COMP 6721 Fall 2023 Project Part 2

Team Name: NS_13

Team Members: Wenhao Gu, Md Sayeed Abid, Archilkumar Dineshbhai Katrodiya

Student ID Numbers: 40203004, 40260779, 40270119

Team Member Specializations:

Data Specialist: Archilkumar Dineshbhai Katrodiya

Training Specialist: Wenhao Gu

Evaluation Specialist: Md Sayeed Abid

Project Repository Link: [GitHub]

Dataset Choices:

classes images	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

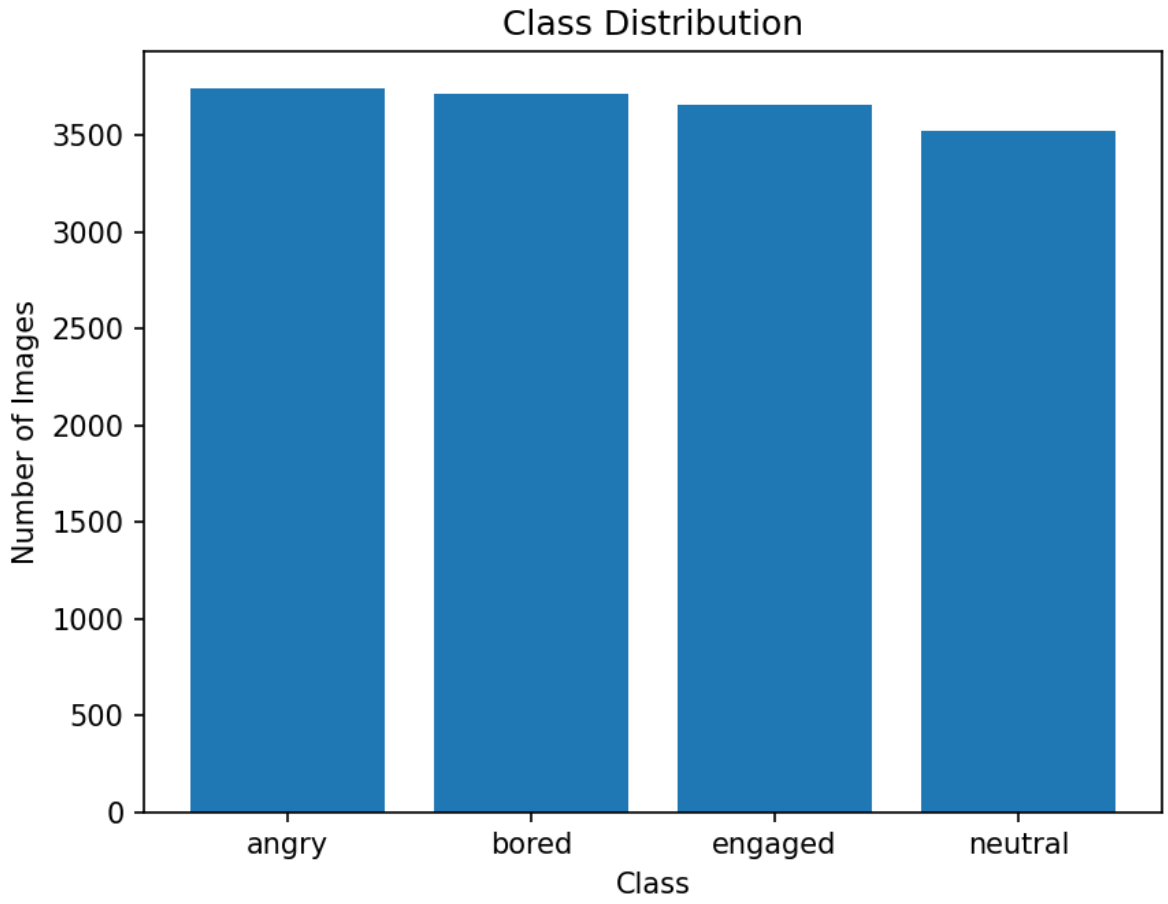
We changed the dataset from part one because of an unbalanced reason.

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.

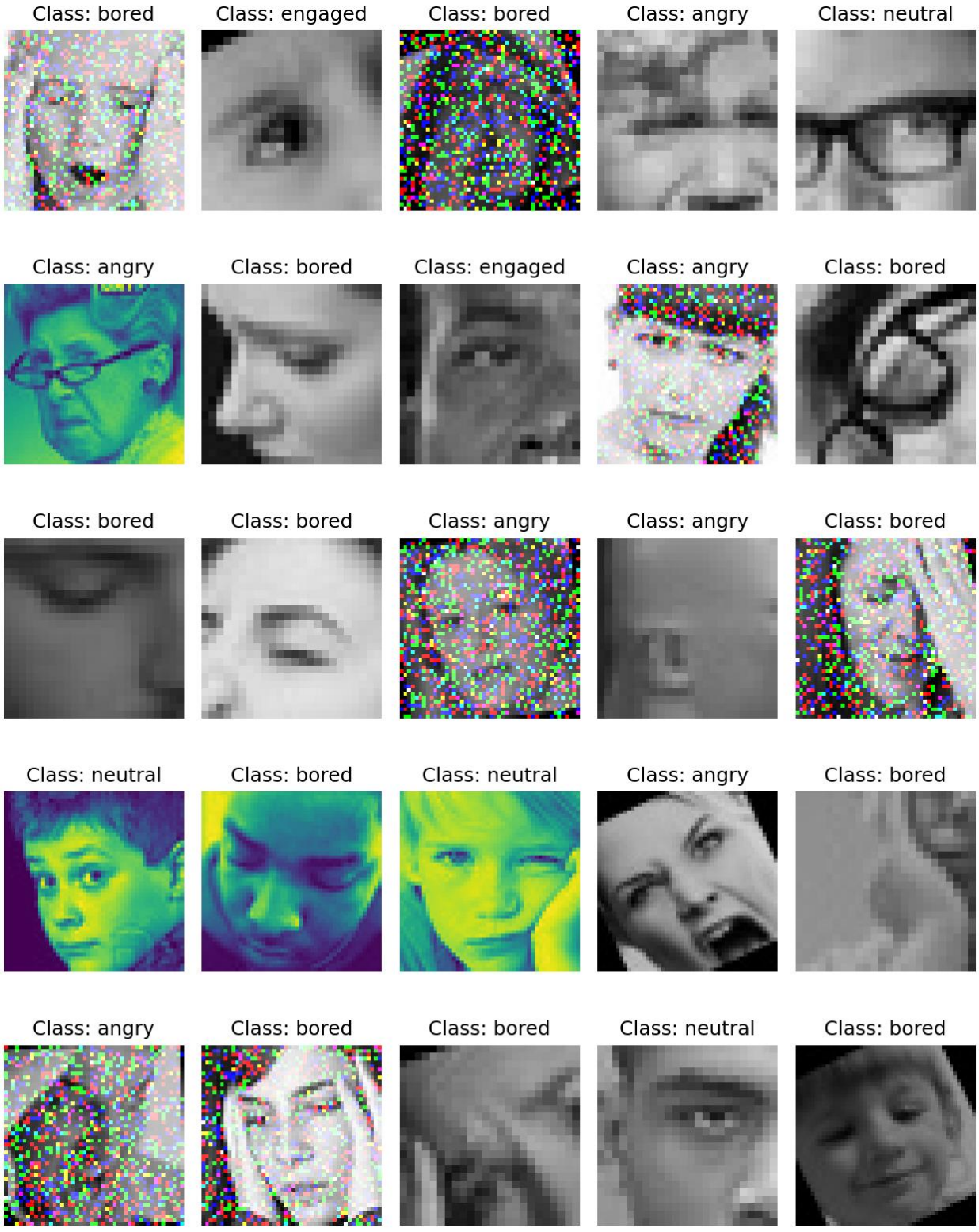
Subsequently, we enhanced the dataset through several image augmentation techniques. Each of the selected images underwent brightness correction and slight rotation. We also cropped the images for better focus on facial expressions and added noise to increase the robustness of our dataset. These augmentation 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.

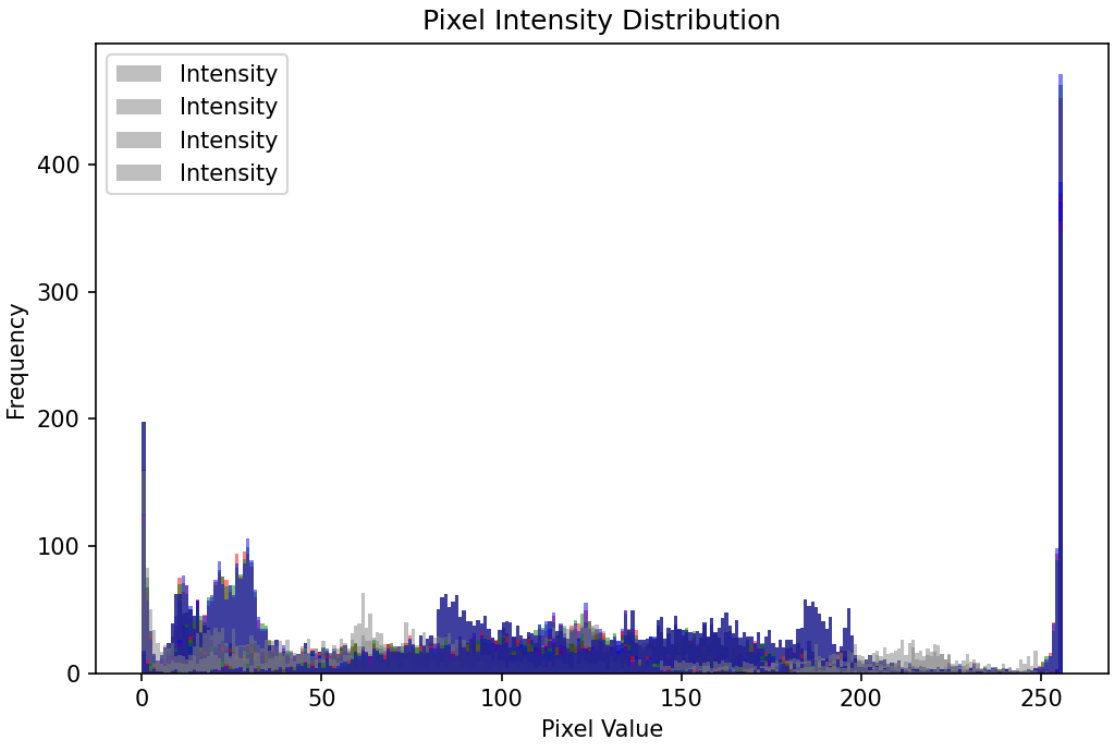
Dataset Visualization:

Class Distribution:



Pixel Intensity Distribution:




Pixel Intensity Distribution
Intensity
Intensity
Intensity
Intensity
Frequency
0
100
200
300
400
0
50
100
150
200
250
Pixel Value

CNN Architecture:

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.

Model Variation 1:

Instead of three, this model has four convolution layers. In the first three convolution layers, input and output channels remain identical; however, kernel size is 3 for all the layers. Activation, pooling, fully connected, and dropout layers are the same. The invocation of activation and Dropout are also the same.

Convolution Layer 4:
Layer with 64 input channels, 128 output channels and kernel size 3 with padding 1.

Model variation 2:

The model has two convolution layers and only one fully connected layer. It uses LeakyReLU activation, dropout and max pooling for generalization.

Convolution Layers:

Layer 1:

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

Layer 2:

Convolution layer with 16 input channels, 32 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 output layer contains 4 neurons that predict the class.

Activation and Regularization:

- ReLU: Applied after each convolutional.
- Dropout: Dropout with a dropout rate of 0.5 is imposed after the flattening.

Training Process:

The models underwent training for 50 epochs using a learning rate of 0.001. We have chosen Cross-entropy loss as the loss function, while Adam is an optimizer.
We experimented with model training by varying the size of a kernel. We have used three different kernel sizes 2, 3 and 5. The model performed the best with a 3x3 kernel with 3 3-layer convolution net.

Results:

2x2 Kernel:

With the above kernel size, after 50 epochs, the model achieved a training accuracy of around 73.7% and a testing accuracy of 61.2%, 3 layers convolution net. It shows that the model experienced underfitting and did not perform as expected on the training or testing set. CNN is not able to extract enough information from the images. To overcome the problem, we increased the kernel size to 5.

```
Epoch: 48    Training Loss: 0.658150    Training Accuracy: 0.731348    Validation Loss:
                0.914086    Validation Accuracy: 0.620404
          Validation loss decreased (0.921847 --> 0.914086).  Saving model ...
Epoch: 49    Training Loss: 0.659886    Training Accuracy: 0.736035    Validation Loss:
                0.930901    Validation Accuracy: 0.612592
Epoch: 50    Training Loss: 0.658409    Training Accuracy: 0.737402    Validation Loss:
                0.923049    Validation Accuracy: 0.620404
                Test Loss: 0.945618, Test Accuracy: 0.612592
```

5x5 Kernel:

With the above kernel size, after 50 epochs, the model achieved a training accuracy of around 83.7% and a testing accuracy of 62.5%, 3 layers convolution net. The model was overfitting on the training set. To overcome the problem, we changed the kernel size to 3.

```
Epoch: 48    Training Loss: 0.441086    Training Accuracy: 0.825000    Validation Loss:
                1.043358    Validation Accuracy: 0.638327
Epoch: 49    Training Loss: 0.425028    Training Accuracy: 0.833984    Validation Loss:
                1.039613    Validation Accuracy: 0.630974
Epoch: 50    Training Loss: 0.419774    Training Accuracy: 0.837402    Validation Loss:
                1.090913    Validation Accuracy: 0.625000
                Test Loss: 1.001778, Test Accuracy: 0.595588
```

Main model 3x3 Kernel:

With the above kernel size, after 50 epochs, the model achieved a training accuracy of around 81% and a testing accuracy of 65%. It performed better compared to the other two kernels.

```
Epoch: 48    Training Loss: 0.491385    Training Accuracy: 0.808008    Validation Loss:
                0.857007    Validation Accuracy: 0.665441
Epoch: 49    Training Loss: 0.503610    Training Accuracy: 0.800293    Validation Loss:
                0.863198    Validation Accuracy: 0.665441
Epoch: 50    Training Loss: 0.486833    Training Accuracy: 0.810059    Validation Loss:
                0.865220    Validation Accuracy: 0.657629
                Test Loss: 0.883289, Test Accuracy: 0.650276
```

Variation 1:

The models change 3 layers convolution to 2 layers, under training for 50 epochs using a learning rate of 0.001 and a kernel size of 3. We have chosen Cross-entropy loss as the loss function, while Adam is an optimizer. The model achieved a training accuracy of around 57.7% and a testing accuracy of 50%. It shows that the model experienced underfitting.

```
Epoch: 48    Training Loss: 0.997496    Training Accuracy: 0.570898    Validation Loss:
                1.125741    Validation Accuracy: 0.483915
Epoch: 49    Training Loss: 0.986199    Training Accuracy: 0.574414    Validation Loss:
                1.104674    Validation Accuracy: 0.503676
Epoch: 50    Training Loss: 0.984567    Training Accuracy: 0.577051    Validation Loss:
                1.108724    Validation Accuracy: 0.496783
                Test Loss: 1.104898, Test Accuracy: 0.509191
```

Variation 2:

The models change 3 layers convolution to 4 layers, underwent training for 50 epochs using a learning rate of 0.001 and a kernel size of 3. We have chosen Cross-entropy loss as the loss function, while Adam is an optimizer. The model achieved a training accuracy of around 87% and a testing accuracy of 52.8%. It shows that the model experienced overfitting.

```
Epoch: 48    Training Loss: 0.326708    Training Accuracy: 0.872559    Validation Loss:
                1.340406    Validation Accuracy: 0.581342
Epoch: 49    Training Loss: 0.332238    Training Accuracy: 0.871680    Validation Loss:
                1.430754    Validation Accuracy: 0.592831
Epoch: 50    Training Loss: 0.338393    Training Accuracy: 0.870898    Validation Loss:
                1.347836    Validation Accuracy: 0.585018
                Test Loss: 1.089433, Test Accuracy: 0.528493
```

Evaluation:

Performance Metrics:

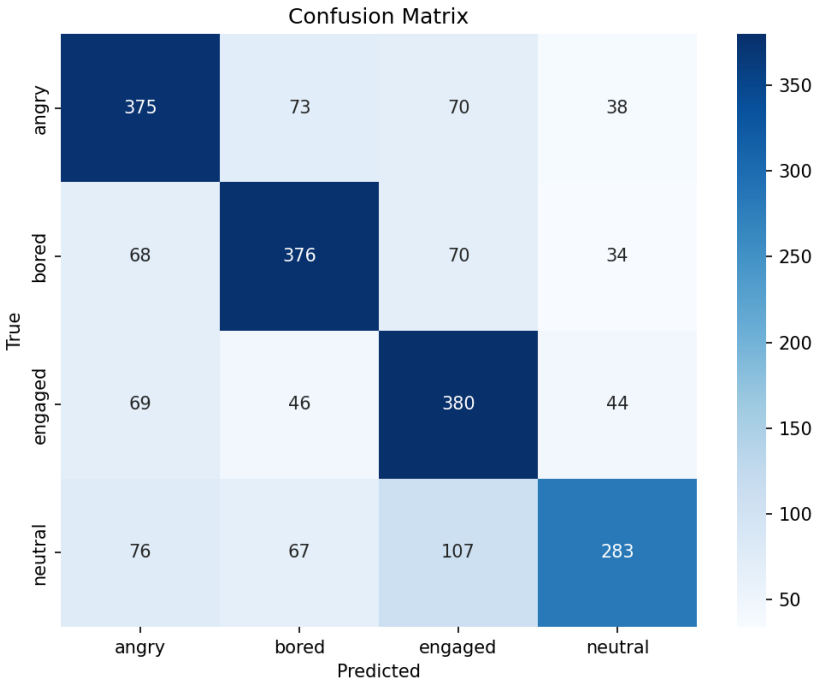
Model	Macro			Micro			Accuracy
	P	R	F1	P	R	F1	
Main Model	65.55	64.98	64.98	64.98	64.98	65.51	64.98
Variant1	50.81	50.72	50.87	50.87	50.87	50.81	50.87
Variant2	55.02	54.41	54.44	54.45	54.57	54.45	54.54

The main model has a training accuracy of 81%, and all the parameters are very close to it, which depicts that the model is generalized. Whereas, in variation 2, all the scores are low compared to its training accuracy of 90%, which shows that the model is overfitting on training data.

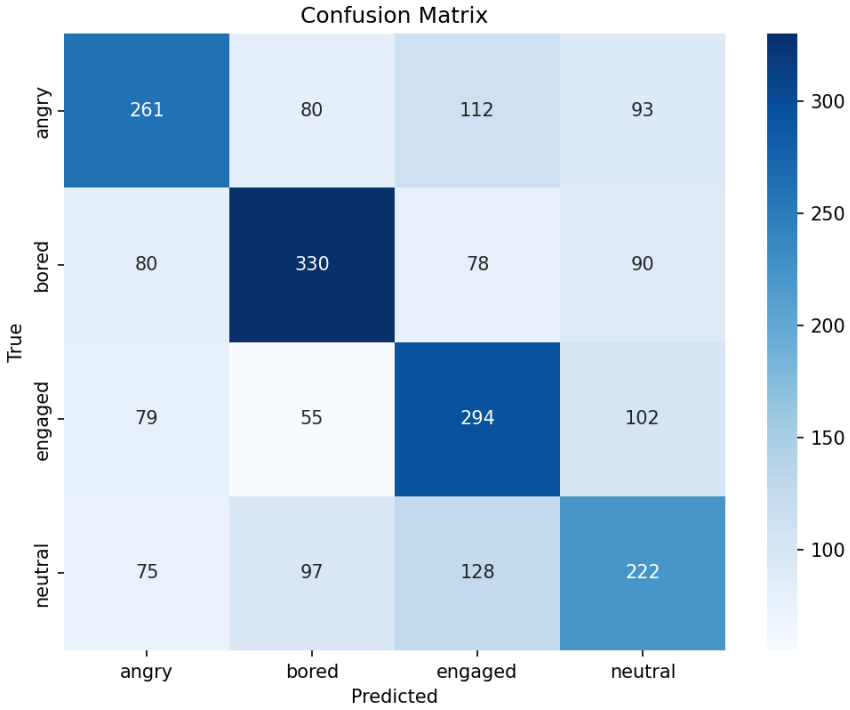
Confusion Matrix Analysis:

Based on the below confusion matrix, the most frequently confused class is natural. The main reason behind this misclassification is a tiny visual difference that increases false negatives. The model performs surprisingly well in recognizing angry and bored faces. Those classes have high visual differences compared to neutral and engaged, which create more discrete feature sets for the model to learn from and eventually reduce the model bias towards those classes.

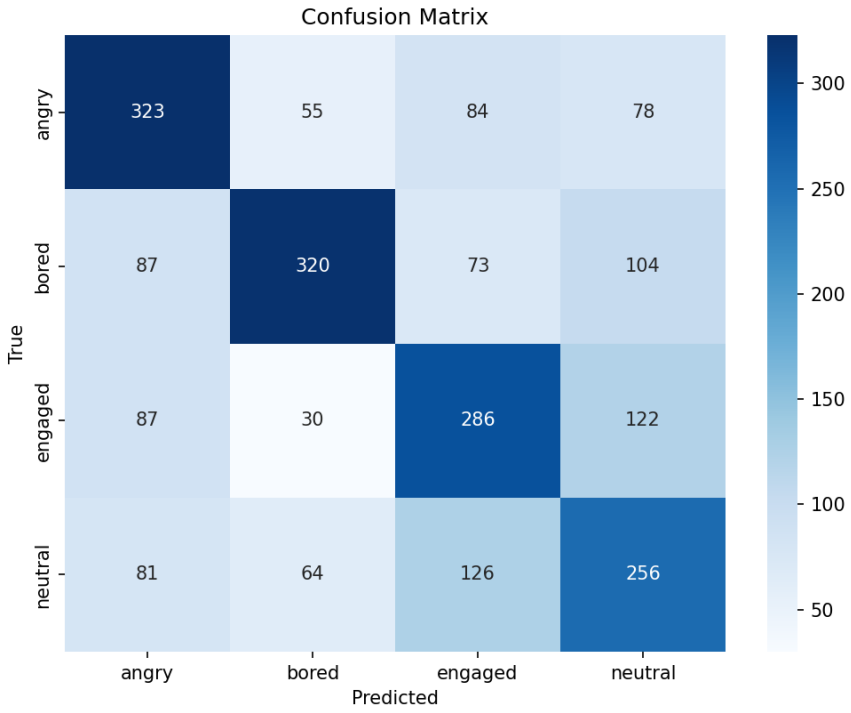
Main model:



Variation 1:



Variation 2:


Confusion Matrix
angry
bored
engaged
neutral
True
323
55
84
78
87
320
73
104
87
30
286
122
81
64
126
256
angry
bored
engaged
neutral
Predicted
300
250
200
150
100
50

Impact of Architectural Variation:

We tried three different models. Variation 1 contains only two convolution layers. The model was unable to capture enough detail for classification. It underwent the problem of underfitting. We decided to increase the number of convolution layers to make the model extract detailed features. We increased the depth of the network by adding two more convolution layers. With 4 convolution layers, the model learned detailed features but lost generalization. We tried the third and our main model, in which we used three convolution layers. This model is capable of learning detailed features and is generalized.

We experimented with model training by varying the size of a kernel. We have used three different kernel sizes: 2, 3, and 5. The model performed the best with a 3x3 kernel. Detailed discussion about these experiments is described above in the **Training Process** part.

Conclusions and Forward Look:

The 3x3 Kernel with 3 Convolution Layers emerged as the best-performing model. It achieved a balance between training and testing accuracy (81% training, 65% testing), suggesting effective feature extraction without significant overfitting or underfitting. Smaller kernels (2x2) were insufficient for feature extraction and lacked the model complexity needed to capture essential features, while larger ones (5x5) led to overfitting. More layers (4 layers) led to overfitting, whereas fewer layers (2 layers) resulted in underfitting. A 3-layer architecture appeared optimal for your dataset. Each change in the model configuration had a significant impact on performance.

Consider employing data augmentation approaches to further increase generalization. This can include rotating, resizing, or flipping photos to provide a more diversified training set, which will help the model generalize better. Consider pre-trained models and fine-tuning them on your specific data for datasets similar to typical image datasets. This technique can make use of learned features from big, heterogeneous datasets. Dropout or L2 regularization should be used to prevent overfitting, especially in models with increased complexity (such as the 4-layer form).