**ASL Skeleton Image Classifier Project Report**

**Project Overview**

The American Sign Language (ASL) Skeleton Image Classifier project is a real-time deep learning system designed to classify ASL letters from webcam video input. It uses MediaPipe to extract hand landmarks, processes them into skeleton images, and predicts the signed letter using a custom-trained Convolutional Neural Network (CNN).

**Key Technologies**

* **Python:** Core programming language.
* **OpenCV:** Captures and processes real-time webcam video frames.
* **MediaPipe:** Detects and renders hand landmarks into skeleton structures.
* **TensorFlow/Keras:** Used to build, train, and deploy the deep CNN model.
* **Scikit-learn:** Implements Random Forest baseline classifier.
* **Deque Buffering:** Smooths predictions for higher stability.

**Components**

**1. Dataset Creation (create\_landmark\_dataset.py)**

* **Image Preprocessing:**
  + Uses TensorFlow's ImageDataGenerator to rescale images and apply augmentations (rotation, zoom, width/height shift).
  + Data split into training (80%) and validation (20%) sets.
* **CNN Model:**
  + Sequential model with four convolutional layers increasing from 32 to 256 filters.
  + Max-pooling layers after each convolutional layer for downsampling.
  + Fully connected dense layer with 256 neurons followed by dropout (0.3) to prevent overfitting.
  + Output layer with softmax activation matching the number of ASL classes.
* **Training:**
  + 10 epochs of training using the Adam optimizer and categorical cross-entropy loss.
  + Training and validation accuracies tracked.
* **Saving:**
  + Trained model saved as asl\_skeleton\_cnn\_best.h5.

**2. Model Training (Alternative Baseline - train\_asl\_classifier.py)**

* **Random Forest Classifier:**
  + Loads landmark coordinate data from dataset.csv.
  + Splits data into training and testing sets.
  + Trains a Random Forest model with 100 trees.
  + Evaluates and prints a detailed classification report.
  + Saves the model as asl\_classifier.joblib.

**3. Real-time ASL Prediction (asl\_webcam\_predictor.py)**

* **Live Webcam Capture:**
  + Frames captured and processed in real time.
* **Hand Detection:**
  + MediaPipe Hands detects and draws hand landmarks.
* **Skeleton Image Construction:**
  + Skeletons rendered on a black canvas of 480x480 pixels.
  + Canvas resized to 128x128 before model prediction.
* **Prediction:**
  + Loads the pre-trained CNN (asl\_skeleton\_cnn\_best.h5).
  + Predicts the ASL letter and associated confidence score.
* **Buffer and Smoothing:**
  + Sliding window buffer (length 8) collects recent predictions.
  + Requires at least 6 consistent predictions with >90% confidence to finalize a letter.
  + Reduces jitter and noise in predictions.
* **Subtitle System:**
  + Builds a subtitle string from recognized letters.
  + Provides a cooldown to avoid repeated letter spamming.
* **User Controls:**
  + Press 'c' to clear the current subtitle.
  + Press 's' to save the subtitle to asl\_subtitle.txt.
  + Press 'x' to copy subtitle text to the clipboard (Windows only).
  + Press 'q' to quit.
* **Display:**
  + Shows live webcam feed side-by-side with the skeleton image.
  + Displays current letter prediction, confidence bar, subtitle text, and prediction buffer.

**Features**

* **Real-time, Stable Predictions:** Leveraging a buffer system to smooth out noisy predictions.
* **High Accuracy:** Custom CNN model trained on augmented datasets ensures robust performance.
* **User Interaction:** Save, copy, and clear subtitles easily.
* **Performance Optimized:** Fast inference even with heavy processing (128x128 input size and efficient CNN).

**Code Structure and Analysis**

* Highly modular, separating model training, baseline testing, and real-time prediction.
* Well-commented with clear step-by-step processes.
* Use of best practices for real-time deep learning inference and GUI display.
* Efficient error handling for webcam issues and file access.

**Conclusion**

The ASL Skeleton Image Classifier project demonstrates a powerful fusion of computer vision and deep learning for accessible communication solutions. Its smooth, accurate real-time predictions and thoughtfully designed user interactions make it a strong prototype for future ASL translation systems or educational tools.