Yoga Pose Identification and Feedback System

Introduction

This project focuses on building a machine learning model to identify yoga poses using human body key points extracted from images or videos. The system also provides feedback on alignment and accuracy of the poses, aiming to enhance yoga practitioners' techniques through automated analysis. The approach combines advanced neural network architecture and preprocessing techniques to ensure high accuracy and usability.

Approach

The system utilizes a pipeline that includes the following key stages:

1. Key point Extraction: Leveraging pose estimation models such as OpenPose or Mediapipe to extract 17 key points representing human body joints.
2. Data Preprocessing: Transforming raw key points into a normalized format suitable for training and inference.
3. Deep Neural Network: Designing and training a neural network for classification and feedback analysis.
4. Feedback Mechanism: Implementing post-processing techniques to evaluate alignment and suggest corrections.

Data Preprocessing

Data preprocessing is critical to ensure the robustness and generalization of the model. Below are the steps followed:

1. Key point Normalization:

- Normalize x and y coordinates of key points based on the dimensions of the input image.
- Calculate relative positions by anchoring key points to the pelvis as a reference point.

2. Handling Missing Data:

- Replace missing or low-confidence key points with interpolated values or a fixed placeholder.

3. Data Augmentation:

- Apply small rotations, flips, and scaling to simulate diverse yoga postures and camera angles.

4. Feature Engineering:

- Compute pairwise distances between key points to capture spatial relationships.

- Calculate angles between limbs to improve pose distinction.

5. Dataset Splitting:

- Split data into training (70%), validation (15%), and testing (15%) sets.

- Use stratified sampling to maintain class balance.

Model Architecture

The model is a deep neural network designed for high accuracy and stability. It includes:

Input Layer:

- Accepts a flattened vector of key point coordinates and engineered features.

Hidden Layers:

1. Dense Block 1:
    a. 1024 neurons, L2 regularization, Batch Normalization, Leaky ReLU activation, Dropout (0.5).
2. Dense Block 2:
    a. 512 neurons, L2 regularization, Batch Normalization, Leaky ReLU activation, Dropout (0.4).
3. Dense Block 3:
    a. 256 neurons, Residual Connection, Batch Normalization, ReLU activation, Dropout (0.3).
4. Dense Block 4:
    a. 128 neurons, Batch Normalization, ELU activation, Dropout (0.3).
5. Dense Block 5:
    a. 64 neurons, Batch Normalization, ReLU activation, Dropout (0.2).

Output Layer:

A SoftMax layer with neurons equal to the number of yoga poses (classes) in the dataset.

Model Optimizer:

Adam optimizer with an initial learning rate of 0.0001 and a cyclic learning rate scheduler.

Loss Function:

Sparse categorical cross-entropy for multi-class classification.

Results

Model Performance:

After training for 100 epochs with early stopping, the model achieved the following metrics:

Metric	Value
Accuracy	95.2%
Precision	94.8%
Recall	95.0%
F1-Score	94.9%

Confusion Matrix:

- Highlighted that most misclassifications occurred between visually similar poses such as Warrior I and Warrior II.

Feedback Accuracy:

- Correctly identified alignment issues in 92% of cases during testing.

Visualization

- Plotted loss and accuracy curves to confirm the model's convergence.

- Visualized pose corrections by overlaying feedback on images.

Next Steps

Data Enhancements:

- Expand the dataset to include more diverse yoga poses and practitioners of varying skill levels.
- Collect video sequences to analyse transitions between poses.

Model Improvements:

1. Transfer Learning:
   - Fine-tune pretrained models like EfficientNet or MobileNet for feature extraction.
2. Advanced Architectures:
   - Experiment with transformer-based models for better spatial understanding.

Real-Time Feedback:

- Integrate the model into a real-time system using TensorFlow.js or a mobile app framework.

User Personalization:

- Develop algorithms to adapt feedback based on user skill level and history.

Deployment:

- Deploy the system as a web or mobile application for broader accessibility.

Conclusion

The Yoga Pose Identification and Feedback System demonstrates a robust approach to leveraging deep learning for pose classification and alignment analysis. While the current model shows promising results, further improvements and real-world deployment will enhance its practical utility.