#### 

#### **University of Information Technology & Sciences**

#### **Department of Computer Science and Engineering**

#### **Course Name: Digital Image Processing Lab**

#### **Course Code: CSE 438**

**Group:8**

**Project Title: Emotion Detection from Images using CNN**

**Project Report**

| **Submitted by:** Arbin zaman  Id: 2125051006  Sohana Afrin  Id: 2125051013  Safin Ahamed Sajid  Id: 2125051022  Md. Fahim Abrar Asif  Id: 2125051116 | **Submitted To:****Audity Ghosh****Lecturer****University of Information Technology & Sciences** |
| --- | --- |

### 

### **1. Theory**

Emotion detection from facial expressions is a significant area in computer vision and artificial intelligence. It involves analyzing a person's facial features to identify and classify their emotional state, such as happiness, sadness, anger, surprise, fear, disgust, and neutral. This task requires understanding subtle patterns and variations in facial muscle movements.

To achieve accurate emotion recognition, **Convolutional Neural Networks (CNNs)** are widely used due to their powerful ability to process and learn from image data. CNNs automatically extract and learn **hierarchical features** from facial images — starting from simple edges and textures to more complex shapes and facial structures. These features help the model to distinguish between different emotional expressions.

In this project, a CNN-based model is developed and trained using the **FER-2013 dataset**, which is a popular benchmark dataset for facial emotion recognition. The FER-2013 dataset contains thousands of labeled grayscale facial images categorized into seven basic emotion classes. The diversity and size of this dataset help in building a robust and generalized emotion classification model.

By leveraging deep learning techniques and the structure of CNNs, the model can effectively learn emotional patterns from facial expressions and predict the correct emotion with high accuracy. This technology can be applied in various fields such as **mental health monitoring**, **human-computer interaction**, **customer service**, and **security systems**, making it a valuable tool in modern AI applications.

### **2. Objectives**

1. To develop a CNN-based system for emotion detection from facial images.
2. To use the FER-2013 dataset to train and evaluate the performance of the model.
3. To allow users to upload images and predict emotions in real time.

### **3. Tools and Environmental Setup**

* **Programming Language**: Python
* **Development Environment**: Google Colab
* **Libraries/Tools Used**:  
  + TensorFlow/Keras
  + OpenCV
  + Matplotlib, NumPy
  + Kaggle API for dataset download

### 

### 

### 

### **4. Methodology**

1. **Install Dependencies**: Kaggle API, OpenCV, TensorFlow.
2. **Data Acquisition**: Download the FER-2013 dataset using the Kaggle API.
3. **Data Preprocessing**:  
   1. Resize images to a fixed shape.
   2. Normalize pixel values.
   3. One-hot encode emotion labels.
4. **Model Building**: A CNN with convolutional, pooling, dropout, and dense layers.
5. **Training & Validation**: Split dataset into training and validation sets, monitor accuracy and loss.
6. **Evaluation**: Predict emotions on test images and visualize predictions.
7. **Real-Time Inference**: Load user image and predict emotion using the trained model.

**5. Gist Code with Comments and Output Images**

#### **Data Preprocessing**

# Normalize the images and one-hot encode labels

X\_train = X\_train / 255.0

X\_test = X\_test / 255.0

y\_train = to\_categorical(y\_train, 7)

y\_test = to\_categorical(y\_test, 7)

#### **CNN Model Architecture**

model = Sequential()

model.add(Conv2D(32, (3,3), activation='relu', input\_shape=(48, 48, 1)))

model.add(MaxPooling2D(2,2))

model.add(Dropout(0.25))

model.add(Conv2D(64, (3,3), activation='relu'))

model.add(MaxPooling2D(2,2))

model.add(Dropout(0.25))

model.add(Flatten())

model.add(Dense(128, activation='relu'))

model.add(Dropout(0.5))

model.add(Dense(7, activation='softmax')) # 7 emotion classes

#### **Model Training**

model.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy'])

history = model.fit(X\_train, y\_train, epochs=30, validation\_data=(X\_test, y\_test))

#### 

#### 

#### 

#### **Emotion Prediction from Image**

def predict\_emotion(img\_path):

img = cv2.imread(img\_path, cv2.IMREAD\_GRAYSCALE)

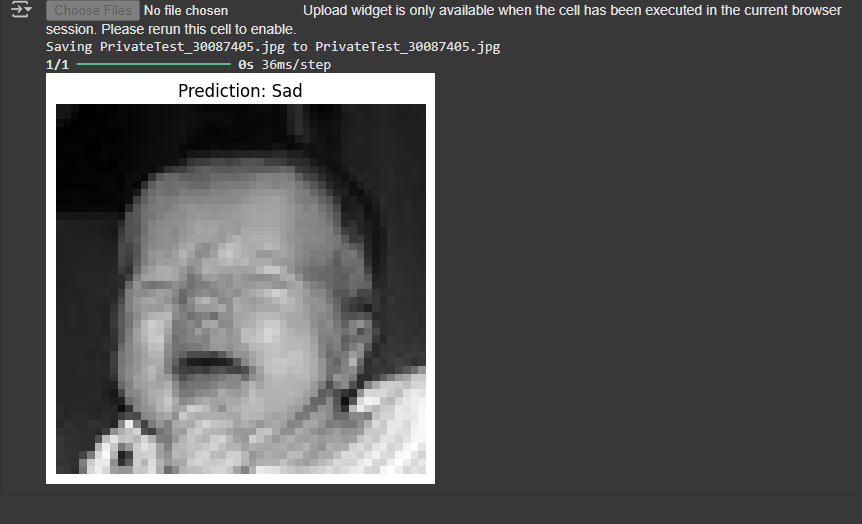
img = cv2.resize(img, (48,48)) / 255.0

img = img.reshape(1, 48, 48, 1)

prediction = model.predict(img)

return emotion\_labels[np.argmax(prediction)]

**Output Image**



### 

### 

### **6. Gist Code Explanation**

1. **Conv2D Layers** extract spatial features from facial images.
2. **MaxPooling2D** reduces spatial dimensions and helps in feature generalization.
3. **Dropout Layers** help prevent overfitting.
4. **Dense Layers** act as classifiers; the final one uses softmax to output probabilities for 7 emotion classes.

### **7. Result Description**

1. Achieved around ~60% accuracy on validation data (exact figure to be inserted from notebook).
2. Successfully predicted emotions on uploaded images with decent confidence.
3. Visual feedback of predictions through matplotlib or OpenCV displays.

### **8. Conclusion (Gap Mention)**

The project successfully demonstrates emotion detection using CNN and FER-2013. However, the model:

1. Struggles with less-represented classes (e.g., fear, disgust).
2. Is sensitive to poor lighting and occluded faces.
3. Was trained on grayscale images only, limiting color-based emotional cues.

### **9. Future Work**

* Integrate face detection for better preprocessing.
* Expand the dataset with colored and higher-resolution images.
* Implement on-device inference using TensorFlow Lite.
* Combine with audio and text modalities for multimodal emotion recognition.

### 

### 

### 

### 

### **10. References**

1. Goodfellow, Ian, et al. "Challenges in representation learning: A report on three machine learning contests." International Conference on Neural Information Processing. Springer, Berlin, Heidelberg, 2013.
2. Kaggle FER-2013 Dataset: https://www.kaggle.com/datasets/msambare/fer2013
3. Chollet, François. *Deep Learning with Python*. Manning Publications, 2017.
4. OpenCV Documentation: https://docs.opencv.org/
5. TensorFlow Keras API: https://www.tensorflow.org/api\_docs/python/tf/keras