IMPLEMENTATION OF YOGA POSE ESTIMATION AND FEEDBACK MECHANISM USING POSE DETECTION FOR SELF LEARNING

An Interim Project Report

Submitted by

DHANUSH VARMA CH [CB.EN.U4CSE18112]
VENKATARAMAN R [CB.EN.U4CSE18162]
HARSHAVARDHAN D [CB.EN.U4CSE18174]
GUHAN K [CB.EN.U4CSE18511]

Under the guidance of

Dr. Raghesh Krishnan K
(Assistant Professor, Department of Computer Science & Engineering)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

AMRITA VISHWA VIDYAPEETHAM

Amrita Nagar PO, Coimbatore - 641 112, Tamilnadu

Review 3 – October 2021

ABSTRACT

Yoga which was developed in ancient India was initially considered as an old age exercise. But because of its many spiritual, physical and mental benefits, Yoga became popular across all age categories. One of the major problems that Yoga and other exercises face is the correct posture in which they must be performed. Even a slight error in the posture may not only nullify the benefits of the exercise but also may result in injuries or lead to structural deformities. So, this puts forward the requirement of an instructor who guides on how the exercise must be done in order to get the maximum from the exercise. But not everyone has the accessibility to an instructor who can supervise and correct the posture whenever the person decides to exercise. Thus, this demands the requirement of a model that will take an asana as input from the user and give an output if the person is doing it correctly or a feedback in case a posture correction is required. This project focuses on the idea of solving the problem of improper posture and allows people to learn and practice exercises correctly by themselves.

Keywords: Yoga, MediaPipe, Pose Assessment, Body Angle, Self-learning, Evaluation

TABLE OF CONTENTS

ABSTRACT ii

LIST OF TABLES iv

LIST OF FIGURES v

ABBREVIATIONS vi

1 INTRODUCTION 1
1.1 Problem Definition 1

2 LITERATURE SURVEY 3
2.1 Classification Models 3
2.2 Comparison between 2D and 3D Models 4
2.3 Summary 4
2.4 Data Set 6
2.5 Software/Tools Requirements 6

3 PROPOSED SYSTEM 7
3.1 System Analysis 7
3.1.1 System requirement analysis 7
3.1.2 Module details of the system 8
3.2 System Design 13
3.2.1 Flow diagram of the system 13
3.2.2 Architecture diagram 14

4 Implementation and Testing 15
4.1 Dataset 15
4.2 Preprocessing Techniques 15
4.2.1 Brightness adjustment: 15
4.2.2 Sharpening Images: 15
4.2.3 Contrast Adjustments: 15
4.2.4 Body Segmentation 16
4.3 Pose Landmarks 16
4.4 Angle Computation 16

5 Results and Analysis 17
5.1 Finding the Best fit ML model 17

6 Conclusion 19

7 Future Enhancement 20

LIST OF TABLES

2.1 Benchmark on YOGI Dataset 3

5.1 Accuracy and F1 scores 18

LIST OF FIGURES

1.1 Classification Model based on distance between joints 2
1.2 Feedback Model based on angles between adjacent points 2

3.1 Gamma Correction Formula 8
3.2 Graphical Representation of Histogram Equalization 9
3.3 Flow diagram of proposed model 13
3.4 Architectural Diagram 14

5.1 Model Selection for the dataset 17
5.2 Accuracy score and F1 score of models 18

ABBREVIATIONS

CNN	Convolutional Neural Network
CSV	Comma separated values
KNN	K-Nearest Neighbors
LSTM	Long Short Term Memory
ML	Machine Learning
SVM	Support Vector Machine

Chapter 1

INTRODUCTION

Human posture assessment is a difficult issue in the control of Personal Computer vision. It manages confinement of human joints in a picture or video to shape a skeletal portrayal. Practice and wellness is one use of posture assessment that has attracted the attention of many experts in this subject. Yoga, a deeply entrenched practise that originated in India but is now widely acclaimed for its multiple profound physical and mental benefits.

The problem with yoga is that it is critical to do it correctly, as any incorrect position during a yoga session can be useless and potentially inconvenient. This necessitates the presence of an instructor to supervise and correct the individual's stance. Not every client has access to a yoga instructor, a computerised reasoning-based program might be used to recognise yoga poses and provide personalised feedback to help people improve their structure.

1.1 Problem Definition

Our project aims to design a Machine Learning (ML) based application based on the MediaPipe Pose library that monitors the pose performed by the practitioner and gives real time feedback for improvement of the yoga posture.

Generally , Yoga poses are classified by applying ML algorithms with the difference between joints as in Figure 1.1 . The diversity in the structure of Humans across the world and no feedback for improvement to the practitioner are a major drawback of using length calculated between joints . Using angle between adjacent joints as in Figure 1.2 would overcome these drawbacks and can give feedback to improve the pose of the practitioner.

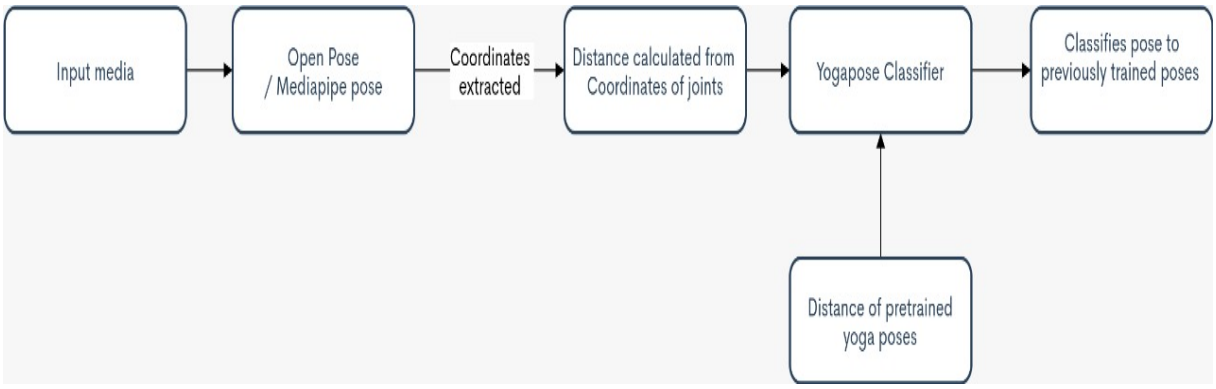

Figure 1.1: Classification Model based on distance between joints

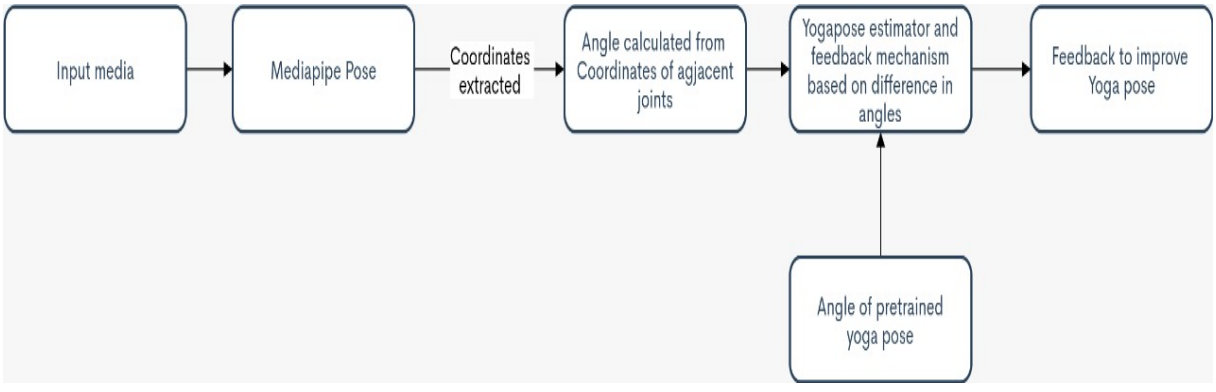

Figure 1.2: Feedback Model based on angles between adjacent points

Chapter 2

LITERATURE SURVEY

ML has profound advancements in real time image processing with pose prediction , autonomous systems . Machine Learning Applications which could help humans in various ways are being developed . Various pose estimation based models that predict and classify human poses are discussed below.

2.1 Classification Models

Abhishek Sharma Agrawal et al. (2020) discusses classification of yoga poses using Machine Learning Algorithms like Logistic Regression, Random Forest, Support Vector Machine (SVM), Decision Tree, Naive Bayes and K-Nearest Neighbors (KNN) based on the YOGI dataset using similarities between the joint coordinates extracted by tf-pose-estimation algorithm.

Table 2.1: Benchmark on YOGI Dataset

Classifier	**Accuracy**
Logistic Regression	0.8215
Random Forest	0.9926
SVM	0.8791
Decision Tree	0.9752
Naive Bayes	0.7475
KNN	0.9725

Muhammad Usama Islam Islam et al. (2017) proposed a model that estimates the accuracy of the pose through data acquisition with the help of a Kinect device (a Kinect device can acquire different types of data in data acquisition steps such as color, depth and skeleton information at the frame rate of 30fps with a resolution of 480*320) then preprocess the data that they acquire and find the acquisitions of the three different weight persons and compare the accuracies of the three persons by three different poses.

Deepak Kumar Kumar and Sinha (2020) classifies the poses using SVM, Convolutional Neural Network (CNN) and CNN+Long Short Term Memory (LSTM) algorithms using the features extracted from the yoga practitioner by OpenPose library based on 12 different joints. CNN+LSTM gave the highest accuracy, compared to the SVM and CNN models. The major drawback with the SVM model was the misclassification between vrikshasana and tadasana since both of them require a standing position and furthermore the underlying posture arrangement is comparative.

A dataset was made from scratch for Kathakali hand gestures by Bhavanam and Iyer (2020), appropriate image processing techniques were applied and compared SVM with CNN for mudra classification.A Dataset with 684 images divided into 70/30 split for train and test data Features extracted from the image dataset given to SVM as input and an accuracy score of 40% was obtained. Low accuracy obtained because of collision in feature extraction results. CNN achieved an accuracy of 74%.

2.2 Comparison between 2D and 3D Models

Sankara Narayanan Narayanan et al. (2021) Y Guo Guo et al. (2021) Yanze Wang Wang and Ye (2021) Bradley J. Wheeler Wheeler and Karimi (2021) extracts 17 different body points and compares the accuracy of classification of 2D and 3D versions of OpenPose models using the features generated for 2D and 3D Comma separated values (CSV) files and finally states that the 3D model of OpenPose Predicts more accurate form results compared to 2D model.

2.3 Summary

The research gap is these models don't provide valuable feedback on how yoga practitioners practice a pose , they are not trained over a diverse dataset and some use very expensive devices which require high maintainability since they are easily damaged. Classification models don't give valuable information for the learner to improve /practise yoga. Thus a system that assesses yoga pose of a learner by detecting the human body first using the webcam, extracting the coordinated points of various joints in the body, calculating the difference of body angles between the trained pose images and

that of the user accurately and finally provide real-time feedback on improvisation of the incorrect parts between learner and the trained images is necessary.

2.4 Data Set

Source of data set :

- https://www.kaggle.com/niharika41298/yoga-poses-dataset
- https://sites.google.com/view/yoga-82/home
- The combined dataset contains- 900 + diverse images (Children , Women and Men) for five different Yoga Asanas.
- Images in different Lightings/orientation for each yoga pose.

2.5 Software/Tools Requirements

- Flask (Python library) - Front End
- OpenCV - Used for feature extraction ,object detection in this case the person and altering the orientation and brightness of images.
- Media Pipe (coordinates extraction) - Media Pipe offers open source cross-platform, customizable ML solutions for live and streaming media.
- Visual Studio Code - code editor redefined and optimized for building and debugging modern web and cloud applications.
- TensorFlow - provides a collection of workflows to develop and train models using Python or JavaScript, and to easily deploy in the cloud, on-prem, in the browser or on-device

Chapter 3

PROPOSED SYSTEM

3.1 System Analysis

3.1.1 System requirement analysis

Software Requirements:

- The person must stand away from the camera such that the webcam is able to capture the entire body.
- The application should be able to provide accurate feedback for the pose performed by the practitioner.
- Only one person should stand in front of the webcam to get the best results.
- The application should also be able to extract body features under special conditions:
  - Different body orientations
  - Low Light Conditions

Hardware Requirements:

- OS: Windows 10 and later
- Processor: Intel Core i5/ Inter Core i7
- Memory: 2GB RAM minimum, 4GB RAM recommended
- Screen Resolution: 1024 * 768 or higher recommended
- Internet: Stable Broadband Internet Connection.

3.1.2 Module details of the system

Prepare Image Dataset from various sources

- Various websites were referred and two different datasets were gathered suitable-for training.
- Dataset Information:
  - Yoga 82 dataset prepared by Osaka University, Japan and IIT Gandhinagar,India
  - The dataset contains a three-level hierarchy including body positions, variations in body positions, and the actual pose name.
  - Yoga Poses Dataset prepared by Niharika Kaggle.
  - The dataset has all the 5 most famous asanas namely downward dog, plankpose, tree pose, goddess pose and warrior-2 pose.
- Both datasets are scanned and the images where there is too much text information, person is too small, person is not visible/ part of person is visible, corruptimages, multiple persons in an image, animated images are removed.
- Finally, both the datasets are combined into one large training dataset with subdirectories for each asana.

Preprocessing the image data set

- Image resizing is done because the images come in different resolutions, and high definition images will be more computationally intensive to process. We need to reduce the storage space by discarding unnecessary pixel information.
- **Brightness Adjustments using Gamma Correction**
  Gamma correction is a non-linear adjustment to individual pixel values. In image normalization, linear operations are carried out on individual pixels, gamma correction carries out a non-linear operation on the source image pixels, and can cause saturation of the image being altered.

$$O = \left(\frac{I}{255}\right)^{\gamma} \times 255$$

Figure 3.1: Gamma Correction Formula

- **Contrast Adjustments using Histogram equalization**
  Histogram Equalization is an image processing technique that adjusts the contrast of an image by using its histogram. To enhance the image's contrast, it spreads out the most frequent pixel intensity values or stretches out the intensity range of the image. By accomplishing this, histogram equalization allows the image's areas with lower contrast to gain a higher contrast.

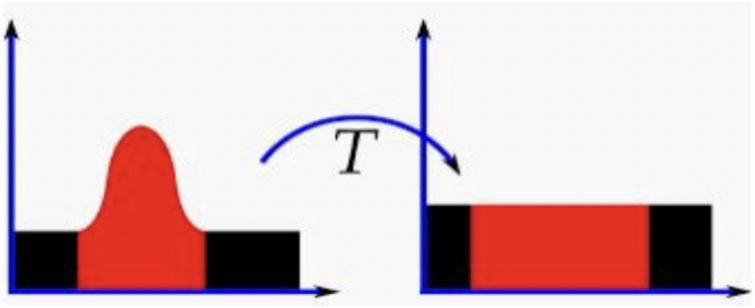

Figure 3.2: Graphical Representation of Histogram Equalization

- **Image Sharpening using Laplacian Filtering**
  Laplacian filter is an edge detector used to compute the second derivatives of an image, measuring the rate at which the first derivatives change. This determines if a change in adjacent pixel values is from an edge or continuous progression. Laplacian filter kernels usually contain negative values in a cross pattern, centered within the array. The corners are either zero or positive values. The center value can be either negative or positive.

Extracting the body features using MediaPipe Pose

- MediaPipe Pose is a ML solution for high-fidelity body pose tracking, inferring 33 3D landmarks and background segmentation masks on the whole body from RGB video frames utilizing BlazePose (a pose detection model developed by Google that can compute (x,y,z) coordinates of 33 skeleton key points).

- Pose detection works best when the subject's entire body is visible in the frame, but it also detects a partial body pose. In that case the landmarks that are not recognized are assigned coordinates outside of the image.

  SEGMENTATION MASK

- The output segmentation mask, predicted only when enable segmentation is set to true. The mask has the same width and height as the input image, and contains values in [0.0, 1.0] where 1.0 and 0.0 indicate high certainty of a "human" and "background" pixel respectively.

OUTPUT: POSE LANDMARKS

- A list of pose landmarks. Each landmark consists of the following:
  - x and y: Landmark coordinates normalized to [0.0, 1.0] by the image width and height respectively.
  - z: Represents the landmark depth with the depth at the midpoint of hips being the origin, and the smaller the value the closer the landmark is to the camera. The magnitude of z uses roughly the same scale as x.
  - visibility: A value in [0.0, 1.0] indicating the likelihood of the landmark being visible (present and not occluded) in the image.
  - Results of all asanas will be stored in a single csv.

Create an efficient feedback based machine learning model

- Data in csv are labelled according to Asana
- Train and test machine learning algorithms (Random Forest, SVM, Decision Tree,and KNN etc) using the data generated to find which model best fits .
- The yoga pose performed by user is classified into any one of the pretrained asana using machine learning algorithm
- Accuracy score to indicate closeness of the poses is displayed
- Feedback based on difference in angles of specific pose is displayed to the practitioner for self correction.

Develop the UI for Web Application

Welcome Page:

- About the app
  - Purpose of the app
  - Features and screenshots
  - Developers Information
- Contact us

Application Page:

- User clicks on the start button
- Users must allow webcam access to start performing. If not, display a warning instructing the person to turn on the webcam.
- Display a short video, on steps to be performed for the asana.

- User is then allowed to perform the asana for 15 seconds.
- At the end of 15 seconds, a screenshot of the user is submitted for processing.
- Finally, the user will get the feedback on how they have performed the pose. If the results are good, the image will be sent as training data for the same asana.

Deploy the application in the Cloud Service:

- The platform used to deploy the application in the cloud service for functioning like AWS.
- **AWS Lambda :**
  - Serverless computing service as part of Amazon Web Services, helps you run your code without managing the underlying infrastructure.
  - Very cheap because you only pay when you invoke the lambda function (that is, when you make prediction requests). Saves a lot of money compared to the cost of running containers or VMs.
  - Monitors real-time metrics including error rates, total requests, function-level concurrency usage, latency, and throttled requests through Amazon CloudWatch.
- **GCP**
  - **Google AI Platform:**
    Google AI Platform provides comprehensive machine learning services. Data Scientists and Machine Learning Engineers can use this platform to work on machine learning projects from ideation to deployment more effectively.
  - **Google App Engine:**
    - Google App Engine is a Platform as a service (PaaS) provided by Google that supports the development and hosting of different scalable web applications.
    - Google App Engine Provides an auto-scaling feature that automatically allocates resources so your web application can handle more requests.

- **AZURE**
  - **Microsoft Azure Functions**
    - Azure Functions is a serverless cloud service provided by Microsoft Azure as a Functions-as-a-service (FaaS).
    - If you have a big machine learning model, then Azure functions are the right choice for you. It supports the deployment of large ML packages such as deep learning frameworks (Tensorflow and Pytorch).

3.2 System Design

3.2.1 Flow diagram of the system

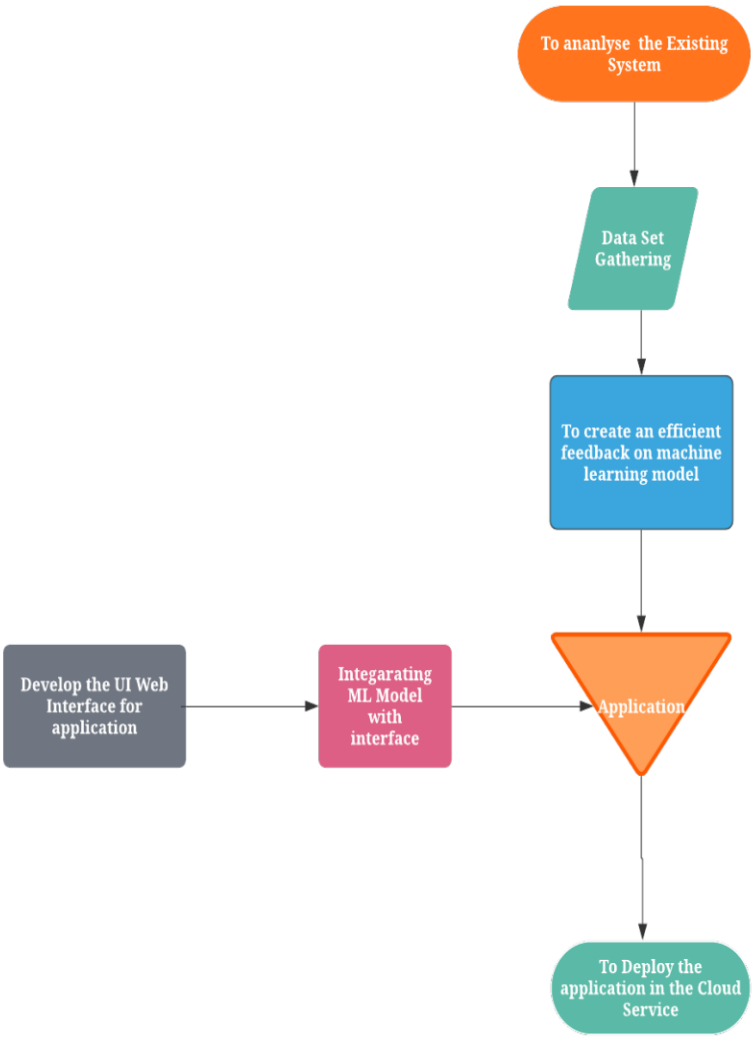

Figure 3.3: Flow diagram of proposed model

3.2.2 Architecture diagram

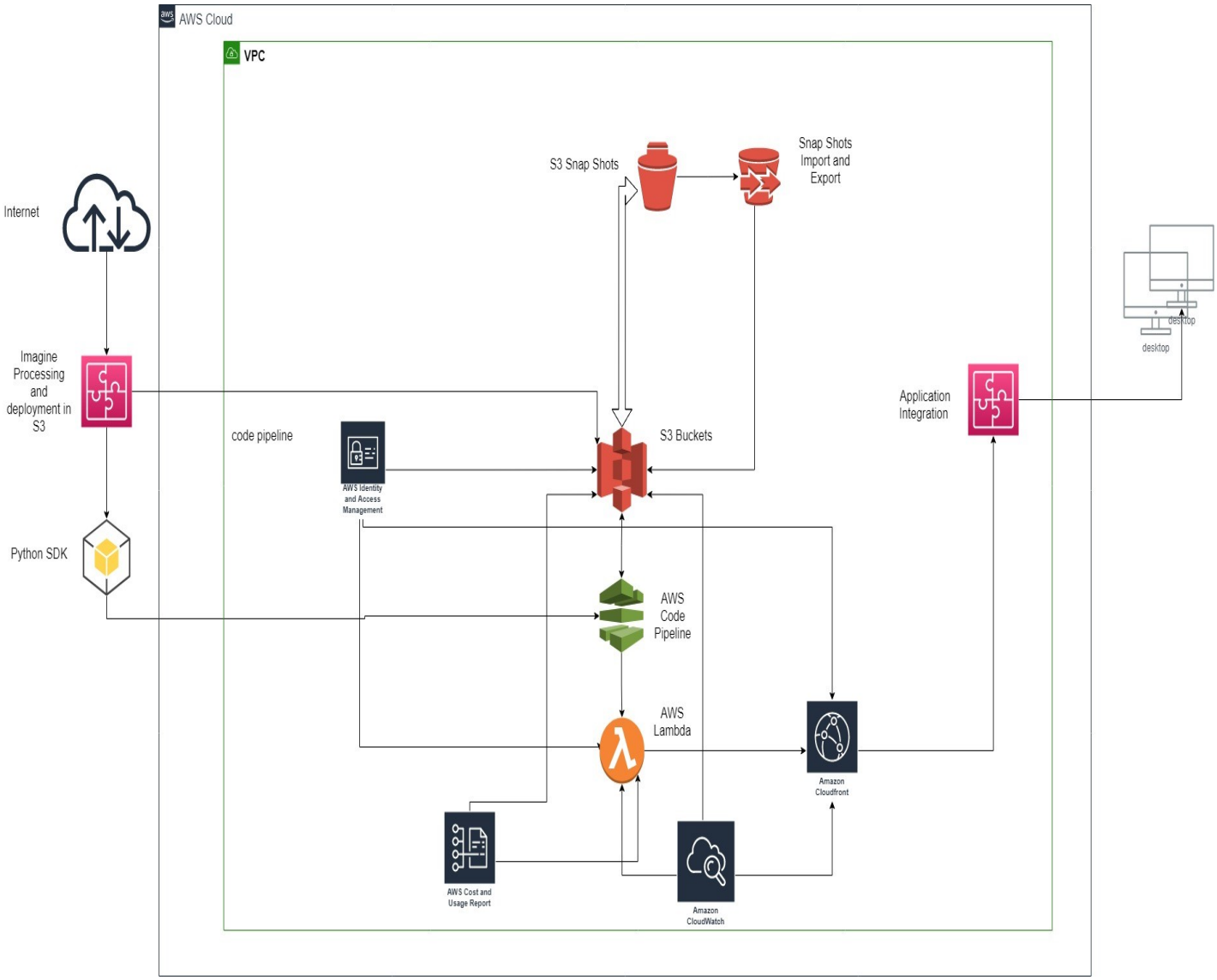

Figure 3.4: Architectural Diagram

Chapter 4

IMPLEMENTATION AND TESTING

4.1 Dataset

The dataset has all the 5 most famous Asanas namely downward dog, plank pose, tree pose, goddess pose and warrior-2 pose. This dataset contains over 1500 images.

4.2 Preprocessing Techniques

4.2.1 Brightness adjustment:

- Images are first resized to reduce computation.
- Gamma correction is a non-linear adjustment to individual pixel values. In image normalization, linear operations are carried out on individual pixels, gamma correction carries out a non-linear operation on the source image pixels, and can cause saturation of the image being altered.

4.2.2 Sharpening Images:

- Edge detector is used to compute the second derivatives of an image.
- This determines if a change in adjacent pixel values is from an edge or continuous progression. Laplacian filter kernels usually contain negative values in a cross pattern, centered within the array. The corners are either zero or positive values. The center value can be either negative or positive.

4.2.3 Contrast Adjustments:

- Adjusts image contrast by its histogram.
- To enhance contrast, spreads out intensity range of image.
- This allows the image's areas with lower contrast to gain a higher contrast.

4.2.4 Body Segmentation

- Media Pipe Segmentation function is used to blur the background of the image
- The mask has the same width and height as the input image, and contains values in [0.0, 1.0] where 1.0 and 0.0 indicate “human” and “background” pixel respectively.

4.3 Pose Landmarks

- Media pipe blaze pose is used to extract 3D coordinates of 33 joints from the image
- x and y: Landmark coordinates normalized to [0.0, 1.0] by the image width and height respectively.
- z: Represents the landmark depth with the depth at the midpoint of hips being the origin, and the smaller the value the closer the landmark is to the camera.

4.4 Angle Computation

- The points extracted from the body are: [left wrist angle ,right wrist angle ,left elbow angle, right elbow angle , left shoulder angle, right shoulder angle, left knee angle, right knee angle, left ankle angle, right ankle angle, left hip angle, right hip angle]
- Key angles at (knee , elbow , shoulder , ankle) are calculated from the points extracted and labelled with respective Asana name.
- Angle at a joint is given by:
  $angle = degrees(atan2(y3 - y2, x3 - x2) - math.atan2(y1 - y2, x1 - x2))$

Chapter 5

RESULTS AND ANALYSIS

5.1 Finding the Best fit ML model

- Train and test machine learning algorithms (Random Forest, SVC, Decision Tree, KNN, Adaboost, RFC) using the data generated to find which model best fits.

```
[17] from xgboost import XGBClassifier
     from sklearn.model_selection import GridSearchCV
     from sklearn.ensemble import RandomForestClassifier
     from lightgbm import LGBMClassifier
     from sklearn.model_selection import cross_val_score
     from sklearn.metrics import f1_score
     from sklearn.ensemble import AdaBoostClassifier
     from sklearn.datasets import make_classification

[18] xgb=XGBClassifier(random_state=90,n_jobs=-1,eval_metric='mlogloss')

[19] rfc=RandomForestClassifier(n_jobs=-1,random_state=42)

[20] adb = AdaBoostClassifier(n_estimators = 100, learning_rate = 0.6)

[21] svmcm = SVC(C=0.1, kernel='poly')

[22] svmrbf = SVC(C=0.1, kernel='rbf')

[23] knnm = KNeighborsClassifier(n_neighbors=4)

[24] dtc = DecisionTreeClassifier()
```

Figure 5.1: Model Selection for the dataset

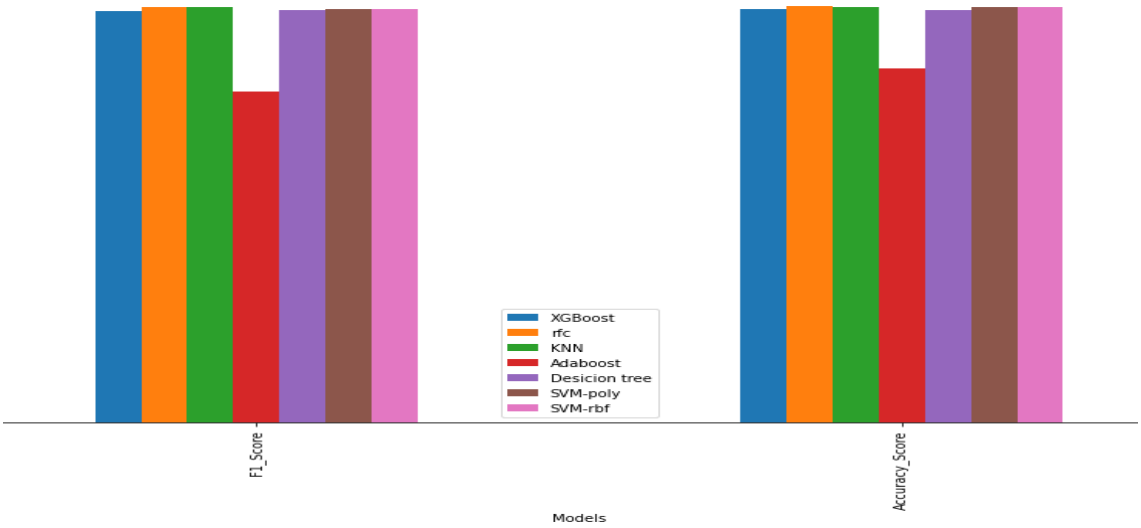

Figure 5.2: Accuracy score and F1 score of models

Table 5.1: Accuracy and F1 scores

Model	Accuracy Score	F1-Score
K Nearest Neighbours	0.9943820224719101	0.9877977644256399
SVM - Poly Kernel	0.9877977644256399	0.9838243875647213
XGBoost	0.9831182901605438	0.9782318267339015
Decision Trees	0.9831460674157303	0.9785827203906257
Random Forest	0.9887640449438202	0.9877453004206302
Ada Boost	0.8423618634886239	0.7874519073526639

- From the table, we can say that K Nearest Neighbours is the BEST FIT ML MODEL for the dataset.

Chapter 6

CONCLUSION

This work proposes a system for Yoga Pose Training and Feedback System to help the self-learning of Yoga for five different poses. The system assesses a Yoga pose of a learner by detecting the human body first using the webcam, extracting the coordinated points of various joints in the body, calculating the difference of body angles between the trained pose images and that of the user accurately and finally provide real-time feedback on improvisation of the incorrect parts between learner and the trained images.

Chapter 7

FUTURE ENHANCEMENT

In the future, we will include much larger data set and interpret the Yoga Asana from a video stream instead of images and also extend the classification beyond 5 Asanas. So the model can classify more and more Asanas and gradually it includes most of the yoga Asanas so that the user can perform all the asanas here itself. In addition, hyper parameter tuning will be applied to the models to achieve even better accuracies.

REFERENCES

1. Agrawal, Y., Shah, Y., and Sharma, A. (2020). "Implementation of machine learning technique for identification of yoga poses." *2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT)*, IEEE. 40–43. https://ieeexplore.ieee.org/document/9115758.

2. Bhavanam, L. T. and Iyer, G. N. (2020). "On the classification of kathakali hand gestures using support vector machines and convolutional neural networks." *2020 International Conference on Artificial Intelligence and Signal Processing (AISP)*, IEEE. 1–6.

3. Guo, Y., Ma, L., Li, Z., Wang, X., and Wang, F. (2021). "Monocular 3d multi-person pose estimation via predicting factorised correction factors." *Computer Vision and Image Understanding*, 103278. https://www.sciencedirect.com/science/article/abs/pii/S1077314221001223.

4. Islam, M. U., Mahmud, H., Ashraf, F. B., Hossain, I., and Hasan, M. K. (2017). "Yoga posture recognition by detecting human joint points in real time using microsoft kinect." *2017 IEEE Region 10 humanitarian technology conference (R10-HTC)*, IEEE. 668–673. https://ieeexplore.ieee.org/document/8289047.

5. Kumar, D. and Sinha, A. (2020). *Yoga Pose Detection and Classification Using Deep Learning*. LAP LAMBERT Academic Publishing. https://ijsrcseit.com/CSEIT206623.

6. Narayanan, S. S., Misra, D. K., Arora, K., and Rai, H. (2021). "Yoga pose detection using deep learning techniques." *Available at SSRN 3842656*. https://papers.ssrn.com/sol3/papers.cfm?abstract$_i d = 3842656$.

7. Wang, Y. and Ye, J. (2021). "Tmf: Temporal motion and fusion for action recognition." *Computer Vision and Image Understanding*, 103304. https://www.sciencedirect.com/science/article/abs/pii/S107731422100148X.

8. Wheeler, B. J. and Karimi, H. A. (2021). "A semantically driven self-supervised algorithm for detecting anomalies in image sets." *Computer Vision and Image Understanding*, 103279. https://www.sciencedirect.com/science/article/abs/pii/S1077314221001235.