Final Project:

Assignment: Building a Facial Expression Recognition Model Using the FER 2013 Dataset

Objective

The goal of this assignment is to build a convolutional neural network (CNN) for recognizing facial expressions using the FER 2013 dataset. You will preprocess the data, design and train the model, evaluate its performance, and document your findings.

Deliverables

- **Code**: A Jupyter notebook containing the complete code for the project.
- **Report**: A detailed report explaining your approach, experiments, results, and conclusions.
- **Video Presentation**: Explaining the project in the video, Max. Video length 10 min.

  Video includes:

  - **Description of the Problem Statement** your project was trying to solve
  - **Proposed Solution**
  - **Results** (can include model accuracy, website demo, etc)
  - Optional - **Code Overview**

Instructions:

1. **Load the Dataset**
   - Load the FER 2013 dataset from the link provided - https://www.kaggle.com/datasets/ahmedmoorsy/facial-expression?select=fer2013.csv

     .

2. **Data Preprocessing**
   - Normalize the pixel values to the range [0, 1].
   - Resize the images if necessary (the original size is 48x48 pixels).
   - Perform data augmentation (e.g., rotations, shifts, flips) to increase the training data's diversity (optional, but it's better if you do it)

3. **Data Splitting**
   - Split the data into training, validation, and test sets. A common split is 80% training, 10% validation, and 10% testing.

Step 2: Model Building

1. **Choose a Model Architecture**
    - Start with a simple CNN architecture. You can explore more complex architectures like VGG or ResNet later if you want.

2. **Define the Model**
    - Use TensorFlow and Keras to define the model. Include convolutional layers, pooling layers, and fully connected layers.

Step 3: Model Training

1. **Compile the Model**
    - Specify the loss function (e.g., categorical cross-entropy), optimizer (e.g., Adam), and evaluation metrics (e.g., accuracy).
2. **Train the Model**
    - Fit the model to the training data, using the validation set to monitor performance and prevent overfitting.
3. **Hyperparameter Tuning**
    - Experiment with different hyperparameters like learning rate, batch size, and number of epochs to optimize the model.

Step 4: Model Evaluation

1. **Evaluate the Model**
    - Test the model on the test set and report the accuracy and other relevant metrics.

Final submission will include you code and a report including the process you followed and results.