Indian Institute of Technology, Kanpur

EE655 Course Project

Supervised by : Dr. Koteswar Rao J.

Presented By:

Nishant Patel – 210673
Srijan Shekhar – 211057
Sahil Panjwani-210897
Tejas Jain - 211108

From - To :

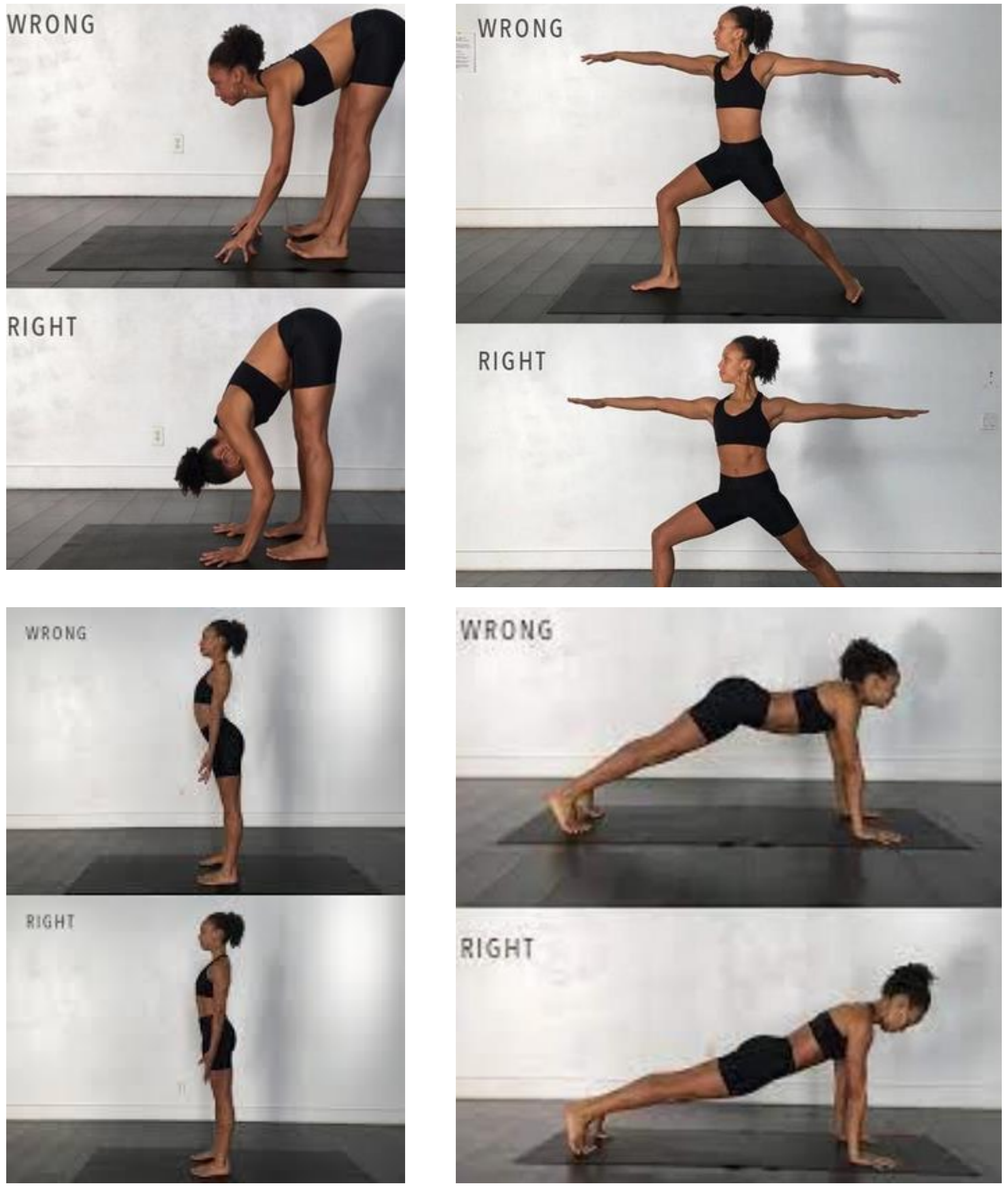
Jan’25 - Apr’25

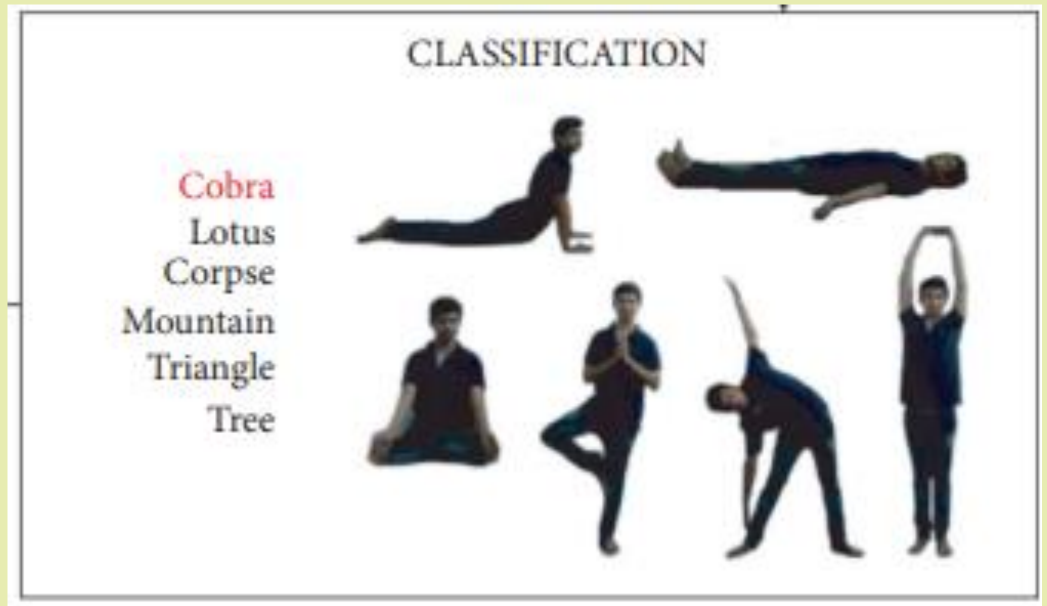
PROJECT TITLE

Pose Detection and Feedback Generation
for different Yoga Asanas

Motivation

- Incorrect yoga postures can cause harm and reduce effectiveness.
- Self-learning is on the rise, but it lacks real-time corrections.
- Need for automated pose detection to assist learners with proper form.



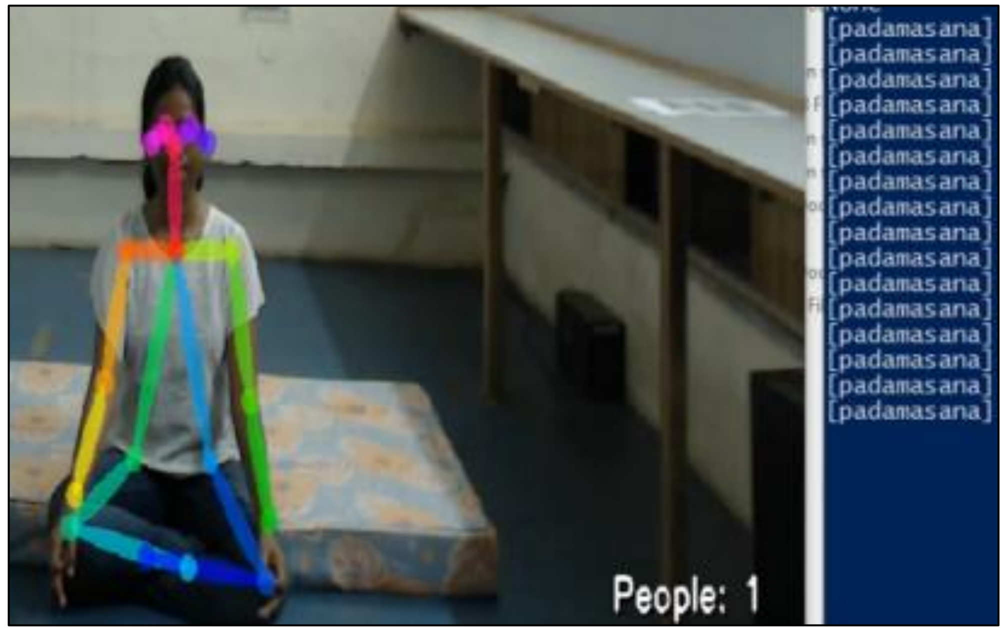


Problem Statement

Create a comprehensive classification framework for yoga poses, with the goal of accurately identifying the specific posture being demonstrated by the participant in a given video recording.

Proposed Solution

- Preprocessed yoga videos into frame-wise JSON files containing body joint coordinates.
- Computed joint angles for each frame to create a compact and meaningful representation.
- Trained a CNN-LSTM model using the joint angle sequences to classify yoga asanas.
- Captured both spatial (joint angles) and temporal (motion across frames) information.
- Focused solely on accurate pose classification from video input.



Classification or Detection of Yoga Asana

Dataset

The following datasets have been used to train our current model for classification and feedback generation

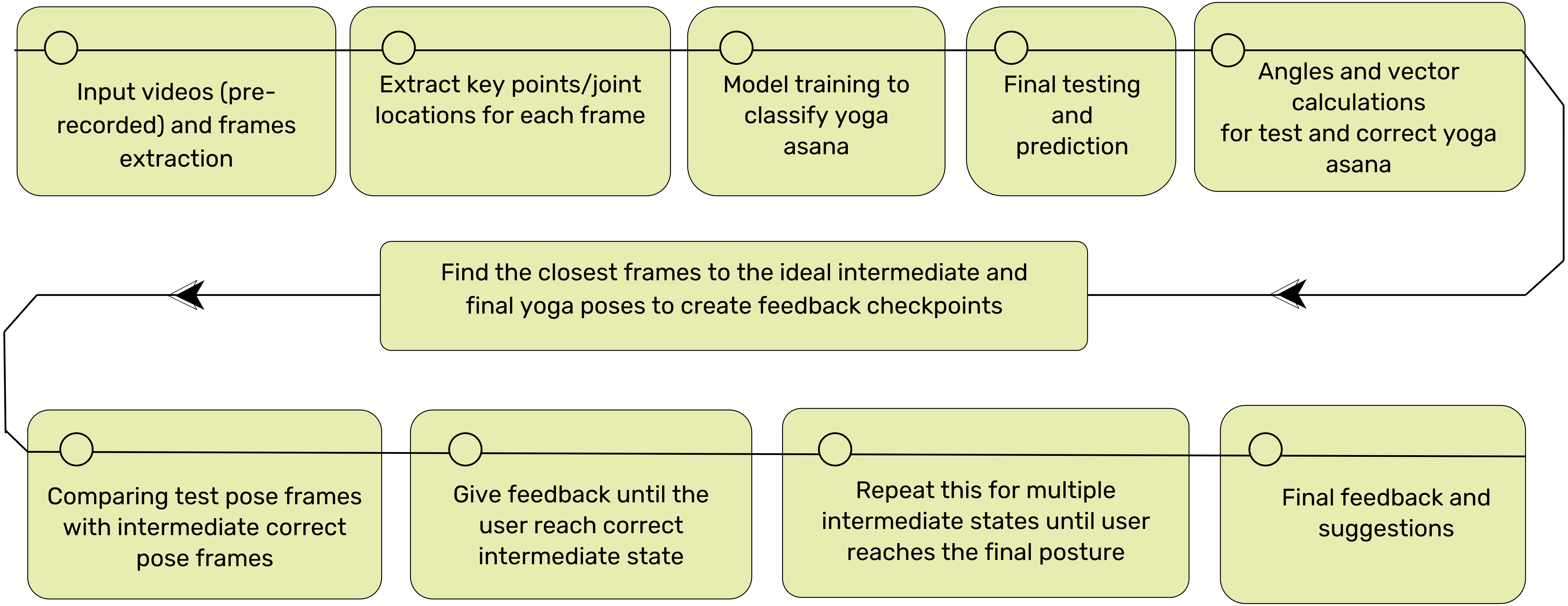
1. A publicly available, online, open-source collection dataset

The dataset comprises of 88 videos of 6 yoga asanas performed by different person. All the videos are collected for more than 45 s in an indoor environment. The total length of 88 videos for training is 1 h 6 min and 5 s at 30 frames per second, that is, a total of about 111,750 frames.

2. YouTube Tutorial Videos - Alo Moves - Online Yoga & Fitness Videos

This dataset includes 48 yoga asanas with full-length tutorials that provide step-by-step guidance We have created our own dataset for six asanas, collecting four correct key frames from the beginning to intermediate to the final position for each asana. This allows us to compare any test asana video and provide feedback.

Pipeline of the project



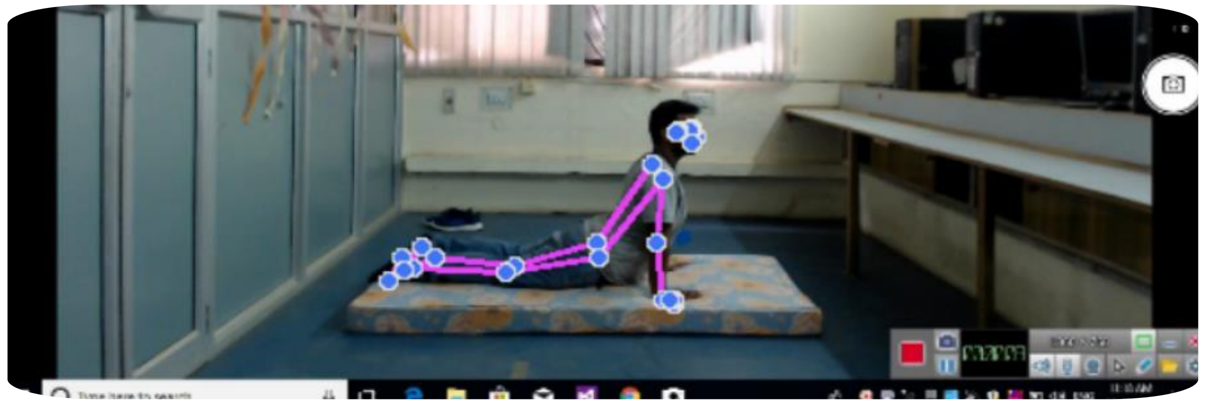
Detailed Description Of The Work Done

Classification of Yoga Asanas

Video Frames Extraction, Automated Pose Landmark Extraction, and JSON Data Formatting:

- We have utilized OpenCV and MediaPipe libraries to automatically extract pose landmark data i.e. 33 keypoints/joint locations from frames of 88 video files (6 Yoga Asanas).
- By iterating through frames, it detects key body landmarks, and we have also skipped 9 frames for efficient processing and optimization.
- This pose data(landmarks) is then aggregated into JSON format for further analysis.

Key-points extraction of single frame



Key-points extraction of a video

Classification of Yoga Asanas

Pre-Processing:

- Loaded pose data from JSON files representing yoga videos.
- Split the dataset into **training**, **validation**, and **test** sets.
- Applied a **sliding window** technique with window_size = 10.
- Created multiple test cases, each with 10 consecutive frames.
- Assigned a label to each test case based on the corresponding yoga asana.
- Prepared the data for input into the CNN-LSTM model.

```python
train_list = []
#list of asanas, each asana has all it's videos, each video has list of s
val_list = []
test_list = []
# print("ok")
for a in asanas:
    print("ok")
    currAsanaTrain = []
    currAsanaVal = []
    currAsanaTest = []
    path = data_path + "\\" +  a
    for i in range(1,17):
        # print("ok")
        currVideo = []
        start =  str(i) + "_"
        for filename in os.listdir(path):
            data = []
            # print(i)
            if filename.startswith(start):
                #get data from file
                # print("o")
                with open(path + "\\" + filename) as json_data:
                    # print("ok1")
                    d = json.load(json_data)
                    # print(d)
                    npdata = np.asarray(d)
                    # print(npdata.shape)
                    # print(npdata.shape)
                    Xdata = npdata[:,:,0]
                    Ydata = npdata[:,:,1]
                    # print(len(Ydata))
                    # print(Xdata)
                    stk = np.dstack((Xdata, Ydata)) #stack vertically
                    currVideo.append(stk)
                    # print(stk)
                # print("ok")
        #print currVideo
        if a == 'vrikshasan' and i == 15:
            # this one has difference and creates noise
            currAsanaTrain.append(currVideo)
        elif (i+1)%5 == 0 and len(currVideo) != 0:
            currAsanaTest.append(currVideo)
```

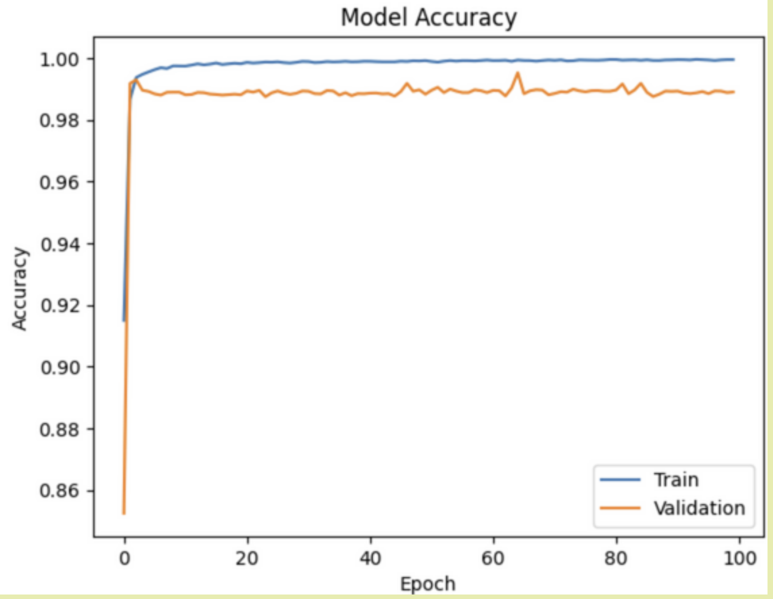
Model Training:

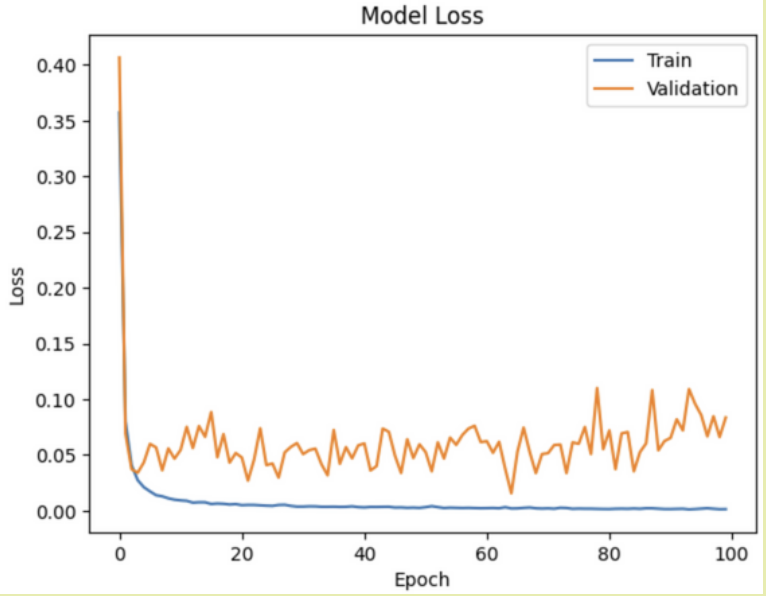
- Then we implemented our human pose recognition deep learning model using Keras with TensorFlow.

```
def get_model():
    model = Sequential([
        TimeDistributed(Conv1D(16,3, activation='elu', padding = "same"),input_shape=XchTrain.shape[1:]),
        TimeDistributed(BatchNormalization()),
        #TimeDistributed(MaxPooling1D()),
        TimeDistributed(Dropout(0.5)),
        #TimeDistributed(Conv1D(64,3, activation='relu',padding = "same")),
        BatchNormalization(),
        #TimeDistributed(Dropout(0.8)),
        TimeDistributed(Flatten()),
        #TimeDistributed(Dense(30,activation='softmax')),
        LSTM(20,unit_forget_bias = 0.5, return_sequences = True),
        TimeDistributed(Dense(6,activation='softmax'))
    ])
    adam = Adam(lr=0.0001)
    model.compile(loss='categorical_crossentropy',
                  optimizer= adam,
                  metrics=['accuracy'])
    return model
```

Model Architecture

- Visualized the training and validation accuracy and loss over 100 epochs using Matplotlib.

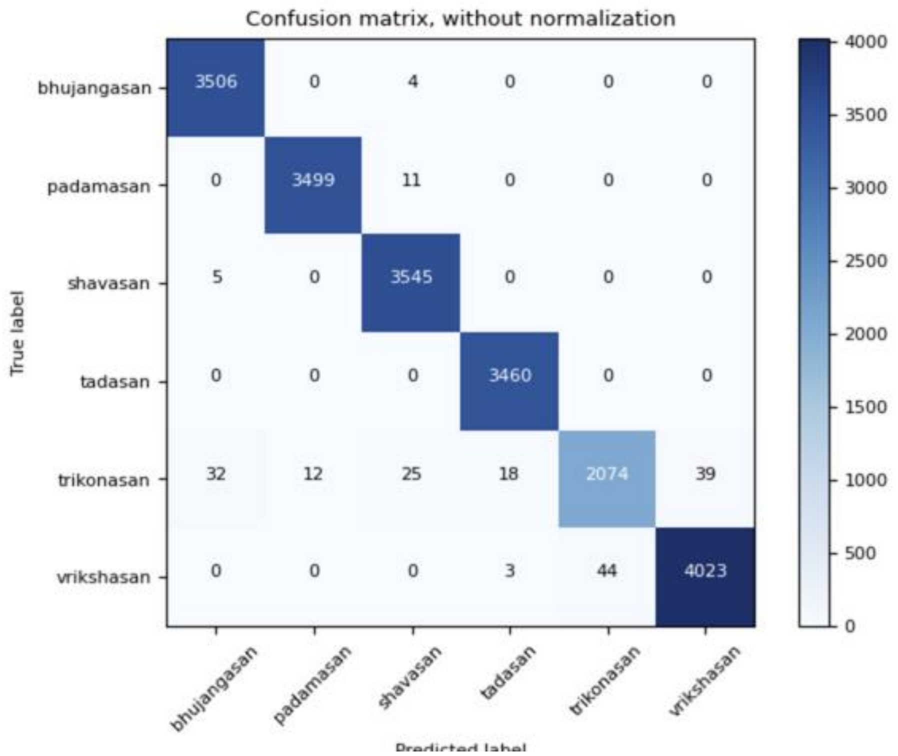


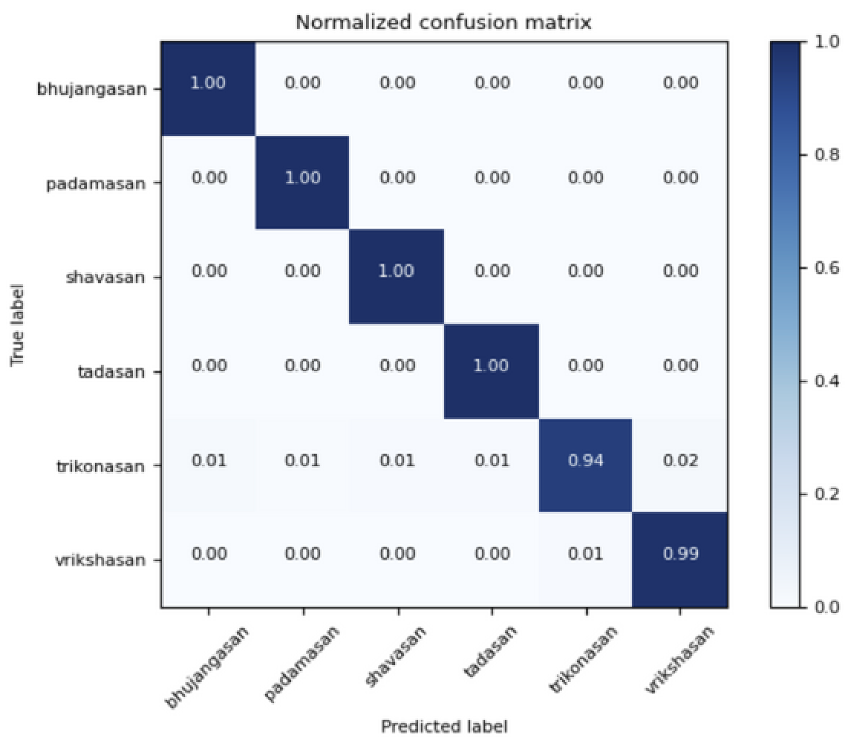


Classification of Yoga Asanas

Model Evaluation:

- The model demonstrated exceptional performance on the test data, achieving a framewise accuracy of 99.05%. Employing polling techniques further improved accuracy to an impressive 99.66%.





Feedback generation for different yoga asanas

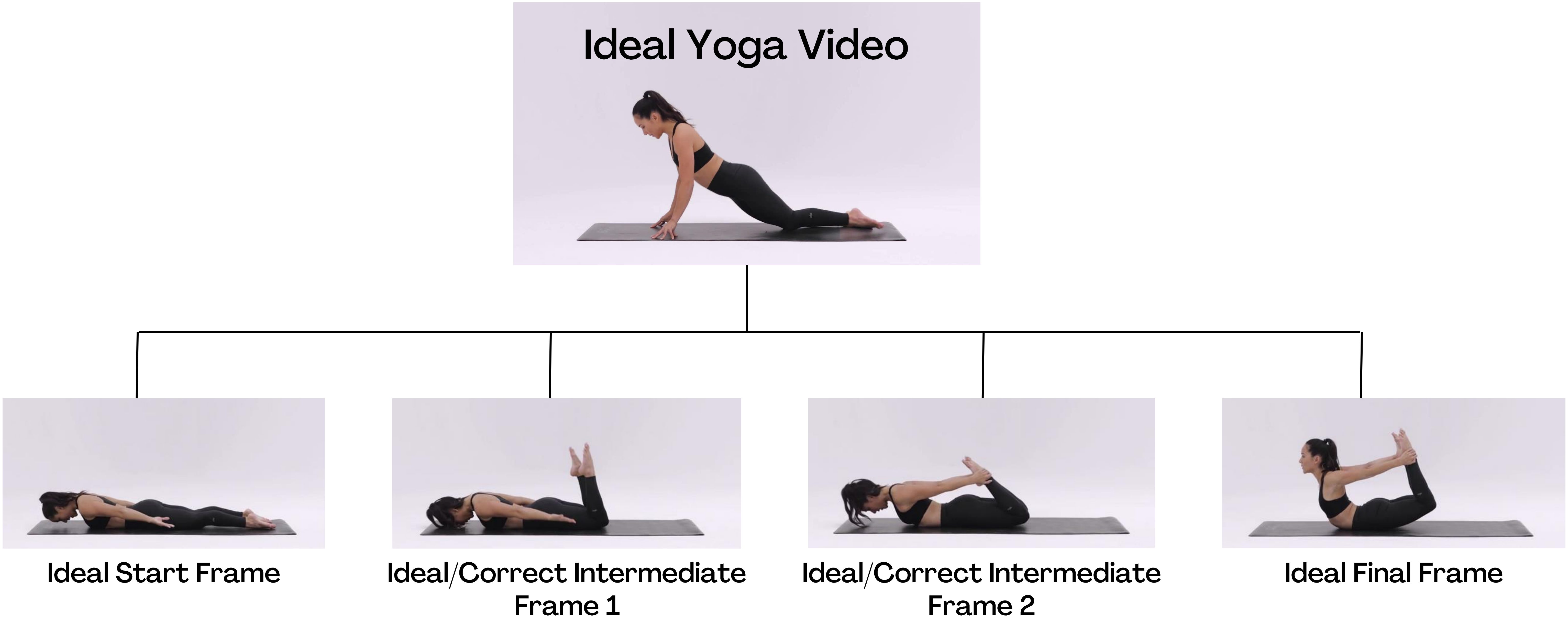
Identifying the best intermediate poses:

- We identified the frames with the closest angles in the test video compared to the correct intermediate poses (as mentioned in dataset 2) and collect these intermediate states.

Feedback Generation:

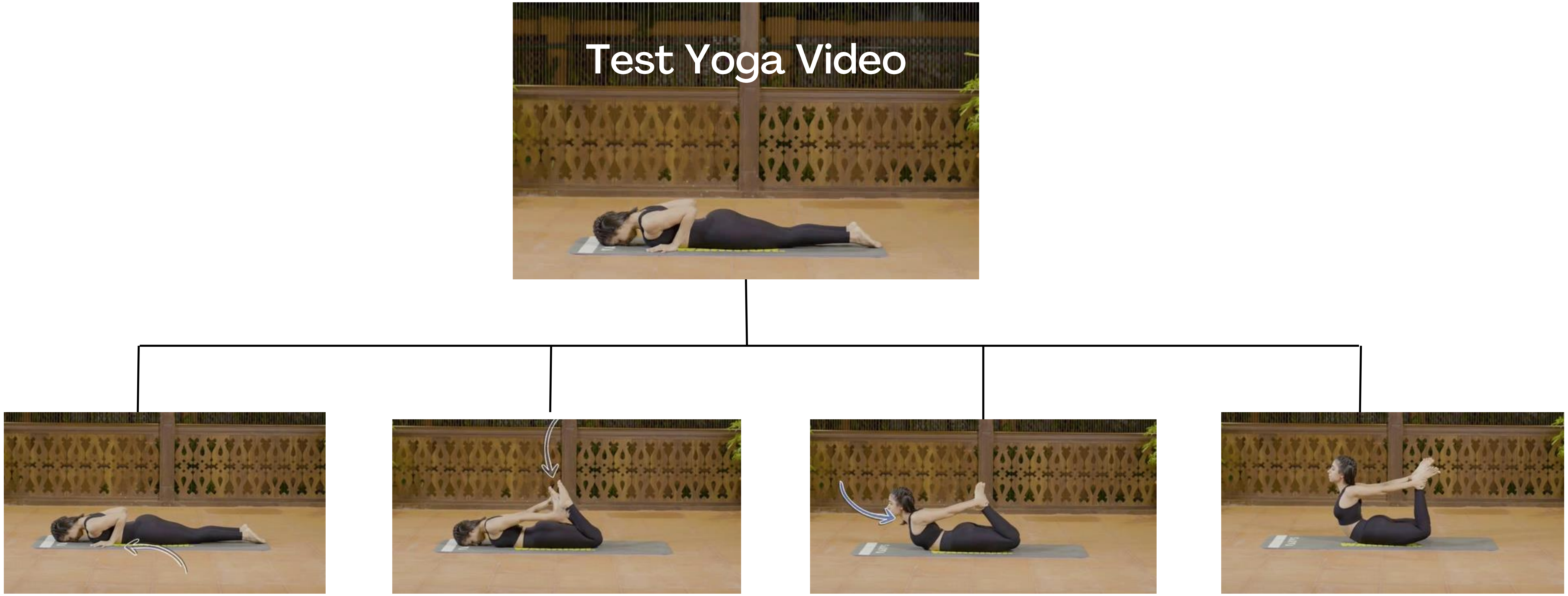
- Performed angular comparison for test video frames to the correct intermediate poses and provide feedback based on how closely the video frames match the intermediate poses.
- Provide feedback on joint angles to help the user improve their form in the test video.

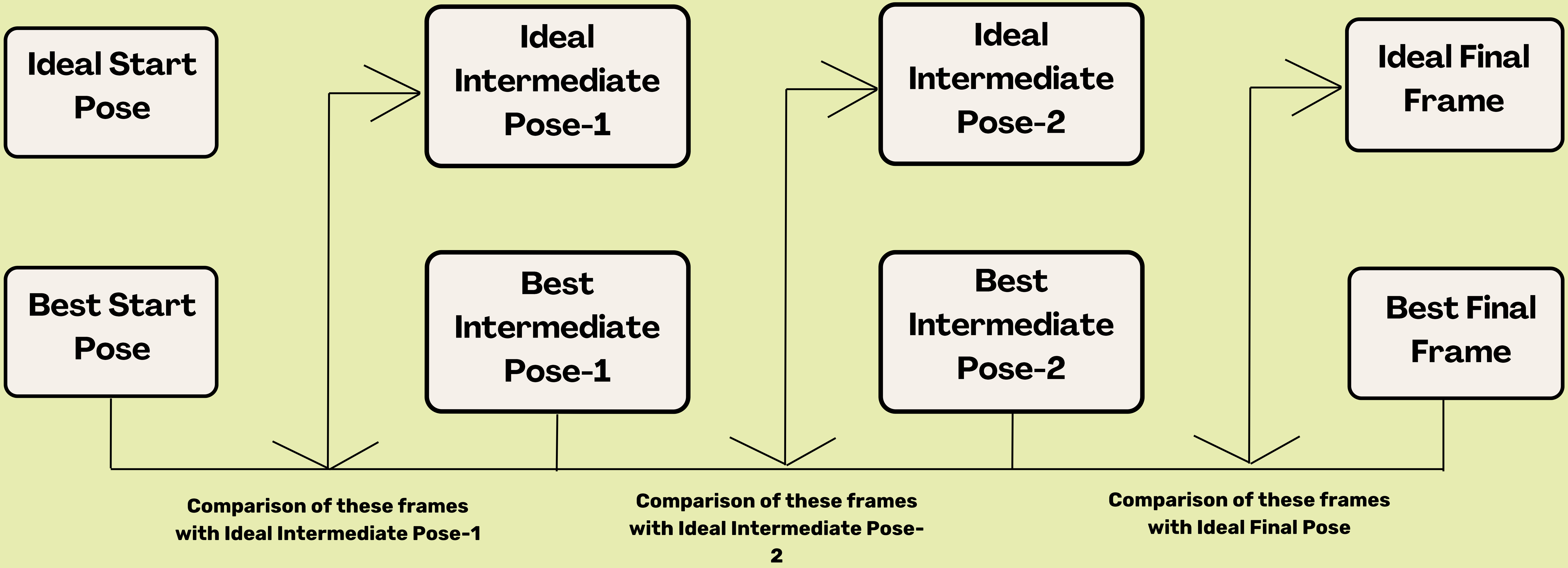
Demonstration of Ideal Dhanuarasana poses



These frames are chosen and stored manually by us for different yoga asanas

Demonstration of generated Intermediate Dhanuarasana poses in a test video





Feedback Generation

Our feedback series begins by comparing initial frames with the first ideal intermediate pose. It advances until the test frame reaches intermediate pose 1. Subsequently, frames are compared with ideal intermediate frame 2 until the next checkpoint, then transitioning again. This systematic process continues until we reach the final yoga pose.

Research Papers

We have primarily used the following reasearch paper to build our model

- Yadav SK, Singh A, Gupta A, Raheja J (2019) Real-time yoga recognition using deep learning.
  https://link.springer.com/article/10.1007/s00521-019-04232-7#Abs1

Other papers used for reference and understanding

- Anand Thoutam, V., Srivastava, A., Badal, T., Kumar Mishra, V., Sinha, G. R., Sakalle, A., ... & Raj, M. (2022). Yoga pose estimation and feedback generation using deep learning.
  https://www.hindawi.com/journals/cin/2022/4311350/

- A. Chaudhari, O. Dalvi, O. Ramade and D. Ambawade(2021), YogGuru: Real-Time Yoga Pose Correction System Using Deep Learning Methods.
  https://ieeexplore.ieee.org/document/9509937

Thank You!