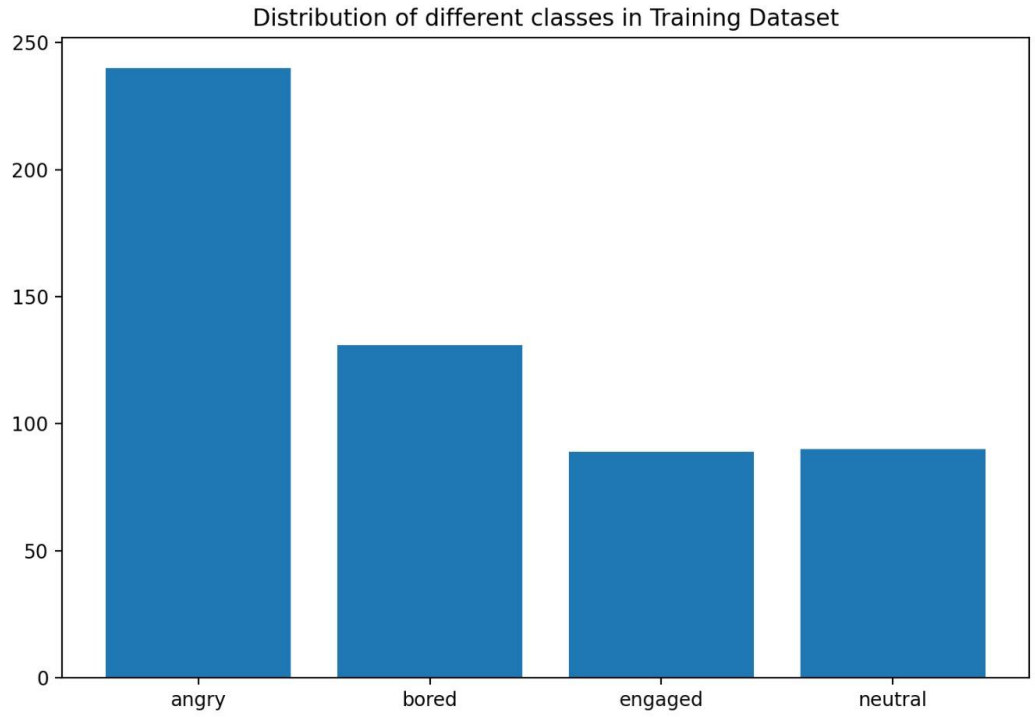

Figure 7. The testing data images of four classes on the Male sub-dataset while on the bias analysis.

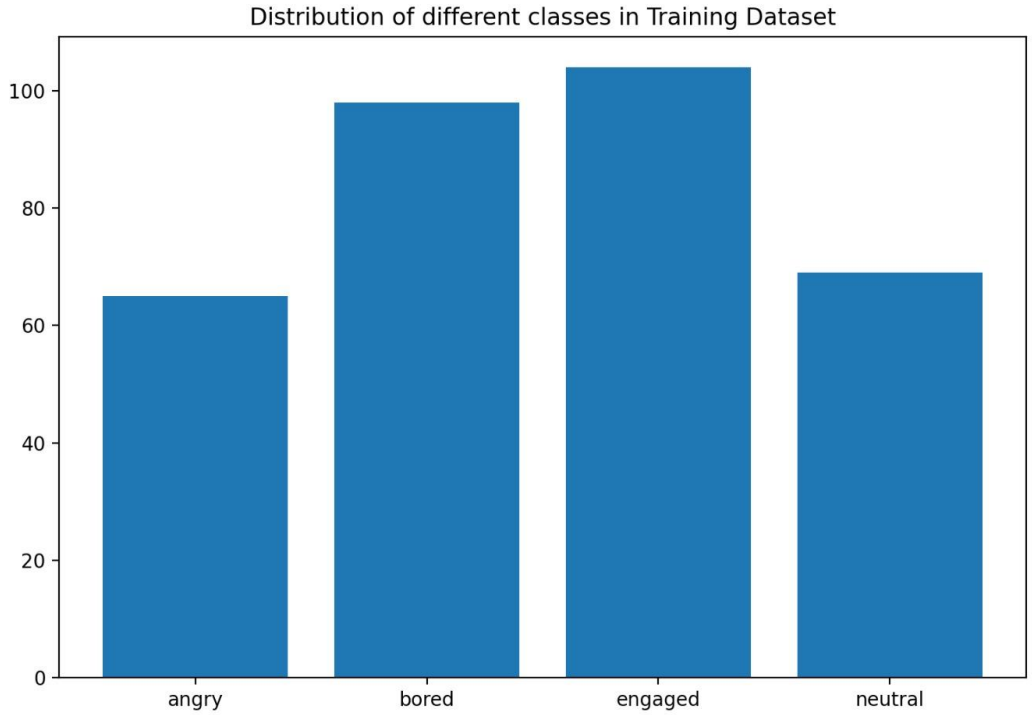

Figure 8. The testing data images of four classes on the Female sub-dataset while on the bias analysis.

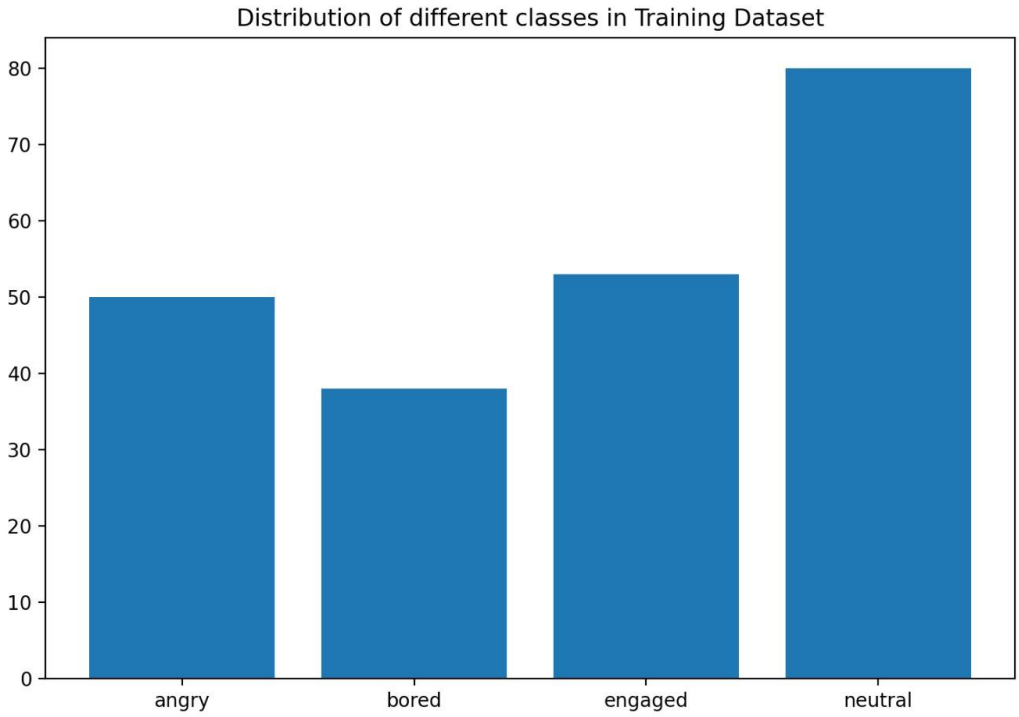

Figure 9. The testing data images of four classes on the Young sub-dataset while on the bias analysis.

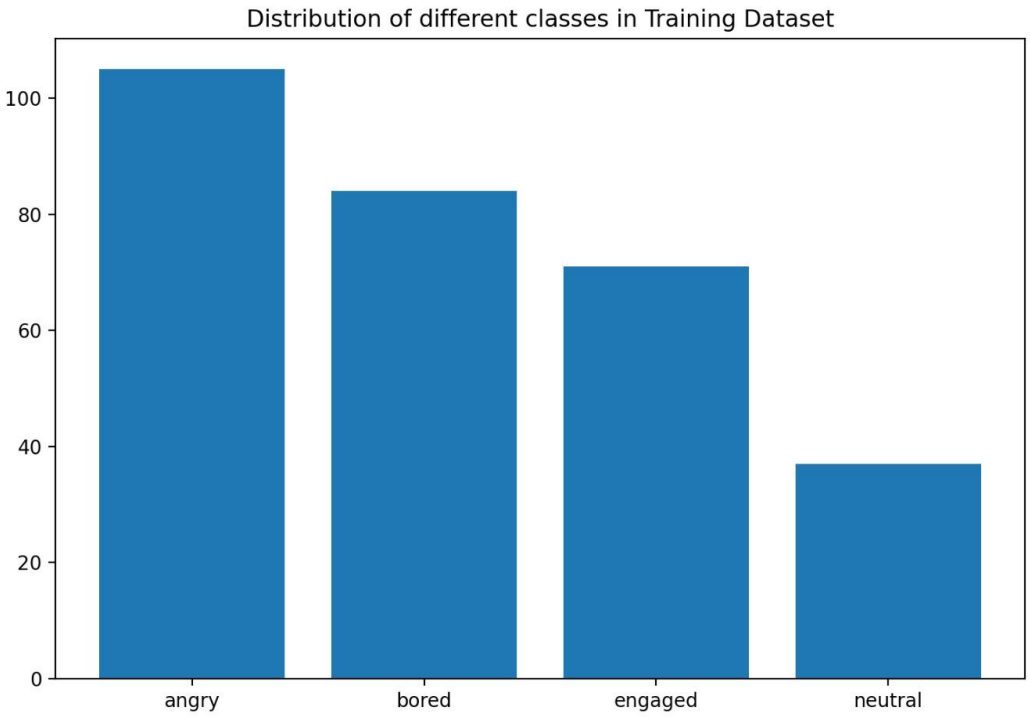

Figure 10. The testing data images of four classes on the middle-aged sub-dataset while on the bias analysis.

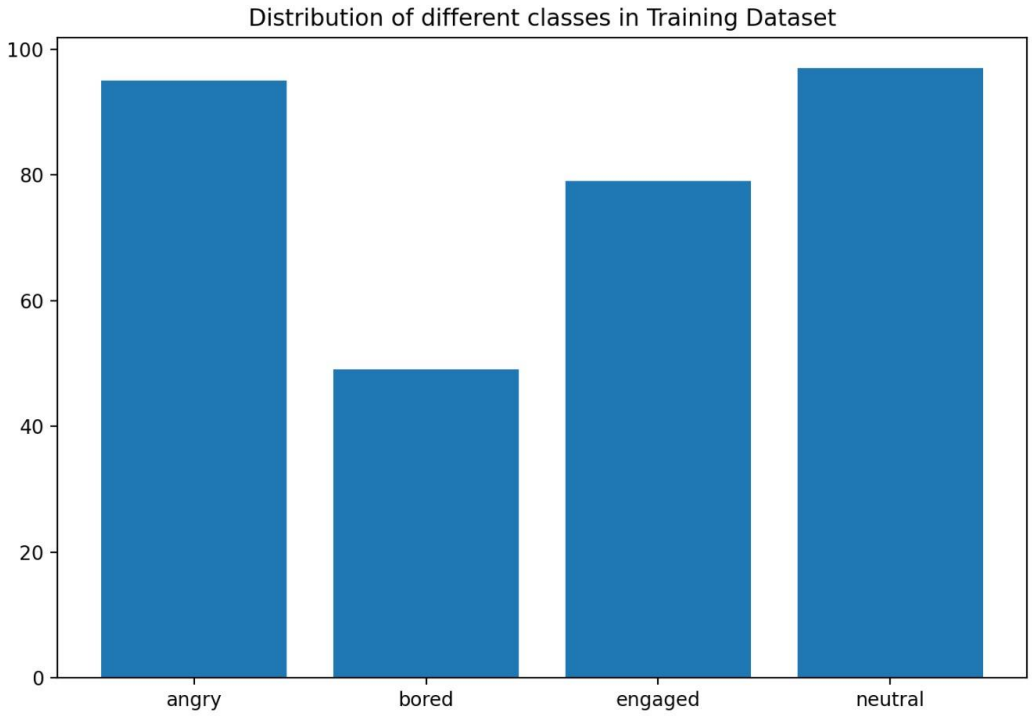

Figure 11. The testing data images of four classes on the senior sub-dataset while on the bias analysis.

4.2 Sample Images

As described and asked in the project guideline, below we listed the outputs of 5x5 grids, each image contains 25 images of four classes one after another.

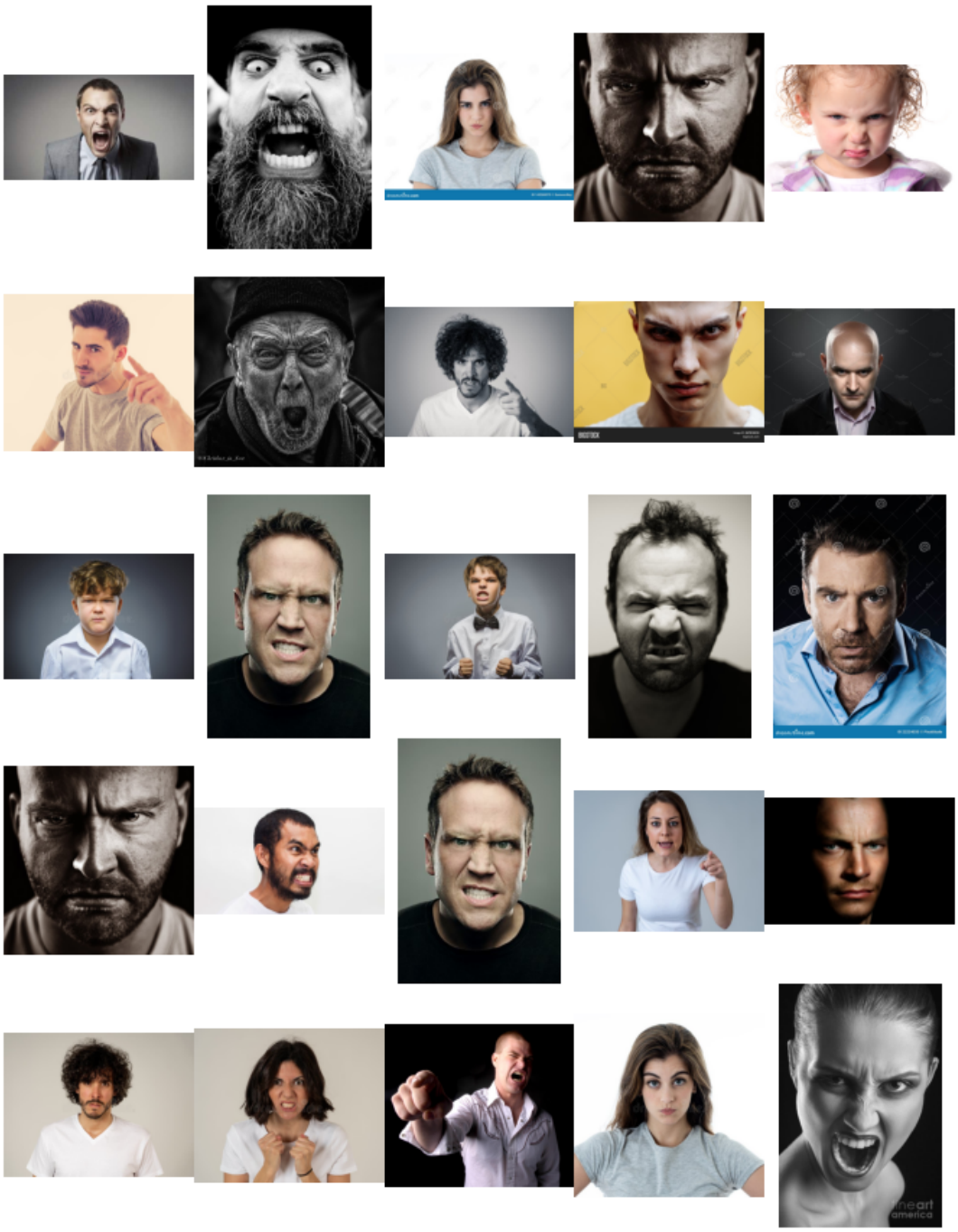

Figure 12. 25 “Angry” images from our dataset in a 5X5 grid.

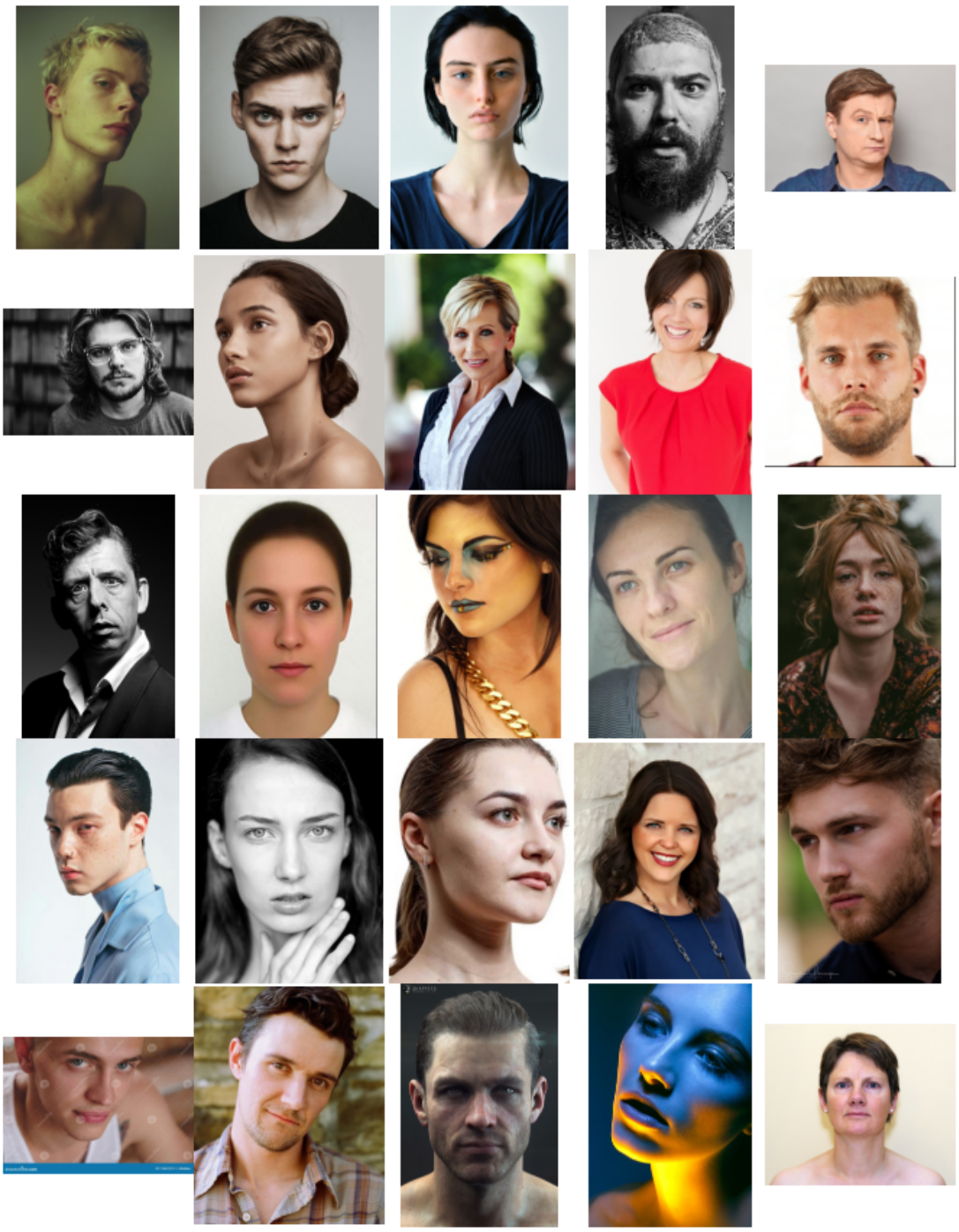

Figure 13. 25 “Engaged” images from our dataset in a 5X5 grid.

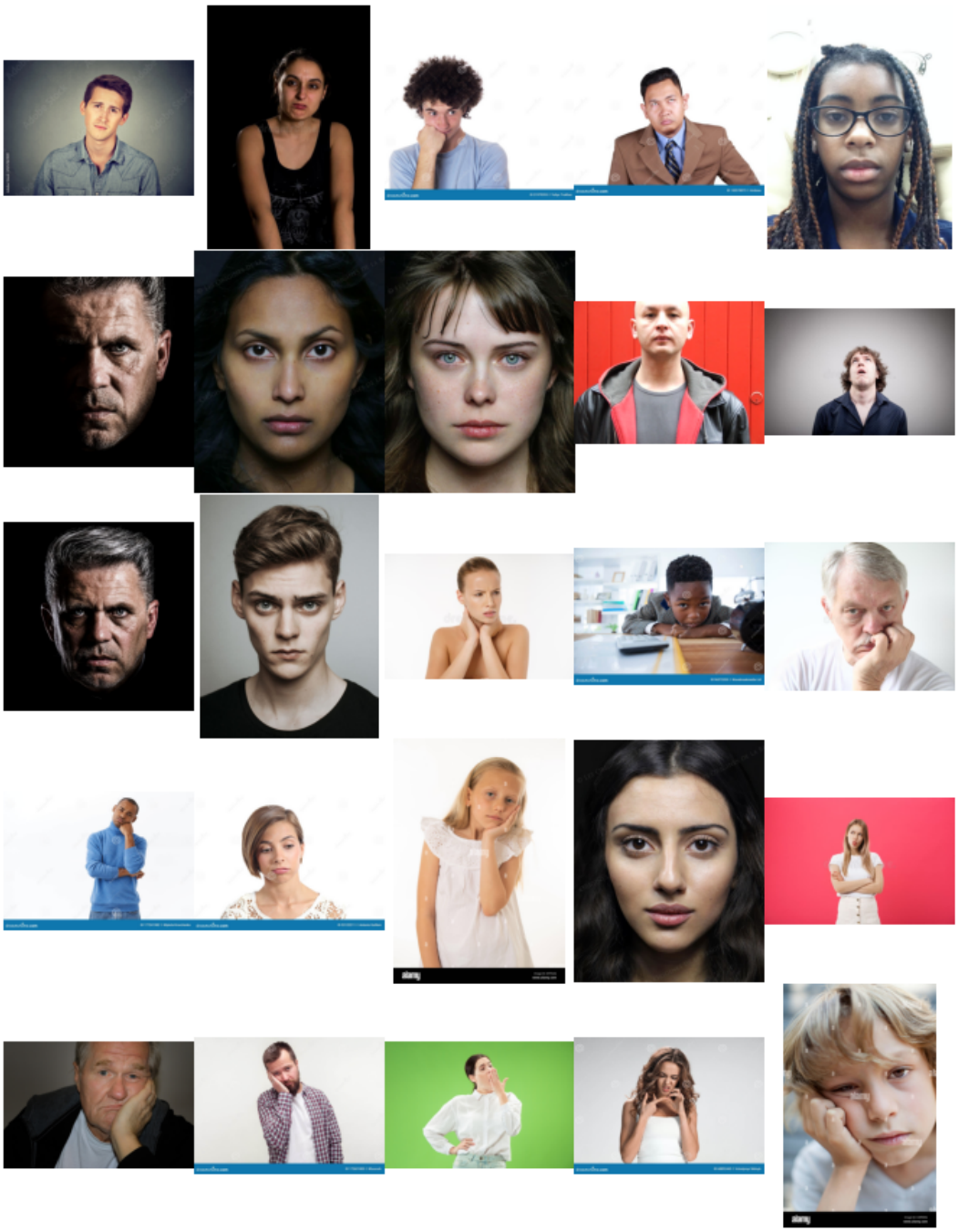

Figure 14. 25 “Bored” images from our dataset in a 5X5 grid.

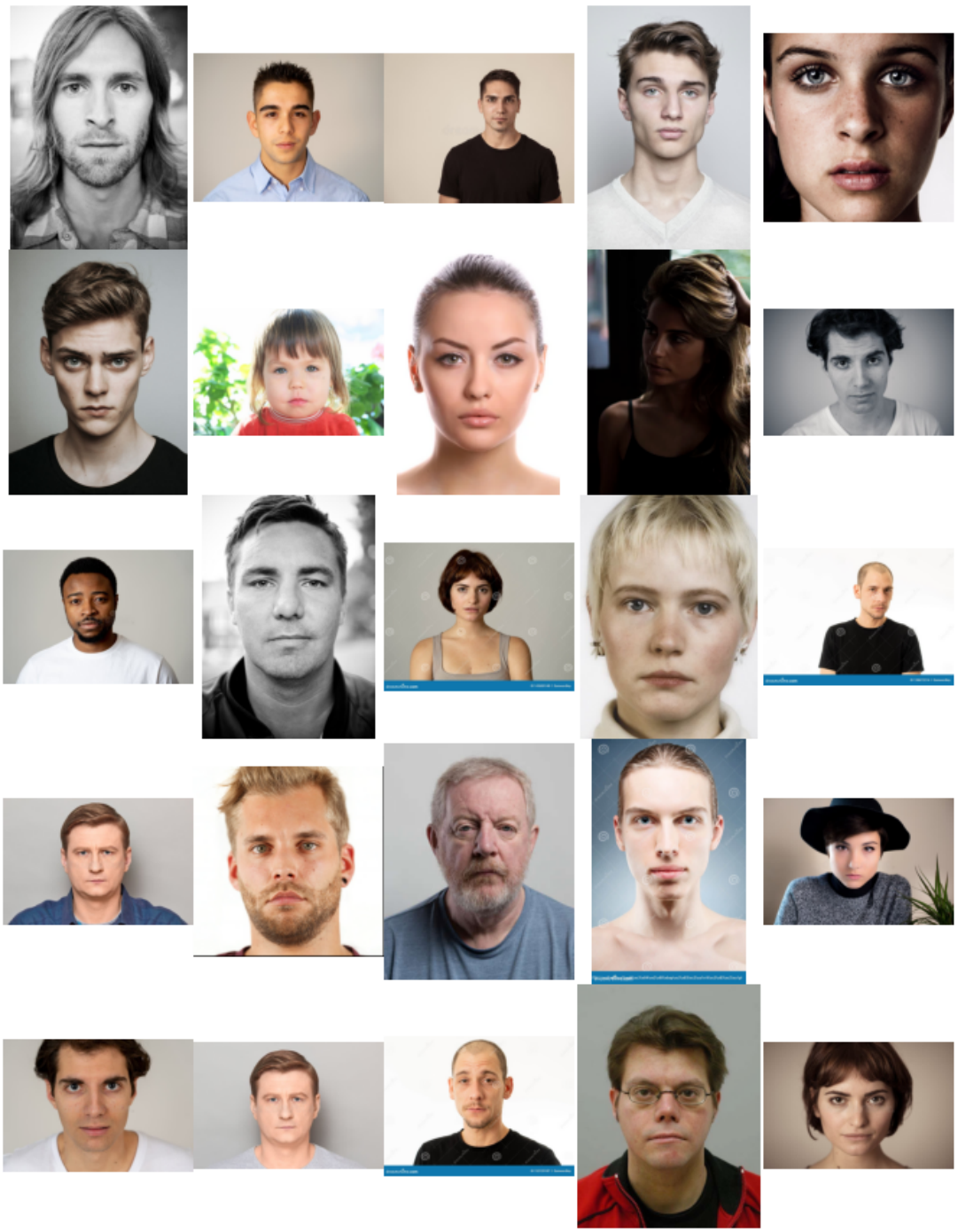

Figure 15. 25 “Neutral” images from our dataset in a 5X5 grid.

4.3 Pixel Intensity Distribution

Below, we have shown the plot that we get from the histogram that shows the pixel intensity distribution of the random images we have shown in section 4.2.

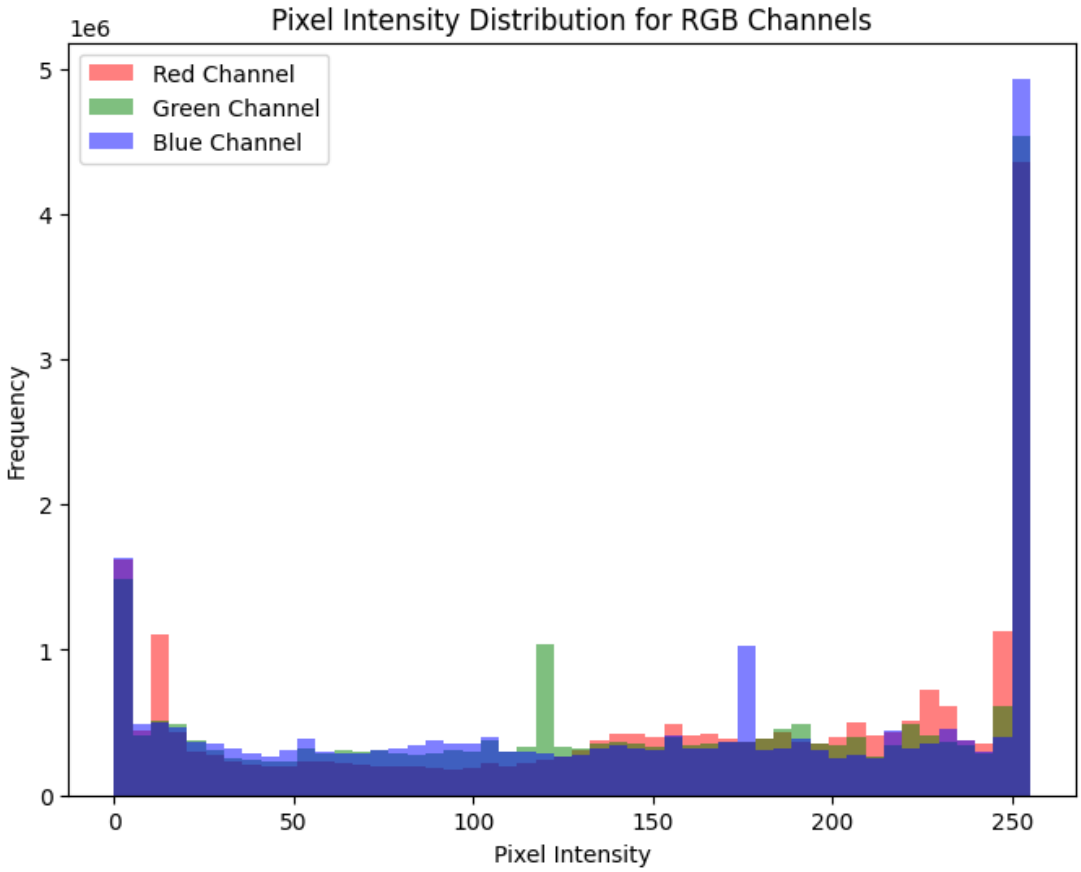

Figure 16. Histogram of random images in RGB color.

5. CNN Architecture

The description of our CNN architecture is listed below.

5.1 Model Overview and Architecture Details

We have implemented three models in total as described in the project description.

5.1.1 Main Model

The architectural composition entails a sequence of three consecutive convolutional layers distinguished by escalating filter depths (32, 64, and 128) and employing 3x3 kernel dimensions. Following each convolutional layer, spatial downscaling is effectuated through 2x2 max-pooling. The strategic integration of batch normalization postulates an optimization mechanism to augment training expediency and robustness. The Rectified Linear Unit (ReLU) is uniformly deployed as the activation function throughout the network, accentuating model non-linearity. In addressing potential overfitting, dropout regularization, with a probability of 0.5, is introduced subsequent to the inaugural fully connected layer. The conclusive classification is orchestrated through fully connected layers, featuring 512, 256, and 4 neurons, respectively.

5.1.2 Variant 1 and Variant 2

In variant 1, three convolutional layers with progressively greater filter depths (32, 64, and 128) and 3x3 kernel sizes make up the architecture. Max-pooling (2x2) is used after each convolutional layer to perform spatial downsampling. After every convolutional layer, batch normalization is used to speed up and stabilize training. The activation function used throughout the network is the Rectified Linear Unit (ReLU), which amplifies the non-linearity of the model. Dropout with a probability of 0.5 is added after the first completely linked layer to avoid overfitting. For the final categorization, the completely linked layers include 512, 256, and 4 neurons, respectively. We did not make any substantial changes in layers between variants 1 and 2 as when we were making changes in layer numbers and kernel size, the model got stuck at some point of training and took a much longer time to train the model.

5.2 Training Process

5.2.1 Main Variant

The training of the main variant is different from the two other variants. We use the Adam optimizer with a learning rate of 0.001, to compute the running average of the gradient and its square we use betas with the values 0.9 and 0.999, for numerical stability in the denominator we use le-8 as epsilon rate, with the weight decay of le-4. The training period of 40 epochs makes the accuracy higher than the other variants. Moreover, before feeding into the neural network for training we use a different transformation that helps to get more accuracy than others.

5.2.2 Variant 1 and Variant 2

During the training of variant 1, Stochastic gradient descent (SGD) optimization with cross-entropy loss as the objective function and a learning rate of 0.01 is used in the training procedure. The model is iteratively trained using batches of data from the training loader during the course of the 60 epoch training period. After every epoch, the validation performance is also assessed, providing metrics for a thorough evaluation that include loss, accuracy, recall, macro/micro precision, F1-score, and a confusion matrix.

In variation 2, we use the Adam optimizer with a learning rate of 0.001 instead of 0.01 in variant 1, replacing SGD with the same set of parameters and hyperparameters. In both variants 1 and 2, we use the cross-entropy loss function to quantify the difference between the true and predicted class labels. The optimizer updates the model weights to minimize the cross-entropy loss during the 60 epoch training procedure, which involves training the model using batches of data from the training loader.

6. Evaluation

6.1 Performance Metrics

Table 1: Evaluation of main model, variant 1 and variant 2

Model	Macro			Micro			Accuracy
	P	**R**	**F1**	**P**	**R**	**F1**	
Main Model	**83**	**83**	**83**	**83**	**83**	**83**	**84%**
Variant 1	45	46	44	44	44	44	44%
Variant 2	50	49	50	50	50	50	50%

The main model in Part III performs better on every assessment parameter. The accuracy of the model is 84%. In terms of other metrics, this model gets 0.805, 0.830, and 0.820 as precision, recall and F1 respectively. On the other hand, it gets 0.801 for every parameter of micro assessment.

Now, if compared with Model 1, Model 2 performs better on a variety of assessment parameters. Model 2 obtains a better accuracy of 50%, whereas Model 1 only manages an approximate accuracy of 44.17%. Additionally, Model 2 performs better than Model 1 in terms of macro accuracy, recall, and F1 scores (0.5063, 0.4972, and 0.4996, respectively) (0.4505, 0.4581, and 0.4451, respectively). Comparably, Model 2's micro accuracy, recall, and F1 scores (0.5 each) outperform Model 1's (0.4417 each). These results, which show increased accuracy as well as enhanced precision, recall, and F1 scores for both macro and micro averages, imply that Model 2 performs better overall in classification on the given face recognition test.

6.2 Confusion Matrix Analysis

Here to mention, in the confusion matrix, the corresponding classes represent the classes below-

Class 0: Angry

Class 1: Bored

Class 2: Engaged

Class 3: Neutral

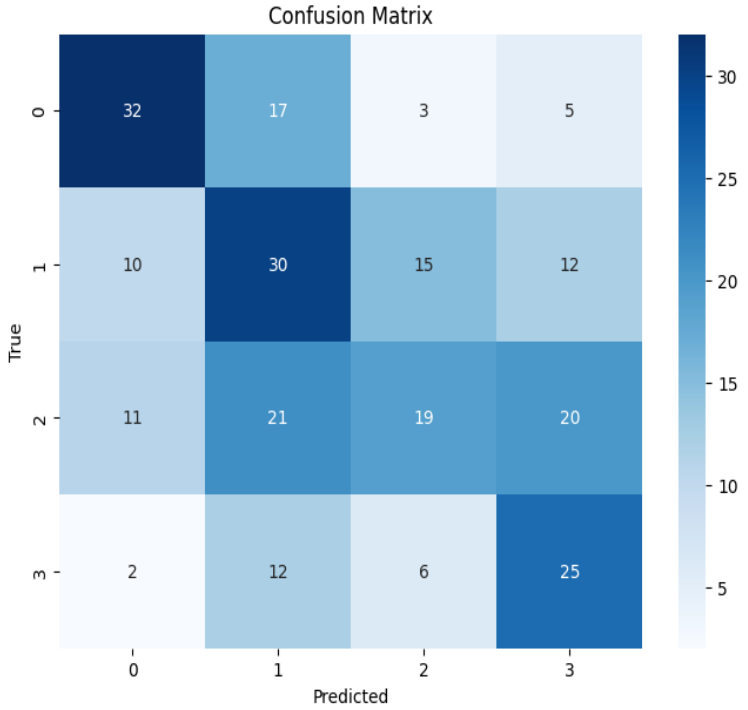

Figure 17. Confusion Matrix of Variant 1

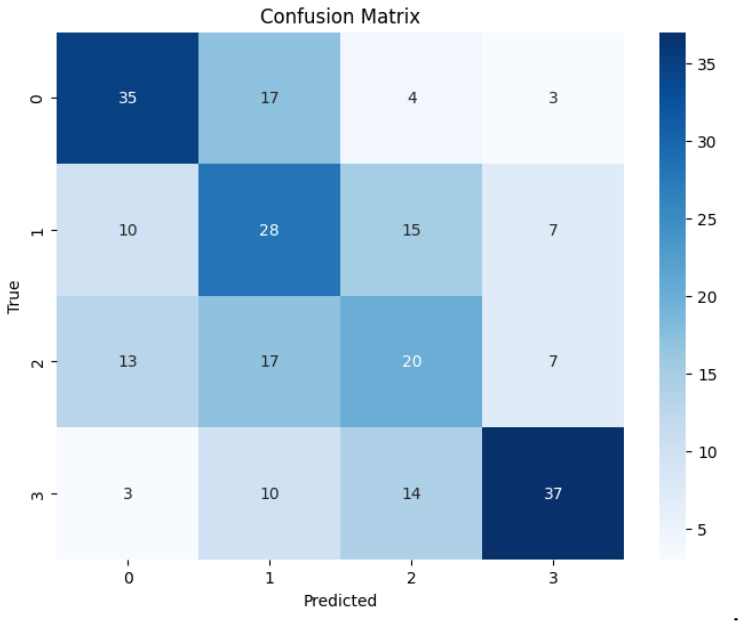

Figure 18. Confusion Matrix of Variant 2

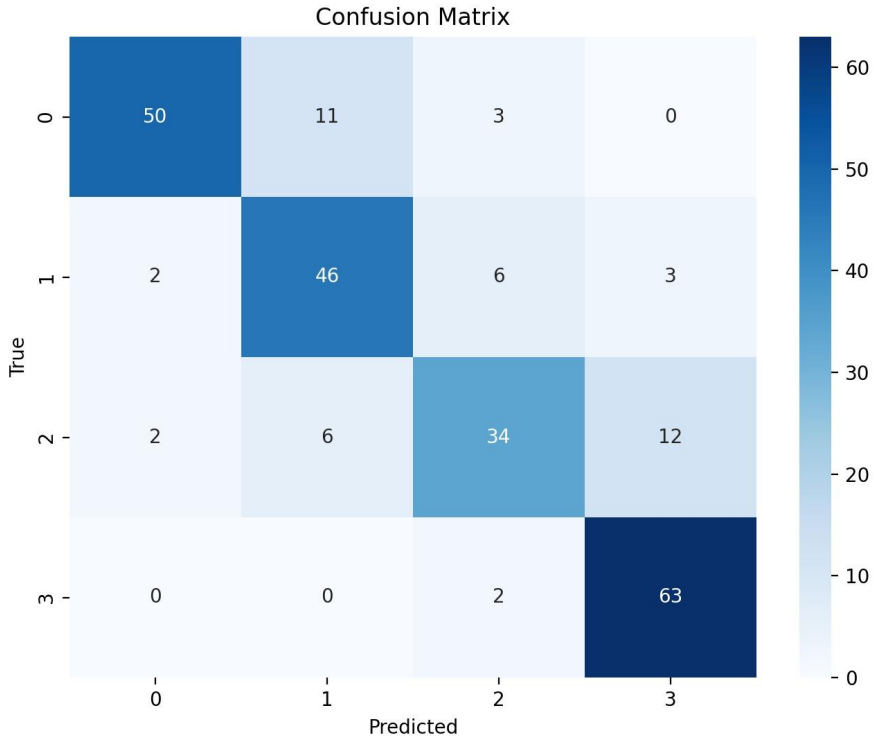

Figure 19. Confusion Matrix of the Main Model

From Figure 17, we can observe that in variant 1, Class 1 is frequently misclassified as Classes 2 and 3, and Class 2 is often confused with Classes 1 and 3.

From Figure 18, in variant 2, Class 1 is misclassified as Classes 2 and 3, and Class 2 is sometimes confused with Class 0.

Both variants struggle with distinguishing Class 1, as evident from misclassifications with other classes. Variant 1 often confuses Class 2 and Class 3. Variant 2 tends to misclassify Class 2, especially as Class 0. Both models perform relatively well in correctly classifying instances of Class 0 and Class 3.

In figure 19 on the final model on part III, we see the model can mostly classify among the correct classes that also validates the efficiency of the model.

6.3 Impact of Architectural Variations

When we increased the number of convolutional layers, it resulted in longer training times. Perhaps it was because of the vanishing gradient problem, in which learning is hampered by very small gradients during backpropagation. Though we have not faced any overfitting, it is a dominant risk factor for more complicated models, particularly when there is a shortage of training data. When a model learns noise from training data, it becomes overfitted and has poor generalization to fresh, untrained data.

In terms of kernel size, our machines got stuck when we tried to increase the kernel size. Hence, we limited the kernel size to 3 only. Large kernel-size deep networks could need more GPU RAM during training. Probably that is the reason we experienced the issues when training as the GPU RAM was not enough in our machines.

6.4 K- Fold Cross Validation

Table 2 provides a detailed breakdown of performance metrics for a classification model across ten folds or instances. The model exhibits relatively consistent results across scenarios, with an average macro F1-score of 23.09, micro F1-score of 33.96, and overall accuracy of 25.89%. The Precision, Recall, and F1-score values for each instance suggest a stable performance with an average Precision, Recall, and F1-score of 58.25%. Notably, the model demonstrates a higher level of precision, recall, and F1-score in some instances, particularly in the sixth fold where precision, recall, and F1-score reach 60%. This indicates that the model performs reasonably well across various subsets of the data, yielding an average balanced F1-score of 58.26%. The consistency in performance metrics across folds suggests that the model maintains a stable and reliable classification performance, particularly in terms of precision and recall.

Table 2: 10-fold cross-validation, together with the average over all folds.

Fold	**Macro**			**Micro**			**Accuracy**
	P	**R**	**F1**	**P**	**R**	**F1**	
1	58	19	33	24	58	58	58
2	19	33	24	58	58	58	58
3	31	36	31	58	58	58	58
4	19	33	24	58	58	58	58
5	31	34	26	58.3	58.4	58.3	58.3
6	33	37	31	60	60	60	60
7	19	33	25	58	58	58	58
8	19	33	25	58	58	58	58
9	19	33	25	58	58	58	58
10	19	33	25	58	58	58	58
Average	23.09	33.96	25.89	58.25	58.25	58.26	58.26

6.5 Bias Analysis

Here, in this part, we choose two attributes - age and gender. We divided the datasets into male and female sub-datasets under gender attribute and also three sub-datasets named- young, middle-aged and senior under age attribute. For this extensive analysis, we divided the datasets under the mentioned categories to get the bias analysis.

Table 3: Result from the bias analysis

Attribute	**Group**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1 (%)**
Age	Young	33	27	27	26
	Middle-aged	29	28	26	26
	Senior	33	27	30	27
	Average	32	28	29	26
Gender	Male	48	29	31	30
	Female	30	33	35	30
	Average	39	31	33	30
Overall System Average		33.5	29	30.14	28
Main Model (Part III)		**84**	**83**	**83**	**83**

Table 3 presents the results of a bias analysis, examining the performance metrics of a classification model across different attributes and groups. The analysis is conducted on two key attributes: Age and Gender. For the Age attribute, the model's accuracy, precision, recall, and

F1-score are evaluated for three groups: Young, Middle-aged, and Senior. The results indicate relatively consistent performance across age groups, with an overall system average accuracy of 32%. In terms of Gender, the model's performance is assessed for Male and Female groups, revealing variations in accuracy, precision, recall, and F1-score. The overall system average for gender attributes is 39%. The final row presents the overall system average across all attributes, showcasing an accuracy of 33.5%, precision of 29%, recall of 30.14%, and an F1-score of 28%.

Additionally, the performance of the main model (Part III) is provided, exhibiting higher metrics across the board with an impressive accuracy, precision, recall, and F1-score of 84%. These results suggest that the model's performance varies across different attribute groups, with notable distinctions in gender-based assessments, and emphasizes the superior performance of the main model in comparison to the attribute-specific evaluations.

6.6 Conclusion and Forward Look

This project demands building out AI-ducation Analytics using a custom-built image dataset on the raw CNN network. However, the task is challenging as the image classes we worked on are very close to each other from the perspective of their visual features. The dataset has no major issues yet the accuracy would be more increased if we can use some pre-trained CNN architecture to train our model. Nevertheless, longer training times are a result of larger pre-trained CNN models requiring more computational power due to their increased layers and parameters. It is critical to determine if the computing infrastructure we are utilizing can handle the added complexity [3]. In future, if we can manage more powerful machines, we will try some popular topologies for such deep learning challenges, including densely connected networks (DenseNet), or residual networks (ResNet) could be more efficient in encountering problems similar to this project. As a future endeavor, we will implement such models in our training that could lead to better results in every evaluation criteria that we get so far in this project.

Reference

[1] “Face expression recognition dataset,” *Kaggle*, Jan. 03, 2019. https://www.kaggle.com/datasets/jonathanoheix/face-expression-recognition-dataset

[2] “CKPLUS,” *Kaggle*, Oct. 16, 2018. https://www.kaggle.com/datasets/shawon10/ckplus/data

[3] Sourav, M.S.U., Wang, H. Intelligent Identification of Jute Pests Based on Transfer Learning and Deep Convolutional Neural Networks. *Neural Process Lett* 55, 2193–2210 (2023). https://doi.org/10.1007/s11063-022-10978-4