National College of Ireland

Yoga Pose Multiclass Classification Using Machine Learning Models

MSc Research Project
Data Analytics

Aishwarya Ubale
Student ID: x22112081

School of Computing
National College of Ireland

Supervisor: Christian Horn

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Aishwarya Ubale
Student ID:	x22112081
Programme:	Data Analytics
Year:	2023-24
Module:	MSc Research Project
Supervisor:	Christian Horn
Submission Due Date:	31/01/2024
Project Title:	Yoga Pose Multiclass Classification Using Machine Learning Models
Word Count:	8723
Page Count:	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Aishwarya Ubale
Date:	31st January 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Yoga Pose MultiClass Classification Using Machine Learning Models

Aishwarya Ubale
x22112081

Abstract

An ancient practice - Yoga is substantially popular for its holistic benefits that involves physical postures and mind control practices. In today's time, leveraging the advancements happening particularly in Machine learning and computer vision field has enhanced various aspects of Yoga. This research paper focuses Machine learning models to classify yoga poses using image recognition techniques. The dataset utilized for the same consists images of popularly known yoga poses.Performing augmentation on this dataset and through application of machine learning models - after performing hyper parameter optimisation, is utilized to perform the classification task. The evaluation of the experiments and performance of the models has been assessed using accuracy, Precision, Recall, F1-Score and discusses the implications of these findings in the context of yoga pose classification. The comparative study highlights the potential improvement in the performance of Support Vector Classifier than other models with accuracy of 87% where as Decision Tree Classifier being the least improved model performing average with 71% accuracy. When contrasted with other existing approaches that do not use augmentation as one of the prepossessing steps and also uses heavy in size deep learning models for yoga pose classification task,the experimental outcome demonstrates the superior accuracy and efficiency of the proposed methodology that utilizes combination of preprocessed - augmented and fine-tuned hyper parameters for machine learning models.

Keywords - Random Forest Classifier, Support Vector Machine, Decision Tree Classifer, K-Nearest Neighbour Classifier, Gradient Boost Classifier, MediaPipe, Computer Vision.

1 Introduction

There has been an increased demand for Yoga practice due to it flexible feature - no need for any equipment or specific environment to perform the exercise using Yoga practice Zhang et al.; 2022. Especially after Covid-19 pandemic, people have been inclined to be fit by implementing Yoga like easy to do practices. This has surged and increase in the development of fitness tracking in the form of mobile apps or applications serving as fitness guide to help improve the posture Kanase et al.; 2021. A significant development in the field of Machine Learning and with the help of Computer Vision has been observed Aravind et al.; 2019. Number of applications for posture correction, pose detection, real-time pose detection and classification has been in process with advancements happening in fast pace. There has been remarkable developments in the field of Deep Learning

techniques for posture detection task, but as compared to the machine learning models, which are comparatively in expensive in the time and size factors, it is important to take in account that how can the task be completed by using optimal resources Gadhiya and Kalani; 2021. Such machine learning models that are not so heavy are more than suitable for Pose classification by performing image recognition is possible. Yoga pose classification comes under the human pose estimation that falls under computer vision techniques which are responsible to track the body points from the supplied input - either from images or videos or even from real-time camera capture format Taware et al.; 2022. To accomplish the task of pose recognition from the images, in the existing research, various libraries have been utilized, such as MediaPipe Pose, OpenPose, PoseNet, MoveNet and many more available in the market. For this research, MediaPipe Pose library will be used to establish pose recognition from the input images.Mediapipe is popular human pose estimation model under Mediapipe library - open source platform. Pose model is capable of locating landmarks from the images even if the image contains incomplete pose or partially visible pose. This unique behaviour is important to get correct outcome when classification will be implemented. In this paper, the Yoga pose detection and classification is based on MediaPipe Pose model and inexpensive machine learning models namely, Random Forest Classifier, Decision Tree Classifier, Support Vector Machine Classifier, K-Nearest Neighbour Classifier and Gradient Boost Classifier.

Research Question:

To what extent can the integration of MediaPipe Pose model coupled with image augmentation and hyper-parameter optimized machine learning models effectively classify the images containing practitioners performing Yoga poses?

The manuscript is organized as follows. Section 2 presents the literature review of yoga pose recognition and classification using machine learning models. Section 3 discusses the research method and specification or methodology of the proposed work, data collection description including image preprocessing - augmentation technique, landmark detection, classification using machine learning models. Section 4 describes design specifications of the proposed methodology.Section 5 presents the implementation of the techniques - experiments performed in the research with technical details. Section 6 presents the evaluation of the experiments conducted in the classification process and discusses the outcome of the study. Section 7 consists of conclusion and future work giving insights gained from the research and future enhancements that are technically feasible.

2 Related Work

Several researchers worked in the field of Pose detection and classification using Machine learning models and the research is still on going. It is important to understand the existing research work due to advancement in the machine learning field. Following sections describe detailed overview of the literature overview completed for this research paper.

2.1 Landmark / Key point detection task

In Gadhiya and Kalani; 2021, the research paper focuses on outlining a real time machine learning framework which performs detection and classification of Yoga poses both combined later for a real-time feedback model. This proposed methodology uses 'BlazePose'

model for the estimation of yoga poses. The model extracts 33 landmarks from the images that will be used for the pose estimation using various machine learning and deep learning models. This is one of the important aspects to look for in this paper. Although it is not clearly mentioned in the paper whether landmark detection is possible when the human in the image is partially or incompletely visible while performing any of the poses like 'Plank' or side facing 'warrior pose'.

In Pauzi et al.; 2021, The purpose of this paper serves to detect a risk-prone movement of the targeted subject and by capturing existing motion of the limbs from the video taken by the camera. This is achieved by implementing one framework that consists of video processing that focusses on joint detection using Mediapipe library. This processing is completed and has acceptable results for real-time scenario , which illustrates that mediapipe is capable of capturing body joints/key points and will be a good fit to be used in the current research for landmark detection. Later the points were used for further angle calculation to detect any harmful posture of any human in the video. This paper is useful to understand the working of MediaPipe library to be implemented for landmark detection and apply it for current scenario with images as input to the model.

In Chung et al.; 2022, the paper illustrates a comparative analysis of different machine learning pose detection models like OpenPose, PoseNet and MediaPipe's Blaze pose, MoveNet and Frame28. By considering the working of each model on the chosen dataset and occurring on the inference by conducting number of experiments, which will convey the best working model from all. The comparison result depicts that the among all the selected models, the Mediapipe's Blaze pose model specifically was able to handle the challenges of the pose detection well, the evaluation metric used was percentage of detected joints (PDJ) that was highest for this method. The model handled challenges like incomplete pose in the image or for video experiments, wrong camera position and able to detect the pose giving acceptable results. This study concludes that using Mediapipe pose model for pose detection will be a good choice to go, especially if the dataset is of type Image.

In Anilkumar et al.; 2021, The paper proposed the methodology for establishing a system for estimating yoga poses. This system achieves pose estimation by using MediaPipe library to track and analyze the poses in real time. The paper discusses the comparison of detected poses using MediaPipe with the reference data available in the dataset which has built-in landmarks. The system further utilises these keypoints along with deep learning models to perform pose detection, hand tracking recognition along with other models used in computer vision. After analysing the dataset, systems gives feedback to the user whether the pose is having correct posture or needs improvement.. An important aspect of this paper is that, it uses only the three points given by MediaPipe as the input and calculates all the angles at joints manually through a function, this helps is getting more accurate results. Augmentation is missing that will increase the performance well.

In Zhang et al.; 2022, the paper proposed a framework for detection of deep squat motion based upon MediaPipe and Yolov5 models. The model presents a 3D detection system. Due to having ambiguity in the bounding box and obtaining incomplete semantic data while performing estimation of each frame, it uses a human body detecter in the first frame for calculating perpectual area and then sends this as an input to the keypoint detection – Mediapipe pose model. This results in using previous keypoints to detect the next frame's keypoints. The paper discussed three experiments – using Yolov5, MediaPipe and Yolov5+MediaPipe combined. The results depicts that the combined version gives better performance, whereas in particular, MediaPipe had better results than Yolov5

when used individually. The class imbalance issue has been encountered in this paper that improved the performance.

In Li et al.; 2022 Taware et al.; 2022 the papers offers a framework that explores how MediaPipe – BlazePose model is able to overcome the limitations of fitness movement recognition which will be a solution for realtime tracking of the movements in the we and mobile applications. This paper stated that it serves an effective solution than the existing fitness apps/webtes available in the market because it utilized MediaPipe as pose detection algorithm. Li et al.; 2022 additionally implements both models - MediaPipe Blaze Pose Full and Blaze Pose lite and as a result it gives number of repetition of the exercise performed too. Taware et al.; 2022 specifically focuses on only one movement Bicep Curl exercise and by calculating the right angle from the obtained keypoints, it counts number of bicep curls performed by the user.

In Halder and Tayade; 2021 The paper focuses on real time recognition of vernacular sign language – used by people with hearing disabilities – deaf people. Th methodology focuses on detecting the hand sign language from the images by detecting the keypoints from the hand using MediaPipe pose model. Detected landmarks will be used further to performing classification of the sign language using Machine learning models. The results showcased that the complex task of identifying hand landmarks was completed by MediaPipe without a need of using Convolutional Neural Network from scratch.

2.2 Classification using Machine learning models

In Sunney et al.; 2023, the paper discusses various experiments conducted for accurate yoga pose estimation using real time feedback by detecting five yoga poses unsupervised practice at home. The poses detection was based on Open Pose model whereas for classification and detection, machine learning and deep learning models were implemented. Based on latency, size and accuracy four machine learning models – Decision Tree, SVM, Random Forest, XGBoost and two deep learning models LSTM and CNN were implemented and analyzed for a comparative analysis. The result indicated that XGBoost gave optimal performance with highest accuracy 95% ,smallest latency 8ms and smaller size than other models. The limitation of this research paper is that it is not suitable for multi-person detection.

In Halder and Tayade; 2021 Unkule et al.; 2022, Sharma et al.; 2014,these papers focus mainly on different sign language recognition using machine learning models. The hand gesture recognition is completed using OpenCV library , the sign language detection is completed using machine learning models – SVM, Random Forest, KNN. K-Fold cross validation was performed to validate all the models and comparison has been made to obtain most optimal algorithm for the chosen task. Results showcased that the KNN and Random Forest performed well than SVM. The limitation to this research is that model is suitable for the limited dataset only hindering the ability to work for generalized ground.

In Kanase et al.; 2021, the paper mainly focuses on extraction of datapoints using OpenPose library from video recording and applying machine learning algorithms to detect the correct posture while performing weight training exercises. As the video recording will vary in its duration, it creates errors in the vector length for every example. To resolve this issue, a novel approach of using DTW distance in KNN classifier instead of Euclidean Distance – fails to detect phase-shifted sequences in time, has been implemented.

In Palanimeera and Ponmozhi; 2021, The yoga pose detection utilized four machine

learning models – KNN, SVM, Naïve Bayes, Logistic Regression for classification of Salutation poses in the Yoga. Among these algorithms, KNN worked giving better results than other algorithms. Limitation to this paper – no clear explanation of evaluation metrics, only the implementation part and results are given in the paper.

In Sheng et al.; 2018,The paper focuses on a learning-based approach for detection of road cracks. The gradient boost algorithm is utilized for the classification task. The models input a dataset of road crack images for the training purpose and these images will be also used as a ground truth for the classification task. The experiments use Precision, Recall, F-Score for the evaluation purpose. The limitation for the research is that the ensemble model is unable to combine the labels as ground truth. A much bigger training dataset is needed to overcome this issue. The current dataset limits itself to static dataset, it should be able to detect the crack from real-time or video datasets that will make the research more useful on generalized ground.

In Aravind et al.; 2019, Ezhilraman et al.; 2019, Das et al.; 2022, the papers focuses on classification task using Gradient Boost Ensemble Decision Tree Classifier for Breast Cancer Detection, classification of Healthy and Rotten leaves of Apple, Brain Image classification respectively. All the three papers use fundamental methodology of utilizing steepest descent function with ensemble classifier for obtaining best classifier results by keeping loss to minimum. This also minimizes the classification time. In Das et al.; 2022 optimized Gradient Descent Classifier is utilized by updating the parameters – learning rate and number of trees by using a customized or novel algorithm proposed in the paper. Accuracy and Classification time were the metrics used to evaluate the model's performance and resulted to get the improved results than the other existing models.

In Shorten and Khoshgoftaar; 2019 Mikołajczyk and Grochowski; 2018, both the papers focus mainly on pre-processing techniques - augmentation of images before classification using deep learning techniques for improvised results. Traditional techniques such as shearing, scaling (zoom in or zoom out), reflection, rotation, horizontal flip are implemented on the input image dataset. The results depicted these techniques prove to be easy to implement and reliable as well as reproducible in nature. The augmented images are then used as input for deep learning models to complete the classification and detection tasks. The limitation of these paper is that it completely relies on the augmented images to improve the effectiveness of the machine learning models and not hyper optimizing these models to increase the performance a bit more.

After getting an overview by analyzing all the papers - the techniques and limitations utilized, this research will purely be experimental based by implementing different machine learning models - KNN,Random Forest, Decision Tree classifier, SVM, Gradient Boost to perform classification of Yoga pose images. By researching all the limitations mentioned in all the above papers it has been clear that none of these papers have used augmentation of images as preprocessing technique along with hyper parameterized machine learning algorithms. The deep learning models being huge in computational size, this research will mainly focus on using relatively small computational sized machine learning algorithms to achieve the classification task by using extracted landmarks from images using MediaPipe library.

In conclusion, given the computational demands of the deep learning models, this research will mainly focus on less computationally costly machine learning models to complete the classification Yoga pose images. The landmarks will be extracted by leveraging the MediaPipe library and experiment with the machine learning models – KNN, SVM,

Decision Tree Classifier, Random Forest Classifier, Gradient Boost Classifier. The papers discussed above leave the gap of fine tuning the hyper parameter for optimization to get better results combined with augmentation of extracted landmarks as preprocessing step necessary for improving the performance of these machine learning models. This novel approach has not been extensively explored in the context of Yoga pose classification task. This research will focus on sidestepping the computational and time complexities associated with deep learning models. Instead will leverage the use of the combined methodology of key point extraction, augmentation of images and fine tuning the hyper parameter optimized machine learning algorithm to achieve balance between computational efficiency and classification accuracy in analyzing the Yoga poses than existing literature.

3 Methodology

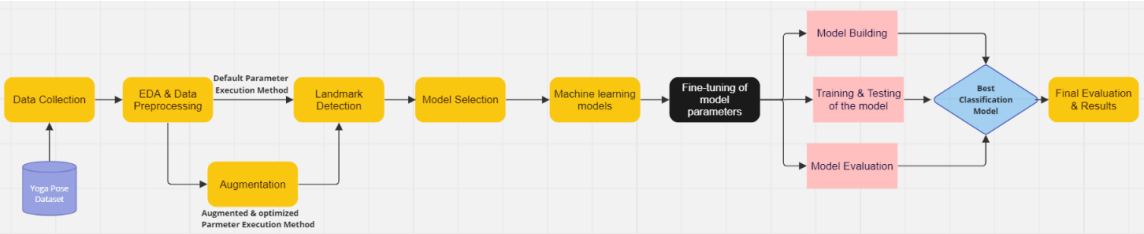

Figure 1: Methodology Work Flow

The fundamental process is utilized as described in above figure 1, for defining the methodology for this research, the process consists of following steps - Data collection, Dataset Preprocessing, Exploratory data analysis, Augmentation, Landmark detection, Model selection, for each model chosen - Model Building, training and evaluation will be eventually repeated for hyper-parameter tuning , Final evaluation. The first step was collection of data. In this scenario, image data has been used which is publicly available on Kaggle website [1]. The dataset has in total five yoga poses which are widely performed and known worldwide – Tree (219 images), Warrior (361 images), downdog (320 images), goddess (260 images), plank (382 images). These images are stored in separate folders named with the name of the poses. The dataset is already divided into train and test folders, each containing the same number of folders having the same poses as the training dataset folder. The overall dataset contains 1542 images with a size of 438MB. Figure -3 illustrates few images printed to get an overview of the dataset.

Once the dataset is finalized, the next step is performing Exploratory Data Analysis (EDA). To understand the distribution of the data or images among five classes, the following are the visualizations performed to get an insight about the selected dataset. figure 2 contains the class distribution where class 'Plank' has more images and class 'tree' has low images where image augmentation is needed. It can be observed that there is a class imbalance issue seen from the class distribution image. Image augmentation is one method to handle this issue or using sample_weights attribute to the model while model is defined.

[1]https://www.kaggle.com/datasets/niharika41298/yoga-poses-dataset/data

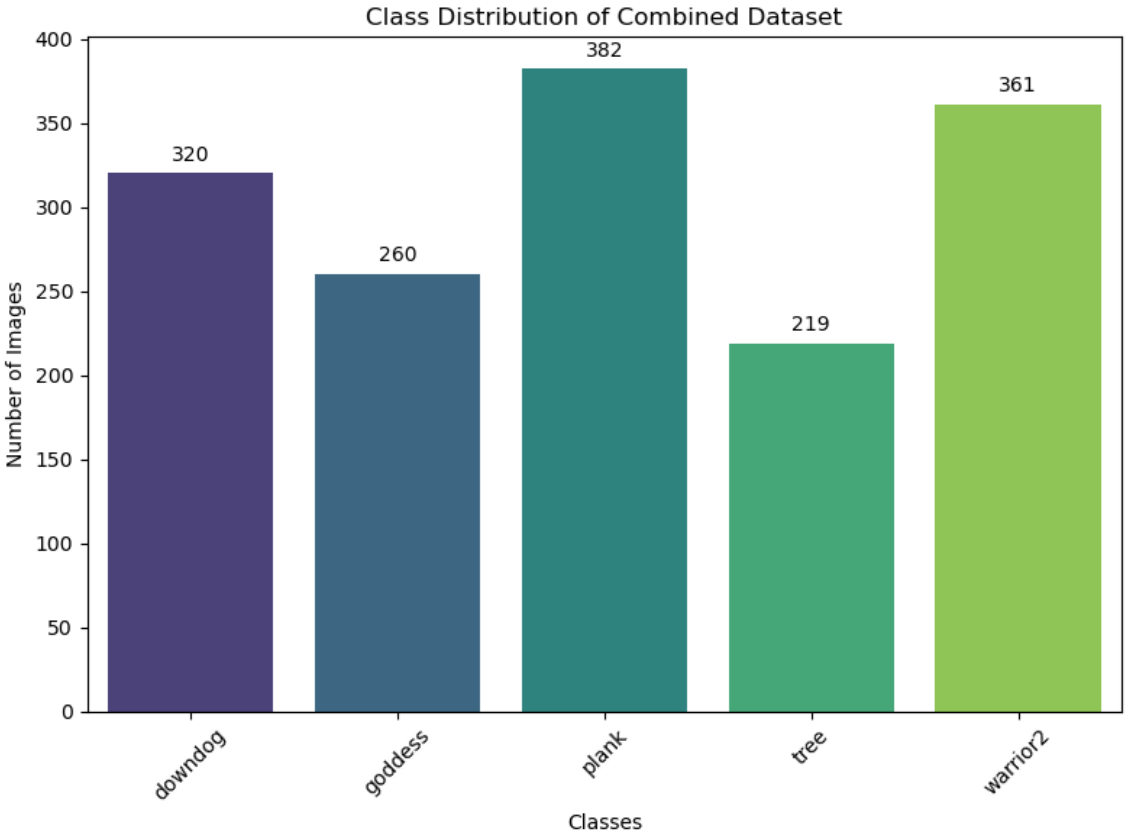

Figure 2: Class distribution Bar Chart

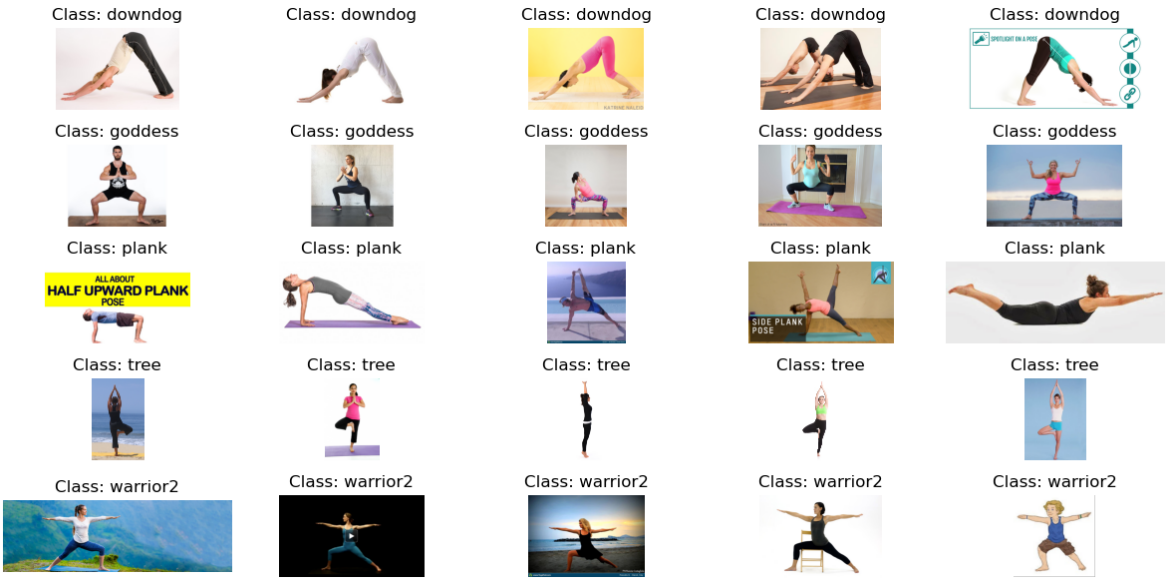

Figure 3: Sample images from each Class

Completion of EDA brings to the next step that is performing preprocessing on the dataset. For image dataset, image augmentation is the most applicable preprocessing technique. As this dataset is a bit critical in nature – has Yoga poses performed in these images, using any image augmentation technique is not applicable to it. If done so, the true meaning of the image will be disturbed leaving the image meaningless to be classified. Hence, augmentation techniques like flipping – horizontally, rotating by few degrees only has been used. Following figure -4 depicts the augmented images:

Figure 4: Images after applying augmentation technique

The number of augmented images is equal to the number of images in the original dataset. This augmented dataset is used for further processing, experimentation, and analysis. Next preprocessing step involves extraction of landmarks or key points from the images. Landmark detection consists of recognizing and tracking keypoints from the human body and generate spatial relationships between them. This process is done using Mediapipe library – an open source or framework or library developed by Google which contains various pre-trained Machine Learning models built for complex tasks like face detection, pose estimation, landmark detection and many more. Keypoints detection involves identification of human body parts such as eyes, nose, mouth, shoulders, elbows, wrists, hips, knees, and ankles. Mediapipe is most applicable for real time processing, supports cross platform working and low latency processing making it more suitable to be used for key point identification.

In this scenario, from the chosen poses, 'plank' is one such pose which has the human body partially visible, in this case the working of the landmark extraction logic – MediaPipe library after processing each image, it attempts to detect all the landmarks for the non-visible body parts too. In the given image, the Plank pose image - fig 1, the skeleton mapped on the image showcases the key points for the other side of the body too. This results in obtaining all the landmarks of the body – 33 points regardless of how much body is visible – partially, incomplete or fully visible in the input images making the classification task more accurate in performance.

The above figure - 5 illustrates the 33 landmarks points required for pose classification. Using these landmarks, it is possible to detect the pose and perform classification task. The models will be trained on the images but on the coordinates obtained from the landmark detection process only. For each of the landmark shown in fig5, four coordinates are computed X,Y,Z,Visibility. These are responsible to locate each landmark accurately on the image.

Model Selection: For pose classification purposes, Machine learning models such as Support Vector Machine, Random Forest Classifier, Decision Tree Classifier, K-Nearest Neighbor Classifier has been used for classifying the images into five poses.

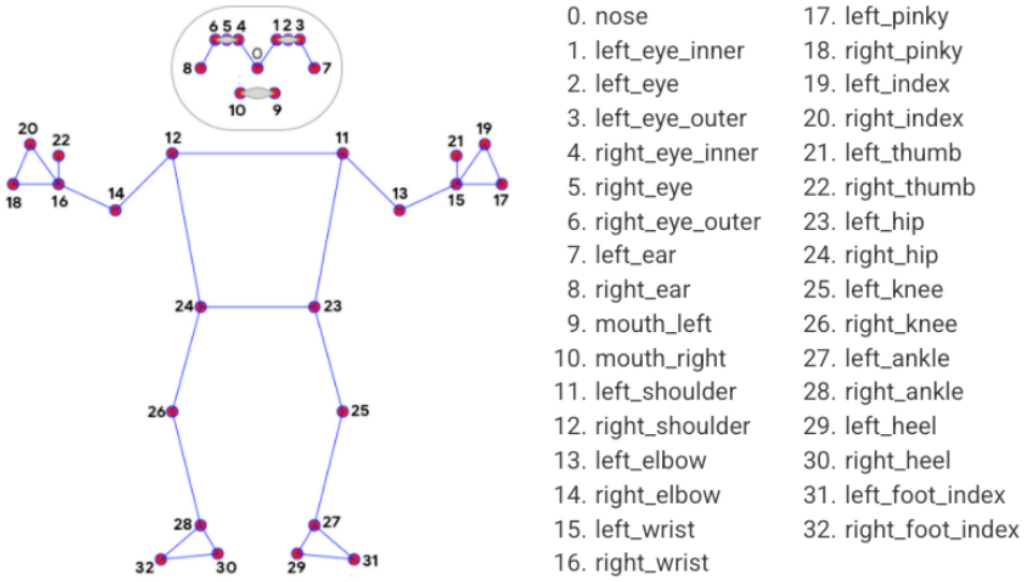

Figure 5: Human Body Landmark Points

• Support Vector Machine Classifier: This model is efficient enough to handle high-dimensionality data and handle complex decision boundaries. Most importantly it is applicable to image dataset because of its ability to map the data to high-dimensionality classes and create best suited decision boundaries for all the classes. This is also applicable in both linear and non-linear classification tasks. Hence this will be implemented in yoga pose classification task.

• Random Forest Classifier: Due to its ensemble learning techniques which combines number of decision trees together to make predictions, this technique is most suited for classification task. This model is also good at handling overfitting problems and capable of extracting diverse features from the images for the classification task.

• Decision Tree Classifier: It is one of the most powerful classification algorithms due to its capability of creating a tree like structure by recursively splitting the data based on its features and making decisions at each node. This model can handle both numerical and categorical data, hence best suited for image classification tasks.

• KNN Classifier: This algorithm utilizes simplest approach; it classifies the data points based on the nearest neighbor's data points from the feature space depending on the similarity within these points. KNN is one of the best suited in pose classification tasks as it can correlate with different poses available in the feature space.

• Gradient Boost Classifier: It is one of the most powerful ensembles learning methods that builds a structure of weak learners – decision trees sequentially to create a strong prediction model. It is most suited because of its work that consists of constructing new models by correcting the errors made by previous models and then improving the accuracy gradually. Applicable to pose classification as it is flexibly able to handle

various data characteristics and minimize the overfitting allowing to learn the relationship between poses effectively and giving better predictions.

4 Design Specification

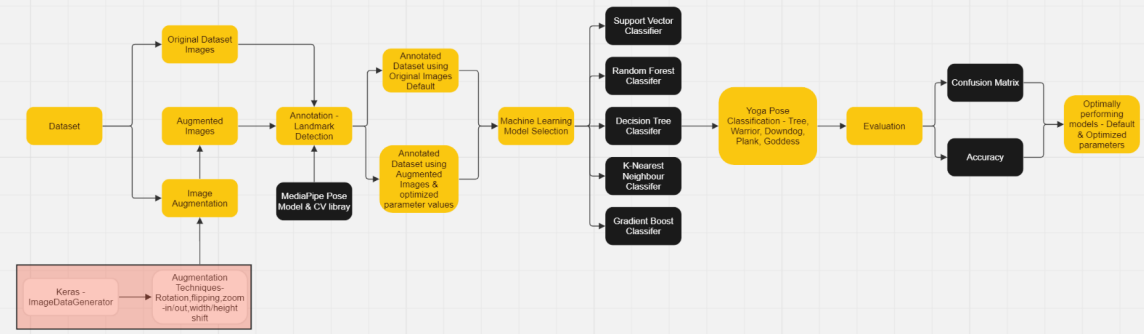

Figure 6: Design - Work Flow Diagram

The above figure - 6 illustrates the work flow for the implementation of proposed methodology. The process is initiated by obtaining the Yoga pose dataset. The said workflow will have two path ways, first method will be using original dataset without image augmentation and second method will be utilizing image augmentation technique - Keras library's 'DataImageGenerator' method that will generate augmented images according to the transformation defined. The augmented dataset and original dataset will now be utilized for completing the landmark detection using MediaPipe and ComputerVision library for reading each image from the dataset for processing. This step will generate two datasets with landmarks detected - original dataset and augmented dataset. Original dataset with default parameters and augmented data coupled with fine-tuning of parameters will now be used for model building using Machine learning models - SVC,Random Forest, Decision Tree, KNN, Gradient Boost classifier. To evaluate the model, Confusion matrix and Accuracy are the two metrics utilized for this research. Final evaluation will be performed by comparing the results obtained from each model eventually concluding with optimal working model from both the scenarios. Following section will focus on the implementation of the proposed design work flow.

5 Implementation

According to the methodology proposed in the above section, the first step in the implementation phase is working on original dataset and performing extraction of landmarks from the images. Media pipe library – pose detection module is utilized in this research. For obtaining the landmarks, Pose Landmarks attribute is used which contains all the predefined - 33 landmarks from the media pipe library. Initially read the image dataset using OpenCV ('cv2') library and processed each image by converting it from BRG to RGB format and identified the height and width of the images needed for further processing. For pose estimation and landmark detection, each image is processed using Pose model from Media Pipe library which returns the result containing 33 landmarks along

with X, Y, Z and visibility coordinates and Pose_Connections attribute returns the connectivity between each key point. The extracted landmarks are then stored in a CSV file for further processing. Next step is to start the second flow, performing Image augmentation on original dataset.For augmentation, Keras library's 'ImageDataGenerator' method is utilized for performing various augmentation techniques like rotation, flipping, horizontal flip, zoom -in/out. Augmented images generated from this process are stored in a folder to be used while performing Landmark detection.The similar processing of landmark detection is performed on the augmented dataset too. At this point, two CSV files containing landmarks - with original dataset and one CSV file containing landmarks after performing augmentation, will be ready to be used for model building and performing classification task. The model building will be performed on the both landmarks dataset computed. For classification purposes, one additional target variable is also appended at the end which contains encoded values for each class respectively. The created CSV file is read, and the data is converted to two classes – X_train – independent variable and Y_train – dependent or target variable containing target column values, similarly, testing data is also imported and X_test , y_test variable is created for further modelling of machine learning models.

Model implementation:

5.1 Support Vector Machine Classifier (SVC)

Experiment 1: Without handling class imbalance issue:
SVC was utilized for pose classification tasks. While defining the model, polynomial kernel was used to capture non-linear relationships in the data. Probability was set to True so that probability estimates for the predicted classes can be gained. Random State = 42 was assigned to ensure reproducible results. Once SVC model was trained with these attributes, the model was trained using training data.
Experiment 2:Handling Class imbalance issue using sample_weight attribute:
To improve the performance of SVC model by handling the class imbalance issue, sample_weight attribute has been utilized for the pose classification task. At the step of model definition, class_weight = 'balanced' and Class frequencies were calculated and sample weights which are inversely proportional to frequencies of the classes. Less frequent classes will get assigned the highest weight so that balance will be established and will improve the model training performance. These calculated weights will get assigned to their corresponding instances present in the training dataset according to their class labels. This will ensure that less frequent classes contribute largely to the model training, this will avoid model results being biased towards classes with majority frequency and solve the issue of class imbalance.

5.2 Random Forest Classifier

Experiment 1 - Utilizing Default values without hyper parameter optimization:
Defined Random Forest Classifier without performing hyper parameter optimization by using 'RandomForestClassifier; from 'sklearn.ensemble' library without modifying its default hyper parameters. Created random forest classifier with 100 trees that is, n_estimators=100 and also assigned random_state = 42 for achieving reproducibility. This the baseline model for analyzing the working of the model without any hyperparameter optimization. The model is then trained using train dataset to understand the patterns between independent and dependent variables.The,tested the working of the model on unseen data using test dataset.

Experiment 2-To improve the model's performance, it is essential to perform hyperparameter optimization on the model
The above experiment showcased the base working of the model. This experiment will try to improve the performance of the model as hyper parameters have control on the model's behavior and performance. 'RandomizedSearchCV' is used to search among various combinations of hyperparameters, as a result to get one best combination for model building. Following are the hyperparameters used to hyper tuning the model: 'Parameter Space Definition ('param_dist') consists of a dictionary with common hyperparameters like n_estimators, max_depth,min_samples_split and min_samples_leaf and their respective value ranges that RandomizedSearchCV will explore. This is a defined parameter space for the model. The RandomizedSearchCV will perform number of iterations 'n_iter' and do cross-validations 'cv' which is based on specific metric like accuracy. Once the search is completed, RandomizedSearchCV will identify the best suited pair of hyperparameters which are optimal and will maximize the scoring metric. The model is trained using random_state=42 for reproducibility and also best_params are sent which are the obtained hyperparameters from RandomSearchCV output. Performed 10 iterations of randomized hyperparameter search having 5-fold cross validation. Then the model is tested using the testing dataset to check the model performance on unseen data.

5.3 Decision Tree Classification

Experiment 1 - Without using hyper parameter optimization:
As the base model, the Decision Tree Classifier was initialized with default parameters without any fine-tuning of the hyper parameters. This will serve as a starting point for the model building and prediction purposes. Once the classifier was defined, it was trained using training dataset and tested against testing dataset to check the performance on unseen data. This experiment serves to obtain the default or base performance of the model without any fine-tuning.
Experiment 2 - Implementing hyper parameter optimization:
Using defined parameters for exhaustive search, the Decision Tree Classifier was fine tuned to get best suited parameters for maximum model performance. 'max_depth' set to 5, 'min_samples_split' set to 2 and 'min_samples_leaf' set to 1 were identified to be most optimal hyperparameter values and the model was initialized with these values so that after training the model and testing on unseen data it will give better classification results than the non-hyper parameterized model.

5.4 K-Nearest Neighbor's Classifier (KNN)

Experiment 1:
To predict the target labels, KNN classifier is implemented and this experiment consists of basic model without any hyper parameter optimisation. The KNN is initialized using 'KNeighborsClassifier' from Scikit-Learn library. The model was configured with 'n_neighbors=5' which interprets that the algorithm will follow the rule of finding 5 nearest data points to make predictions. The model was then trained using training data and tested against unseen testing data to verify the model performance.
Experiment 2:
After obtaining base model in experiment 1, this experiment will perform hyper parameter optimization by fine-tuning the model using these parameter values. The KNN model was finetuned by searching in the specified hyper parameter grid. This grid consists of 'n_neighbors=8,10,12 and 14' which represents the number of nearest neighbors

which will perform classification, 'weights = uniform and distance' both the attributes are used here. The uniform attribute to make sure that all the neighbors are treated equally and distance weights each neighbor by the inverse of their distance. To calculate the nearest distance, various algorithms were computed like 'auto', 'ball_tree', 'kd_tree', 'brute'. To perform the Grid Search and cross validation, 'GridSearchCV' method was used which will properly assess each hyper-parameter defined in the parameter grid using 5-fold cross validation to find the best suited configuration for model to give better performance. Once the best model for KNN classifier was found, it was trained using training data and tested against testing dataset.

5.5 Gradient Boost Classifier

Experiment 1:
This experiment is employed to get the base model performance without any fine-tuning of hyper-parameter optimization for Gradient Boost Classifier. The model was initialized with default settings – random state as 42 for the reproducibility keeping as only parameter while defining the parameter. The model was then trained using training dataset which will help model to understand the relationship between the data-points required for the classification. The model then was tested against the testing dataset to check the performance of the model for unseen data. This was just the basic model that will be considered as starting point to improve the performance of the model in further experiments.
Experiment 2:
This experiment is computed to perform fine-tuning of Gradient Boost parameters to obtain better performing than the base model computed in experiment 1, The n_estimators=100, learning_rate=1.0, max_depth=1,random_state=42 was used as parameters to define the model. Next step is to train the model using training dataset and test the model against the testing dataset to check the model's prediction for unseen data. subcaption

6 Evaluation

This section discusses the results, evaluation metric used for each experiment performed as a part of this research. All the experiments involve implementation of models with default parameters and with hyper parameter optimization techniques applied coupled with augmented dataset. Confusion matrix, Accuracy score, Classification report - F1-score, Precision, Recall, Cross Validation score are the evaluation metrics utilized to evaluate the model's performance.Each cell's color intensity represents the number of instances where the actual class (y-axis) was predicted as the corresponding predicted class (x-axis).

6.1 Support Vector Machine Algorithm (SVC)

Referring to the figure - 7, Both models show relatively high accuracy along the diagonal, especially for the main diagonal elements, indicating accurate predictions for those classes. The optimized model has slightly improved accuracy for some classes, such as Class 1-'Plank', Class 2-'Warrior', and Class 4-'Downdog'. Miss-classifications for Class 5-'Goddess' seem to have reduced in the optimized model compared to the base model. The optimized model shows reduced miss-classifications for Class 2-'Warrior' as Class 5-'Goddess' compared to the base model. Notably, miss-classifications for Class 3-'Tree'

as Class 5-'Goddess' have decreased in the optimized model as well. Class 4-'Downdog' in the optimized model seems to have a higher accuracy compared to the base model, as it has lower off-diagonal values. However, there might be trade-offs, such as a slightly higher miss-classification of Class 2-'Warrior' as Class 5-'Goddess' in the optimized model compared to the base model. Overall, the optimized SVC model demonstrates improvements in accuracy and reductions in miss-classifications for several classes compared to the base model.

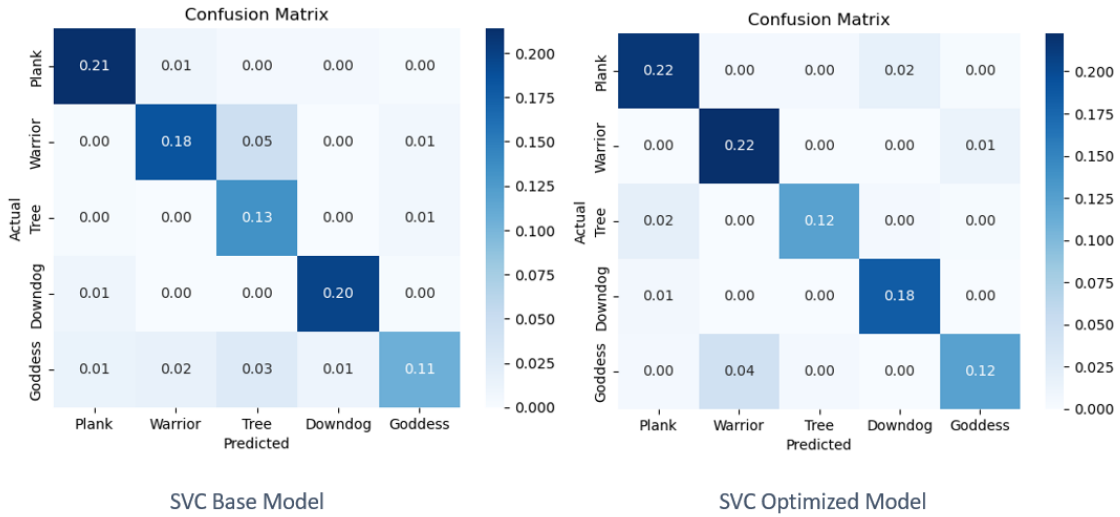

Figure 7: SVC Confusion Matrix

6.2 Decision Tree Classifier Algorithm (DTC)

Referring to figure 8, the optimized model shows a decrease in miss-classifications for Class 1-'Plank' as Class 2-'Warrior' compared to the base model. Miss-classifications for Class 3-'Tree' as Class 5-'Goddess' seem to have decreased in the optimized model compared to the base model. The optimized model exhibits increased miss-classifications for Class 5-'Goddess' as Class 2-'Warrior' compared to the base model. Miss-classifications for Class 1-'Plank' as Class 5-'Goddess' have also increased in the optimized model. Class 2-'Warrior' in the optimized model seems to have improved accuracy compared to the base model, as it has lower off-diagonal values. However, there might be trade-offs, as certain classes show increased miss-classifications while others exhibit improved accuracy in the optimized model compared to the base model. Overall, the optimized Decision Tree Classifier model shows improvements in accuracy for certain classes but might demonstrate increased miss-classifications in other classes compared to the base model.

6.3 KNN Classifier Algorithm (KNN)

Referring to figure 9, the optimized model has slightly higher accuracies – correct predictions for Class 1-'Plank' and Class 4-'Downdog' as compared to the base model. Whereas the base model has higher accuracies for Class 2-'Warrior' and Class 5-'Goddess' compared to optimized model. In case of miss-classification, optimized model has higher miss-classification percentages for Class 2-'Warrior' than Class 5-'Goddess' compared to

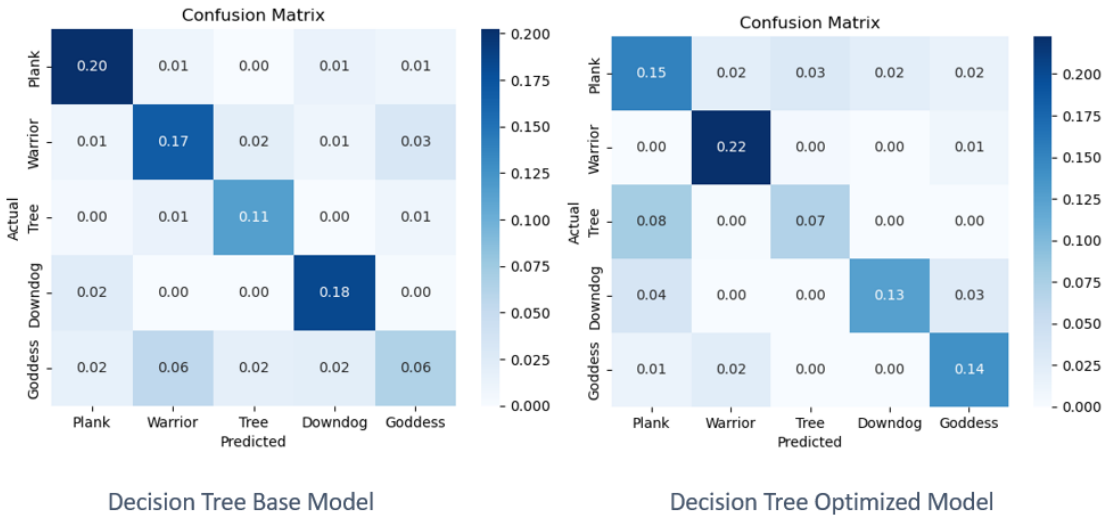

Figure 8: Decision Tree Classifier Confusion Matrix

base model. Whereas base model shows higher miss-classification percentages for Class 3-'Tree' as Class 2-'Warrior' compared to optimized model. Overall, the optimized model performs better for class 1-'Plank' and Class 4-'Downdog', while base model performs better for Class 2-'Warrior' and 5-'Goddess'. Class 2-'Warrior' appears to be particularly challenging for both models.

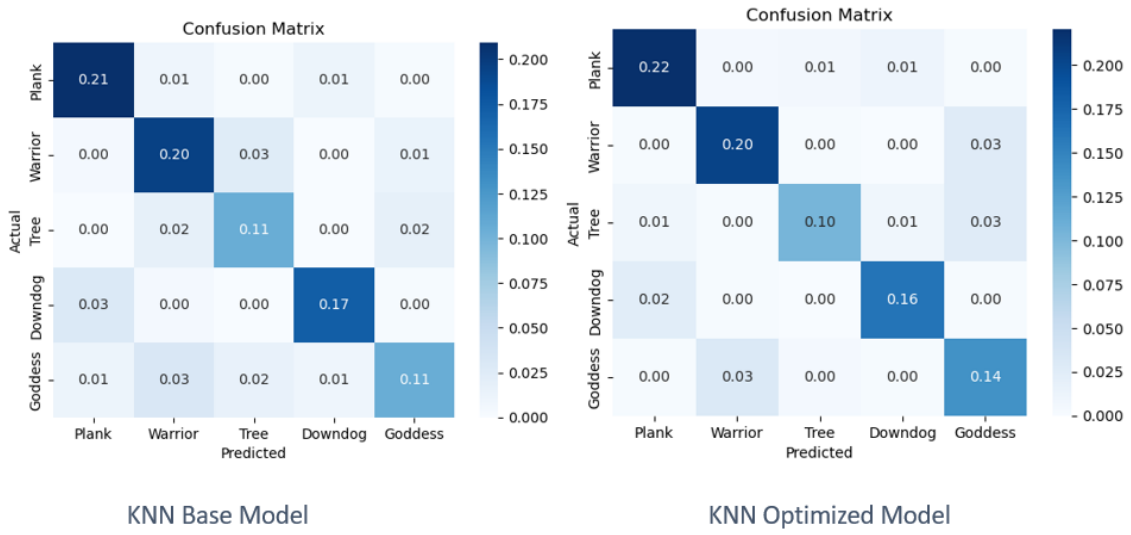

Figure 9: KNN Confusion Matrix

6.4 Gradient Boost Classifier Algorithm(GBC)

Referring to the figure 10,the optimized model generally shows improvements by having better accuracy for certain classes. For instance, the optimized model has higher accuracies for Class 1-'Plank', Class 2-'Warrior', and Class 5-'Goddess' compared to the

base model. The optimized model seems to have reduced miss-classifications compared to the base model for several classes, like Class 1-'Plank' being miss-classified as Class 2-'Warrior' or Class 4-'Downdog'. Notably, miss-classifications for Class 3-'Tree' and Class 5-'Goddess' have been reduced in the optimized model. Classes like 1-'Plank', 2-'Warrior', and 5-'Goddess' exhibit noticeable improvements in accuracy in the optimized model, indicating better performance for these classes compared to the base model. The optimized model appears to have focused improvements on certain classes (Class 1-'Plank', Class-2-'Warrior') where accuracy has notably increased. In summary, the optimized Gradient Boost Classifier model shows improvements in accuracy and reductions in miss-classifications for several classes compared to the base model.

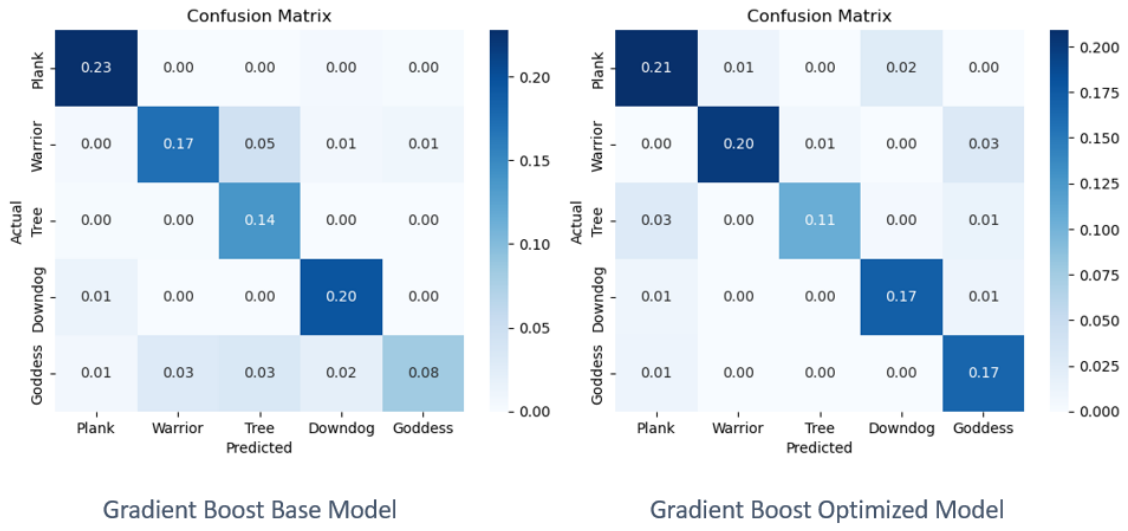

Figure 10: Gradient Boost Classifier Confusion Matrix

6.5 Random Forest Algorithm (RFC)

Referring to figure 11, both models show relatively high accuracy along the diagonal, indicating accurate predictions for those classes. The optimized model seems to have reduced miss-classifications for Class 1- 'Plank'as Class 2 - 'Warrior'compared to the base model. miss-classifications for Class 5-'Goddess'as Class 2-'Warrior' have decreased in the optimized model compared to the base model. The optimized model shows increased miss-classifications for Class 3- 'Tree' as Class 5-'Goddess'compared to the base model. Notably, miss-classifications for Class 4- Downdog' as Class 1-'Plank' have increased in the optimized model. Class 2-'Warrior' in the optimized model seems to have a higher accuracy compared to the base model, as it has lower off-diagonal values. Overall, the optimized Random Forest model shows improvements in accuracy for certain classes but might exhibit increased miss-classifications in other classes compared to the base model.

Evaluation using Accuracy Metric:

Following is the image figure - 12 depicting the accuracy of each machine learning model when implemented by using default parameters/ base model and when computed hyper-parameter optimization. It can be observed that for SVC,KNN,GBC the optimization

Figure 11: Random Forest Confusion Matrix

of parameters improved the accuracy of the model whereas, for RFC and DTC the performance did not improve as expected. To get detailed information, the table - 13 given below illustrates the accuracies of training and testing dataset for each model.

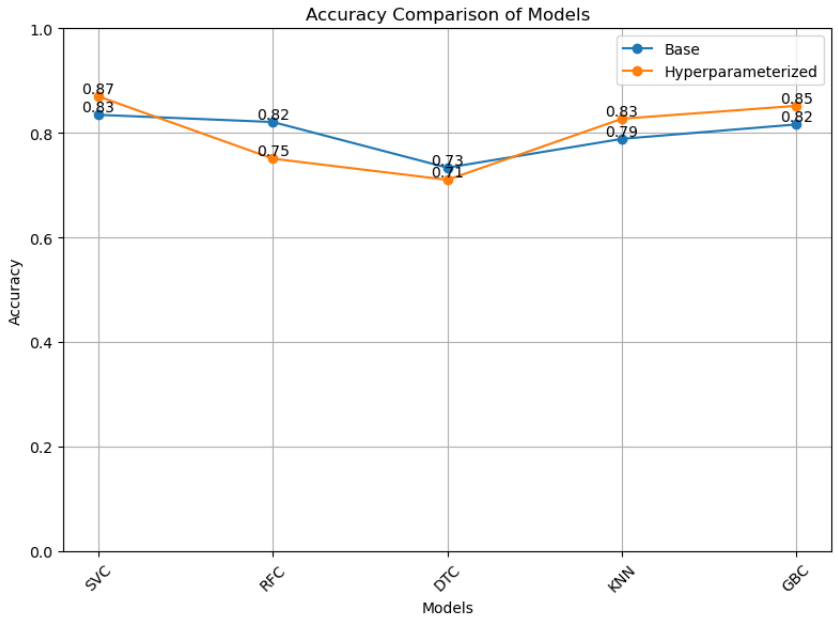

Figure 12: Accuracy line chart across both implementations

Test Dataset Accuracy Evaluation		
Machine Learning Models	Base Model	Optimized Model
Support Vector Classifier	83%	87%
Random Forest Classifier	82%	75%
Decision Tree Classifier	73%	71%
K-Nearest Neighbour Classifier	79%	83%
Gradient Boost Classifier	82%	85%

Table 13: Accuracy Table

6.6 Discussion

As a result of all the experiments being conducted on the Yoga Pose dataset, after comparatively analyzing the results of each experiment, it can be observed that some of the Machine learning model improved its performance after performing hyper-parameter tuning. To be specific among all the models, the best performing model among all is the SVC optimized model giving highest accuracy – 87%, whereas least performing model is the Decision Tree classifier optimized model with the accuracy of 71%. For base model, Random Forest Classifier and Gradient Boost Classifier has performed optimally with accuracy of 82%. The fundamental reason for improvement not seen in the performance of optimized Decision Tree Classifier and Random Forest is because of the augmentation not correctly done coupling with better optimization of parameters needed. For the specific models, more image augmentation is needed that will aid in increasing the model performance. The findings of this paper when compared to the existing literature, in Sunney et al. (2023), the paper focused on SVC algorithm performance improvement without using Augmentation, the results were improved.Similarly, in this paper, all the machine learning models after implementing with the said methodology in this paper - utilizing augmented dataset and applying fine-tuned hyper-parameter optimization, the results are improved and acceptable for SVC, KNN, Gradient Boost Classifier whereas, improvised augmentation and optimization is needed for Random Forest and Decision Tree Classifier.

7 Conclusion and Future Work

This paper proposes a learning-based approach to perform a comparative analysis for classification task of Yoga images using different machine learning algorithms. The use of MediaPipe Pose library completed the landmark detection task by giving acceptable results, even if the image contains images that are partially or incompletely visible, the model was able to extract all the 33 landmarks almost correctly that served an important aspect for getting accurate results in the further modelling of machine learning model for the classification task .All the experiments improved the model's performance gradually. Few models - SCV and Gradient boost with default parameters gave low accuracy results,

when the same model was applied on augmented dataset the performance was good, when the augmented model was further fine-tuned with hyper parameters the results got better in terms of accuracy. Several issues such as class imbalance was encountered in the process, using appropriate parameters like sample_weights, the issue was handled giving desired results. There are models like Decision Tree and Random Forest algorithm that did not improve its performance significantly.The reason behind is requirement of the augmentation and optimisation to improve the performance. To answer the research question mentioned in the introduction part, the research was capable to find few of the more optimized models working with the said methodology. In future works, it would be beneficial to understand the limitation of these algorithms showcasing not so good improvement in their performance. This can be completed using more fine-tuned hyper parameters, applying various class imbalance handling techniques or by updating the dataset.Applying real-time processing for successful models given better results, is also one of the major future works that will eliminate the static image dataset limitation for realistic performance of the models.

References

Anilkumar, A., KT, A., Sajan, S. and KA, S. (2021). Pose estimated yoga monitoring system.

Aravind, K., PraylaShyry, S. and Felix, Y. (2019). Classification of healthy and rot leaves of apple using gradient boosting and support vector classifier, *International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN* pp. 2278–3075.

Chung, J.-L., Ong, L.-Y. and Leow, M.-C. (2022). Comparative analysis of skeleton-based human pose estimation, *Future Internet* **14**(12): 380.

Das, A., Mohapatra, S. K. and Mohanty, M. N. (2022). Brain image classification using optimized extreme gradient boosting ensemble classifier, *Biologically Inspired Techniques in Many Criteria Decision Making: Proceedings of BITMDM 2021*, Springer, pp. 221–229.

Ezhilraman, S. V., Srinivasan, S. and Suseendran, G. (2019). Breast cancer detection using gradient boost ensemble decision tree classifier, *Int. J. Eng. Adv. Technol* **9**: 2169–2173.

Gadhiya, R. and Kalani, N. (2021). Analysis of deep learning based pose estimation techniques for locating landmarks on human body parts, pp. 1–4.

Halder, A. and Tayade, A. (2021). Real-time vernacular sign language recognition using mediapipe and machine learning, *Journal homepage: www. ijrpr. com ISSN* **2582**: 7421.

Kanase, R. R., Kumavat, A. N., Sinalkar, R. D. and Somani, S. (2021). Pose estimation and correcting exercise posture, *ITM Web of Conferences*, Vol. 40, EDP Sciences, p. 03031.

Li, X., Zhang, M., Gu, J. and Zhang, Z. (2022). Fitness action counting based on mediapipe, pp. 1–7.

Mikołajczyk, A. and Grochowski, M. (2018). Data augmentation for improving deep learning in image classification problem, *2018 international interdisciplinary PhD workshop (IIPhDW)*, IEEE, pp. 117–122.

Palanimeera, J. and Ponmozhi, K. (2021). Classification of yoga pose using machine learning techniques, *Materials Today: Proceedings* **37**: 2930–2933.

Pauzi, A. S. B., Mohd Nazri, F. B., Sani, S., Bataineh, A. M., Hisyam, M. N., Jaafar, M. H., Ab Wahab, M. N. and Mohamed, A. S. A. (2021). Movement estimation using mediapipe blazepose, pp. 562–571.

Sharma, M., Pal, R. and Sahoo, A. K. (2014). Indian sign language recognition using neural networks and knn classifiers, *ARPN Journal of Engineering and Applied Sciences* **9**(8): 1255–1259.

Sheng, P., Chen, L. and Tian, J. (2018). Learning-based road crack detection using gradient boost decision tree, *2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, IEEE, pp. 1228–1232.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning, *Journal of big data* **6**(1): 1–48.

Sunney, J., Jilani, M., Pathak, P. and Stynes, P. (2023). A real-time machine learning framework for smart home-based yoga teaching system, pp. 107–114.

Taware, G., Kharat, R., Dhende, P., Jondhalekar, P. and Agrawal, R. (2022). Ai-based workout assistant and fitness guide, *2022 6th International Conference On Computing, Communication, Control And Automation (ICCUBEA*, IEEE, pp. 1–4.

Unkule, P., Shinde, C., Saurkar, P., Agarkar, S. and Verma, U. (2022). Cnn based approach for sign recognition in the indian sign language, pp. 92–97.

Zhang, S., Chen, W., Chen, C. and Liu, Y. (2022). Human deep squat detection method based on mediapipe combined with yolov5 network, pp. 6404–6409.