# Education MindA.I.lytics:
# Facial Expression Recognition Dataset

COMP 472-NN

Moodle Team Name: FS_10

Aman Nihaal Nuckchady, 40249877, Data Specialist
Ferdous Hasnat, 40112912, Evaluation Specialist
Lucas Kim, 40174336, Training Specialist

Project Repository: https://github.com/yogurtshake/COMP-472-Project

**“We certify that this submission is the original work of members of the group and meets the Faculty's Expectations of Originality."**

Aman Nihaal Nuckchady, 40249877
Ferdous Hasnat, 40112912
Lucas Kim, 40174336

# Dataset

## Overview

For this project, we focused exclusively on utilizing the FER-2013 dataset due to its comprehensive collection of facial images annotated with various attributes.

FER-2013 Dataset:

| | |
|---|---|
| Total Number of Images | 2,906 |
| Total Number of Images in Training Set | 2,321 |
| Total Number of Images in Testing Set | 585 |
| Average Number of Images per Class in the Training Set | 580 |
| Average Number of Images per Class in the Testing Set | 146 |

Characteristics:

- The FER-2013 dataset is organized into labeled subfolders, with each subfolder corresponding to a specific facial expression category, including surprise, neutral, happy, and engaged.
- FER-2013 primarily consists of frontal face shots with diverse facial expressions, poses, and backgrounds.
- It also includes a variety of image conditions, such as occlusions, different lighting conditions, and various facial orientations.

## Justification for Dataset Choices

The FER-2013 dataset was chosen as the primary dataset for several reasons:

- Relevance to Project: The project's primary objective is facial expression recognition, and FER-2013 offers a rich set of annotations for various facial expression categories namely: angry, sad, surprise, neutral, happy, angry, disgust, and fear, making it well-suited for this task.
- Diversity and Complexity: FER-2013 contains a diverse range of facial images with different expressions, poses, lighting conditions, and backgrounds, providing a challenging and representative dataset for training and testing.
- Availability and Accessibility: FER-2013 is widely used in the research community and is readily available for non-commercial research purposes, ensuring easy access and reproducibility of results.

## Provenance Information

The table below provides detailed provenance information for the FER-2013 dataset:

| Dataset | Source | Licensing Type | Additional Information |
|---|---|---|---|
| FER-2013 | Kaggle: https://www.kaggle.com/datasets/msambare/fer2013/data | Database: Open Database, Contents: Database Contents (DbCL) | FER-2013 Dataset |

This information includes the source of the dataset, the licensing type governing its usage, and any additional relevant details. It ensures transparency and compliance with licensing agreements while acknowledging the contributions of the original data providers.

# Data Cleaning

The data cleaning process consisted of ensuring standardization throughout the dataset by enforcing a consistent image size of 48x48 pixels, removing irrelevant data, as well as manually sorting through each image to find those that were too dark. Once all dark images were sorted, light augmentation was applied using the Python Imaging Library and a Python script.

## Challenges Faced

The main challenge of the data cleaning process was dealing with images that were too dark. Many of the images in the dataset were visible but a little too dark such that they significantly skewed the average pixel intensity distribution of each class and presented problems such as outliers. The approach we took to deal with this challenge was to manually list the darker images and then write a Python script to increase the brightness of these images. Another small challenge was dealing with irrelevant/corrupt data, but there were only 1 or 2 images that were corrupt/not showing a face, which were simply removed from the dataset.

## Manual Sorting & Light Augmentation Process

Each image in the dataset was manually examined to determine whether or not it was too dark. These images were copied and pasted into a directory with an identical structure to the original dataset, as copying and pasting the files was quicker than manually taking down their file names. Once the sorting was complete, a Python script was written to traverse the dark images directory and write all of those file paths to a .txt file. Then, another Python script was used to read the .txt file, open each image from its path, and increase the brightness of the image by a scaling factor of 1.5. PIL, Python Imaging Library, was used to achieve this.

## Tools Used

Python was used to create the 2 scripts, Python Imaging Library was used to apply the brightness transformation to the selected images, and basic image-viewing software (Microsoft Photos) was employed to facilitate the manual inspection process.

## Image Size Standardization & Additional Transformations

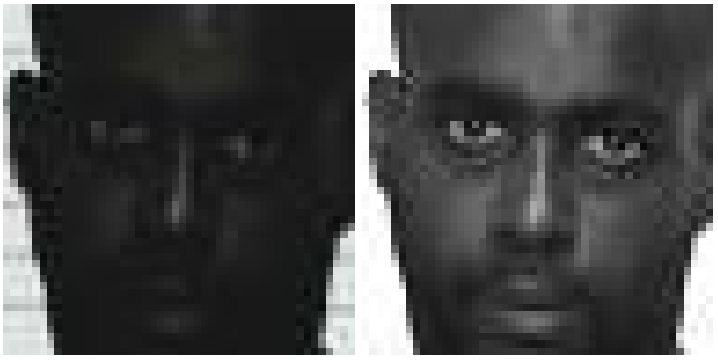
Additional image transformations were considered as well to increase the robustness of the dataset but were ultimately disregarded as they seemed unnecessary in the end. A script was created to enforce a 48x48 pixels image size and to randomly apply transformations such as slight rotations, image mirroring, and so on. This was done using Pytorch and torchvision. The Matplotlib library was also used to visualize the before and after snapshots of the transformations. All in all, this script was used to apply the 48x48 pixels image size

standardization but the additional transformations seemed unnecessary and were not used (although the code remains in the repository), as the main challenge with cleaning the data was the variance in image brightness.

## Example Images - Before & After

Here are some examples of images before and after their light augmentation:

***Before*** ***After***



***Before*** ***After***

# Data Labeling

In labeling the dataset, a manual approach was adopted to ensure accuracy and consistency in assigning labels to facial expressions. The following methods and tools were used in this process:

## Manual Labeling Process

Each image in the dataset was carefully examined individually to determine the appropriate label based on the appropriate facial expression. Labels were assigned to images according to predefined categories, including surprise, neutral, happy, and engaged. Images were sorted into labeled subfolders corresponding to their respective categories for organization and ease of access during training.

## Tools Used

Basic image-viewing software (Microsoft Photos) was employed to facilitate the manual inspection and labeling process. This allowed for zooming in on details and ensuring accurate classification of facial expressions.

## Handling Irrelevant Categories

Irrelevant categories such as sad, angry, disgust, and fear were removed from the dataset to streamline the focus on relevant facial expressions: engaged, happy, neutral, and surprised. The ‘engaged’ category, which originally encompassed both neutral and angry expressions, was refined by merging the ‘angry’ set with the ‘neutral’ set. This consolidation aimed to better represent engaged facial expressions while eliminating redundancy.

## Challenges Faced

One of the main challenges encountered was ensuring the removal of images with irregularities or artifacts that could potentially impact the accuracy of the dataset.

Artificial faces generated through filters or digital manipulation were identified and excluded from the dataset to maintain authenticity. Additionally, images depicting extreme facial orientations exceeding 75 degrees were removed to ensure consistency and relevance to the project's objectives.

Balancing the removal of obstructed facial features with preserving images that still effectively conveyed the intended emotion posed a challenge. Some images with minor obstructions were retained if the overall facial expression remained clear and consistent with the label.

Overall, the manual labeling process involved meticulous examination of each image, coupled with thoughtful decision-making to ensure the integrity and quality of the labeled dataset for subsequent analysis and model training.

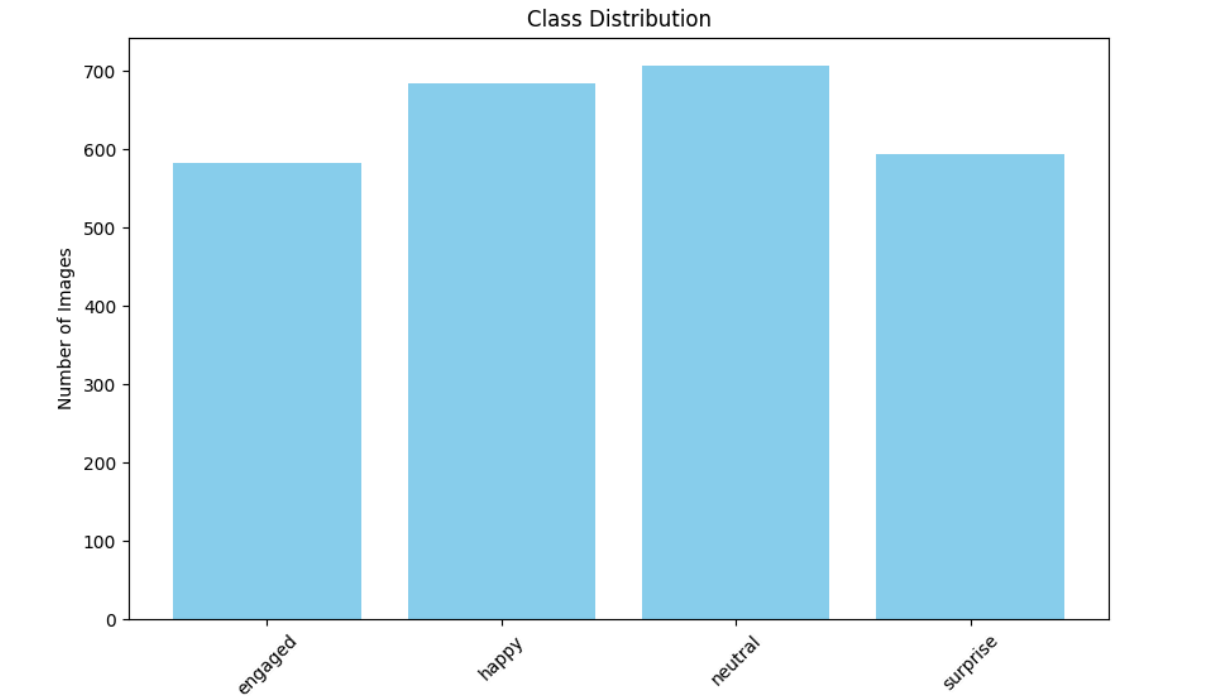
# **Dataset Visualization** (updated)

In this part of the project, we present a visual analysis of a facial expression dataset, which includes emotions such as engaged, happy, neutral, and surprised. The objective is to understand the dataset's distribution, uncover any imbalances, and inspect individual pixel intensity distributions that might highlight variations in image quality, such as lighting differences.

*NOTE: All graphs and visualizations are updated to represent the most recent dataset from Part 3.*

## Class Distribution

The class distribution bar graph indicates that the classes in our dataset are fairly balanced, with 'happy' and 'neutral' showing a slightly higher presence compared to 'engaged' and 'surprised'. No class is excessively over or underrepresented, which is favorable for training unbiased machine learning models.




## Sample Images

The visualization of sample images from each class provides a qualitative sense of the data. Each emotion category—'engaged', 'happy', 'neutral', and 'surprise'—displays distinct facial

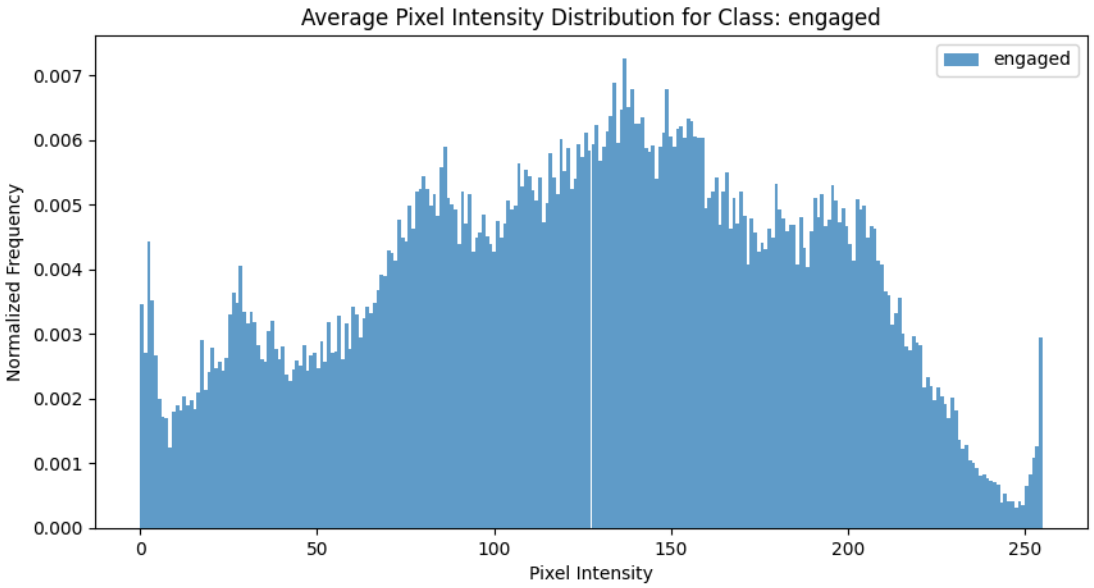
expressions. This representation ensures that there are no noticeable anomalies or mislabeling that could mislead the training of the model.

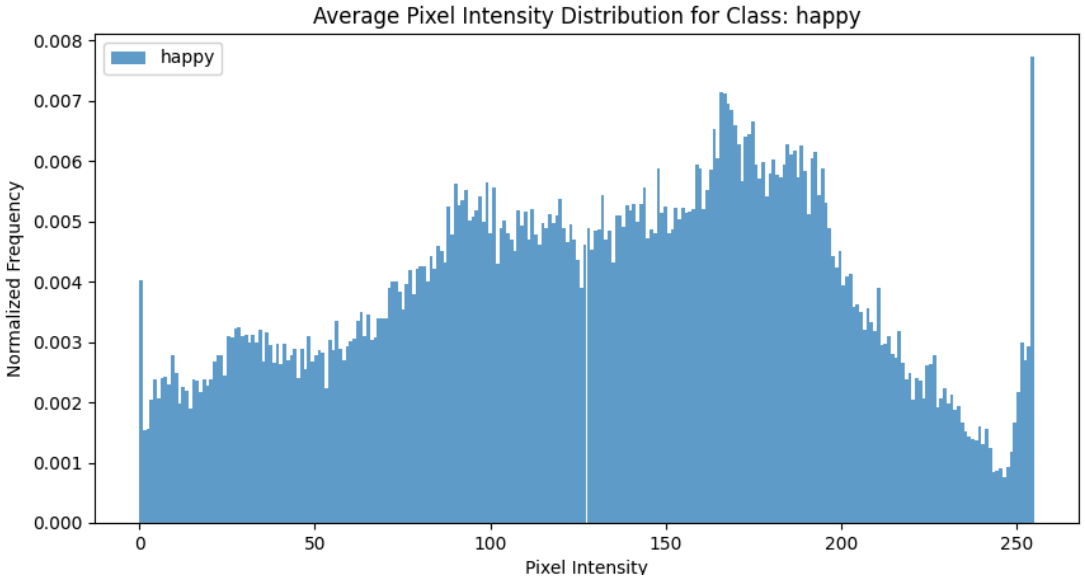

Sample Images for Class: neutral

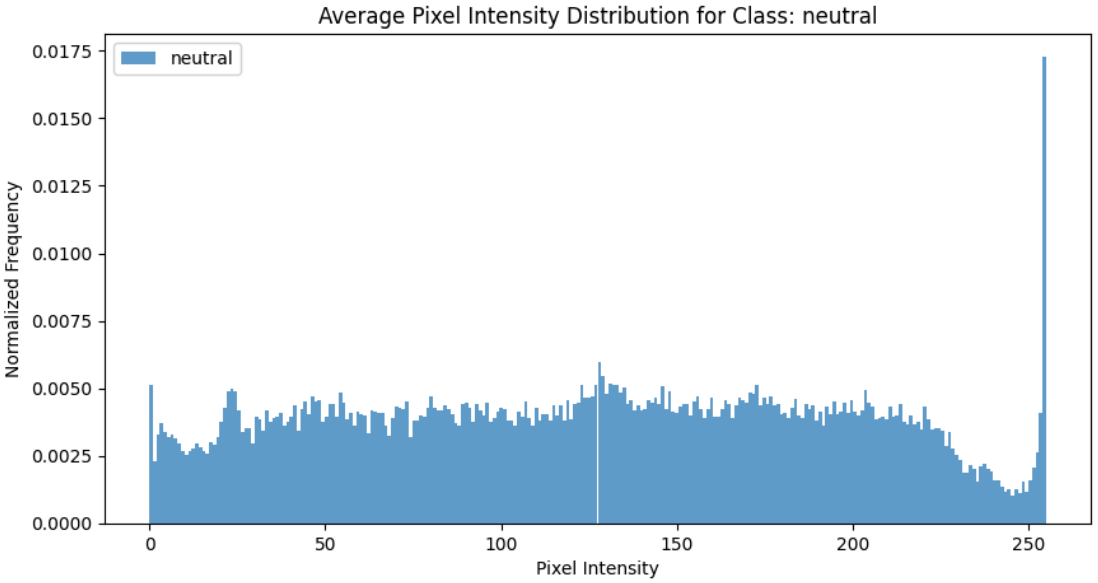
Sample Images for Class: surprise

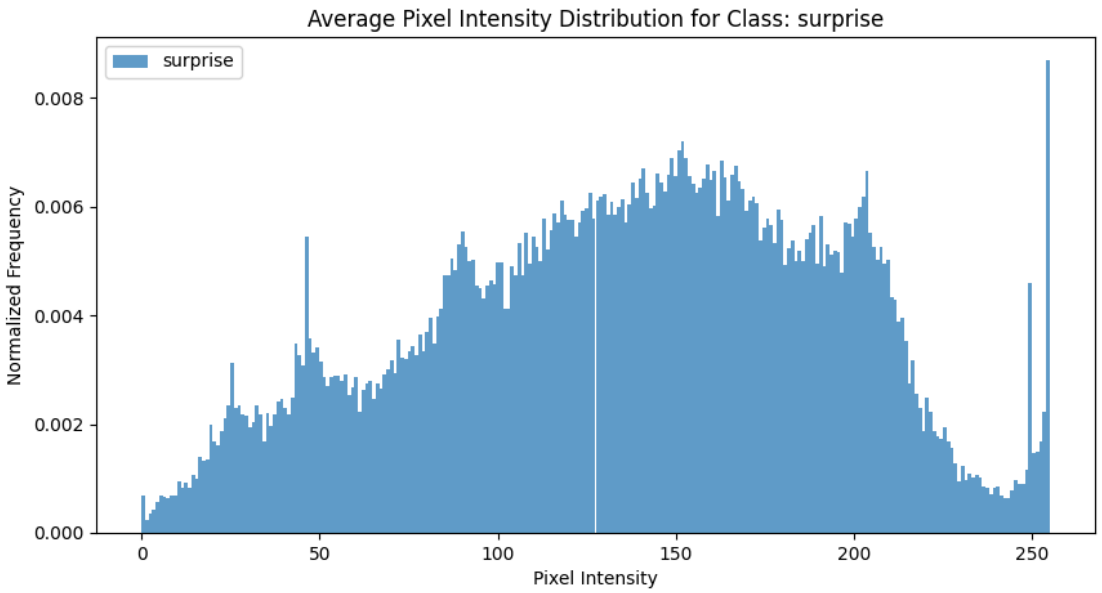
# Pixel Intensity Distribution

The pixel intensity histograms are pivotal for understanding the variations within our image data. It is observed that the pixel intensity for most classes is centered around mid-range values which indicate well-distributed lighting conditions across the images. However, the spikes towards the extreme ends of the spectrum hint at the presence of some dark or bright images.




Average Pixel Intensity Distribution for Class: neutral
neutral
Normalized Frequency
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
50
100
150
200
250
Pixel Intensity


Average Pixel Intensity Distribution for Class: surprise
surprise
Normalized Frequency
0.008
0.006
0.004
0.002
0.000
0
50
100
150
200
250
Pixel Intensity

# CNN Model for Facial Expression Recognition

## Model Overview and Architecture Details

The convolutional neural network (CNN) developed for the recognition of facial expressions is designed to classify images into four categories: "engaged," "happy," "neutral," and "surprised." The model, named SimpleCNN, is a pyramidal structure where the complexity and depth increase as the spatial dimensions decrease, which is characteristic of CNNs used in image recognition tasks.

### 1. Architecture

**Input Layer:** The network accepts grayscale images with a resolution of 48x48 pixels.

**First Convolutional Layer:** Consists of 32 filters with a kernel size of 3x3 and padding of 1. This layer is followed by a ReLU activation function and a max-pooling layer with a kernel size of 2x2, which reduces the spatial dimensions by half.

**Second Convolutional Layer:** Similar to the first but with 64 filters, doubling the depth to capture more complex features.

**Flattening Layer:** After the second pooling operation, the data is flattened to transition from 2D feature maps to 1D feature vectors.

**Fully Connected Layer:** The flattened data is then passed through a dense layer with 128 neurons, accompanied by a ReLU activation and a dropout regularization with a rate of 0.5 to reduce overfitting.

**Output Layer:** A final fully connected layer with 4 neurons corresponds to the number of classes, with no activation function since this is handled by the loss during training.

### 2. Regularization Techniques

**Dropout:** A dropout layer with a rate of 0.5 is used after the first fully connected layer to prevent overfitting by randomly setting neuron outputs to zero during training.

**Data Normalization:** Input data is normalized to have a mean of 0.5 and a standard deviation of 0.5, which helps with the convergence during training.

## 3. Variant Models

Two variant models based on the original SimpleCNN model were developed by slightly adjusting hyperparameters with the goal of finding the best-performing model.

The first variant model varies from the original model in its number of convolutional layers whereas the second variant model varies from the original model in its layers' kernel sizes.

Both models were trained and evaluated on the same datasets as the original SimpleCNN model.

**Variant 1: Vary the Number of Convolutional Layers**

The first variant model has nearly the same architecture as the original model but for the fact that it has 1 extra convolutional layer. In both models, the first layer has 32 filters, kernel size of 3x3, and a padding of 1. Also in both models the second layer is the same but has double the filters of the 1st layer. In this variant, the third convolutional layer follows suit of the second with everything the same and a doubled depth once again at 128 filters.

Another difference in this variant is that due to the extra layer, the input size of the fully connected layer is changed. In the original model, this is (64 * 12 * 12), whereas in this variant it is changed to (128 * 6 * 6) due to the difference in structure from the extra convolutional layer.

**Variant 2: Experiment with Different Kernel Sizes**

The second variant model once again has nearly the same architecture as the original model but for the fact that the kernel sizes are different in the first two layers. Unlike Variant 1, this variant has 2 convolutional layers just like the original model.

In the original model, both the first and second convolutional layers have a kernel size of 3x3. In this variant model, we experimented with a few different kernel sizes with the most recent version of the model using a kernel size of 5x5 in its first layer and 2x2 in its second layer.

# Training Process

## 1. Methodology

The training process is executed over a minimum of 10 epochs, utilizing the Adam optimizer with a learning rate of 0.001. The model employs the cross-entropy loss function, a standard choice for multi-class classification problems.

## 2. Hyperparameters

**Batch Size:** 64 images per batch allow for efficient computation while providing a reliable estimate of the gradient.

**Learning Rate:** Set at 0.001, which is a common starting point for the Adam optimizer, balancing the speed of convergence and the risk of overshooting minima.

**Optimizer:** The Adam optimizer is known for combining the benefits of two other extensions of stochastic gradient descent: Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp).

## 3. Optimization Techniques

**Early Stopping:** The training loop includes an early stopping mechanism, where training can be halted if the validation loss does not improve after three consecutive epochs. This is to prevent overfitting and ensure that the model generalizes well to unseen data.

**Best Model Saving:** Instead of keeping the final model at the last epoch, the training script saves the state of the model with the lowest validation loss, ensuring that the best generalizing model is retained.

## 4. Training Results

The model demonstrated a performance of 55-65% accuracy on the validation set across different training runs. This variability is an expected outcome due to factors inherent to the training of neural networks, including but not limited to random weight initialization, stochastic gradient descent, and the non-deterministic nature of GPU computations.

Despite these variations, the model consistently classified over half of the images correctly, highlighting its ability to generalize from the training data to unseen samples. The validation loss and accuracy were carefully monitored after each epoch, guiding the early stopping and model-saving mechanisms.

# Evaluation (Revised)

## Performance Metrics

| **Model** | **Macro** | | | **Micro** | | | **Accuracy** |
|---|---|---|---|---|---|---|---|
| | P | R | F | P | R | F | |
| Original Model | 53.80% | 52.28% | 51.37% | 53.85% | 53.85% | 53.85% | 53.85% |
| Variant 1 | 56.56% | 56.69% | 54.00% | 58.46% | 58.46% | 58.46% | 58.46% |
| Variant 2 | 56.57% | 56.99% | 56.49% | 58.80% | 58.80% | 58.80% | 58.80% |

**Performance Overview:**

- The Original Model shows the lowest performance across almost all metrics, with a Macro Precision of 53.80%, Macro Recall of 52.28%, Macro F1-Score of 51.37%, and an Accuracy of 53.85%. The Micro Precision, Recall, and F1-Score mirror the Accuracy at 53.85%, indicating a relatively balanced performance across the different classes.
- Variant 1 shows improved performance over the Original Model across all metrics. It has a Macro Precision of 56.56%, a Macro Recall of 56.69%, and a Macro F1-Score of 54.00%. The Micro measures and Accuracy all stand at 58.46%. This demonstrates a consistent improvement in classification performance across all classes when compared to the Original Model.
- Variant 2 displays a similar trend of improvement over the Original Model. It has a Macro Precision of 56.57%, Macro Recall of 56.99%, and a Macro F1-Score of 56.49%. It surpasses the Original Model in Accuracy at 58.80%, as well as in Micro Precision and F1-Score, both at 58.80%, and Micro Recall at 58.80%.

**Our Understanding:**

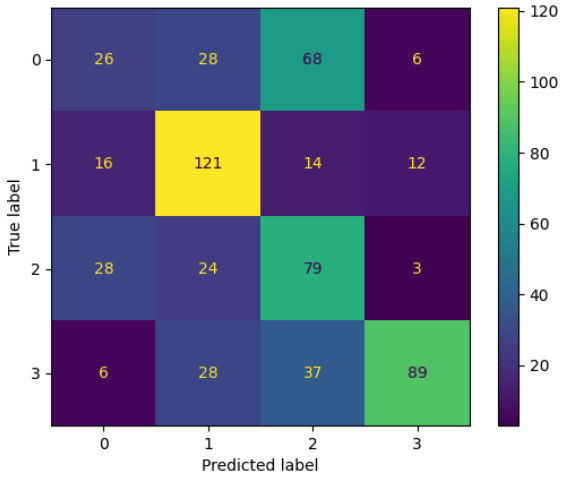
- The increase in the Macro Precision and Recall from the Original Model to Variants 1 and 2 suggests that the variants are better at predicting positive samples for each class and are also less likely to miss positive samples.
- The Micro averages being close to the Macro averages indicate a balanced dataset, or that the models perform equally well across classes of different sizes.
- The consistent Micro and Macro F1-Scores suggest that there is a balance between the precision and recall in these models, which is often sought after in multi-class classification problems.
- The equal Micro metrics and Accuracy for Variant 1 suggest that this model performs consistently across different instances and classes, which might be beneficial if each class is equally important.
- Despite Variant 1 and Variant 2 having the same Accuracy, Variant 2 has a higher Macro Recall but lower Macro Precision, suggesting a trade-off between these two metrics.

# Confusion Matrix Analysis

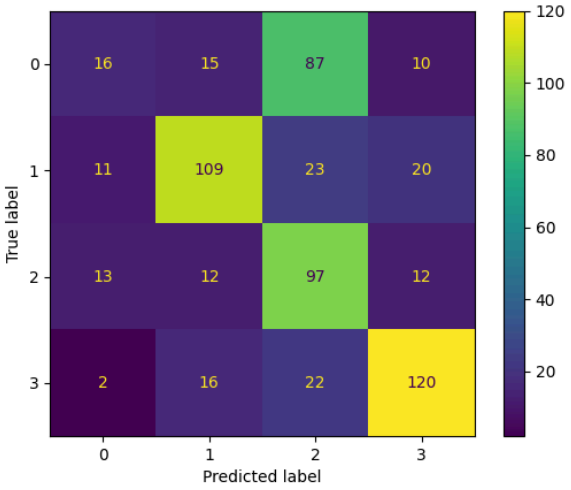
## PART II Confusion Matrices (Revised)

Labels:
**‘0’ - engaged**
**‘1’ - happy**
**‘2’ - neutral**
**‘3’ - surprised**

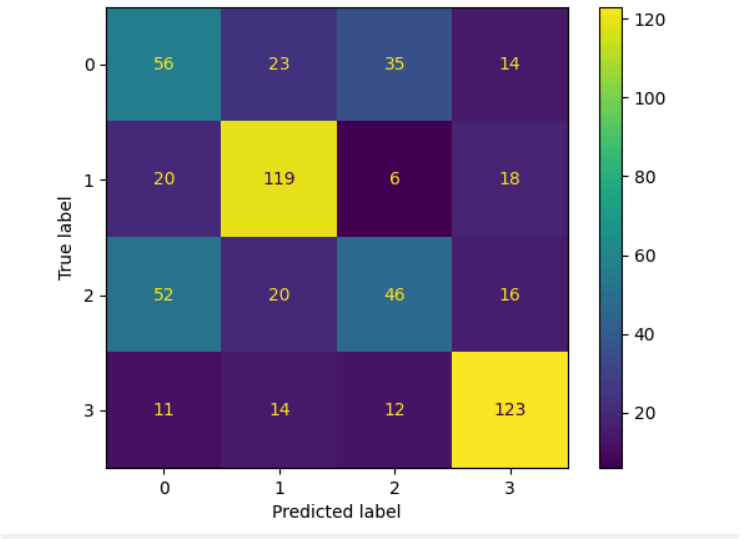
### Confusion Matrix for Original Model:




### Confusion Matrix for Variant 1 Model:

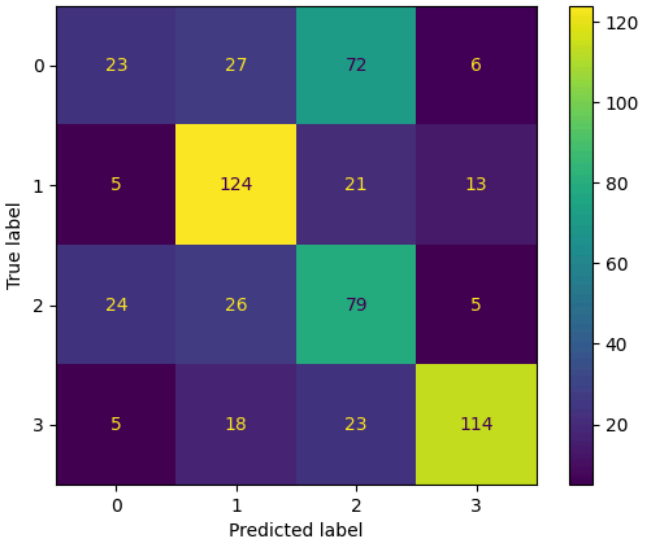


**Confusion Matrix for Variant 2 Model:**




## PART III Confusion Matrix - Best Model - Variant 2 Retrained

Labels:
**‘0’ - engaged**
**‘1’ - happy**
**‘2’ - neutral**
**‘3’ - surprised**



## Impact of Architectural Variations

### 1. Depth of Model (Number of Convolutional Layers)

The original model has two convolutional layers, while Variant 1 introduces an additional convolutional layer.

As per the metrics, Variant 1 has a higher macro precision, recall, and F1 score compared to the original model.

The additional convolutional layer in Variant 1 might allow it to capture more detailed features compared to the original model. The performance improvement suggests that the deeper architecture slightly enhanced the model's ability to discriminate between different facial expressions.

There's no clear evidence of overfitting based on the results, as the accuracy and other metrics do not indicate significant deviations between training and testing performance. However, further investigation would be necessary to confirm this.

### 2. Kernel Size Variations

Variant 2 uses different kernel sizes compared to the original model, with a larger 5x5 kernel for convolutional layer 1 and a smaller 2x2 kernel for convolutional layer 2.

Despite these variations, the performance metrics of Variant 2 are slightly higher than those of Variant 1, with only minor differences in precision, recall, and F1 score. Kernel size variations can influence the model's ability to recognize different facial features. Larger kernel sizes might capture broader features, while smaller kernel sizes can focus on finer details.

The performance of Variant 2 seems to be significantly affected by the kernel size variations.

## K-fold Cross-validation

| Fold | Macro | | | Micro | | | Accuracy |
|---|---|---|---|---|---|---|---|
| | P | R | F | P | R | F | |
| 1 | 72.6% | 72.1% | 70.9% | 70.8% | 70.1% | 70.8% | 70.84% |
| 2 | 72.1% | 72.1% | 71.1% | 70.8% | 70.1% | 70.8% | 70.88% |
| 3 | 67.6% | 67.1% | 66.1% | 68.6% | 68.5% | 68.5% | 68.57% |
| 4 | 68.4% | 68.5% | 66.4% | 66.9% | 66.9% | 66.9% | 66.98% |
| 5 | 73.3% | 70.9% | 69.7% | 71.1% | 71.1% | 71.1% | 71.11% |
| 6 | 70.8% | 71.2% | 70.4% | 70.5% | 70.5% | 70.5% | 70.47% |
| 7 | 70.9% | 70.4% | 69.7% | 71.1% | 71.1% | 71.1% | 71.11% |
| 8 | 72.1% | 72.2% | 71.5% | 72.1% | 72.1% | 72.1% | 72.06% |
| 9 | 68.7% | 68.3% | 67.9% | 69.8% | 69.8% | 69.8% | 69.84% |
| 10 | 66.6% | 67.1% | 65.4% | 66.1% | 66.1% | 66.3% | 66.03% |
| **Average** | 70.3% | 69.9% | 68.9% | 69.8% | 69.8% | 69.7% | 69.79% |

**Table 1**: Part 2 Model

| Fold | Macro | | | Micro | | | Accuracy |
|---|---|---|---|---|---|---|---|
| | P | R | F | P | R | F | |
| 1 | 79.1% | 78.3% | 77.3% | 75.9% | 75.4% | 75.6% | 75.94% |
| 2 | 77.2% | 73.8% | 73.8% | 74.8% | 74.6% | 74.8% | 74.68% |
| 3 | 77.7% | 75.8% | 75.7% | 77.1% | 77.7% | 77.4% | 77.77% |
| 4 | 77.4% | 75.5% | 75.9% | 78.9% | 78.1% | 78.1% | 78.09% |
| 5 | 80.64% | 79.7% | 79.3% | 80.6% | 80.5% | 80.5% | 80.63% |
| 6 | 81.76% | 79.6% | 79.2% | 80.0% | 80.0% | 80.4% | 80.00% |
| 7 | 76.6% | 73.1% | 72.9% | 76.5% | 76.5% | 76.6% | 76.50% |
| 8 | 81.1% | 79.6% | 78.7% | 78.7% | 78.7% | 78.7% | 78.73% |
| 9 | 78.8% | 75.1% | 75.7% | 77.1% | 77.1% | 77.4% | 77.14% |
| 10 | 81.1% | 77.3% | 78.1% | 80.0% | 80.1% | 80.5% | 80.00% |
| **Average** | 79.2% | 76.8% | 76.9% | 77.9% | 77.8% | 77.6% | 77.95% |

**Table 2:** Part 3 Model

## Observations Across Different Folds

Across different folds, we can observe that the performance metrics of Macro Precision, Recall, F1-Score, and Accuracy show consistency in the range of their values, indicating stability in the model's predictive capability across various subsets of data. In Part II and Part III, there is no significant fluctuation between the folds, which suggests that the model generalizes well and is not overfitted to a specific part of the data. For Part II, accuracy ranges from approximately 66% to 72%, whereas for Part III, it is slightly higher, ranging from approximately 74% to 80%. This suggests that the updates made in Part III have contributed to a more robust model that performs better across different subsets of the data.

## K-fold Cross-validation vs. Original Train/Test Split

**Original Train/Test Split:**
Original Model Metrics:
Macro Precision: 53.80%
Macro Recall: 52.28%
Macro F1-Score: 51.37%
Accuracy: 53.85%
Micro P/R/F: 53.85%

Variant 1 Metrics:
Macro Precision: 56.56%
Macro Recall: 56.69%
Macro F1-Score: 54.00%
Accuracy: 58.46%
Micro P/R/F: 58.46%

Variant 2 Metrics:
Macro Precision: 56.57%
Macro Recall: 56.99%
Macro F1-Score: 56.49%
Accuracy: 58.80%
Micro P/R/F: 58.80%

**K-fold Cross-validation (Part II Model):**
Average Macro Precision: 70.30%
Average Macro Recall: 69.95%
Average Macro F1-Score: 68.93%
Average Accuracy: 69.79%
Average Micro P/R/F: 69.79%

## Analysis:

The k-fold cross-validation results exhibit a significantly higher performance compared to the original train/test split evaluation, with average accuracies and other metrics around the **69-70%** range compared to approximately **54-59%** from the original evaluation.

**Variance in Data**: The k-fold method evaluates the model across different data subsets, reducing the bias that might come from a single data split. The higher metrics suggest that the model is more stable across varying data when the evaluation is more comprehensive.

**Data Imbalance:** If the original test set had a class imbalance or data points that were not representative of the general population, it might have skewed the performance metrics. K-fold helps smooth out this issue by ensuring each fold accurately represents the whole.

In conclusion, the k-fold cross-validation results suggest that the model's performance is likely better than originally estimated. It underscores the importance of using multiple evaluations to mitigate the effects of data-specific issues and provides a clearer understanding of the model's ability to generalize.

# Bias Analysis

## Introduction

In this bias analysis, we chose the two bias attributes: age and gender. Each demographic group had several subcategories: the age category was divided into three categories/groups: young people, middle-aged people, and seniors while gender was divided into male and female groups.

We manually segmented the raw dataset used for training the models in Part II by handpicking the appropriate images and placing them into their subcategories/groups (where each subcategory had the four emotion categories - engaged, happy, neutral, and surprised.) Each group contains the same amount of images as the other groups and it was ensured that the dataset was balanced to detect biases in the model.

A script then loads the pre-trained CNN model (Variant 2), evaluates its performance on age and gender classification tasks using image datasets, and saves the evaluation results to CSV files for further analysis.

## Bias Detection Results

| Attribute | Group | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|---|
| Age | Young | 74.77% | 73.97% | 74.06% | 73.11% |
| | Middle-aged | 75.44% | 77.54% | 75.11% | 74.36% |
| | Senior | 57.11% | 61.99% | 56.37% | 55.16% |
| | **Average** | 69.11% | 71.17% | 68.51% | 67.54% |
| Gender | Male | 75.38% | 77.48% | 75.56% | 75.10% |
| | Female | 77.70% | 77.70% | 77.74% | 76.97% |
| | **Average** | 76.54% | 77.59% | 76.65% | 76.04% |
| **Overall System Average** | | 72.08% | 73.74% | 71.77% | 70.94% |

*Model from Part 2*

It can be seen from the table above, that the model performs significantly differently for the "senior" category compared to the "young" and "middle-aged" groups.

The notably lower scores for "Seniors" could be due to the model being trained on a dataset having originally a smaller sample size of this category, leading to less accurate performance metrics.

Conversely, the model's scores for the male and female groups were nearly identical, and so they did not indicate any sort of bias in terms of gender.

## Bias Mitigation Steps

To mitigate bias from the dataset, we adopted a multi-step approach. Initially, we recognized the need to diversify the dataset to ensure fair representation across different demographic groups. We manually curated additional images from publicly available datasets, particularly leveraging the Facial Expression Recognition 2013 (FER 2013) dataset. These additional images were incorporated into our training dataset to provide more comprehensive coverage of various demographic groups.

Upon analysis, we observed that the demographic group "seniors", had a significantly lower representation in the dataset compared to others. To solve this imbalance, we ensured that each emotion category for the "seniors" group had an equal representation by including 130 images for each emotion category. This adjustment aimed to address the underrepresentation of the "seniors" demographic and promote fairness in model training.

However, during subsequent evaluations, we noticed a disproportionate impact on the accuracy of the model for the "middle-age" group. Further investigation revealed that the increased sample size for this group inadvertently skewed the model's performance, leading to higher accuracy but potentially introducing bias. To mitigate this issue, we implemented a manual removal process, wherein we systematically balanced the dataset by removing excess images from the "middle-age" group.

Following these adjustments, we conducted a final bias analysis test to assess the effectiveness of our mitigation efforts. The results indicated a notable improvement in the model's fairness and accuracy across different demographic groups, demonstrating the efficacy of our bias mitigation strategies.

## Comparative Performance Analysis

| Attribute | Group | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|---|
| Age | Young | 80.00% | 80.18% | 79.56% | 79.32% |
| | Middle-aged | 78.72% | 82.77% | 79.31% | 78.79% |
| | Senior | 77.64% | 78.49% | 77.14% | 77.33% |
| | **Average** | 78.79% | 80.48% | 78.67% | 78.48% |
| Gender | Male | 80.65% | 83.43% | 80.47% | 80.59% |
| | Female | 82.50% | 84.02% | 82.47% | 81.46% |
| | **Average** | 81.58% | 83.73% | 81.47% | 81.03% |
| **Overall System Average** | | 79.90% | 81.78% | 79.79% | 79.50% |

*Model from Part 3 (retrained)*

The results in this table above show the scores from our best model (variant 2) which was retrained on our dataset after we adjusted it with our bias mitigation efforts. The model was tested on the same exact data splits of Age and Gender as the model from Part 2, ensuring that any differences in scores directly reflect the models' differences after retraining.

One should notice immediately that the model's performance for the Senior group is higher by roughly 20% across all scores. This significant improvement shows that our efforts of including 130 images for each emotion category for "senior" worked to mitigate bias on that account. Another point of note in regards to the "senior" group is that the recall score had the highest improvement at 21%, suggesting that the bias mitigation really affected how well the model can correctly find all images of a class within "senior".

The numbers also show how much more balanced the dataset became after bias mitigation. Beforehand, the largest gap in performance on each group was a concerning 18.33% difference between "senior" and "middle-aged", whereas the largest gap for the retrained model is a mere 2.36% between "senior" and "young".

Once again, just as for the previous model, there is no significant difference between the performance for the Gender groups of "male" and "female", although there is a slight increase in average performance on Gender.

Finally, not only does the retrained model show less bias but it also shows better overall performance than the model from Part 2, as seen in its 7.82% increase in overall system average. This is likely due to a number of factors including successful bias mitigation and further balancing of classes in the dataset.

# References

[Manas Sambare], "FER-2013," [Online]. Available: https://www.kaggle.com/datasets/msambare/fer2013. [Accessed: March 10, 2024].