**Real-time Facial Landmark Detection**

**Project Domain / Category**

Image Processing

**Abstract / Introduction**

Facial landmark detection is a computer vision task in which a model needs to predict key points representing regions or landmarks on a human’s face – eyes, nose, lips, and others. Facial landmark detection is a base task which can be used to perform other computer vision tasks, including head pose estimation, identifying gaze direction, detecting facial gestures, and swapping faces.

This project includes developing a real-time facial landmark detection system using Python. This project will involve identifying key points on a person's face, such as the eyes, nose, and lips, in images. This project will provide a valuable tool for various computer vision applications, making it easier to detect and analyze facial landmarks in real-time, opening the door to a wide range of possibilities in fields like biometrics, human-computer interaction, and emotion recognition.

**Functional Requirements:**

1. **Data Collection**

Gather a dataset of facial images with annotated landmarks. You can use publicly available datasets like 300-W, COFW, or create your own dataset.

1. **Preprocessing**

Resize and Normalize the data to enhance the accuracy of landmark detection.

Augment the dataset with transformations like rotation, scaling, and flipping for better model generalization.

1. **Model Selection**

Choose a deep learning model architecture suitable for facial landmark detection, such as a Convolutional Neural Network (CNN). Pre-trained transfer learning models like ResNet or MobileNet can also be used.

1. **Data Splitting:**

Split the dataset into training, validation, and test sets to train and evaluate the model's performance.

1. **Model Training**

Train the selected model using the training dataset. The loss function should be designed to minimize the difference between predicted and ground-truth landmark locations.

1. **Validation and Hyperparameter Tuning**

Validate the model's performance using the validation set and fine-tune hyperparameters like learning rate, batch size, and network architecture to achieve the best results.

1. **Testing and Evaluation**

Assess the model's accuracy on the test dataset to ensure it generalizes well to unseen data. Confusion Metrics can be used to evaluate performance.

1. **Real-time Implementation**

Integrate the trained model into a real-time Python application using libraries like OpenCV for video capture and display. The application should continuously detect facial landmarks from a webcam feed.

1. **Face Recognition**

Implement face recognition using the detected facial landmarks. You can use techniques like face embeddings or existing libraries like OpenCV or dlib for face recognition.

1. **User Interface**

Create a user-friendly interface for the application, allowing users to select functionalities of face recognition.

**Important links and Tutorials:**

* Python
  + - * <https://www.w3schools.com/python/>
      * <https://www.tutorialspoint.com/python/index.htm>
* Image processing
  + - * <https://regenerativetoday.com/some-basic-image-preprocessing-operations-for-beginners-in-python/>
      * <https://www.section.io/engineering-education/image-preprocessing-in-python/>
      * <https://www.tensorflow.org/tutorials/load_data/images>
* Deep Learning
  + - * <https://www.simplilearn.com/tutorials/deep-learning-tutorial/guide-to-building-powerful-keras-image-classification-models>
      * <https://www.analyticsvidhya.com/blog/2020/02/learn-image-classification-cnn-convolutional-neural-networks-3-datasets/>
* Transfer learning
  + - * <https://towardsdatascience.com/transfer-learning-for-image-classification-using-tensorflow-71c359b56673>
      * <https://www.kaggle.com/code/kmkarakaya/transfer-learning-for-image-classification/notebook>

**Hardware Requirement:**

* Processor –Core i3
* Hard Disk – 160 GB
* Memory – 12GB RAM
* Monitor

**Tools:**

**Language:** Python (Only python language)

**Framework:** Anaconda

**IDE:** JupyterNotebook, Pycharm, Spyder, Visual Studio Code, etc.

**Supervisor:**

Name: Madiha Faqir Hussain

Email ID: [madiha.hussain@vu.edu.pk](mailto:madiha.hussain@vu.edu.pk)

Skype ID: madiha.akhtar74