Human Pose Detection: A Machine Learning Based Approach with Keypoints Extraction and Ensemble Methods

Harshitha Polsani
Computing and Data Science
Sam Houston State University
Huntsville,USA
hxp029@shsu.edu

Venkata Ramya Vardineni
Computing and Data Science
Sam Houston State University
Huntsville, USA
Orcid: 0009-0002-2550-4929

Anusha Yarrasani
Computing and Data Science
Sam Houston State University
Huntsville, USA
Orcid: 0009-0001-3356-6743

***Abstract*—Human pose detection is an important task in computer vision with various applications such as fitness tracking, sports analysis, and rehabilitation. In this paper, we present a machine learning based approach for human pose detection. We use two datasets, a primary data set of yoga poses images and a Yoga-82 data set, to create a comprehensive data set of six target classes. We employ the pretrained Mediapipe model for feature extraction and also employed under sampling and oversampling techniques to make the data balanced.**

We use Support Vector Machine (SVM), Random Forest Classifier(RF), K-Nearest Neighbors (KNN) Classifier, Bagging Classifier, Gradient Boosting Classifier (GBC) , and employed ensemble on various classifiers for classification task. Our approach achieves an accuracy of 92% using Ensemble methods and Support Vector Machine, demonstrating promising results for real-world applications of human pose detection.

Keywords—Pose detection, Mediapipe, Sampling, Machine Learning

I. INTRODUCTION

Human pose detection has gained significant attention in recent years due to its potential applications in various fields such as healthcare, sports, and entertainment. Human pose detection is a computer vision task that involves identifying the location and orientation of a person's body parts in an image or video. The ability to accurately detect human poses has significant potential applications in healthcare, sports, and entertainment industries. For instance, in healthcare, human pose detection can be used for diagnosis and treatment of various musculoskeletal disorders. In sports, it can provide valuable insights into athletes' movements and help trainers in designing personalized training plans. In entertainment, it can be used for virtual reality and animation.

In this paper, we present our work on human pose detection that aims to detect six different poses - reclining, sitting, standing, inverted, balancing, and wheel poses. We used two different datasets, the primary data set being the Yoga Image Classification data set from Kaggle, which has 107 directories of poses, and the other is Yoga 82 data set. Our goal was to create a comprehensive data set that includes as many different poses as possible. After merging and preprocessing the data, we ended up with a data set consisting of 6219 instances of six target classes. However, the data was imbalanced, and we used sampling techniques to make it balanced. We then split the dataset into train, validation, and test sets and applied various machine learning algorithms, including Support Vector Machine, Random Forest Classifier, K-Nearest Neighbors (KNN) Classifier, Bagging Classifier, Gradient Boosting Classifier, and employed ensemble on various classifiers.

In the following sections, we describe our methodology and the results of our experiments in detail. The paper is organized into four sections. The second section presents a literature survey on the related work done in the field of human pose detection. The third section discusses the data collection process and how we merged and pre processed the data. The fourth section explains our methodology, including the machine learning algorithms used, the hyper parameters tuning, and the ensemble techniques employed. Finally, we present the experimental results and conclude the paper with some future directions.

II. RELATED WORK

Marko Linna et al. [3] demonstrated a Convolutional Neural Network- based approach for real-time multi human pose estimation from video. This method utilizes a person detector to locate individuals in an image before performing pose estimation on each individual separately. This approach can be seen as an end-to-end system, as it incorporates all necessary steps for pose estimation from any source image. However, this approach is that it relies heavily on the performance of the person detector used. If the detector fails to accurately locate individuals in the image, the subsequent pose estimation step will also be inaccurate. Additionally, the method may struggle with images containing crowded or overlapping individuals. Despite these limitations, the approach proposed by Linna et al. represents a significant advancement in the field of human pose estimation and has the potential to be useful in a wide range of applications.

Vivek Anand Thoutam et. al. [4] proposed a system for estimating yoga poses and generating feedback using deep learning. The system uses a dataset of yoga pose images and corresponding labels to train a Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) for pose estimation. The CNN predicts the joint locations for keypoints on the human body, which are then used to estimate the yoga pose. The system also generates feedback on the user's pose based on a set of predefined rules. The methodology is evaluated on a publicly available dataset consisting of six yoga poses, including Cobra, Tree, Mountain, Lotus, Triangle, and Corpse. The dataset contains a total of 70 videos, and the authors use 320 instances for training and 30 instances for validation. For testing, the authors create a separate dataset consisting of 30 instances, with 5 instances for each pose. However, the proposed pose

estimation algorithm extracts only 18 body keypoints, where each keypoint consists of x and y coordinates of body points. The study utilized a small dataset for evaluation, which may not fully capture the diversity of yoga poses and their variations.

Bhosale et al. [5] present a methodology for yoga pose detection and correction using Posenet and K Nearest Neighbors (KNN). They collected a dataset consisting of nearly 400 distinct images of each yoga pose, which were optimized to similar pixels and size and then blended into a single video. The proposed model was trained using the video, where KNN classifier was used to detect the pose by detecting the key-points of the limbs of human. The authors used webcam as input feed to get real-time yoga pose of the user, and KNN classifier identified significant points and classified them into one of the yoga poses. The dataset used for this study was collected from Kaggle and other open sources and consisted of images of 5 different yoga poses. A total number of 1578 images were collected for the study. A potential drawback of this research is the limited scope of the dataset utilized for assessment, which may not adequately reflect the diverse range of yoga poses and their variations. Furthermore, while the authors employed Posenet for detecting the key points, our model uses mediapipe for this purpose.

III. Data Collection and Pre-processing

In this section, we describe the process of collecting and pre processing the data used in our study.

A. Data Collection:

To create a comprehensive dataset for human pose detection, we used two different datasets: the Yoga Image Classification dataset from Kaggle, and the Yoga-82 dataset [1]. The Yoga Image Classification dataset, obtained from Kaggle, consisted of 107 directories of poses, while the Yoga-82 dataset was used to extract images of six poses not present in the Yoga Image Classification dataset through web scraping. The six target poses, namely Standing, Sitting, Reclining, Inverted, Balancing, and Wheel Poses, were selected for the study. Images of all other poses were grouped into their respective superclass categories following the reference paper by Verma et al. [1].

The presented figure 1 illustrates the organization of data, specifically the images of poses, under their respective super classes.

B. Pre-Processing and Key Feature Extraction

The images from both datasets were preprocessed to remove any invalid file formats, and then we used the pre-trained Mediapipe pose detection model [2] to extract the key features for each pose. This model detects 33 keypoints on a person's body and provides their respective (x,y) coordinates, their depth (z), and a measure of their visibility. The (x,y) coordinates correspond to the pixel location of each keypoint in the image. The depth (z) coordinate represents the distance of the keypoint from the camera. The visibility measure indicates whether a keypoint is visible in the image or not, and is represented as a value between 0 and 1, where 0 indicates that the landmark is not visible and 1 indicates that it is clear.

These keypoints represent specific points on a person's body, such as the head, shoulders, elbows, wrists, hips, knees, and ankles. By using these keypoints, we were able to capture important spatial relationships between different parts of the body and identify specific human poses based on these relationships. The landmark data for each pose was stored in its respective CSV file.

Once we had extracted the key features for each pose, we assigned a label to each pose and merged all the individual datasets into one final dataset containing 6219 instances. This final dataset contained the keypoint features, which included 33 keypoints, each with x, y, z, and visibility coordinates. As a result, there were 132 features in the dataset, and the dataset contained six target classes.

However, the dataset was imbalanced, with some classes having significantly more instances than others. The Standing pose had 1579 instances, Sitting had 1561 instances, Reclining had 1394 instances, Inverted had 386 instances, Balancing had 611 instances and Wheel Pose had 688 instances. To address this imbalance, we used undersampling of majority classes (Standing, Sitting and Reclining) and oversampling using SMOTE (Synthetic Minority Over-sampling Technique) for minority classes (Inverted, Balancing and Wheel). As a result, each class had 1036 instances, making the dataset balanced.

The data was split into train, test, and validation sets in a ratio of 60:20:20, respectively. The training set consisted of 3729 instances, the validation set had 1243 instances, and the test set had 1244 instances. Machine learning algorithms, such as Support Vector Classifier, Random Forest Classifier, KNN, Bagging Classifier, and Gradient Boosting Classifier, were applied to the data. Ensembles of these classifiers were also employed, and they outperformed the individual models used.

Overall, the data collection and preprocessing process helped us to create a comprehensive and balanced dataset, which was used for our human pose detection study.

C. Mediapipe Pose Detection Model

Mediapipe is an open-source framework developed by Google, designed for building machine learning pipelines to process multimedia data such as images and videos. One of the key features of the Mediapipe framework is the Mediapipe Pose Detection Model, which is a pre-trained deep learning model capable of detecting human poses from images and videos. The model uses a complex set of algorithms to accurately identify the landmarks and keypoints of human body parts, such as the head, shoulders, elbows, hips, and knees. The keypoints are detected by analyzing the relative positions of pixels in the image, and the model can output these keypoints in a variety of formats, including JSON and CSV files. With its advanced pose detection capabilities, the Mediapipe Pose Detection Model has a wide range of applications, including virtual try-on, fitness tracking, and motion capture.

In our study, we used the Mediapipe Pose Detection Model to extract key points and landmarks from images.

Specifically, we used the mppose library from Mediapipe to detect the poses in the images. This was done by processing the images and detecting the landmarks in each pose. We then drew the landmarks on a white background using the mpDraw library, and extracted the coordinates and visibility of each landmark. This information was then stored in a dataset in the form of a CSV file for further analysis.

Using the Mediapipe library provided us with an efficient and accurate way to detect and extract the pose information from images, which was essential for our study of classifying different human poses. Overall, the data collection and preprocessing process helped us to create a comprehensive and balanced dataset, which was used for our human pose detection study.

The figure 1 demonstrates the conceptual outline of Proposed Methodology.

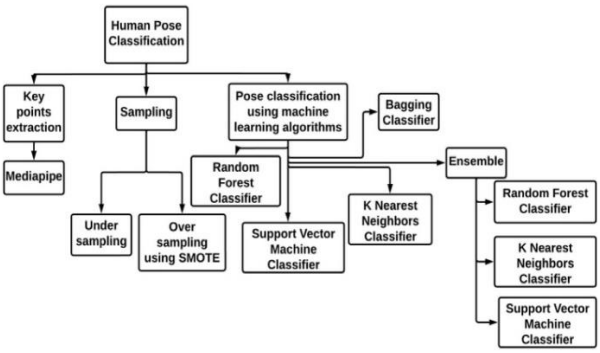

Figure 1. Conceptual outline of topics

IV. METHODOLOGY

In this paper, we present a comparative study of different classification algorithms, namely K-Nearest Neighbor (KNN) Classifier, Random Forest Classifier, Support Vector Machine (SVM) Classifier, Bagging Classifier, and Gradient Boosting Classifier, and Ensemble method for combining the predictions of these algorithms. The aim of this study is to classify poses using the classification algorithms and evaluate the performance of these algorithms on a given dataset and determine which algorithm(s) perform best for this dataset.

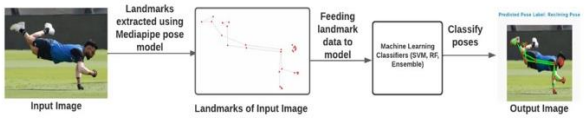

Figure 2. Overview of proposed methodology

A. Support Vector Machine:

In our study, we used Support Vector Machine (SVM) for pose classification. SVM is a classification algorithm that uses a hyperplane to separate the data into different classes. We implemented the SVM algorithm with a Radial Basis Function (RBF) kernel and used grid search to tune the hyperparameters. The hyperparameters we tuned were the regularization parameter C, the kernel type, and the decision function shape. Our goal was to achieve high accuracy in pose classification, and we evaluated the performance of our model on the test set. Our results showed that the SVM algorithm with tuned hyperparameters achieved an accuracy of 91.15% in pose classification.

Our study investigated the effectiveness of SVM with RBF kernel in pose classification. One important aspect of SVM is its regularization parameter, C. The regularization parameter determines the trade-off between the complexity of the model and its ability to fit the training data. A higher value of C allows the model to fit the training data more closely, while a lower value of C encourages the model to have a simpler decision boundary that may not fit the training data as closely.

In our study, we used grid search to tune the hyperparameters of the SVM model, including the regularization parameter C. We searched for the optimal value of C that maximizes the accuracy of the model on the validation set. Our grid search revealed that the optimal value of C was 25.

We then trained the SVM model on the training set using the optimal value of C and evaluated its performance on the test set and validation set. Our results showed that the model achieved an accuracy of 91.15% on both the validation and test set, indicating that the model is able to generalize well to unseen data.

To further evaluate the performance of our model, we performed k-fold cross-validation. Our cross-validation results showed that the model achieved a high precision score of 92.32%, recall score of 92.25%, and F1 score of 92.21%. These results indicate that our model is performing well in terms of both precision and recall, and it is able to achieve a good balance between the two.

In summary, our study demonstrates the effectiveness of SVM with RBF kernel in pose classification. We have shown that tuning the regularization parameter C using grid search can lead to a well-performing model that is able to generalize well to unseen data. Our model achieved high accuracy and good balance between precision and recall, indicating its potential usefulness in practical applications.

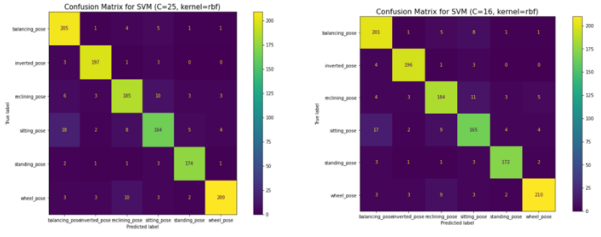

Figure 3. Displays the confusion matrix for the SVM model trained with a radial basis function (RBF) kernel and decision function shape of "one-vs-one" (OvO) with various values of the regularization parameter C. The matrix visualizes the model's performance in classifying the different poses. The rows represent the true labels of the poses, while the columns represent the predicted labels. The diagonal elements indicate the number of correctly classified poses for each class. The off-diagonal elements represent the misclassified poses.

B. Random Forest Classifier:

Random Forest is an ensemble learning method that combines multiple decision trees to create a more accurate and robust model. It works by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Each tree is constructed using a random subset of the features and a random subset of the data. This randomness helps to prevent overfitting and increases the generalizability of the model.

To make a prediction, the Random Forest takes the input features and passes them through each of the decision trees in the forest. Each tree independently produces a prediction, and the class with the most votes or the average prediction is returned as the final prediction.

In our study, We experimented with different values of n_estimators, which represents the number of decision trees in the forest, and evaluated the model's accuracy on the validation and test sets for classifying poses. Our results show that increasing the number of estimators from 1000 to 1500 improved the validation accuracy from 0.9107 to 0.9115. However, further increasing the number of estimators to 2000 did not result in significant improvement in the validation accuracy. The test accuracy of the model with 2000 estimators was 0.8939.

In addition, we also performed 10-Fold Cross-Validation on the Random Forest model with 1000 estimators. The mean precision score was 0.9182, with a mean recall score of 0.9172 and a mean F1 score of 0.9171. The overall cross-validation score had a mean of 0.9172.

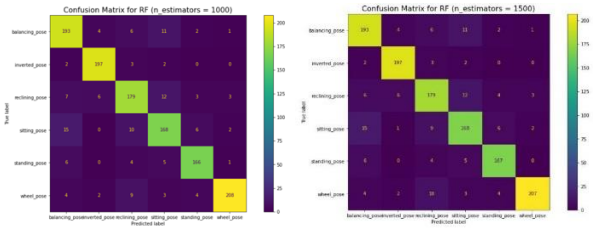

Figure 4. Displays the confusion matrix of the Random Forest model with 1000 and 1500 estimators, which were evaluated on the test set. The confusion matrix provides a visual representation of the performance of the model, showing the number of true positives, true negatives, false positives, and false negatives for each class.

C. K Nearest Neighbors Classifier :

The K-Nearest Neighbors (KNN) algorithm is a non-parametric machine learning method that can be used for classification tasks. In our study, we used KNN to classify human poses based on their extracted keypoint features. KNN works by finding the k-nearest neighbors to a data point and using the most common class among the neighbors as the predicted class for the data point. We preprocessed the data and split it into training and testing sets. We then trained the KNN model using the training set with the number of neighbors set to 3. For each data point in the test set, the model finds the 3 nearest neighbors and predicts the class based on the most common class among those neighbors.

We evaluated the performance of the KNN model by computing its accuracy, precision, recall, and F1-score. We also plotted the confusion matrix to visualize the model's performance. Additionally, we varied the value of k and evaluated the model's performance based on its accuracy in predicting the correct class of poses on the test set.

Our results show that the KNN algorithm is simple to implement and can be effective in classification tasks, especially when the number of features is small. However, it may not perform well when the number of features is large, and it can be sensitive to outliers in the data. We obtained an accuracy of 88.96% with a precision of 89.28%, a recall of 88.97%, and an F1-score of 88.87% using the KNN model with n=5. These results were obtained through 10-fold cross-validation, which helps to reduce the bias and variance of the model.

Overall, our study demonstrates the usefulness of the KNN algorithm in classifying human poses based on keypoint features. However, further studies are needed to evaluate the performance of the algorithm on larger datasets with more complex features.

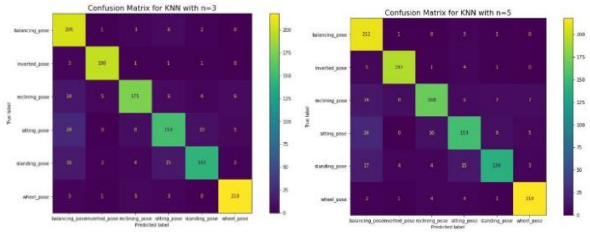

Figure 5. Confusion matrix for the KNN algorithm with n=3 and n=5. The matrix illustrates the correct and incorrect predictions made by the algorithm for each pose. The rows represent the true labels, while the columns represent the predicted labels. The numbers in the diagonal represent the number of correctly classified poses, while the off-diagonal elements represent the misclassified poses.

D. Bagging Classifier:

Bagging classifier is a popular ensemble method that can be used to improve the performance of a classification algorithm. In our study, we used Bagging Classifier as an ensemble learning method to classify poses. The Bagging Classifier creates multiple bootstrap samples from the original training dataset and trains a base classifier on each sample. The method combines the predictions of the base classifiers to improve the performance of the classification algorithm. Bagging can help to reduce overfitting and improve the generalization of the model.

We implemented the Bagging Classifier with a base estimator of Random Forest Classifier, which used 100 trees. We trained the Bagging Classifier on a dataset of pose images and evaluated its performance using a test dataset. The Bagging Classifier achieved a classification accuracy of 0.9075, demonstrating its ability to improve the accuracy of pose classification.

To further evaluate the performance of the Bagging Classifier, we experimented with different hyperparameters. We varied the number of estimators in the Bagging Classifier and the number of trees in the Random Forest Classifier. For instance, we evaluated the Bagging Classifier with 10 estimators and a base estimator of Random Forest Classifier with 1000 trees. The accuracy score obtained was 0.8867, which was lower than the accuracy score obtained with 100 trees.

We also used Support Vector Machine (SVM) with a radial basis function (RBF) kernel as the base estimator for the Bagging Classifier. We experimented with different regularization parameter (C) values for the SVM. For example, we evaluated the Bagging Classifier with a base estimator of SVM with C=25, C=27, and C=19. The Bagging Classifier achieved the highest accuracy score of 0.9108 with a base estimator of SVM with C=27.

We also performed 10-fold cross-validation on our Bagging Classifier with a base estimator of SVM with a radial basis function kernel and a regularization parameter (C) of 27. The mean accuracy score obtained was 0.9197, indicating that the Bagging Classifier was able to accurately classify the poses.

We also evaluated the precision, recall, and F1 scores for the Bagging Classifier using cross-validation. The mean precision score was 0.9224, the mean recall score was 0.9224, and the mean F1 score was 0.9212. These scores demonstrate the effectiveness of the Bagging Classifier in accurately identifying the poses.

In summary, our results show that the Bagging Classifier with a base estimator of SVM with C=27 is an effective method for classifying poses, with good accuracy, precision, recall, and F1 scores. The 10-fold cross-validation further validates the performance of our model, indicating that it can generalize well to new data.

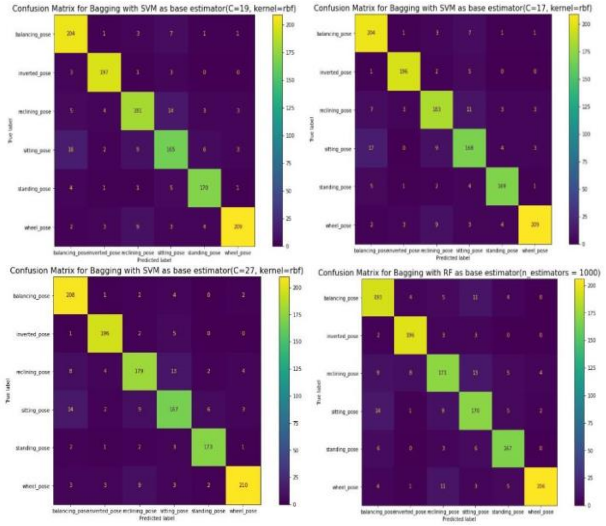

Figure 6. The figure shows the confusion matrix of the Bagging Classifier with a Support Vector Machine (SVM) as the base estimator for C=17, C=27, C=19, and a Random Forest Classifier with n_estimators = 1000.

E. Gradient Boosting Classifier:

Gradient Boosting Classifier (GBC) is an ensemble learning algorithm that combines the predictions of multiple weak classifiers to form a strong classifier. It is a tree-based model that builds an ensemble of decision trees sequentially, where each tree corrects the errors made by the previous tree. In other words, the algorithm trains a series of decision trees on the residuals (i.e., the differences between the actual and predicted values) of the previous tree, with the goal of minimizing the overall error.

The GBC algorithm has several hyperparameters that can be tuned to optimize its performance. The number of estimators is the number of trees that the algorithm builds, and the learning rate determines the contribution of each tree to the final prediction. The maximum depth of the trees controls their complexity, and the subsample parameter specifies the fraction of samples to use for each tree. Other hyperparameters include the loss function, which measures the difference between the predicted and actual values, and the criterion for splitting nodes in the trees.

In this study, we used the Gradient Boosting Classifier (GBC) to classify different poses based on the key features extracted from the images. We set the hyperparameters of the GBC model to their default values except for the random state parameter, which we set to 2 to ensure reproducibility of the results. The default hyperparameters include 100 estimators, a learning rate of 0.1, a maximum depth of 3, and a subsample of 1.0, among others. However, these hyperparameters can be tuned to optimize the performance of the model for specific tasks and datasets.

F. Ensemble Model:

Ensemble learning is a technique used in machine learning that combines the predictions of multiple models to improve the accuracy and robustness of the predictions. In our study, we explored two different ensemble methods using three and five classifiers, respectively. Specifically, we used the Random Forest Classifier, Support Vector Machine Classifier, and K-Nearest Neighbors Classifier to build our ensembles.

The first ensemble method was implemented using a custom function, which takes the predictions from the three classifiers as input and returns the final prediction based on a voting scheme. The function first checks if all three classifiers predict the same class. If yes, the function returns the predicted class. If two of the classifiers predict the same class, the function returns the predicted class. If none of the above conditions are met, the function returns the class predicted by the support vector machine. This method achieved an accuracy of 92%, which is higher than the accuracy achieved by the individual classifiers.

The second ensemble method was implemented using a custom function, which takes the predictions from the five classifiers as input and returns the final prediction based on a similar voting scheme. Specifically, for each sample, the function counts the number of votes for each class from the

five classifiers and returns the class with the highest number of votes as prediction. This method achieved an accuracy of 91%, demonstrating the effectiveness of ensemble learning in improving the performance of machine learning models.

To evaluate the performance of our ensembles, we performed 10-fold Cross-Validation on the dataset using the first ensemble method. The mean accuracy of the model was 0.8938 with a standard deviation of 0.0121. Additionally, we reported the precision, recall, and F1 scores of the model, with mean scores of 0.9294, 0.9278, and 0.9275, respectively. These ensemble results indicate the better performance compared to individual classifiers

This study provides insights into the performance of several classification algorithms for pose classification and demonstrates the effectiveness of ensemble methods in improving the accuracy and reliability of the classification results.

G. Multi Layer Perceptron:

Multi-Layer Perceptrons (MLPs) are a type of feedforward neural network, which consists of multiple layers of neurons, with each layer connected to the previous layer. In the MLP model, the input layer receives the input features, and the output layer produces the predictions. The hidden layers perform non-linear transformations on the input features, allowing the model to learn more complex patterns in the data.

In our experiment, we used an MLP model with three layers. The first hidden layer consisted of 64 neurons with a ReLU activation function, which is a popular choice because it is computationally efficient and helps to prevent the vanishing gradient problem. The second hidden layer consisted of 32 neurons with a ReLU activation function. The output layer consisted of six neurons, corresponding to the six target classes, with a softmax activation function, which is commonly used for multi-class classification tasks because it produces a probability distribution over the classes.

The categorical cross-entropy loss function used in the model measures the difference between the predicted and true class probabilities. The Adam optimizer is an adaptive learning rate optimization algorithm that uses a combination of momentum and adaptive learning rates to optimize the model parameters.

We preprocessed the data by encoding the target variable using the LabelEncoder and one-hot encoding it. We trained the model on the training set using a batch size of 32 and for 500 epochs.

The model achieved a training accuracy of 0.997 and a validation accuracy of 0.899. The test accuracy of the model was 0.882. Although the MLP model achieved a test accuracy of 0.882, indicating good generalization to new data, it is important to note that there may be other models that perform even better on this task. It is always recommended to explore different models and compare their performances to determine the best approach for a given dataset and problem.

H. Convolutional Neural Network:

Convolutional Neural Networks (CNNs) are a type of neural network commonly used in image classification tasks. They have been proven to be highly effective in recognizing patterns in images and are widely used in computer vision applications. In this project, we have used CNNs to classify pose using the landmark data extracted from images. Instead of using raw image data, the model is trained on 132 landmark features extracted from images. Each landmark feature consists of the x, y, and z coordinates of a keypoint, as well as a visibility value that indicates whether the keypoint is visible in the image or not.

To train the model on this data, the input data consists of 11x12 matrices with a single channel, where each pixel represents a particular landmark feature. The target variable is one of six possible poses, encoded using one-hot encoding.

The CNN architecture used in this project consists of two convolutional layers, each followed by max-pooling layers, a flatten layer, and two fully connected layers. The first convolutional layer has 32 filters with a kernel size of 3x3 and uses the ReLU activation function. The first max-pooling layer has a pool size of 2x2. The second convolutional layer has 64 filters with a kernel size of 3x3 and also uses the ReLU activation function. The second max-pooling layer has the same pool size as the first. The flatten layer converts the output of the second max-pooling layer into a one-dimensional vector. The first fully connected layer has 128 neurons with the ReLU activation function, and the second fully connected layer has six neurons with the softmax activation function.

The model is trained using the Adam optimizer with a categorical cross-entropy loss function and an accuracy metric. The model is trained for 500 epochs with a batch size of 32. During training, the input data is randomly augmented to improve the generalization performance of the model. This is achieved by applying random transformations such as rotation, scaling, and flipping to the input data. The model achieves a train accuracy of 0.997 and a validation accuracy of 0.894. The test accuracy is 0.880.

In this project, we have shown that CNNs can be effectively used to classify pose based on data extracted from images. By using landmark data instead of raw images, we have reduced the complexity of the input data and made the training process more efficient. The trained model achieves high accuracy on both the training and validation sets and generalizes well to new data. While there may be other models that perform well on this task, our results show the potential of CNNs to be used in a wide range of computer vision applications beyond traditional image classification tasks.

I. Convolutional Neural Network(AlexNet):

Convolutional Neural Networks (CNNs) have proven to be highly effective in various image classification tasks. In this study, we implemented the AlexNet architecture. The AlexNet model code is a deep neural network architecture that has shown impressive performance in image classification tasks. In this implementation, the model is trained on direct images instead of landmark data for classifying poses.

To handle the class imbalance, present in our dataset, we defined class weights based on the frequency of each class. We then used these class weights during training to give more importance to the underrepresented classes.

We preprocessed the images using various data augmentation techniques, such as rotation, shifting, zooming, flipping, and changing brightness levels. This helped to increase the diversity of the training data and make the model more robust to variations in the input images.

The proposed model is based on AlexNet architecture which consists of 3 convolutional layers, 2 fully connected layers, and an output layer. The input images are of size 227x227x3.

The first convolutional layer has 96 filters of size 11x11 with a stride of 4 and uses the ReLU activation function. The output of this layer is fed to a max pooling layer of size 3x3 with a stride of 2. A zero-padding layer of size 1x1 is added before the next layer.

The second convolutional layer has 256 filters of size 5x5 with a stride of 1 and uses the ReLU activation function. The output of this layer is fed to a max pooling layer of size 3x3 with a stride of 2. A zero-padding layer of size 1x1 is added before the next layer.

The third convolutional layer has 384 filters of size 3x3 with a stride of 1 and uses the ReLU activation function. A zero-padding layer of size 1x1 is added before passing the output to a fully connected layer.

The output of the third convolutional layer is flattened and passed to the first fully connected layer with 4096 neurons and a ReLU activation function. A dropout layer with a rate of 0.5 is added to prevent overfitting. The output of the first fully connected layer is then passed to the second fully connected layer with 4096 neurons and a ReLU activation function. Another dropout layer with a rate of 0.5 is added before the final output layer with the number of units equal to the number of classes in the dataset. The softmax activation function is used in the output layer.

The model is trained using the Adam optimizer with a categorical cross-entropy loss function and accuracy as the evaluation metric. The training is performed for 150 epochs with a batch size of 32. The class weights are used to balance the imbalanced classes in the dataset. The training and validation data are generated using data augmentation techniques. The model achieves a validation accuracy of 0.548 and test accuracy is 0.692.

Based on the evaluation of our proposed model, we have achieved satisfactory results in image classification tasks. While the accuracy may not be the highest compared to other state-of-the-art models, our model shows potential for further improvements and can be fine-tuned for specific applications.

V. Real-time Implementation of Pose Detection System

In this paper, we present the real-time implementation of our pose detection system using Mediapipe and Support Vector Machines (SVM) algorithm. Our method is capable of detecting six different poses namely: Balancing Pose, Inverted Pose, Reclining Pose, Sitting Pose, Standing Pose, and Wheel Pose.

Mediapipe pose library is used to process the video frames and detect the pose landmarks. The pose landmarks are then processed to extract the features required for classification. The feature vector is fed into the SVM model trained and tested on the pose landmark data , which predicts the target pose class.

This implementation is able to handle real-time video processing with minimal latency. The SVM model enables the system to classify poses accurately and in real-time. It can also handle multiple people in the frame, although it will only classify the pose and detect the keypoints of a single person closest to the camera.

The real-time implementation of pose detection using Mediapipe and SVM classification provides an efficient and accurate solution for detecting human poses in real-time. The implementation has the potential to be utilized in various applications, such as fitness tracking, physiotherapy, and rehabilitation. The system's accuracy and efficiency make it a viable option for these applications, as it can detect poses accurately and in real-time.

VI. Experimental Results

To evaluate the performance of our human pose detection system, we employed various performance metrics including accuracy, Precision, Recall, and F1 score. Furthermore, we conducted 10-fold cross-validation to ensure the reliability and robustness of our results.

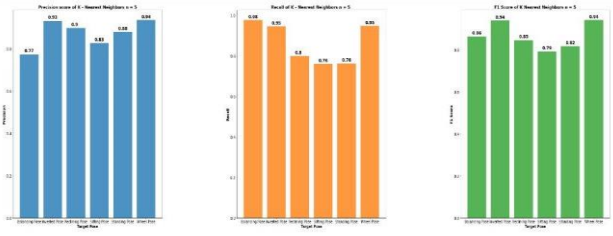

Figure 7. The figure displays the precision, recall, and F1 scores for each individual target pose using KNN for 5 nearest neighbors.

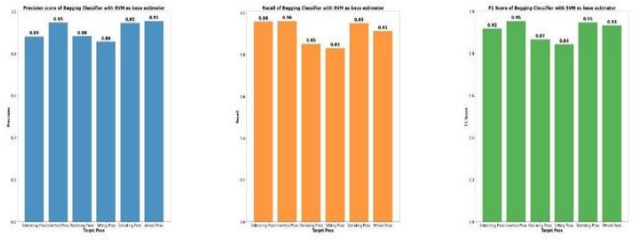

Figure 8. The figure displays the precision, recall, and F1 scores for each individual target pose using Bagging Classifier with base estimator as SVM for C = 27.

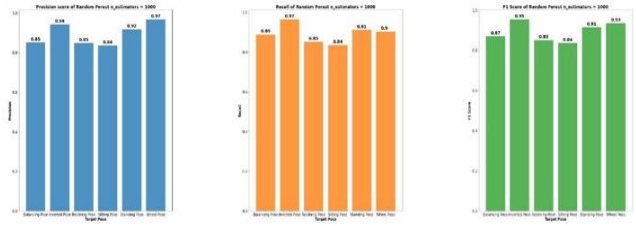

Figure 9. The figure displays the precision, recall, and F1 scores for each individual target pose using Random Forest for 1000 trees.

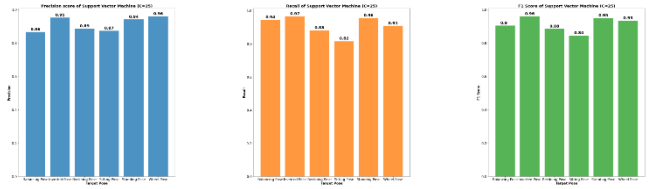

Figure 10. The figure displays the precision, recall, and F1 scores for each individual target pose using SVM for C =25 with OvO.

As this is a multi-class classification problem, we evaluated the performance of our models separately for each pose, and the x-axis of the figures 7,8,9 represents the different target poses while the y-axis represents the precision, recall, and F1 scores respectively.

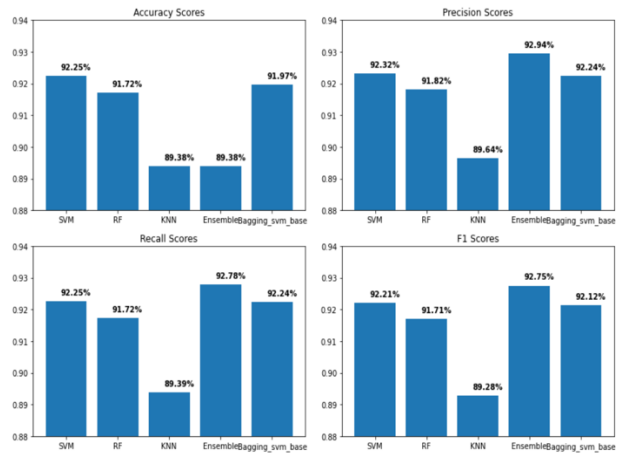

Figure 10. Figure shows the performance evaluation of each model for our human pose detection after performing 10-fold cross-validation.

In order to visually demonstrate the effectiveness of our study about human pose detection, we have included a set of output images that show the detected pose names and their corresponding keypoints. These images in Figure 12. serve as further evidence of the accuracy and robustness of our system in identifying human poses in various situations.

After evaluating the performance of our human pose detection models using various performance metrics and conducting 10-fold cross-validation, we conclude that our proposed models have shown high accuracy, precision, recall, and F1 score across a diverse range of poses and scenarios. The inclusion of output images further supports the robustness and accuracy of our study, highlighting its potential for various applications in human pose analysis and action recognition. Our experimental results demonstrate the effectiveness and reliability of our proposed human pose detection implementation, which can be utilized for real-world applications such as sports training, healthcare, and human-computer interaction.

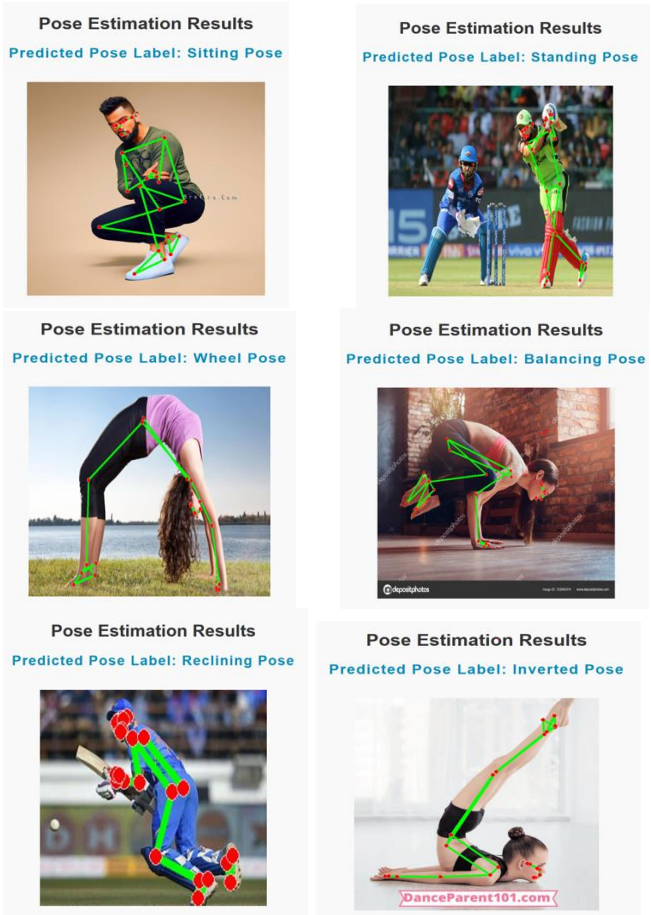

Figure 12. Figure shows the experimental results of the Human Pose Detection.

VII. Conclusion

Our proposed human pose detection models have demonstrated high performance and robustness in detecting different target poses under various conditions. We utilized the Mediapipe library for landmark extraction and trained various machine learning models, including Support Vector Machines, Random Forest, Bagging, Gradient Boosting, and K-Nearest Neighbors, on the extracted landmarks to classify human poses.

However, we acknowledge that the accuracy of our models can be affected by factors such as image quality, lighting conditions, and occlusion. In particular, we have observed limitations in the detection of pose landmarks in blurry images. Nonetheless, our results have shown the potential of our models.

Moving forward, we plan to test and apply our models in Various scenarios to evaluate them. Additionally, we aim to explore the possibility of integrating our models with other computer vision technologies to enhance their performance and expand their range of applications. We also intend to investigate ways to improve the accuracy of our models in detecting poses under challenging conditions such as low-light environments and occlusion.

Overall, our work highlights the potential of using Mediapipe for landmark extraction and ensemble learning based approaches in human pose detection and analysis. We believe that further research and development in this area will lead to significant advancements in the field of computer vision and human-machine interaction. Our results suggest that various machine learning techniques, when combined with Mediapipe, can yield promising results in detecting different target poses.

VIII. FUTURE SCOPE

There are several avenues for future research in this field. One direction is to improve the robustness of our models to challenging image conditions, such as low lighting, occlusion, and blur. This can be achieved through the integration of more advanced image processing techniques. Another direction is to extend the scope of our models to include more complex and dynamic poses, such as those observed in sports and dance activities. Furthermore, the integration of deep learning techniques can potentially enhance the accuracy and speed of our models.

To further extend the scope of our research, we plan to explore the potential of our models for multi-person pose detection. This will involve addressing the challenges of detecting and tracking multiple individuals simultaneously, which requires robustness to occlusion and varying pose scales. Additionally, we believe that our models can be applied in a range of industries, such as healthcare and robotics, for applications such as physical therapy and human-robot interaction. Overall, our work highlights the potential of machine learning and computer vision in advancing the field of human pose detection.

IX. REFERENCES

[1] “Yoga-82: A New Dataset for Fine-grained Classification of Human Poses, ” Manisha Verma, Sudhakar Kumawat, Yuta Nakashima, and Shanmuganathan Raman, in ARXIV Workshops, 2020.

[2] “Yoga Pose Detection and Correction using Posenet and KNN,” Varsha Bhosale, Pranjal Nandeshwar, Anhishishek Bale, and Janmesh Sankhe, in IRJET, 2022.

[3] “Yoga Pose Classification Using Deep Learning,” Robert Chun, Katerina Potika, and Susmit Gaikwad, in IEEE, 2020.

[4] “Yoga Pose Estimation and feedback Generation Using Deep Learning,” Vivek Anand Thoutam, Anugrah Srivastava, Tapas Badal, Vipul Kumar Mishra, G. R. Sinha, Aditi Sakalle, Harshit Bhardwaj, and Manish Raj. in IEEE, 2022.

[5] “Real-time Human Pose Estimation fro Video with Convolutional Neural Networks,” Marko Linna, Juho Kannala, and Esa Rahtu. in ARXIV, 2016.

[6] “Depth – Based 3D Hand Pose Estimation: From Current Achievements to Future Goals,” Shanxin Yuan, Guillermo Garcia – Hernando, Tae – Kyun Kim, Bjorn Stenger, Iason Oikononidis, Antonis Argyros, Gyeongsik Moon, Ju Yong Chang, Kyoung Mu Lee, Pavlo Molchanov, Jan Kautz, Sina Honari, Liuhao Ge, Junsong Yuan, Xinghao Chen, Guijin Wang, Fan Yang, Kai Akiyama, Yang Wu, Qingfu Wang, Meysam Madadi, Sergio Escalera, Shile Li, Dongheui Lee, Iason Oikonomidis. in ARXIV, 2018.

[7] “BlazePose: On – device Real-time Body Pose tracking,” Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, Matthias Grundmann. in ARXIV, 2020.

[8] “Yoga Pose Detection Using Deep Learning Techniques,” S. Sankara Narayanan, Devendra Kumar Misra, Kartik Arora, Harsh Rai. in ICICC, 2021.

[9] “Lifting from the Deep: Convolutional 3D Pose Estimation from a Single Image,” Denis Tome, Chris Russell, Lourdes Agapito. in ARXIV, 2017.

[10] “OpenPose: realtime Multi – Person 2D Pose Estimation using Part Affinity Fields,” Zhe Cao, Gines Hidalgo, Tomas Simon, Shih – En Wei, Yaser Sheikh. in ARXIV, 2019.

[11] “Convolutional Pose Machines,” Shih – En Wei, Varun Ramakrishna, Takeo Kanade, Yaser Sheikh. in ARXIV, 2016.

[12] “Simple Baselines for Human Pose Estimation and Tracking,” Bin Xiao, Haiping Wu, Yichen Wei. in ARXIV, 2018.

[13] “DensePose: Dense Human Pose Estimation in the Wild,” Riza Alp Guler, Natalia neveravo, Iasonas Kokkinos. in ARXIV, 2018.

[14] “The Progress of Human Pose Estimation : A Survey and Taxonomy of Models Applied in 2D Human Pose Estimation,” Tewodros Legesse Munea, Yalew Zelalem Jembre, Halefom tekle Weldegebriel, Longbiao Chen, Chenxi Huang, Chenhui Yang. In IEEE, 2020.

[15] “Human Pose Estimation and its Application to Action Recognition: A Survey,”Liangchen Song, Gang Yu, Junsong Yuan, Zicheng Liu. in JVCIR, 2021.