MODEL DEVELOPMENT DOCUMENT

MODEL NAME

CNN-powered Emotion Detection System

OBJECTIVE

To develop a robust and accurate deep learning model capable of classifying facial expressions into seven basic emotions (anger, disgust, fear, happiness, sadness, surprise, and neutral) using transfer learning with MobileNetV2. The model should achieve high accuracy on unseen data and be deployable for real-world applications such as human-computer interaction, emotion analysis, and mental health monitoring.

BRIEF

This project focuses on developing an emotion detection model using deep learning. The model leverages transfer learning with MobileNetV2 to classify facial expressions into seven basic emotions: anger, disgust, fear, happiness, sadness, surprise, and neutral. It is trained on a large dataset of facial images and evaluated on its accuracy in recognizing emotions from unseen data. This technology has potential applications in various fields, such as human-computer interaction, emotion analysis, and mental health monitoring.

DELIVERABLES OF THE PROJECT

- Analysing facial expression whether it is sad, happy, angry.
- Understanding emotion status

DATASET SOURCE

Dataset: Emotion Detection FER

Source: Kaggle

Description:

- The dataset contains approximately 35,000 grayscale images of faces.
- Each image is labeled with one of seven basic emotions: anger, disgust, fear, happiness, sadness, surprise, and neutral.
- The images are preprocessed by resizing them to (48, 48) pixels.
- The dataset is split into training and testing set

MODEL ARCHITECTURE

Model:

Transfer learning approach using MobileNetV2 as the base model.

Base Model:

MobileNetV2 pre-trained on ImageNet.

Custom Layers:

- GlobalAveragePooling2D to reduce spatial dimensions.
- Dropout for regularization.
- Dense layers for classification (64 units with ReLU activation and 7 units with softmax activation).

Freezing:

Base model layers are initially frozen (base_model.trainable = True).

Training

- **Optimizer:** Adam optimizer.
- **Loss Function:** Sparse categorical cross-entropy.
- **Metrics:** Accuracy.
- **Epochs:**15 .

Batch Size: 16.

Callbacks:

- Early stopping to prevent overfitting (patience=3).
- Model Checkpoint to save the best model ('best_mobilenet_model.h5').

Evaluation

- **Validation Set:** The model's performance is evaluated on the validation set using model evaluate
- **Metrics:** Loss and accuracy are reported.
- **Visualization:** Training and validation accuracy are plotted over epochs.

The code utilizes transfer learning with MobileNetV2 to build an emotion detection model. Here's a breakdown of the key steps:

Data Acquisition and Preparation:

- Downloads the "emotion-detection-fer" dataset from Kaggle and extracts it.
- Creates training and validation datasets from the images, resizing them to (160, 160).
- Preprocesses images using MobileNetV2's preprocess input function.

Model Building:

- Loads a pre-trained MobileNetV2 model without its classification layers (include top False).
- Adds a global average pooling layer, two dropout layers for regularization, and two dense layers for classification.
- The final layer uses softmax activation for multi-class emotion prediction.

Model Compilation:

Compiles the model with the Adam optimizer, sparse categorical cross-entropy loss (suitable for multi-class problems), and accuracy as the evaluation metric.

Training and Validation:

- Trains the model using the training dataset and validates it with the validation dataset for 15 epochs.
- Implements early stopping to prevent overfitting and saves the best model weights.

Visualization and Evaluation:

- Plots the training and validation accuracy over epochs for visual analysis.
- Evaluates the trained model on the validation dataset to assess its performance on unseen data, printing the final loss and accuracy.

SOFTWARE

1. Google Colab: The primary environment for developing, training, and evaluating your emotion detection model. It provides the necessary computational resources and pre-installed libraries.
2. Python: The core programming language for implementing your model, data preprocessing, and visualization.
3. TensorFlow: The deep learning framework used for building and training your model. It provides functionalities for defining neural network architectures, optimizing model parameters, and evaluating performance.
4. Keras: A high-level API built on top of TensorFlow, simplifying model development and training. It offers a user-friendly interface for creating and managing neural networks.
5. MobileNetV2: A pre-trained convolutional neural network architecture used as the base model for your emotion detection model. You leveraged transfer learning to utilize its learned features for recognizing emotions.
6. Kaggle API: Used for downloading the "Emotion Detection FER" dataset from Kaggle, providing the necessary training data for your model. It enables programmatic access to Kaggle's resources.
7. Matplotlib: A plotting library used for visualizing training progress and model performance. You likely used it to create plots of training accuracy and other metrics.

INDIVIDUAL DETAIL

Name: Arun Gourh

Email-id: arungourh12935@gmail.com

CONCLUSION

This project successfully demonstrated the feasibility of building an emotion detection model using transfer learning with MobileNetV2. While there is room for further improvement, the achieved accuracy and insights gained from this study provide a solid foundation for future research and development in this area. The model has the potential to be utilized in various applications, such as human-computer interaction, affective computing, and mental health monitoring