FACE MASK DETECTION

CSD TEAM -07

TEAM MEMBERS

M.SHANMUKHA RAM(216Q1A4433)

CH.VENKATA SANDEEP(216Q1A4420)

CH.BHANU SANKAR(216Q1A4407)

G.KALYAN REDDY(216Q1A4409)

B.DHANUNJAY(216Q1A4439)

Abstract of Face Mask Detection:

Face mask detection involves recognizing whether individuals in an image or video are wearing a mask or not. It combines image processing and deep learning techniques for object detection and classification. The primary workflow is as follows:

1. **Image/Video Input:** The system captures images or videos from a camera feed.
2. **Face Detection:** Using computer vision techniques (e.g., Haar cascades, HOG + SVM, or deep learning-based models like YOLO or SSD), the system detects the regions in the image containing faces.
3. **Mask Classification:** Once faces are detected, a deep learning classifier, typically a Convolutional Neural Network (CNN), is used to classify whether each detected face is wearing a mask or not.

4. **Output Results:** The system provides the classification results (e.g., "Mask," "No Mask") and highlights the identified faces in the input using bounding boxes.
5. **Alert Mechanisms (Optional):** If a person is detected without a mask, the system can trigger alerts, such as sending notifications or sounding alarms.

Technologies Used in Face Mask Detection:

1. **Programming Languages:**
    - **Python:** Most widely used for implementing machine learning and computer vision models due to its robust libraries.
    - **JavaScript:** For web-based implementations (e.g., real-time mask detection in browsers using TensorFlow.js).
2. **Deep Learning Frameworks:**
    - **TensorFlow/Keras:** For building and training deep learning models.
    - **PyTorch:** An alternative for creating neural networks and fine-tuning models.
    - **OpenCV:** For face detection and image preprocessing.
3. **Pre-Trained Models:**
    - **MobileNet, ResNet:** Lightweight and efficient models often used as backbones for classification tasks.
    - **YOLO (You Only Look Once):** For real-time face detection and classification.
    - **SSD (Single Shot Detector):** For object detection tasks, including mask detection.
4. **Libraries for Deployment:**

- **Flask/Django:** For creating backend APIs to serve the model.
- **TensorFlow Lite/ONNX:** For deploying models on edge devices or mobile platforms.
- **OpenCV:** For handling real-time video streams and displaying results.

5. **Hardware:**
   - **GPU/TPU:** For faster training and inference of deep learning models.
   - **Edge Devices:** Raspberry Pi or Jetson Nano for deploying the solution in real-world scenarios.
6. **Datasets:**
   - Public datasets such as the **"Face Mask Detection Dataset"** or custom datasets with labeled images of individuals wearing and not wearing masks.
7. **Tools for Training and Visualization:**
   - **Jupyter Notebook/Google Colab:** For building and testing models.
   - **Matplotlib/Seaborn:** For visualizing results like training accuracy and loss.
