COMP 6721 Fall 2023 Project Part 3

Team Name: NS_13

Team Members:

Wenhao Gu [40203004]

Md Sayeed Abid [40260779]

Archilkumar Dineshbhai Katrodiya [40270119]

Team Member Specializations:

Data Specialist: Archilkumar Dineshbhai Katrodiya

Training Specialist: Wenhao Gu

Evaluation Specialist: Md Sayeed Abid

Project Repository Link: [GitHub]

Introduction:

In this project we developed an AI system to detect human facial expressions to detect activities during online class. Throughout this project, we have developed and experimented with several CNN architectures to finalize the most optimal model for our project. We created a standardized dataset to train and test our model. As there were no proper pre-labeled datasets available we labeled our dataset and augmented it to prepare it for the project. Finally, to increase the robustness of our system we implemented 10-fold cross-validation to adapt our model more robustly, and we also performed bias-analysis on various attributes to determine our model's generalizing ability. Finally, we were able to fulfill all the requirements and show satisfactory results.

Dataset Choices:

Classes	Image Source	Licensing Type	Relevant	Information
Neutral:	3522	FER-2013 Dataset	Publicly available	The dataset can be accessed at kaggle
Engaged/Focused:	3654	FER-2013 Dataset (choose by hand)	Publicly available	The dataset can be accessed at kaggle
Bored/Tired:	3714	FER-2013 Dataset (choose by hand)	Publicly available	The dataset can be accessed at kaggle
Angry/Irritated:	3744	FER-2013 Dataset	Publicly available	The dataset can be accessed at kaggle

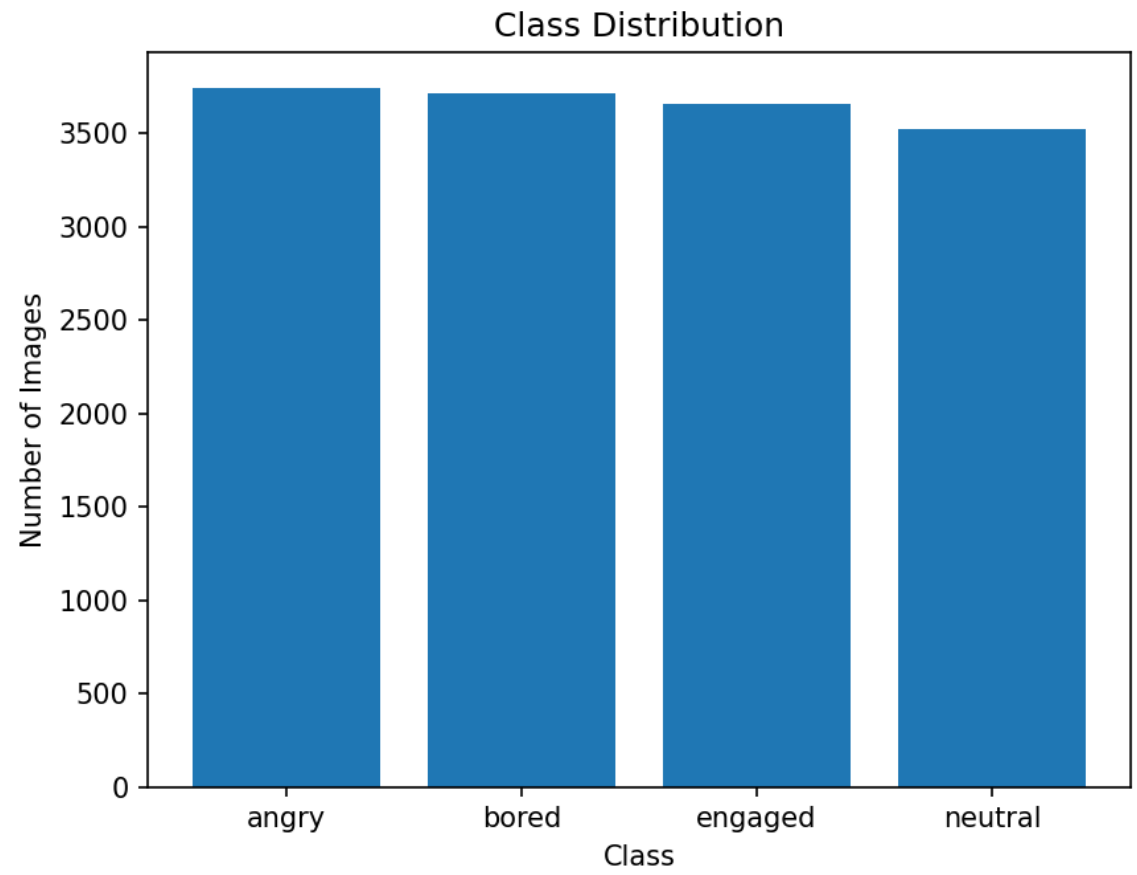
We changed the dataset from part one because of an unbalanced reason.

In our part2, we focused on identifying 'engaged' and 'bored' emotions within the FER-2013 Dataset, which also includes Sad, Disgust, Fear, Surprise, and Happy categories. We handpicked approximately 700 images for each emotion (engaged and bored) from these categories. Our initial step was data cleaning, which involved carefully reviewing each image to ensure accurate categorization into 'engaged' or 'bored', as some images had ambiguous emotional expressions.

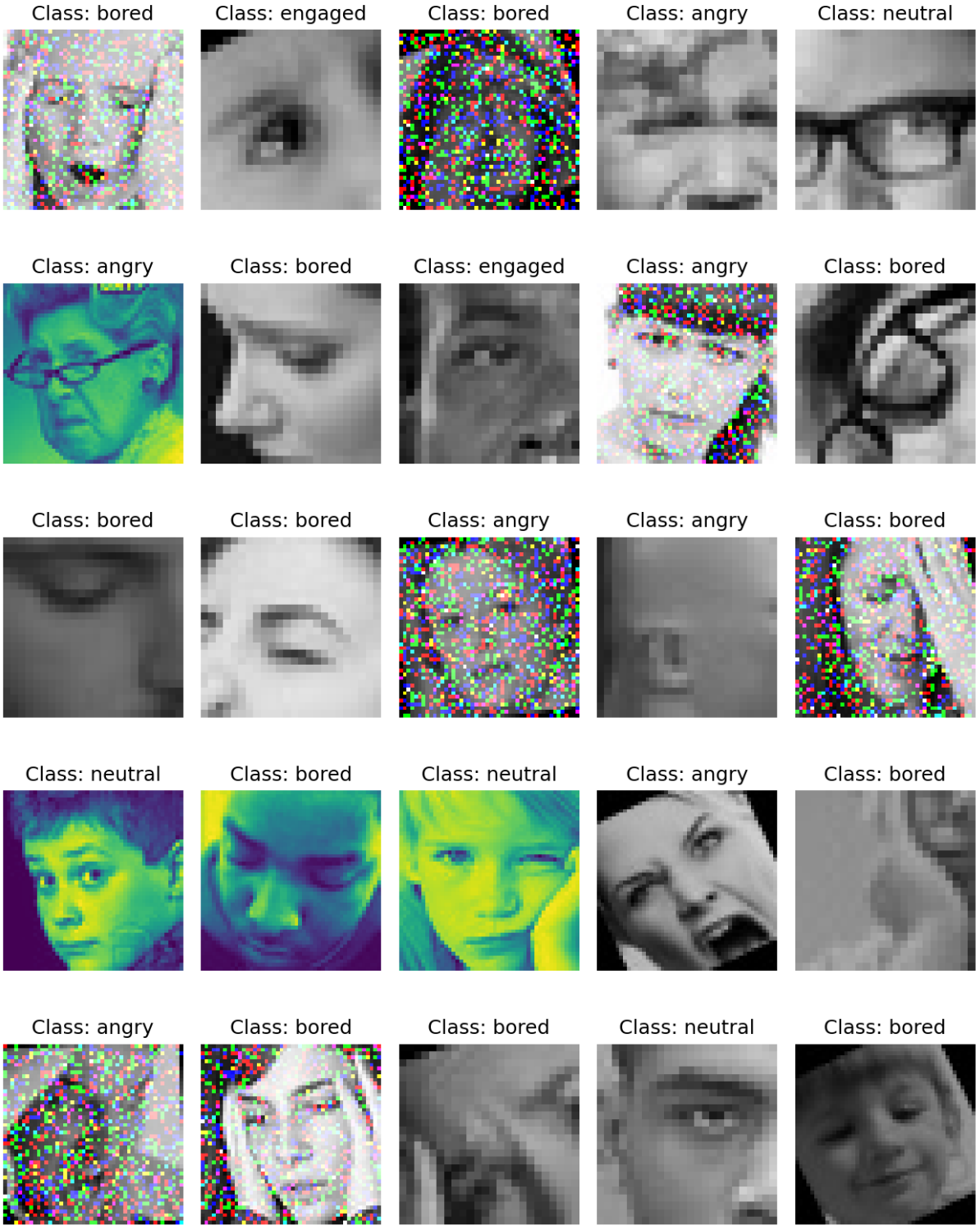
Subsequently, we enhanced the dataset through several image augmentation techniques. Each of the selected images underwent brightness correction and slight rotation. We also cropped the images for better focus on facial expressions and added noise to increase the robustness of our dataset. These augmentation techniques expanded our initial set of approximately 700 images per class to around 3600 images per class, enriching the dataset for more effective emotion recognition research. We adapted some techniques for data augmentation from [3] and implemented it in our project.

Dataset Visualization:

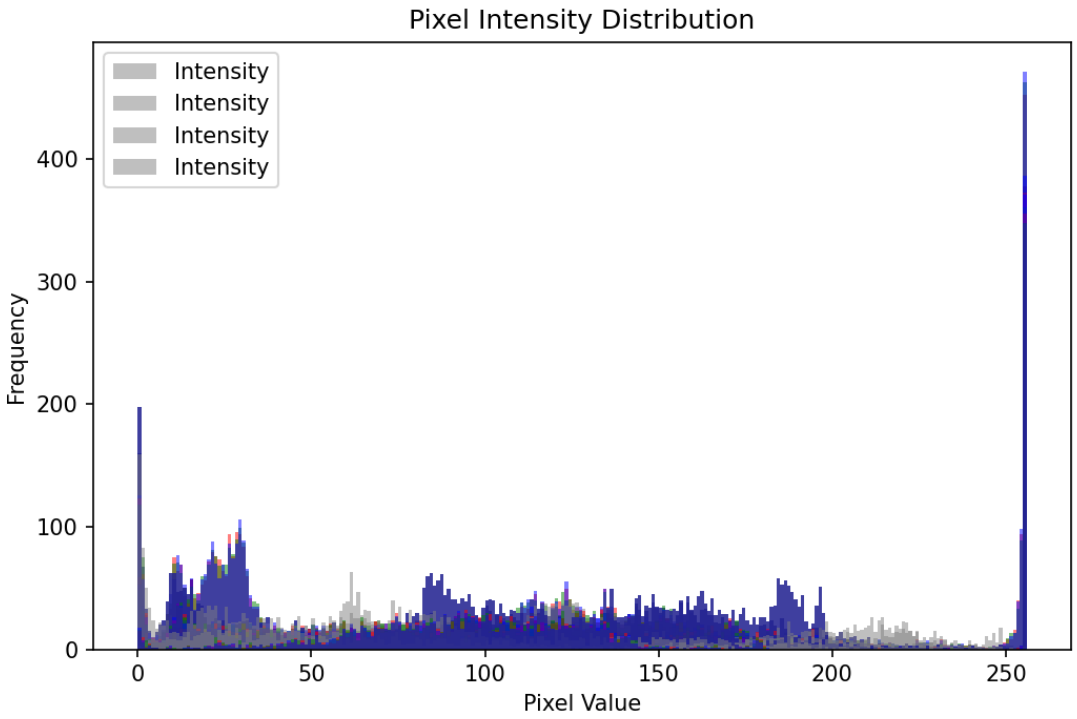
Class Distribution:



Pixel Intensity Distribution:



Sample Images:



Data Cleaning:

For the bias analysis, we segmented data based on two attributes: age and gender. Both segments have two subsegments in each, in age, it's young and old. In gender, it's Male and Female. We manually labelled these data based on visual inspection of the images. We have built a hierarchical folder structure with appropriate class names and segments for the dataset, which makes it easier to load data and parse labels. We also applied the same preprocessing steps mentioned above, including augmentation, brightness corrections and noise addition to make it more generalized.

The below mentioned structure below shows labeling and the hierarchy of the dataset.

Labeling:

We labeled the data structure like below, we chose the Age and Gender for our bias analysis.

```
testdata/
├── angry/
│   ├── Age/
│   │   ├── old/
│   │   │   ├── 1_0 .jpg
│   │   │   ├── 1_1 .jpg
│   │   │   └── ...
│   │   ├── young/
│   │   │   ├── 1_0 .jpg
│   │   │   ├── 1_1 .jpg
│   │   │   └── ...
│   ├── gender/
│   │   ├── Female/
│   │   │   ├── 1_0 .jpg
│   │   │   ├── 1_1 .jpg
│   │   │   └── ...
│   │   ├── Male/
│   │   │   ├── 1_0 .jpg
│   │   │   ├── 1_1 .jpg
│   │   │   └── ...
├── bored/
│   └── ...
├── engaged/
│   └── ...
└── neutral/
        └── ...
```

CNN Architecture:

Change form part2:

We define a transform to normalize the data and play with the normalized parameters to suit our model best.

Main model:

The model has three convolutional layers for feature extraction and two fully connected layers for classification, 3*3 kernel size. We also added max-pooling and Dropout layers that enhance the model's generalization ability. We have used LeakyReLU as an activation function.

Convolution Layers:

Layer 1:

The first layer is the Convolution layer with kernel size 3 and padding 1. It produces output with 16 channels.

Layer 2:

Convolution layer with 16 input channels, 32 output channels and kernel size 2 with padding 1.

Layer 3:

Convolution layer with 32 input channels, 64 output channels and kernel size 3 with padding 1.

Pooling Layer:

We have used MaxPooling with Kernel size 2 and stride 2.

Fully Connected Layers:

The Hidden Layer contains 512 neurons, and the output layer contains 4.

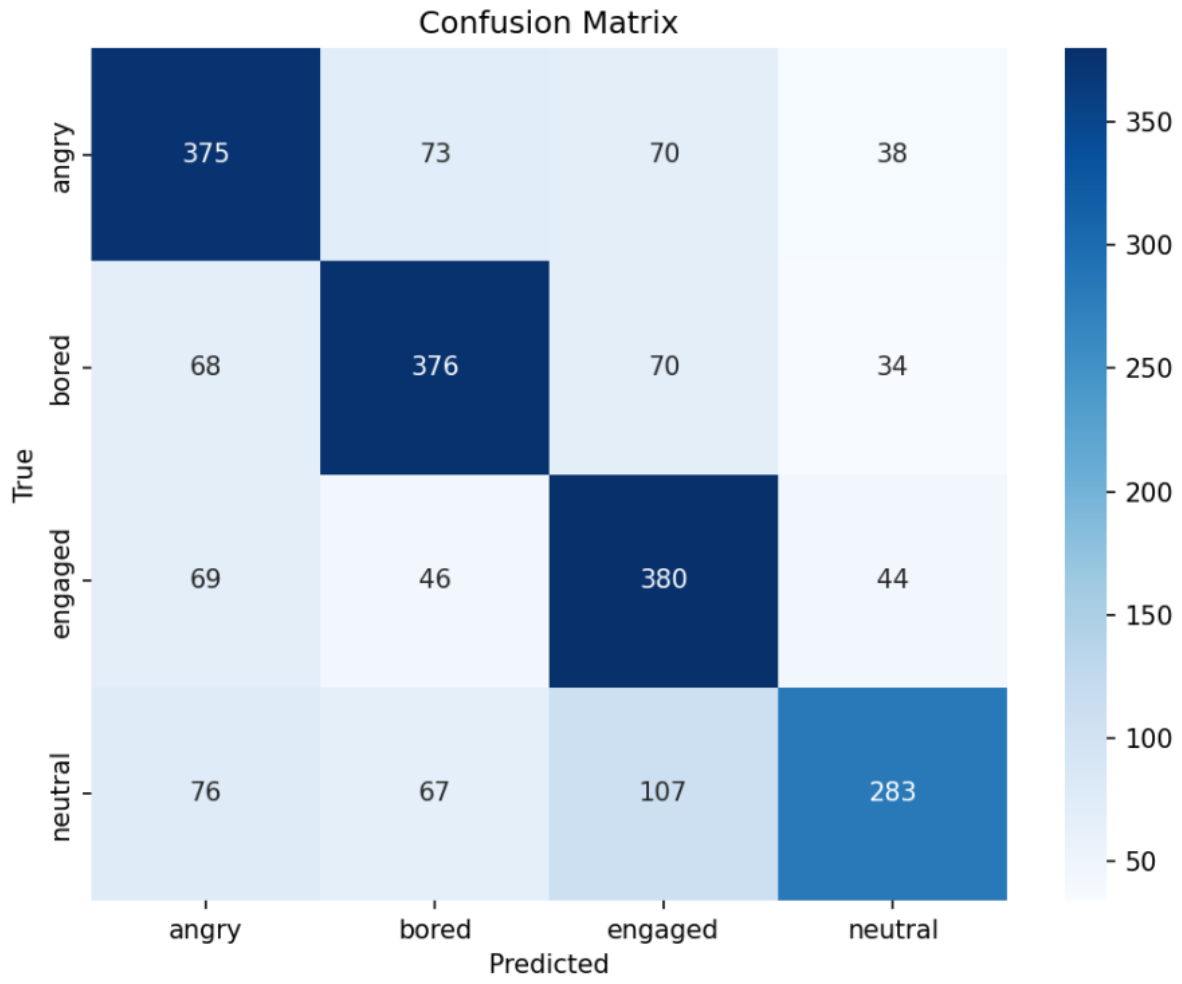
Activation and Regularization:

- ReLU: Applied after each convolutional and fully connected layer except for the output layer.
- Dropout: Dropout with a dropout rate of 0.5 is imposed after the flattening and the first fully connected layer.

Evaluation:

Confusion Matrix:

Here is the confusion matrix of our final model:



From the confusion matrix, we can see that our model is performing quite well in the angry, bored, and engaged classes, but it sometimes fails to classify the neutral. In terms of facial expressions, if we look closely at our dataset, we can see that the engaged and neutral classes are very close, and as our dataset is not very large and does not have huge variations and it fails in that area sometimes. But the overall performance is still satisfactory.

K-fold Cross Validation:

The tables below will explain all the results of the k-fold cross-validation for both of our models. To implement K-fold cross-validation we took help of python's scikit learn library [4]:

Fold	Macro			Micro			Accuracy
	P	R	F	P	R	F	
1	50.07	53.33	51.61	90.21	91.20	90.86	90.26006
2	50.07	53.33	51.61	90.21	90.21	90.21	90.26032
3	50.07	53.33	51.61	90.21	91.20	90.21	90.26076
4	51.67	54.02	52.77	90.56	90.98	90.56	91.12206
5	50.11	53.33	51.64	90.33	91.03	90.33	90.33110
6	96.93	56.96	58.53	90.86	90.08	90.86	90.86112
7	96.89	56.06	56.92	89.44	89.03	89.89	89.44367
8	63.60	56.93	58.23	90.86	90.86	90.86	90.86322
9	60.11	53.33	58.64	90.33	90.41	90.30	90.86718
10	96.97	57.91	50.29	90.99	91.30	91.30	90.99672
Average	66.49	55.71	55.22	90.66	91.08	90.74	90.56771

Table 1: K-fold cross-validation of our final model

Fold	Macro			Micro			Accuracy
	P	R	F	P	R	F	
1	59.97	59.88	59.80	59.95	59.95	59.90	59.95
2	59.93	59.94	59.90	59.91	59.91	59.89	59.91
3	59.81	59.92	59.95	59.93	59.92	59.92	59.90
4	59.97	59.97	59.93	60.01	60.06	59.99	59.98
5	59.83	59.87	59.91	59.85	59.88	59.89	59.87
6	59.93	59.99	59.93	59.91	59.90	59.90	59.94
7	59.90	59.97	59.89	59,92	59.83	59.99	59.89

8	60.11	60.08	59.90	60.12	60.22	60.38	60.13
9	60.02	60.10	59.98	60.10	60.13	60,13	60,01
10	60.04	59.90	59.93	59.83	59.88	59.80	59.90

Table 2: K-fold cross-validation from Part II

As we can see from the table that our final model has outperformed the initial one. Though the performance in Macro precision, recall, and f1-score is almost the same as before in Micro precision, recall, and accuracy, it performs very well. This difference in performance metrics between macro and micro averages is because of the nature of the dataset. The model is performing well in the majority class while struggling with minority classes. As we augmented our dataset to increase the sample images and make the dataset balanced so in some minority classes, it struggles to classify correctly.

If we look at the result from our previous model and the result obtained from the k-fold cross-validation, we can see that there is a big difference in the result. There are several reasons for this. First, the dataset is more balanced, and it covers a wider range of data variations. Secondly, the significant difference in performance between a single train-test split and 10-fold cross-validation indicates that the initial train-test split was not representative of the overall dataset. Cross-validation helps in mitigating this issue by using multiple train-test splits. So that all the classes and data variations are represented equally. Thus, we get higher test accuracy than before.

Bias Analysis:

Introduction and Result:

For bias analysis, we have chosen age and gender as the attributes to analyze biases throughout different classes. The table below shows Accuracy, Precision, Recall, and F1-measure for both attributes and for both of our models:

Attribute	Group	Accuracy%	Precision%	Recall%	F1-Score%
Gender	Male	0.0	0.0	0.0	0.0
	Female	97.86	97.86	97.20	97.86
Age	Young	94.59	93.92	93.56	93.59
	Old	0.0	0.0	0.0	0.0
Overall System Average		42.18	42.51	42.18	41.61

Table 3: Bias analysis among different attribute using initial model

Attribute	Group	Accuracy%	Precision%	Recall%	F1-Score%
Gender	Male	44.27	46.56	44.27	43.78
	Female	52.34	51.87	52.34	51.79
Age	Young	51.56	50.20	51.56	50.59
	Old	47.65	48.19	47.65	47.09
Overall System Average		49.37	49.01	49.06	48.81

Table 4: Bias analysis among different attribute using Final model

Detecting and Mitigating Bias:

From table 4 we can see that our model performs equally among various classes and attributes. It indicates that our model is generalized well. Among the 4 attributes, the male class performs inferiorly among the others, having an accuracy of 44.27%, and the female class is the strongest in terms of performance, with a 52.34% accuracy.

However, when we initially tried to do a bias analysis, we got very low accuracy. Some of the attributes gave very high accuracy, but some were near zero. So, to mitigate this issue, we relabeled our dataset appropriately. Another issue was that we were previously using an online train-test-validation split for our training, but that resulted in some data leakage, and thus, the performance was not satisfactory. Finally, we did a proper data split, re-trained our model, and got these results.

All these bias mitigating methods have resulted in an improvement in not only our bias analysis but also the overall performance of the model. If we look into the overall model performance from Table 3 and Table 4, we can see that there is a 7.19% improvement in our model's test accuracy, which is a big achievement.

Conclusion

In this final part of our project, we improved a lot in machine learning and deep learning, especially with Convolutional Neural Networks. Also, we did important work in solving ethical issues in AI.

We focused on fixing age and gender biases, which are big problems in AI. These biases can make AI unfair. We carefully checked our training data and algorithms to find and reduce these biases. We relabeled the data with ages and genders and made algorithms to lessen bias. We keep checking for age and gender bias to make sure our AI is fair.

During evaluation, using the K-fold method, we found flaws in our model. So, we changed our data and model. We defined a transform to normalize the data. We adjusted the normalization parameters to best fit our model. This was important to improve the model's performance. This step was crucial for making our AI better and more reliable.

Reference

[1] Sambare, M. (n.d.). FER-2013 [Dataset; Kaggle]. In Learn facial expressions from an image (Version 1). https://www.kaggle.com/datasets/msambare/fer2013/data
[2] Dey, J. (n.d.). Student-engagement [Dataset; Kaggle]. In predict student engagement in online classes (Version 1). https://www.kaggle.com/datasets/joyee19/studentengagement
[3] *Image Augmentation*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/04/image-augmentation-using-3-python-librarie/
[4] *Cross-validation: Evaluating estimator performance*. Scikit Learn. https://scikit-learn.org/stable/modules/cross_validation.html