A MAJOR PROJECT II REPORT

ON

WORKOUT POSE RECOGNITION

Submitted in Partial Fulfillment of the requirements

for the award of the Degree of

MASTER OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

By

AMAN REHMAN

2K22/CSE/02

Under the Supervision of

PROF. SHAILENDER KUMAR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daulatpur, Main Bawana Road, Delhi-110042. India

MAY, 2024

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

CANDIDATE'S DECLARATION

I, Aman Rehman, Roll No. 2K22/CSE/02, student of M.Tech Computer Science and Engineering, hereby declare that the Major Project - II titled "WORKOUT POSE RECOGNITION" which is submitted to the Department of Computer Science and Engineering, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of degree of Master of Technology, carried out during the period from 2022 to 2024, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi **Aman Rehman**

Date: 20/05/2024

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

CERTIFICATE

I hereby certify that the Major Project - II titled "WORKOUT POSE RECOGNITION" which is submitted by Aman Rehman, Roll No. 2K22/CSE/02, to the Department of Computer Science and Engineering, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of the degree of Master of Technology, carried out during the period from 2022 to 2024, is a record of the project work carried out by the student under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi

Prof. Shailender Kumar

Date: 20/05/2024

SUPERVISOR

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

ACKNOWLEDGEMENT

I wish to express my sincerest gratitude to Prof. Shailender Kumar for his continuous guidance and mentorship that he provided during the project. He showed the path to achieve the targets associated with this project by explaining all the tasks to be done and the importance of this project as well as its industrial relevance. He was always ready to help and clear my doubts regarding any hurdles in this project. Without his constant support and motivation, this project would not have been successful.

Place: Delhi Aman Rehman

Date: 20/05/2024

Abstract

Physical exercise is a key part of a healthy lifestyle, encompassing a wide array of activities from dance to weightlifting and sports. Proper form and posture are critical to the ensure the safety of these exercises and making them effective. Accurate assessment of workout poses can provide invaluable feedback to individuals and fitness professionals alike, enabling adjustments for optimal performance. Performing exercises correctly also reduces the risk of injury, especially for people new to exercises. Additionally, fitness professionals can utilize this to remotely guide clients in virtual training sessions.

The emergence and evolution of deep learning techniques has revolutionized computer vision tasks, offering excellent performance in various tasks. In the context of workout pose estimation, this can be used for automating the process of assessing body positions during exercises. Multiple machine and deep learning techniques have been used in this domain with excellent results. Although a comparative study has not been performed, and the datasets chosen have seen extreme variety. Our approach involves leveraging advanced architectures, machine learning methods, and ensemble learning methods like Random Forest, XGBoost, to develop an efficient and accurate system capable of precise pose estimation across a diverse range of exercises with quick inference.

Further objectives of the research include implementing and comparing the performance of the above mentioned models with fine-tuning for workout pose estimation, evaluating their performance to understand the strengths and weaknesses of each architecture, collect and present the information gained, and suggest models that give high performance and quick inference. We also aim to shed light of some of the less utilized methods to explore the above problem.

Our approach also captures the difference in using plain deep learning networks, as opposed to keypoint based machine learning and ensemble methods.

List of Publications

1. Aman Rehman, Shailender Kumar, "A Comparison of Transfer Learning Inspired CNN Architectures for 2D Image Workout Recognition", presented at **IEEE 9th International Conference for Convergence in Technology (I2CT) 2024**, Pune, Maharashtra, India.

2. Aman Rehman, Shailender Kumar, "Leveraging MediaPipe and YOLO Keypoint Detection in Ensemble Approaches for Workout Pose Recognition", accepted at **IEEE International Conference on Advancement in Computation & Computer Technologies (InCACCT) 2024**, Chandigarh, Punjab, India.

Contents

List of Tables

List of Figures

List of Symbols, Abbreviations

CNN	Convolutional Neural Networks
SVM	Support Vector Machine
ROC	Receiver Operating Characteristic
AUC	Area Under Curve
GBM	Gradient Boosting Machine
MLP	Multi Layer Perceptron
$YOLO$	You Only Look Once
NB	Naive Bayes
XG	eXtreme Gradient
KNN	K Nearest Neighbors
$LSTM$	Long Short Term Memory
PAF	Part Affinity Fields
RGB	Red Green Blue
$STDCNN$	Semi Transfer Deep Convolutional Neural Network
VGG	Visual Geometry Group

Chapter 1

INTRODUCTION

Pose identification means determining the way a person positions their body when performing any activity. That position is then associated with that of a particular exercise, given the context learned from the image or video or real-time camera. Accurate identification of exercises is necessary for a variety of use cases, for example, supporting virtual personal trainers in guiding users through exercises with individualized recommendations, allows fitness tracking programs [1] to monitor workout routines and provide real-time feedback on form, and help with rehabilitation monitoring [2]. It also improves gesture detection in fitness games for immersive experiences, makes it easier to analyse athletes' actions for sports performance analysis [3], and can also connect with health monitoring devices to track physical activity levels and offer personalized recommendations.

1.1 Problem statement

The accurate assessment of workout poses is essential for ensuring the safety and effectiveness of physical exercises. Despite the advances in deep learning techniques for computer vision tasks, there have been 2 particular shortcomings identified. One, there has not been a comprehensive comparative research on the effectiveness of various machine learning models for workout pose estimation. Second, the lack of ensemble learning approaches in this task is surprising. This research aims to leverage advanced architectures and ensemble learning methods to develop an quick and effective system for precise pose identification/classification across diverse exercises, with the goal of understanding and comparing the performance of different models.

1.2 Benefits of exercise

One of the most crucial things for the human health is exercising. It is also necessary for all aspects of life. It is even more important than ever, especially for today's youth because the junk food they eat on a daily basis may negatively impact their quality of life.

It is hard to function well when we are ill. If we stretch it over a longer period, then exercise is one of those long term treatments to avoid many potential helath issues later. Exercise is therefore necessary to overcome all of these issues. Also, it is something that everyone should do, not just the young generation. For instance, a desk job necessitates lengthy hours without breaks in a single sitting position. This makes for an extremely unhealthy lifestyle. They sit all day and then return home to sleep, so we can see their

physical activity is minimal. As a result, exercise is crucial to living a healthy lifestyle.

But it is also important to perform exercises correctly. Adjustments are usually necessary for any workout, especially if you are new to exercising, whether it is a squatting, hinging, or other type of action. There are many reasons for using the right technique. Although it may seem that all of the minor tweaks are unnecessary, but performing incorrect movements over a long term can have lasting impact.

When exercises are performed correctly, the weight load is applied to the correct muscles and joints we want to target. Performing the same exercises improperly, we would fail to load the joints and muscles completely thereby reducing the effectiveness. Moreover, it is possible that one is overstretching other muscles and this can put a significant amount of strain on the body, increasing the risk of injury. If we execute an exercise improperly for a long time, we are instilling negative habits in our body that might result in injuries as well as subpar training results, particularly in the early stages.

Anything less than perfect execution increases the likelihood of accidents or injury. This is particularly true for lifting weights. It is important to get the exercises right as early as possible so that we minimize the risk of injury. Exercises like deadlifts and other heavy weighted exercises exert a fair amount of force on the body, so we need to always perform them with proper form. People who maintain perfect form also find that workouts are easier. This is because it is highly possible that poor technique makes the individual perform more work than necessary. This is only going to make the exercise more difficult in addition to giving subpar results. It is known fact that people exercise to improve their bodies, in terms of strength, flexibility and mobility. Also, it is the desire of every individual that the results are fast and visible. There are many factors that go into this. One of these factors include performing exercises optimally with the correct posture. So, it is also important to realize the benefit and need of such systems. This is also because many gyms do not have adequately trained or qualified trainers, and people are often slightly apprehensive about going to the gym because of insecurities or body-image issues. These solutions are not meant to discourage anyone from going to a gym, but act as a motivator to begin or continue exercising. It is also our belief that such systems can be integrated with professionals and in professional setting to augment the training regime.

1.3 Technological intervention in workouts

Many exercise image recognition implementations use deep learning solutions, like CNNs, that is, Convolutional Neural Networks which are well-known for their effectiveness in classification and keypoint identification tasks [4]. The efficiency of these hinges significantly on the variety of the dataset. Ensuring that the model can represent the subtleties present in the given setting requires a strong dataset that covers a broad range of exercises, and environemental characteristics. Further, many applications of exercise pose recognition may require fast inference and analysis, for example real-time analysis [5]. In such cases, the processing capacity of the underlying hardware is also an important consideration. This is because some people may prefer working out themselves with the help of a mobile application. Or they may be unable to hire a personal trainer. As a result, it is essential to find stability within accuracy of the model and computing resource limits because the application should be efficient for the given context.

Architecturally, implementing popular CNN architectures, specifically VGG16, VGG19, ResNet50, DenseNet121, DenseNet201, MobileNetV2, InceptionV3, Xception and EfficientNetB0 are used in the form of transfer learning to extract important features from images. All of these models and possibly more, have been used extensively for classification. Each one comes with its own pros and cons, but in general all of these are quite popular and have performed extremely well on large datasets. The same models mentioned above have also been used in the area of action recognition or classification. Next, the models are adjusted to the subtleties of pose identification by training, hyperparameter tuning, and fine-tuning specific layers or the network as a whole. The introduction of all these techniques, along with engineering the models to cater to specific contexts of use allows for great performance and reduced overfitting. It is also possible that the optimal models are chosen to be deployed for large-scale use.

Deploying the best model and continuously checking its performance allows possible improvement in accuracy and reliability in real-world applications. Because people may use the model in a variety of scenarios, it is important for the model to generalize and be able to perform optimally in all the use cases and environments. Deep learning techniques have not been the only methods used for this task. Machine learning extensively has been used, like Decision Tree, Linear SVM, Multi-Layer Perceptron, KNN and GaussianNB have also been used previously in the domain of action recognition, as will be detailed later in the literature review. Further, it is also clear that the domain of exercise pose recognition has a lot of variety in it. The variety comes from the type of exercises that are being performed, the location where they are being performed, the people performing those exercises. There is also variation in the lighting condition and angles in which the image has been taken. Based on this knowledge, we observed that the exploration of ensemble learning methods was slightly limited.

Chapter 2

LITERATURE REVIEW

A thorough literature review was conducted into the techniques that have been previously used for the problem of workout pose detection and similar use cases. The use cases include physical activity recognition or monitoring, activity recognition, sports image classification, etc. All the solutions require identifying the pose being performed in the image and hence caters to our use as well. Since there are a lot of techniques that previous research has used, a categorization of the techniques used has been done. Under each technique, the relevant papers have been mentioned. This is done to show the variety of techniques that have been used in this domain, and also to easily discern which paper has used which technique. It can also help to identify if certain techniques are more popular than others. We have attempted to review recently published papers only. However, we acknowledge that it is obvious that many more techniques may have been used for the same problem. But, since those techniques have not been used in the recent window of 1-4 years within our publication, we have not considered those in our review.

2.1 CNN & Transfer Learning

Research on the identification of actions in still photos has regularly been conducted. Every study that pertains to the following makes use of still pictures. A residual neural network is used in Sreela et al. [6]'s image action recognition model to extract features efficiently. Succeeding in 15 Pascal VOC action courses, the model showed sensitivity to contextual changes and real-world imagery. By utilizing a residual neural network and a support vector machine classifier, the enhanced model achieved 66.1% accuracy in identifying human activities in a two-layer classification model. The research [7] uses ResNet as a feature extractor to identify human actions in still photos with accuracy of 61.07%. Activity recognition is augmented by approximating postures, and classification is handled by SVM. But these systems suffer in nuanced positions, truncations, and occlusions in complex situations.

The two-stream convolutional neural network described in study [8] improves action recognition by using long-term fusion pooling to capture the temporal complexity of actions. Their method, called STDDCN, combines a knowledge distillation module with dense connectivity and multiscale information. Though it is successful in comprehending actions, it suffers from difficulties with computational load and large memory needs because of frame processing, even though it achieves great performance with fewer parameters. Their model yields an accuracy of 92.1%.

Singh et al. [9] put forth a novel approach to position estimation and identification which did not require wearable technology. To reliably categorize different postures, two

pre-trained models' performances were compared with those of a bespoke CNN called DeepPose, which was trained using both image and keypoint datasets. The outcome demonstrated that the suggested model performed better than the image dataset when using the keypoint dataset.

Pose identification and correction using machine learning methods has been extensively researched. It involves either entering exercises manually before performing them or automatically identifying and classifying user motion into an activity. There are studies that have used either images, or videos or both. In paper [10], the authors evaluated the motion quality of the workout performed by a user against the known correct motion. The authors used the HPTE dataset consisting of eight exercises performed by 5 different people from different backgrounds and angles. The proposed approach used PoseNet model in the backend which used two-dimensional data that has confidence linked to the body's important points and produces a heatmap for stance assessment. The system reached an overall accuracy of 80%. The user received feedback upon the completion of the task.

Jose et al. [11] achieved 85% accuracy using the VGG-16 model to recognize 10 distinct yoga poses in still photos. Podgorelec et al. [12] presented a CNN architecture based on the VGG-19 model with differential evolution method for classifying images. The CNN model's hyperparameters are optimized using the DE (differential evolution) approach after the training data has been fine-tuned using the TL method prior to classification. With 4 sports categories in their unique dataset, they achieved 81.31% accuracy. There is room for improvement in accuracy. Farhad et al. [13] used a pre-trained VGG-16 model to classify 18 sports in a custom sports picture dataset. With this approach, they successfully classified the images in the dataset with 93% accuracy.

The study [14] used a neural network for sports picture classification across 6 categories using a pre-trained InceptionV3 for feature extraction. In sports picture categorization, their method yielded 96.64% accuracy rate. Joshi et al. [14] attempted to classify six different sports categories with respect to athlete's body movement. Inception V3 was used for derivation of feature and then neural networks for pose categorization. On comparison with other machine learning classifiers, it validated the framework's efficacy. The suggested method successfully detected and classified a variety of sports data, evident by its remarkable 96.64% average accuracy.

Bhat et al. [15] used a technique for detecting joints for which adjustment was needed while the exercise was being performed. For this, joint angles' distribution of dataset's exercises was found. Then detecting poses in which some joint angles witness aberration. The researchers created skeleton layouts using a variety of datasets based upon Yoga, Kungfu and Pilates. With the Kungfu-7 dataset, the DenseNet+Random Forest classifier delivered the best result, 86% validation accuracy.

Bhat et al. [15] detected joints that needed adjustment during exercise. This was done by finding the distribution in joint angles of exercises in the dataset. Then detecting poses in which some joint angles witness aberration. The researchers created skeleton layouts using a variety of datasets based upon Yoga, Kungfu and Pilates. With the Kungfu-7 dataset, the DenseNet+Random Forest classifier delivered the best result, 86% validation accuracy.

2.2 Sensor-Based Methods

Das et al. [16] developed a system that counts repetitions, analyzes comfort using heart rate data, and recognizes activities performed at home and at the gym. For indoor exercise recognition, they employed k-NN, SVM, and decision trees. They achieved remarkable accuracy of 99.4% for counting repetitions and 95.3% for activity detection. This technique is effective for usage in the gym or at home since it assists in selecting the proper weights for workouts. Decision trees were used as the classifier in this case,

Zhou et al. [17] used AdaBoost, a confidence-based algorithm. They used inertial pressure sensors to detect motion during multiple sports activities. More emphasis was put on leg exercises. A total of six participants participated in twenty-four leg workouts. The accuracy achieved was 81.7%. The utilization of sensor-based technologies in this technique shows potential for precise monitoring and analysis of leg exercises. The authors stressed the possibility of accurately measuring exercise time spent on routine tasks. They used SVM and Random Forest (RF) to reach an accuracy of about 96.2%.

[18] made use of data gathered from special wearable wristband sensors, demonstrating the potential to identify workout periods. Nunavath et al. [19] classified multiple physical activities using sensor data, especially using accelerometer data. They applied sensor-based methods on the UCI-HAR dataset, such as DNNs, that is Deep Neural Networks as well as RNNs, that is Recurrent Neural Networks. Based on the sensor data that was gathered and then analyzed, it was found that this achieved a 84.89% accuracy in the classification of these activities.

Vimala et al. [19] offered an artificial intelligence-based solution for identifying patients' physical activity types. First, an RNN was used, that is a recurrent neural network. Secondly, a DNN was used, which is deep feed-forward neural network. There were also two datasets that were used. The first one consisted of 14 daily life activities which were captured from wrist-worn sensors. The second dataset comprised of 10 activities captured using hip-worn sensors. The data gathered from volunteers wearing these sensors were then used for model evaluation. The RNN model performed better with 84.89% accuracy. The F1-score was 82.56%. This suggested the model's capacity to accurately understand as well as track the physical activities of patients.

Capecci et al. [20] focuses on creating an interactive monitoring tool for physical therapy that is performed from the comfort of users' home. A stage for processing videos and evaluating workout performance are included in the software platform. A device called the Kinect v2 sensor is used to extract information from the image. It is elaborated to get an exercise score. Additionally, the instrument gives physiotherapists a numerical assessment of their patients' workout performances. 5 people and 5 distinct exercises were used in the testing of the suggested tool.

In another paper, Capecci et al. [21], in a rehabilitation setting, the Kinect v2 sensor's accuracy evaluation is examined. In terms of joint locations/angles during dynamic postures in therapy for low-back pain, an accuracy analysis is done. While some research has examined the validity of joint angles and positions, the results of these studies only take into account static postures. In contrast, rehabilitation exercise monitoring takes into account dynamic movements involving a broad range of motion as well as joint tracking. Joint locations and angles in this work are clinical characteristics to assess the subject's motions. The accuracy of both space and time is examined in comparison to the state-of-the-art or gold standard, which is a stereophotogrammetric system.

2.3 Keypoints

Using MediaPipe, Pham et al. [22] were able to extract skeletal sequences from RGB footage that was taken using a commercial camera. The DD-Net deep architecture is utilized for action recognition in the assessment section. The model's average accuracy is 98.33%. In an another approach, Yubin et al. (2021) [23] proposed a Mediapipe based computer vision-based technique for rating yoga poses utilizing two datasets: Dataset A, which they took from Kaggle and Dataset B, which the researchers created. Using coarse as well as fine contrastive samples, the method produced the greatest accuracy for Dataset A, 0.8321.

Rangari et al. [24]focused on individual exercise pose recognition using 2D coordinates for body keypoints recorded by an RGB camera. 18 major body joints' coordinates were identified and used as the model's attributes for exercise classification. Human volunteers of various ages and heights were taken to create a benchmark dataset. When tested on that dataset, the model outperformed previous methods with an impressive accuracy of 97.01%, demonstrating the efficacy of their proposed strategy. Faisal et al. [15] used a deep learning-based model that can recognize five distinct poses with significantly less data. The variability in the dataset is compensated by improving the diversity in the images. They evaluated performance of their model against many image classification models, including ResNet.

Pham et al. [25] develops a comprehensive framework that identifies physical exercises and assesses thier quality as well, as it is being performed by users. This framework uses a custom dataset which targets 9 common exercises, including arm circles, squats, jumping jacks. The dataset encompasses data from nine subjects. The study explores the efficacy of two distinct networks for HAR: the compact DD-Net and the high-performance graph CNN, FF-AAGCN. For real-time classification, a sliding window method is integrated with these networks. Results on the dataset show that this framework is able to achieve high accuracy in exercise recognition, with DD-Net achieving accuracies and F1-scores of 99.24% and 99.23%, respectively, while FF-AAGCN attains 98.48% and 98.32%.

2.4 Hybrid CNN Architecture

A CNN-based technique for predicting human exercise postures was proposed by Haque et al. [26] used a dataset with five distinct workout courses to achieve this. Their model properly identified these exercise postures with an accuracy of 82.68%. A self-prepared dataset as well as a publicly available one were used to demonstrate the findings of paper [27] for WorkoutNet, a deep learning network for exercise action identification. It surpassed models like as XceptionNet, VGG-19, and DenseNet121, with a validation accuracy of 92.75%. It was verified with F1-measure and confusion matrix to properly classify exercises into 10 groups.

In [28], a boxing experiment is used to make a dataset. Supervised classification is used to categorize the limb and, for each strike, the method employed. Also, the performer's level of skill was classified. A boxing bag was instrcuted to be struck by both novice and expert kickboxers from various distances. This data was captured, resulting in a dataset of about 4000 strike trajectories. Then multi-class linear support vector and K-nearest neighbors were used to classify these. With 99% accuracy, both systems could identify the limb employed in the strike. Further, with 86% accuracy, both systems were able to classify the employed methods. By classifying each limb separately using hierarchical

classification, the accuracy was further increased. What was remarkable is that to obtain the above mentioned prediction accuracy, just 10% of the dataset was needed for training. KNN was able to successfully classify strikes by skill level with 73.3% acuuracy.

2.5 Machine learning techniques

Whelan et al. [29] used IMUs that were positioned on the shank, thigh, and lumbar spine. Global and customized classification methods were contrasted in the study. The results were surprising, with accuracy (AC) at 64%, sensitivity (SE) at 70%, and specificity (SP) at 28% for binary classification and 59%, 24%, and 84% for multi-class classification, respectively, the results showed that global classifiers performed badly. Individualized classifiers performed noticeably better even with just one IMU. In binary classification, they obtained 81% AC, 81% SE, and 84% SP; in multi-class classification, they obtained 69%, 70%, and 89%, respectively. These results draw attention to the shortcomings of global classification techniques and the improved precision of customized methods for squat technique monitoring and evaluation.

In, [30] 3D skeletal data of a patient is recorded. A recurrent neural network is used in combination with long-short-term memory is used, to gauge time-frequency properties in the skeletal domain. This is done using approximation coefficients of the Discrete Wavelet Transform (DWT) to identify the exercise performed. Every exercise instance is then divided into segments. To investigate muscle and/or joint progress, the best performing of these are utilized as a comparison image, and different versions of the activity are compared with the reference. Lastly, indicators are assessed at the conclusion of every engagement session to do joint performance analysis.

Sarwat et al. [31] demonstrates a cheap technique that attempts to diagnose strokes. It is a major cause of death and hard to diagnose as well. This system uses a machine learning algorithm coupled with common tools that are open-source only. Using the inertial measurement unit of smartphones which is built-in along with gradient boosting open-sourced algorithms, it was possible to diagnose a patient by coming up with a score on 3 tasks in the Fugl-Meyer upper-extremity assessment. An accuracy of 95.56% was obtained using 5-fold cross-validation evaluation.

Soekarjo et al. [28], using a dataset collected from a kickboxing experiment, supervised classification is used to categorize the limb and, for each strike, the method employed. Also, the performer's level of skill was classified. A boxing bag was instrcuted to be struck by both novice and expert kickboxers from various distances. This data was captured, resulting in a dataset of about 4000 strike trajectories. Then multi-class linear support vector and K-nearest neighbors were used to classify these. With 99% accuracy, both systems could identify the limb employed in the strike. Further, with 86% accuracy, both systems were able to classify the employed methods. By classifying each limb separately using hierarchical classification, the accuracy was further increased. What was remarkable is that to obtain the above mentioned prediction accuracy, just 10% of the dataset was needed for training. KNN successfully was able to classify strikes by skill level with 73.3% acuuracy.

Kianifar et al. [32] aimed at giving doctors a numerical representation of SLS performance. An automated squat quality assessment method based on inertial measurement units (IMUs) was suggested. The joint angles, velocities, and accelerations of the squatting leg were estimated using a set of three IMUs. Support vector machines were used to train classifiers based on the most informative features. The findings demonstrated

that internal rotation at the hip and ankle, as well as knee flexion at the, are distinguishing characteristics. Moreover, training multiple classifiers separated by gender results in better classification. The results revealed high accuracy of 95.7% for "poor" or "good" squatting posture quality and 94.6% for distinguishing between a high and low risk of injury.

2.6 Gaps in existing research work

A fair number of papers were reviewed and as shown above, it is evident that the problem of workout pose recognition, and even action recognition in general is a challenging task. A variety of ML and DL models have been on multiple datasets. Based on the extensive literature review, some research gaps were identified, which are as follows

- The previous research generally takes into consideration a small set of exercises, taking into account about five to ten popular poses. So, the number of categories included is less.
- Moreover, the chosen poses in many of the research papers are significantly different from one another in terms of body position. The issue with this is that any model used may find it easy to find discriminating features in the images and/or the keypoints. This will consequently result in high accuracy model. But these may not generalize to a large extent or when exercises with similar range of motion are introduced.
- It is also possible that people perform exercises in different environments, and that there are different people present in the dataset performing different exercises. The environment, angle and lighting condition should also be taken into account, which some papers miss out on in their dataset.
- It was also observed that there was limited cross-cultural and demographic variability in the datasets used, which is something we attempt to address in our approach.
- More diversified datasets are required in order to train and evaluate the systems.
- Most of the papers in this area have used supervised learning approaches. The use of unsupervised methods may also help this domain. This is outside the scope of our current study.
- There is limited discussion into the application of these systems and the way in which they will be implemented. For instance, we focus on the context wherein mobile applications may be used for aid and they may have limited compute power.
- Also, due to the inherent variety of exercises and similarity in movements, exercise pose classification may benefit from ensemble learning approaches, which has been underutilized in this area. Ensemble learning models train several smaller models on different groupings of the data. This may help to capture the high level of variety present in the dataset.

Chapter 3

METHODOLOGY

The study's methodology used a two-step approach to the problem of identifying exercise poses in images. Taking into account the complexity of the task, a thorough and rigorous methodology was created, utilizing the advantages of both deep learning and keypoint-based methodologies. Through the exploration of deep learning architectures, and ensemble-based techniques, the research aimed to create a resilient framework that could precisely identify exercise positions in various environments. The next paragraphs go into slightly more detail about the process, emphasizing the analysis done and its reasoning.

The dataset used was carefully constructed to include a wide variety of exercise poses taken in different settings with different lighting, angles, people, poses, etc, guaranteeing a thorough portrayal of actual exercise scenarios.

In this first stage, deep learning architectures specifically for image recognition were applied to the image data. Models were thoroughly trained and fine-tuned in order to achieve effective pose detection. Evaluation was performed on multiple metrics.

In the second phase, keypoints detection algorithms, namely, YOLOv8 and MediaPipe were used for the purpose of classification of workout poses. Modern object detection system YOLOv8 was first implemented with a range of machine learning and ensemble-based techniques. The same was then repeated using MediaPipe keypoints. And again a thorough analysis and comparison was performed.

The complex nature of workout pose recognition required an extensive analysis to improve applications in the given context. Given the variability in performing exercises, the dynamic motion, and environmental factors, a robust solution was needed. By exploring multiple techniques and evaluating their performance rigorously, the study aimed to develop a highly accurate and reliable system capable of accurately identifying workout poses from images, using lightweight techniques that give fast inference as well.

3.1 Tools used

Python version 3.10 was used to conduct the study. Hardware requirements include an Nvidia RTX 3080 GPU, a 4-core Intel i7 processor, 32GB DDR4 RAM, and the use of Google Colab's GPU for faster processing.

3.1.1 Python 3.10

Python is extensively popular among and used for ML and DL programs. It has also been extensively developed and used for many large-scale projects. Being an high-level

interpreted language, it is executed by an interpreter on the go. This means it processes the program each line at a time. It has extensive support of libraries and regular updates, great for beginners, and a programmer can focus on what to do, and less on how to do that.

3.1.2 Tensorflow

Developed by Google, TensorFlow enables researchers and developers to work on ML and DL tasks. It was released in 2015, and since then has seen constant updates and improvements, making it a popular choice for deep learning workloads. Its support for a variety of models also makes it a popular choice. What is even better is that it is open source. It comes under the Apache Open-Source license. We can use, contribute and redistribute it for free. This is also heavily supported by the developer community. TensorFlow architecture involves pre-processing, model build, train and evaluate.

3.1.3 Pandas

Pandas is a fundamental tool for data preparation and analysis in machine learning. It enables data cleaning, transforming, and analyzing. Pandas is also heavily supported and feature rich to perform a variety of tasks on our data. For instance, as mentioned before, we can clean, transform, and analyses the data whichever way we like. It supports multiple data formats as well, like CSV, Excel, etc. Pandas extracts data to make a Dataframe. On this dataframe then users can calculate statistics, like correlation, average, distribution of data. One can modify data by removing missing values or imputing them, filtering data based on some conditions, visualize data using multiple libraries, and also export the transformed data in multiple formats.

3.1.4 Scikit-Learn

Scikit-learn is an extensively used machine learning open-source library written for Python. It provides many supervised and unsupervised learning algorithms. It focuses on robustness and support requirements in production systems, ensuring ease of use, quality coding, performance, and comprehensive documentation. Built on NumPy, SciPy, and matplotlib, scikit-learn is known for its simplicity and efficiency for data analysis and modeling, making it accessible for both beginners and those experienced in machine learning.

3.1.5 Google Colab

Google Colab is a cloud-based platform that allows users to execute Python in a Jupyter notebook environment for free. We can run our workloads using computational hardware like GPUs and TPUs, which are also provided by Colab itself. Some of the hardware is behind a paywall. There are mutiple GPUs available, depending on the type of length of the workload to be executed. This ease of availability of compute power and storage makes Google Colab a popular an ideal choice for machine learning tasks as well as data analysis. Colab offers other features as well including integration with Google Drive, and the option to share and collaborate on notebooks with others.

3.2 Proposed Approach: Phase 1

The processes used to tackle the given problem of workout pose recognition consists of multiple phases. To put it another way, it includes gathering data, preparing it, choosing a model, training and honing it, assessing and then possibly deploying it.

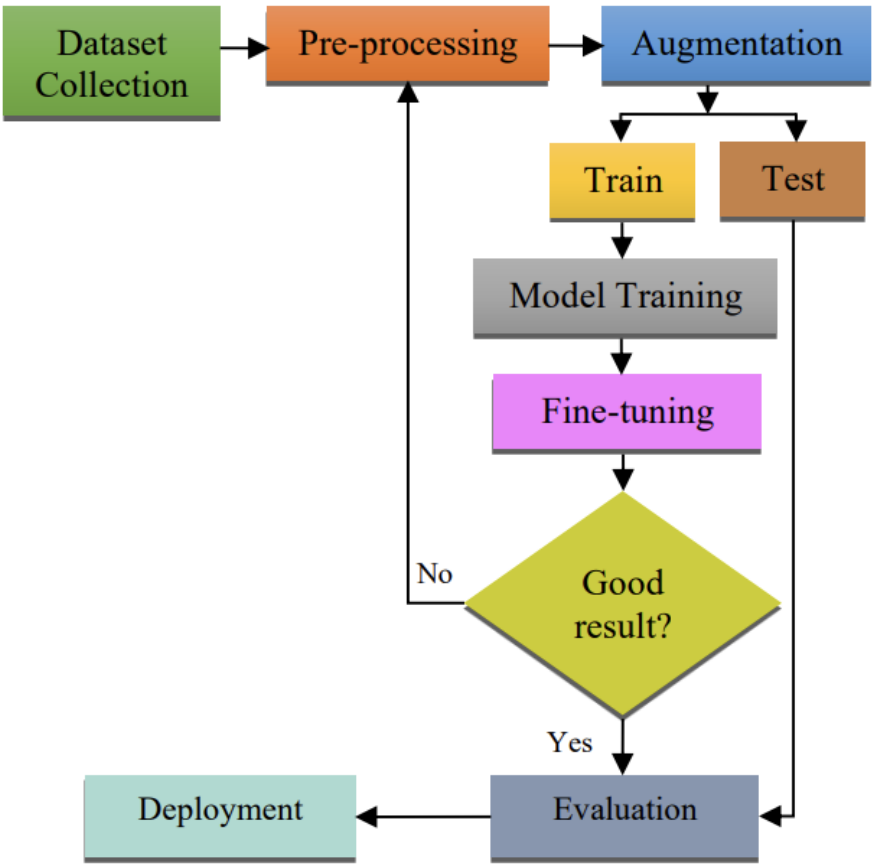


Figure 3.1: Workflow

3.2.1 Dataset Collection

We have taken various exercise categories each containing 2D pose images which were pulled manually from Google search engine using Kaggle as a reference. Thus, our dataset contains 1058 photos in total, divided into 20 classes, each with 55–60 images. As described above, the dataset attempts to capture a wide variety of exercises and different people performing them in various settings. The 20 exercise categories are as follows: Benchpress, bow, bridges, child's pose, cobra, crane pose, crunches, cycling, deadlift, donkey kicks, down dog, flutter kicks, lat-ulldowns, leg-extensions, plank, pullup, shoulder press, squat, standing toe reach and tricep-pushdown. The exercises are chosen to incorporate a variety of bodyweight and weight training exercises as well as stretches.

Figure 3.2: Sample images from dataset

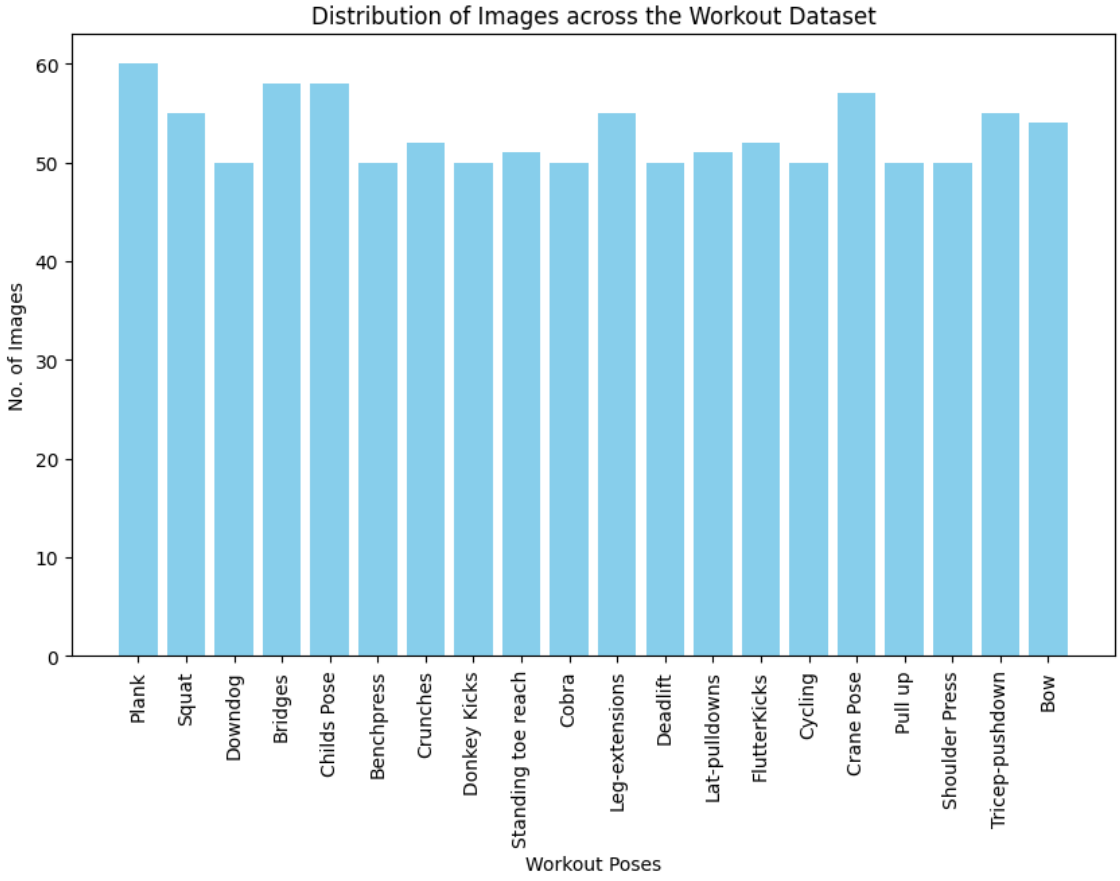


Figure 3.3: Sample images from dataset

3.2.2 Data Processing and Augmentation

Images are loaded using Tensorflow and a standard size of 224x224 pixels is maintained. To avoid class-imbalanced splits, each image—labeled according to the exercise in the directory structure—is shuffled. The dataset split is 80:20 where 80% is for training, and the remaining to test, after normalizing to [-1, 1]. Before an image is fed into the models, it undergoes a predetermined set of augmentations.

For pose recognition tasks, image data augmentation can be essential to boosting the resilience and generalization of deep learning models. Augmentation simulates real-world scenarios by introducing variations in the existing dataset, which increases the diversity of the training dataset. The model is better able to learn features and fluctuations due to this. In our approach, augmentations including contrast and brightness fluctuation, random rotation, zooming, and flipping of the horizontal and vertical axes, were applied to increase variability and facilitate generalization.

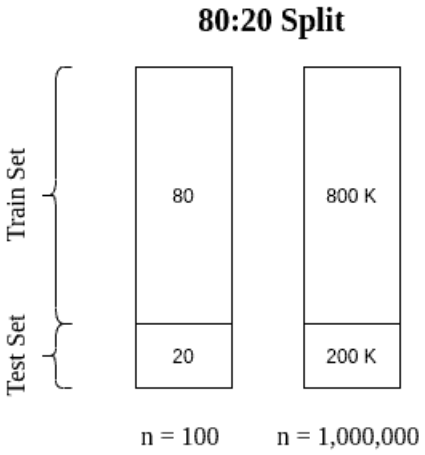


Figure 3.4: Train-test-split

3.2.3 Model Architecture and Training

Multiple deep learning CNN architectures have been used to conduct a thorough study. ImageNet pre-trained models were downloaded from Tensorflow. The advantage is that we can use the information already learnt by these models. To reduce spatial dimensions and focus attention on significant features, Global Average Pooling (GAP) layers were introduced. Also dropout layer is added to avoid overfitting. Later, dense layers are added with softmax activation function. The models are then trained and evaluated using accuracy metric. Categorical cross entropy loss function was used along with Adam optimizer. As mentioned before, multiple deep learning architectures were used in this phase. The models used are as follows:

3.2.3.1 VGG16

Convolutional neural networks with the architecture VGG-16 are renowned for being simple and efficient in image recognition. The network may learn complicated features by stacking the convolutional layers, which use tiny 3x3 filters with a stride of 1. VGG-16 has demonstrated exceptional accuracy in classifying objects in images. It is still a widely used deep learning benchmark and option for transfer learning, despite its complexity and processing demands.

For VGG16, the 16 layers are divided into 3 fully connected and 13 convolutional layers. It is often recommended for its simplicity and uniform structure. It uses small 3x3 convolutional filters throughout the network. For our task, we had a total of 14,724,948 parameters and all of them were kept trainable. VGG19 is a larger variant of VGG16, with 3 additional convolutional layers. There are total of 20,034,644 parameters and all of them were kept trainable.

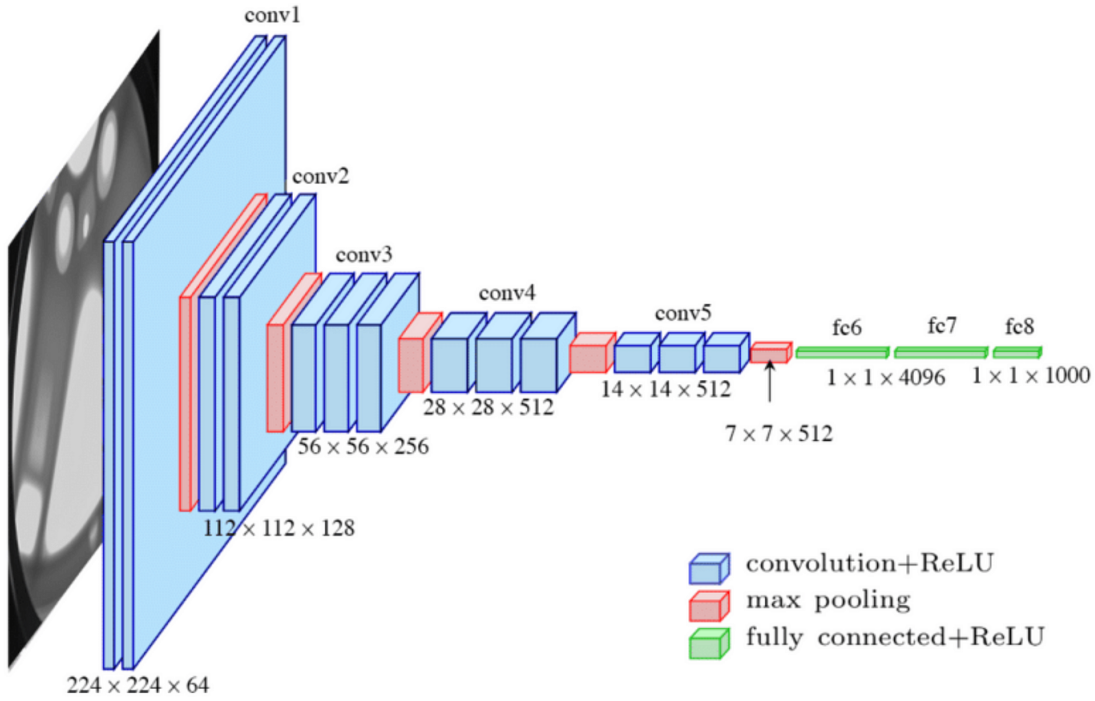


Figure 3.5: VGG16 architecture

3.2.3.2 ResNet50

ResNet50 is a deep CNN that has gained recognition for the introduction of the innovative and useful residual connections. The advantage of these connections is that they enable easier training of quite deep networks with increased effectiveness. It is a 50-layer architecture, has residual blocks with skip connections to avoid some convolutional layers, hence addressing the vanishing gradient issue. This architecture has achieved the best overall results on problems globally like ImageNet task, and has had a major impact on a variety of computer vision problems. The architecture is a fundamental component of many deep learning frameworks, demonstrating the ability of residual learning for training.

ResNet-50 introduces residual learning with 50 layers, including GAP, fully connected, and 48 convolutional layers. Residual blocks use a bottleneck design, consisting of a series of 1x1, 3x3, and 1x1 convolutions which preserves representational strength and decreases computational complexity. There are a total of 23,628,692 parameters, out of which 23,575,572 were kept trainable.

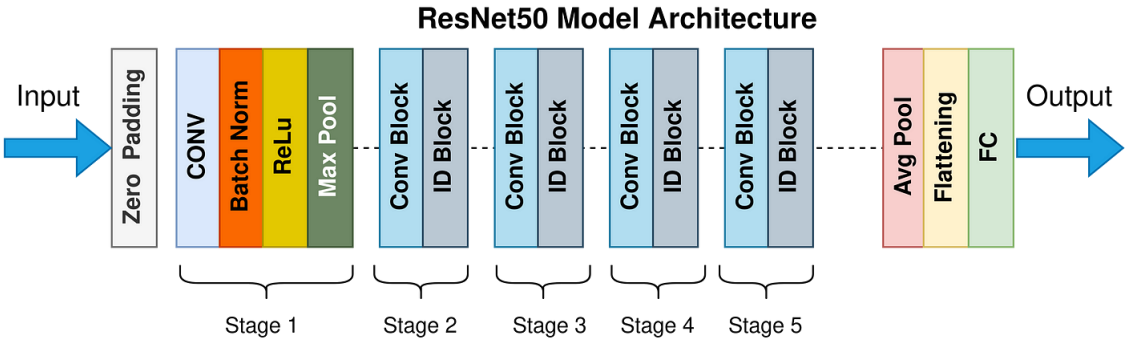


Figure 3.6: ResNet50 architecture

3.2.3.3 DenseNet

DenseNet, or Densely Connected Convolutional Network, is a neural network architecture distinguished by its densely connected layers. It connects every layer in a feed-forward manner to every other layer, in contrast to conventional convolutional neural networks where each layer is only connected to the next following layer. This design helps with gradient flow and feature reuse, which gives an advantage of learning with fewer parameters compared to other networks. In further detail, DenseNet is made up of dense blocks. The feature reuse is encouraged by giving each layer input of feature maps from all levels that came before it. Transition layers are also used by DenseNet to limit the expansion of feature maps and reduce computing expense. This architecture has gained popularity in deep learning research and applications due to its remarkable performance on a multitude of image classification tasks.

DenseNet contains densely connected layers, which ensure that each layer gets information directly from all layers before it. This promotes the reuse of features and eases information transfer across the network. For DenseNet121, there were a total of 7,633,108 parameters out of which 6,047,572 were kept trainable. For DenseNet201, there were a total of 20,995,604 parameters out of which 6,475,092 were kept trainable.

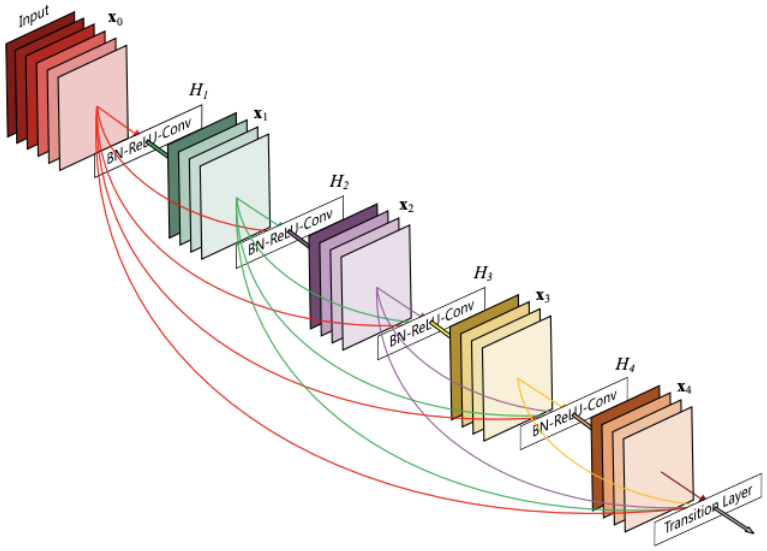


Figure 3.7: Dense blocks in DenseNet's architecture

3.2.3.4 MobileNetV2

A lightweight convolutional neural network, MobileNetV2 was created especially for embedded devices and mobiles with limited processing capability. It expands on the original MobileNet architecture by adding various upgrades for effectiveness and performance. For example, the concept of depthwise separable convolutions are utilized here. This maintains the same representational capacity but reduces computational complexity. These type of convolutions separate the spatial and channel-wise convolutions. Moreover, it adds inverted residual blocks with shortcut connections and also linear bottlenecks to help with gradient propagation. Due to these improvements, MobileNetV2 is a popular

option for deploying deep learning models on smartphones with limited resources since it can achieve impressive levels of performance on tasks like object detection and image classification.

MobileNetV2 introduces depth-wise separable convolutions. Because of its reduced computation and parameter counts, it is ideal for mobile and edge devices and real-time applications with constrained processing capacity. There were a total of 2,283,604 parameters out of which 2,249,492 were kept trainable

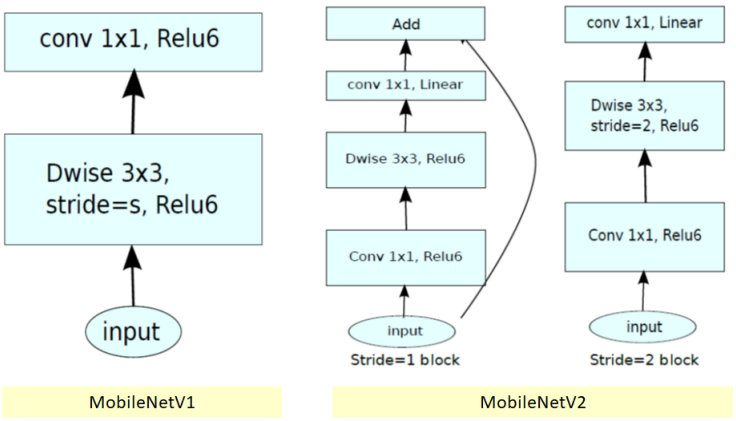


Figure 3.8: MobileNetV2 key feature

3.2.3.5 InceptionV3

Convolutional neural network architecture InceptionV3 is a member of Google's Inception family. Its goal is to maximize accuracy and processing efficiency in computer vision tasks. It consists of several inception modules, pooling operations in its convolutional layers. The filter sizes vary (1x1, 3x3, and 5x5), and parallel feature extraction routes are present. To speed up training and enhance generalization, it also integrates batch normalization technique. For classification, InceptionV3 uses fully connected layers and global average pooling. Because of its sophisticated architecture and effective design, it performs admirably on benchmarks, which has led to its widespread adoption.

InceptionV3 aims to capture multi-scale features efficiently using inception modules. They incorporate convolutions of varying filter sizes and concatenate feature maps. There were a total of 21,843,764 parameters out of which 21,809,332 were kept trainable.

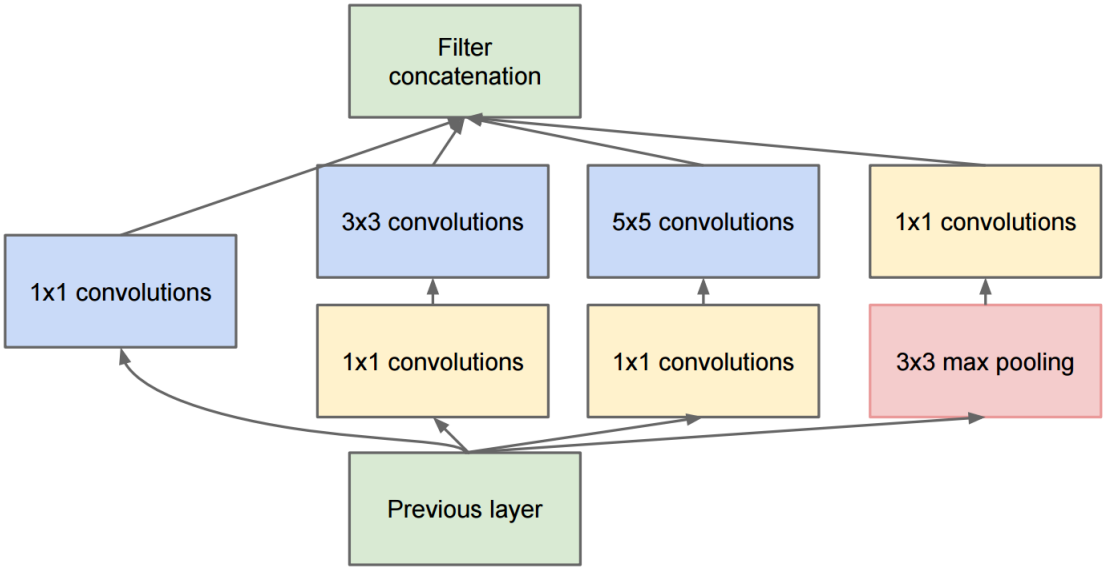
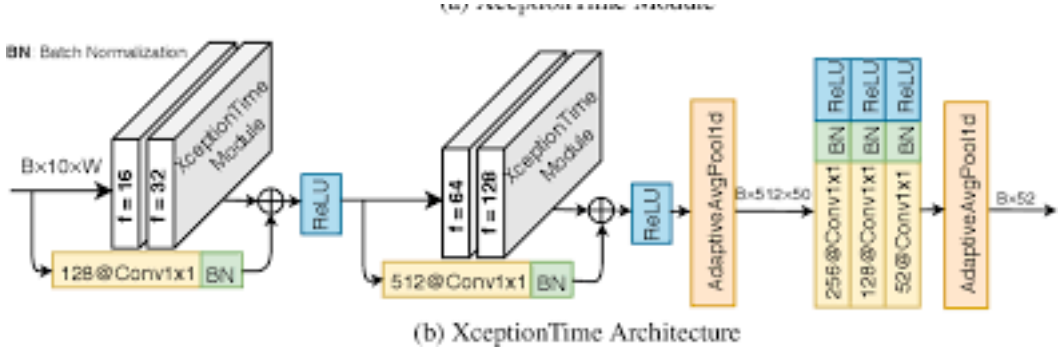


Figure 3.9: InceptionV3's key feature

3.2.3.6 Xception

XceptionNet, or "Extreme Inception," was presented by the man behind the Keras deep learning framework. Though it goes beyond the concept of depthwise separable convolutions, it is influenced by the design of Inception. A sequence of depthwise separable convolutions are used by XceptionNet to replace the conventional Inception modules. Compared to conventional convolutional layers, this separation enables the network to collect spatial and channel-wise correlations independently, resulting in a more economical use of parameters and computations. Its main goal is to boost the network's capacity while minimizing the amount of parameters and computational overhead. XceptionNet achieves state-of-the-art performance with lower risk of overfitting and enhanced efficiency on a variety of computer vision applications.

In Xception, depth wise separable convolutions are used. This minimizes the amount of parameters while capturing spatial interdependence efficiently and encouraging better information flow. The total number of parameters were 20,902,460 out of which 20,847,932 were kept trainable.



(b) XceptionTime Architecture

Figure 3.10: XceptionNet architecture

3.2.3.7 EfficientNet

EfficientNet is a family of convolutional neural networks developed by researchers at Google. It is engineered to attain cutting-edge results on image classification, while maintaining a high degree of computational and parameter efficiency. Compound scaling, which consistently and logically scales the network breadth, depth, and resolution, is the main breakthrough of EfficientNet. It performs better by scaling all of these dimensions at once rahter than the conventional method of just scaling one while maintaining others fixed. The foundational architecture strikes a compromise between depth, width, and resolution within a specified computational budget. The model's size is then gradually increased via compound scaling to get a higher level of accuracy. EfficientNet is known for its exceptional performance on image classification benchmarks like as ImageNet, EfficientNet models use a considerable reduction in parameters and floating-point operations (FLOPS) due to which it is well suited for contexts with limited resources, including mobile and edge devices.

EfficientNet introduces the concept of compound scaling. This means that it scales the network's depth, resolution and width uniformly by a factor which can be learned. This guarantees a computationally efficient model which balances efficiency and complexity, improving performance under various resource restrictions. We have used EfficientNetB0 model whose architecture (figure 4.4) consistently demonstrates good performance on benchmark datasets when it comes to image classification. We added our own GlobalAveragePooling2D layer, dropout layer (at 20% rate) and dense layers with softmax activation for 20 exercise categories. There were a total of 4,075,191 parameters out of which 4,033,168 were kept trainable.

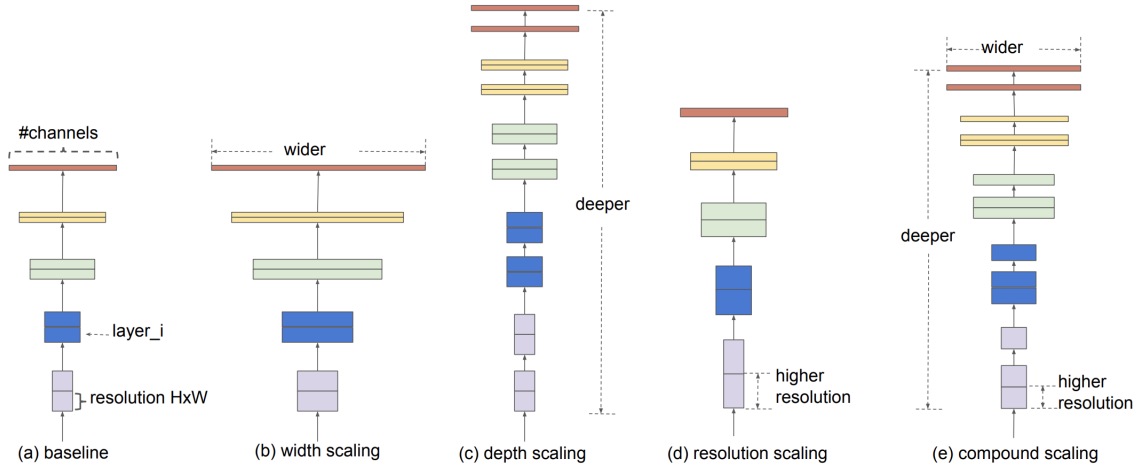


Figure 2. **Model Scaling.** (a) is a baseline network example; (b)-(d) are conventional scaling that only increases one dimension of network width, depth, or resolution. (e) is our proposed compound scaling method that uniformly scales all three dimensions with a fixed ratio.

Figure 3.11: EfficientNet's key feature: Compound Scaling

3.2.4 Model Evaluation and Fine-Tuning

Model evaluation is performed on the test set. For a detailed analysis, we have used multiple methods on each model. We calculated the accuracy, the confusion matrices, class-wise accuracy and generated classification reports for each model. Training and

testing accuracy and loss curves were plotted to analyse the models for overfitting or underfitting. For the process of fine-tuning, we unfroze some of the pre-trained layers. After unfreezing, the parameters associated with those layers also become part of the training process. Multiple tests with different number of frozen layers were performed. After this, the models are further trained for more epochs. Varying improvement in accuracy was observed.

3.2.5 Comparative Study

A comparative study was done to obtain a more thorough understanding of the performance of each model used in the task of pose recognition. Through a thorough analysis of the advantages and disadvantages of various strategies, insights were obtained regarding their effectiveness in a variety of assessment metrics and real-world contexts. Then, the best-performing model found having the best test accuracy was chosen, and its weights were saved for deployment later.

3.2.6 Model deployment

As mentioned before, the best weights for the best performing model, along with all the model parameters were saved for future deployment. To demonstrate the same, the gradio module available in Python was used.

Gradio is a open-source library that enables creating user interfaces which are web-based, for machine learning models. Developers can easily share web-based interfaces for their models, and users can interact with the interface and see the outputs. Gradio is often used to create interactive demos for many machine learning models, and also featuring multiple inputs like image, text, and audio.

3.3 Proposed Approach: Phase 2

There are several steps in the procedure used for the assigned work. The proposed approach is demonstrated below in figure 3.12.

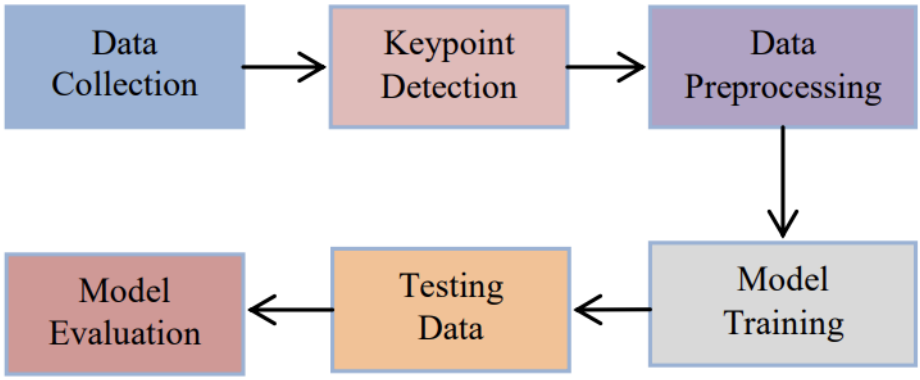


Figure 3.12: Proposed Approach: Phase 2

3.3.1 Dataset Collection

Popular exercises were selected, and for each exercise, images were manually retrieved using Kaggle [16] as a guide and Google search engine. Consequently, the dataset has 1058 images, divided into 20 classes, each including 55–60 images. The twenty workout categories are as follows: bridges, bench press, bow, crane pose, crunches, cycling, cobra, child's pose, donkey kicks, downdog, deadlift, flutter kicks, pullup, plank, standing toe reach, shoulder press, squat, and triceps pushdown. A variety of stretches, bodyweight exercises, and weight training exercises are included in this set.

Tensorflow is then used to load images with a standard size of 224x224. Every image has a label based on the exercise being performed. Before loading, the dataset is shuffled to prevent imbalanced splits, following which a 80:20 split is made for training and testing respectively.

3.3.2 Keypoint Detection

YOLOv8 and MediaPipe algorithms have been used as the keypoint extraction algorithms. For YOLOv8, a grid is created using the input image, which gives cells. Now for each grid cell, bounding boxes and corresponding confidence ratings are predicted as part of a single-stage object recognition approach. The architecture usually comprises of many convolutional layers followed by upsampling and concatenation operations. Anchor boxes are used to increase localization accuracy, while feature fusion and skip connections are used to record spatial relationships.

The second method, MediaPipe extracts keypoints by first preprocessing the input image, then utilizing a convolutional neural network model trained for predicting the location of specific keypoints by detecting and localizing keypoints on the human body, including joints and landmarks. Features are extracted from the image at different scales, followed by regression techniques to predict the coordinates of each keypoint. Post-processing techniques may be applied to refine the results. That extracted keypoints act as the feature

for each image. An example of keypoint detection using MediaPipe on a test image taken from the dataset as shown in figure 3.13.

Figure 3.13: Keypoint detection in an exercise image

3.3.3 Data Preprocessing

Preprocessing involves augmentations, which are required to increase the dataset's diversity to improve generalization. SMOTE (Synthetic Minority Over-sampling) Technique is a method frequently used in data augmentation to reduce class imbalance. By fabricating synthetic records for the minority class, it mitigates imbalance issues. First, a sample is selected at random from the minority class, and then its k-nearest neighbors are found. After calculating a vector between the chosen neighbor with the current data point being examined, it is multiplied with a random number ranging from 0-1. For every image in the dataset, this procedure is repeated. The labels were categorical in nature. They are encoded into numerical values using LabelEncoder.

3.3.4 Model Training

Thereon multiple algorithms have been trained on both the above skeletal data to compare the performance between them. For each algorithm, GridSearchCV was used to find the best hyperparameters. It is a method for systematically tuning hyperparameters of machine learning models. It works by searching through a specified list of parameters and also utilizes cross-validation to evaluate each combination's performance.

3.3.4.1 K-Nearest Neighbors(KNN)

K-Nearest Neighbors is a non-parametric regression and classification approach. For classification, the majority class is taken and for regression, the average of the K closest data points is taken, to predict a new data point. Based on a distance metric, the "nearest" data points are identified. Although it is simple to use and intuitive, how well it performs depends on how the number of neighbors (K) and distance measure are chosen.

KNN could be used to identify exercise positions using information taken from images or videos. For a multitude of types of data, KNN can be a good choice to consider. For example, accelerometer data, joint angles, or key points, KNN may be trained to identify poses. The functioning of calculating the distances between feature vectors for classification still remains the same.

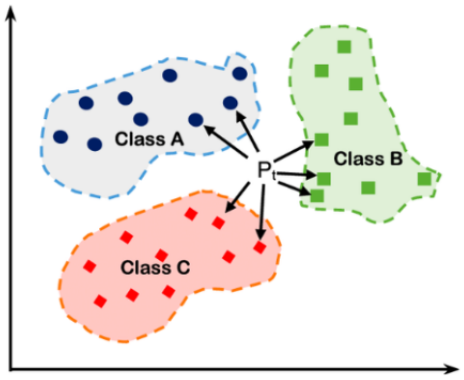


Figure 3.14: KNN

3.3.4.2 GaussianNB

It is based on Bayes' theorem, assumes feature independence, Gaussian Naive Bayes (GaussianNB) is a simple but powerful probabilistic classifier. Classification is done using class probabilities which are calculated using likelihood of features and the each class priors. GaussianNB frequently performs well even though it is simple in its principle and also assumes feature independence.

GaussianNB could be used as a classifier for workout pose recognition to distinguish between exercise poses based on images. It might be trained to categorize features including joint angles or keypoints. Then classify them into various exercise poses. It is also a good choice for applications involving real-time inference, for example real-time pose recognition, because of its simplicity, ease of use and effectiveness, specially in contexts with limited resources.

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Figure 3.15: GaussianNB

3.3.4.3 Linear SVM

Linear Support Vector Machine (SVM) looks for the best linear border in the feature space between classes. It is effective for linearly separable datasets because it maximizes the margin between the closest data points (support vectors) from each class.

It can be used to categorize exercise poses. Through labeled training data representing various workout poses, the SVM learns to accurately categorize fresh instances into the appropriate categories.

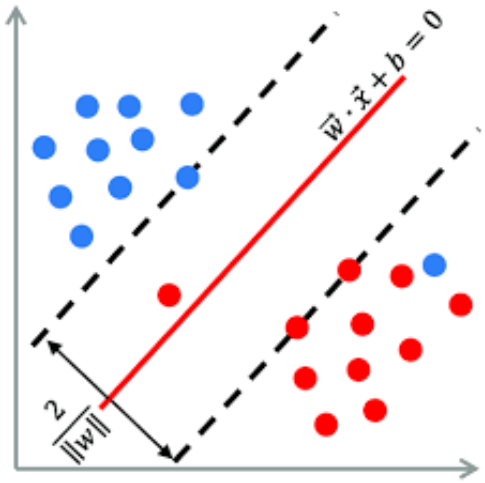


Figure 3.16: SVM explanation

3.3.4.4 Decision Tree

For classification, a flexible and easy-to-understand supervised learning approach is the decision tree. It is based on the input features values, it divides the feature space with a recursive technique. This is used to make smaller subsets of the dats. Each split attempts to maximize information gain. If we visualize, and we can also see in Figure 3.17, a tree-like structure is created, where leaf node represents class labels, and every internal node indicating a choice.

Decision trees could be used for workout posture recognition to categorize exercise poses according to image characteristics. This can be achieved by decision tree partitioning the feature space of, for instance, keypoints to determine which pose a given instance represents. They are useful for predicting and evaluating exercise poses and giving users useful feedback because of its easy understandable visualization.

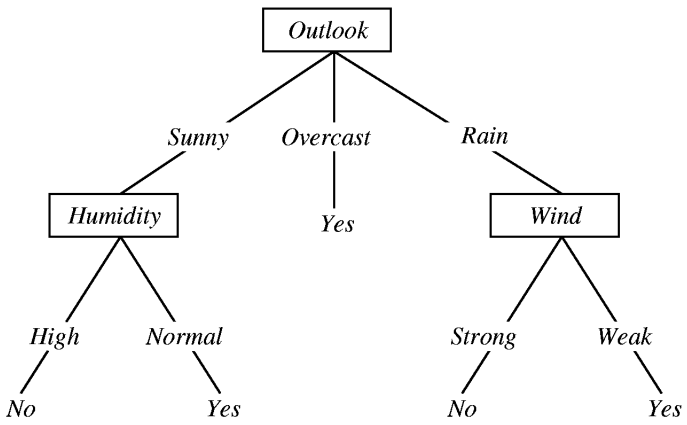


Figure 3.17: Decision Tree working

3.3.4.5 Random Forest

Based on decision trees, Random Forest is an ensemble learning technique. It builds multiple independent decision trees during training and votes together their predictions. To create variation among the trees, each decision tree for making the Random Forest is

trained using a random selection of training data. This random selection improves the model's ability to generalize and also reduces overfitting. Random Forest is popular for its versatile nature, ability to scale well, and the power of high-dimensional data handling.

Random Forest could be used to classify exercise poses because of its collective nature. By this we mean that since numerous decision trees are constructed on different subsets of the data, therefore by using this collective knowledge of trees, workout poses could be effectively classified. Random Forest works well for difficult pose identification tasks in workout monitoring systems because of its robustness against overfitting.

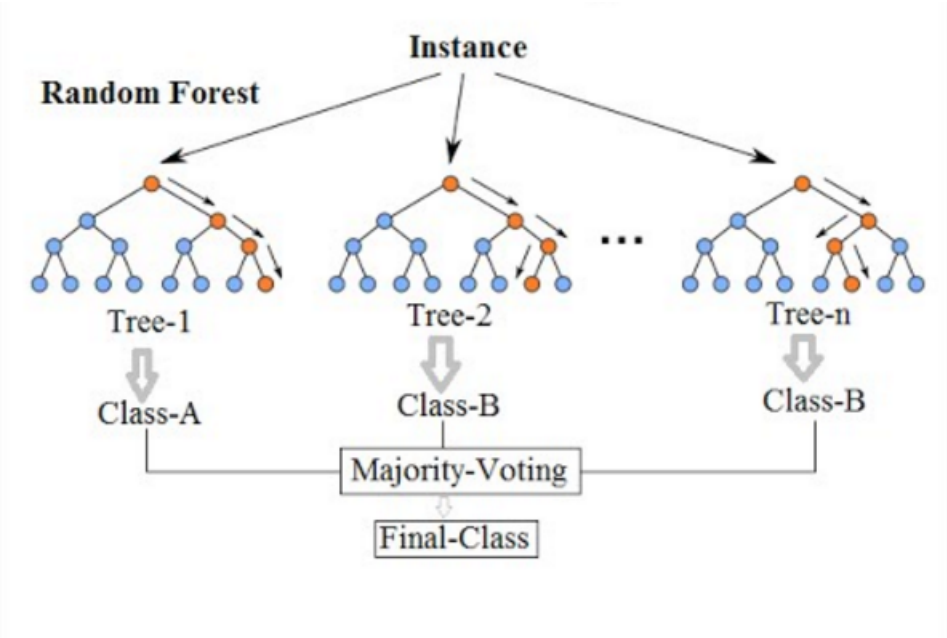


Figure 3.18: Random Forest working

3.3.4.6 LightGBM

LightGBM is a high-performance gradient boosting algorithm. It focuses on speed, accuracy, but with efficiency. It uses a innovative algorithm, using decision trees, which builds the tree using a leaf-wise splitting method. It chooses the leaf with the greatest reduction in loss, or it technical terms, the greatest reduction in delta loss. This helps achieve great performance while also reducing training time substantially and limited memory use. This makes for a scalable solution for large-scale datasets. It is also an advantage for such datasets because of it support for distributed and parallel computation.

LightGBM could be used as a classifier in the context of workout pose recognition through labeled training data with different workout poses and associated features, LightGBM is able to learn intricate patterns and generate precise predictions. Because of its effectiveness and scalability, LightGBM is a good fit for real-time fitness monitoring systems, where prompt and precise position detection is crucial.

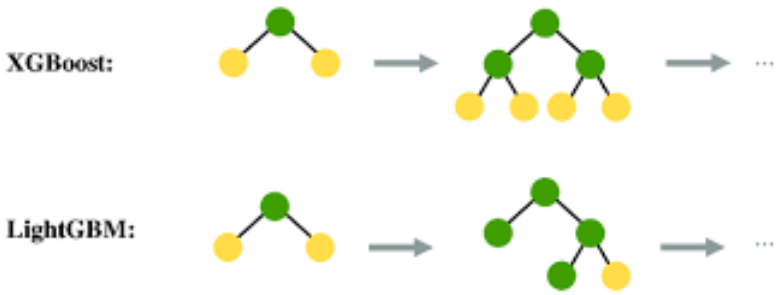


Figure 3.19: LightGBM vs XGBoost: Leafwise splitting

3.3.4.7 XGBoost

XGBoost is a boosting algorithm that is highly optimized. It enhances the generalization capabilities of the model by combining the benefits of gradient boosting with multiple regularization techniques integrated. This techinque consequently helps prevent overfitting. Because it also offers suppoert for parallel computing, it has the ability to handle massive datasets and is hence scalable as well. It is clear that it can be and also is a well-liked option for several tasks, including classification, regression because of its speed, accuracy, high generalizability and being less prone to overfitting.

XGBoost could be used as a classifier for workout pose recognition through training data labeled with different workout poses. XGBoost is able to understand intricate patterns and produce precise predictions thanks to its . The posture recognition system is made more robust and less prone to overfitting thanks to XGBoost's optimization approaches.

3.3.4.8 Bagging

Machine learning models can be made more accurate and stable by using an ensemble learning technique called bagging, which is short for bootstrap aggregating. It operates by using replacement sampling to train several versions of the same base learning algorithm. This is done on varying subsets of the training data. Each model independently trains to predict the target variable. For classification, this is done by voting. By increasing model variety, it lowers variance and hence prevents. The predictions made using bagging technique are often more reliable and accurate.

Bagging can be used to enhance the performance of classifiers that recognize exercise positions. This is achieved by improving the overall accuracy through training multiple classifiers on distinct portions of the train data. By enhancing generalization and reducing the likelihood of overfitting, bagging can result in more dependable workout pose identification.

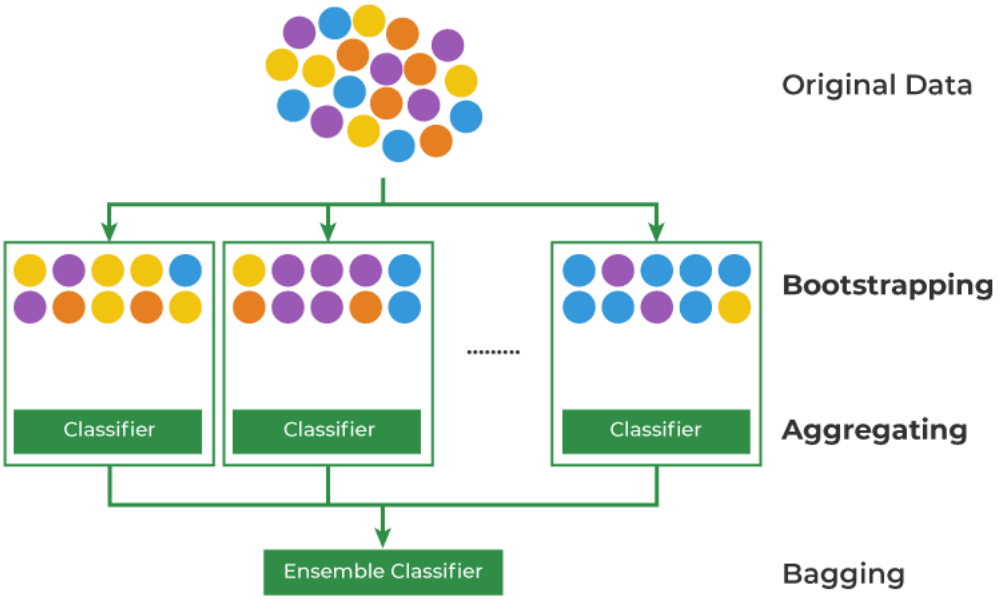


Figure 3.20: Bagging explanation

3.3.5 Model Evaluation

For evaluation and analysis of the models, a test set was used. On each model, several techniques were applied for a thorough analysis. Then their test accuracy, precision and recall were determined. A comparative analysis was conducted in order to further understand each models' performance with respect to the keypoint detection techniques used. The model that performed the best and had the highest test accuracy was then selected, and its weights were saved.

Chapter 4

RESULTS and DISCUSSION

4.1 Phase 1

For all the 9 models, the learning rate has been kept at 0.001, with Adam optimizer while fine tuning with different number of layers and using categorical cross entropy loss.

We conducted an analysis of all the models used in our study for workout image recognition. Table 4.1's benchmark results demonstrate EfficientNetB0's superior performance, with the model attaining the maximum accuracy of 91% for top-1 classification. Among the best performers, DenseNet201 stood out with a 99.05% accuracy rate for top-5 classification closely followed by EfficicentNetB0 with 98.9%. We also present information on each model's precision, recall and F1 score.

Table 4.1: Benchmark result of 9 T-L models

Models	**Before fine-tuning**	**After fine-tuning**				
	Initial Accuracy	**Top-1 Accuracy**	**Top-5 Accuracy**	**Precision**	**Recall**	**F1-Score**
VGG16	70.62%	79.62%	97.16%	0.88	0.81	0.81
VGG19	81.52%	83.89%	98.10%	0.85	0.84	0.84
ResNet50	76.30%	81.99%	97.16%	0.84	0.82	0.81
DenseNet121	66.35%	85.31%	98.00%	0.87	0.87	0.86
DenseNet201	72.51%	86.73%	99.05%	0.89	0.88	0.88
MobileNetV2	72.64%	76.42%	92.92%	0.81	0.76	0.75
InceptionV3	73.11%	84.43%	95.75%	0.85	0.84	0.83
Xception	76.30%	81.52%	96.68%	0.84	0.83	0.82
EfficientNetB0	86.73%	**91.00%**	98.90%	0.92	0.92	0.91

We can see that EfficientNetB0 gave us the best performance for top 1 accuracy. For top-5 accuracy also, the performance is closely matched to the best performing DenseNet201. It is remarkable that even with much fewer parameters than DenseNet201, EfficientNetB0 is able to give competitive performance. We can also see with the precision, recall and F1 score values that EfficientNetB0 can be considered to be the overall best performing model. The scalability and efficiency of EfficientNetB0 is evident here and it can also be argued that the inference times using this model can be faster than the other models mentioned here. This demonstrates the strength of architectural creativity over parameter count. While it may not boast the large number of parameters of some of the other deep learning architectures, it still emerges triumphant in performance in our case, showcasing a superiority that defies expectations. This also lends well to our use case of limited compute power. For scenarios where the computing power is limited, for example on mobile devices, a solution is needed that provide fast inference using lightweight models. And EfficientNetB0 fits nicely in this context. We have given a detailed architecture of EfficientNetB0 in Figure 4.1 to demonstrate its functioning, how an image goes through the various layers and how it is modified.

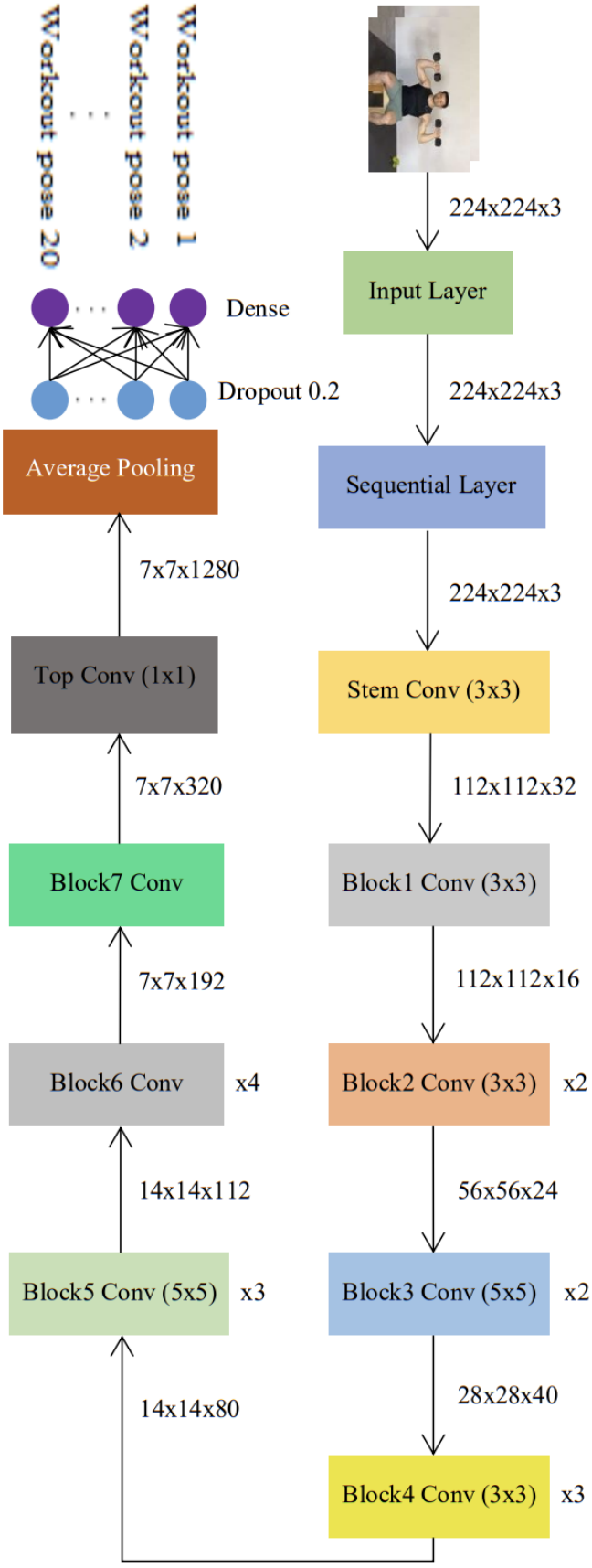


Figure 4.1: Detailed architecture of EfficientB0: the best model found

In the first training phase of EfficicentNetB0, the best model discovered, took place across 25 epochs, yielding an initial testing accuracy of 86.73% and a testing loss of 0.43. After then, the 150th layer was the focus of 10 more epochs of fine-tuning with the learning rate set to 0.0001. With the use of checkpointing and optimal accuracy monitoring mechanisms, the procedure produced an astounding 99.65% fine-tuning training accuracy and 0.012 fine-tuning loss. Testing accuracy for the optimized model was 91.00%, and testing loss was 0.43, which was the same as during the first training phase. The model's accuracy was significantly increased overall through the fine-tuning procedure, indicating the success of the modifications made at this stage. The accuracy and loss curves are also provided to illustrate the model's development with fine-tuning; they show a clear tendency towards improved accuracy and lower loss. The accuracy and loss curves in figures 4.2 and 4.3 graphically illustrate the model's development with finetuning; they show a clear tendency towards improved accuracy and lower loss.

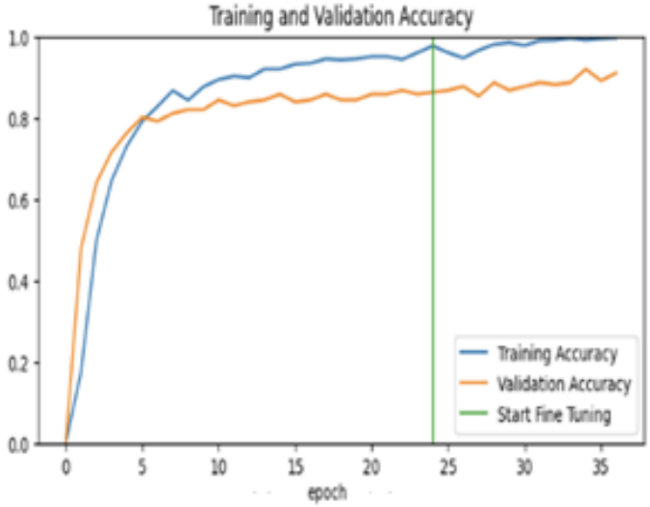


Figure 4.2: Accuracy curve of EfficientNetB0 model

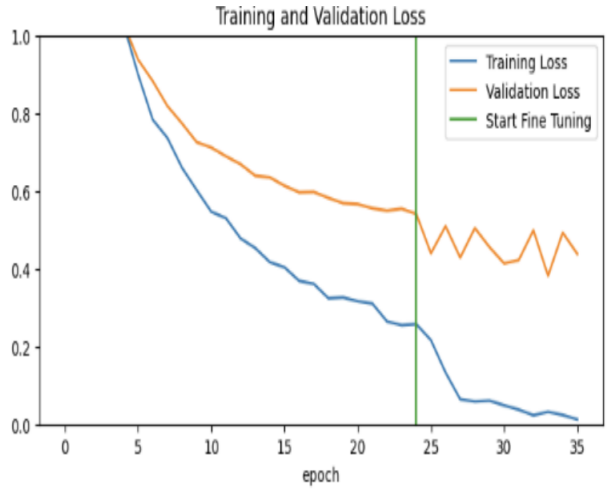


Figure 4.3: Loss curve of EfficientNetB0 model

In order to provide a more nuanced view of exercise-specific accuracy, we have included Table 4.2 that shows EfficientNetB0's accuracy for each workout class. The model performs well in a variety of exercises, with accuracy levels ranging from 69.23% to 100.00%. Most exercises yield greater than or equal to 80%, only tricep-pushdown shows a low performance of 69.23%. Examining the AUC-ROC curve as shown in figure 4.4 offers a thorough evaluation and shed light on the model's discriminative ability in this multi-class situation. The model's competency is demonstrated by notable precision, particularly the 100.00% accuracy for classes like Bow, Bridges, Child's Pose, Cobra, Crane Pose, Cycling, FlutterKicks and Pullup.

Table 4.2: Pose-wise accuracy of EfficientNetB0 model

Workout Pose	**Accuracy**
Benchpress	91.66%
Bow	100.00%
Bridges	100.00%
Child's Pose	100.00%
Cobra	100.00%
Crane Pose	100.00%
Crunches	90.00%
Cycling	100.00%
Deadlift	80.00%
Donkey kicks	90.00%
Downdog	84.61%
FlutterKicks	100.00%
Lat-pulldowns	88.88%
Leg-extensions	93.33%
Plank	82.35%
Pullup	100.00%
Shoulder Press	90.90%
Squat	90.00%
Standing toe reach	90.00%
Tricep-pushdown	69.23%

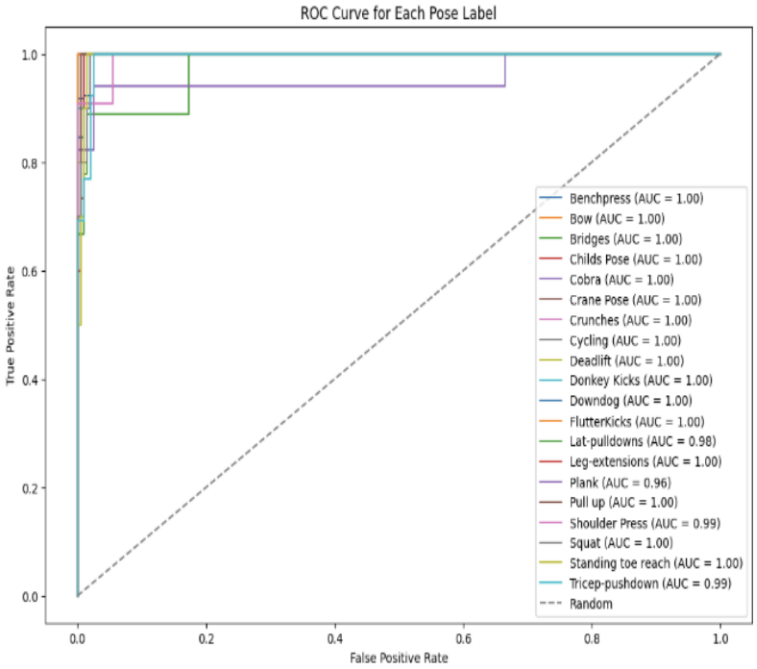


Figure 4.4: AUC-ROC curve of EfficientNetB0 model

A breakdown of classifications is given by the confusion matrix in figure 4.5. Examining measures relevant to each class, such as precision and recall, we obtain detailed understanding of how well the model understands different types of exercises. Diagonal elements show the model correctly predicting poses. The off-diagonal elements point out areas that require improvement. It shows which incorrect class the model predicts for each exercise. The same follows for all the classes. This could be of interest as it may indicate similarity of movement or range of motion between exercises which may require more discriminating data to distinguish classes. This method provides a detailed insight of the strengths and limitations of the model. The confusion matrix is a very useful tool for evaluating any machine learning model. For a binary or a multi-class problem, though it is much more useful in a multi-class setting, this tool allows a simple visual that can highlight the initial shortcomings of the model. For example, if some classes are being very poorly classified, then that warrants further investigation. Just looking at accuracy will not give us that detailed information. Further, if a particular class is being misclassified as some other class repeatedly, then that is also an interesting observation. There must be some similarity between the 2 classes which causes this confusion. So, confusion matrix allows this per-class analysis.

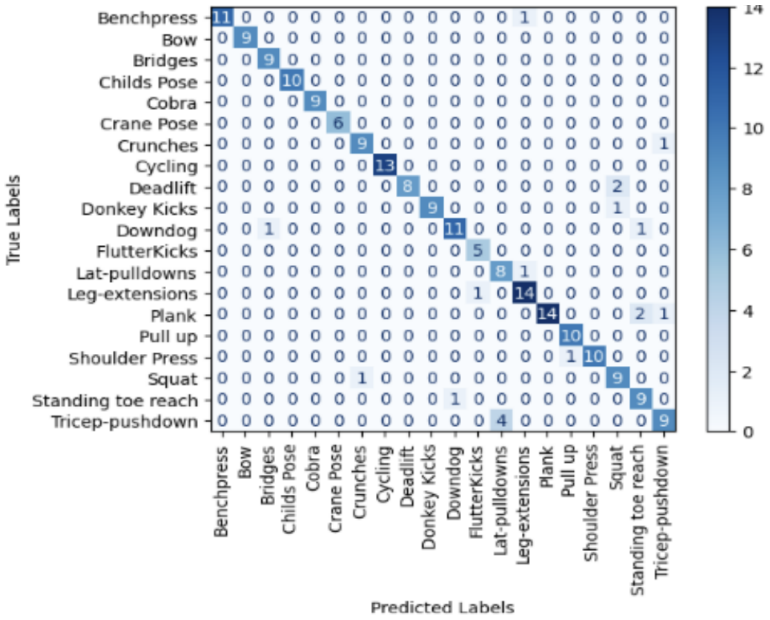


Figure 4.5: Confusion-matrix for EfficientNetB0 model

4.1.1 Outputs

Figure 4.6 shows the true and predicted labels for images of different workout stances from the testing dataset. As demonstrated in the confusion matrix, there are incorrect predictions in the result as well. For classes where lower accuracy is obtained, it is useful to analyze possible similarities between images which may cause confusion.

Figure 4.6: Actual vs Predicted pose of some test images

Gradio was used in the development of an online interface. It is an open-source toolkit that makes it possible to create machine learning model user interfaces. It is frequently used to produce interactive demonstrations with many inputs, including text, audio, and images. Figure 4.7 and figure 4.8 demonstrate this interface in action. Figure 4.7 shows what the interface looks like. Figure 4.8 demonstrates the scenario wherein either a user uploads an image to the model, or the application can also be integrated with a camera that periodically clicks images and feeds it to the model. Based on the what the model has learnt, it outputs a prediction.

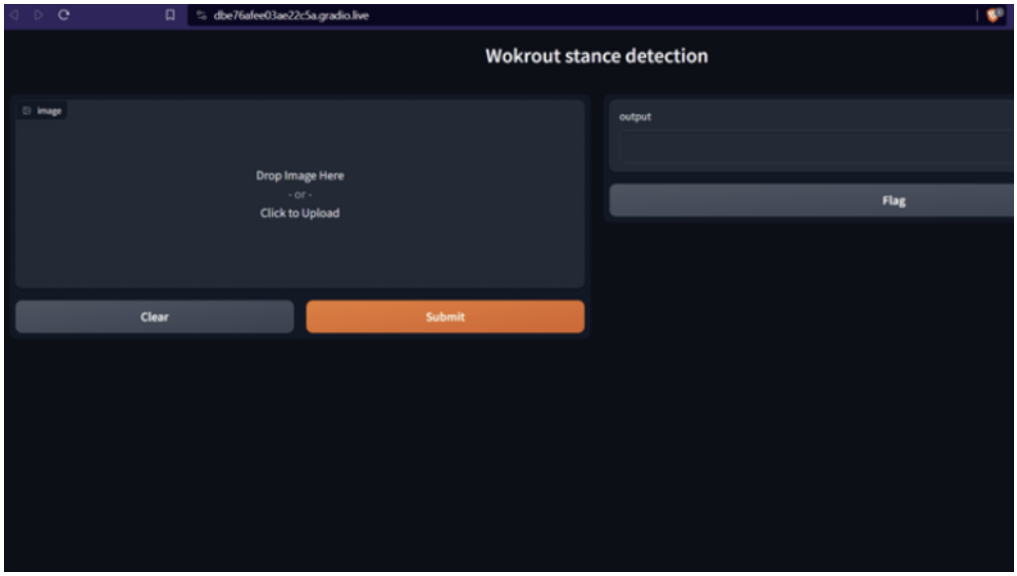


Figure 4.7: Gradio interface

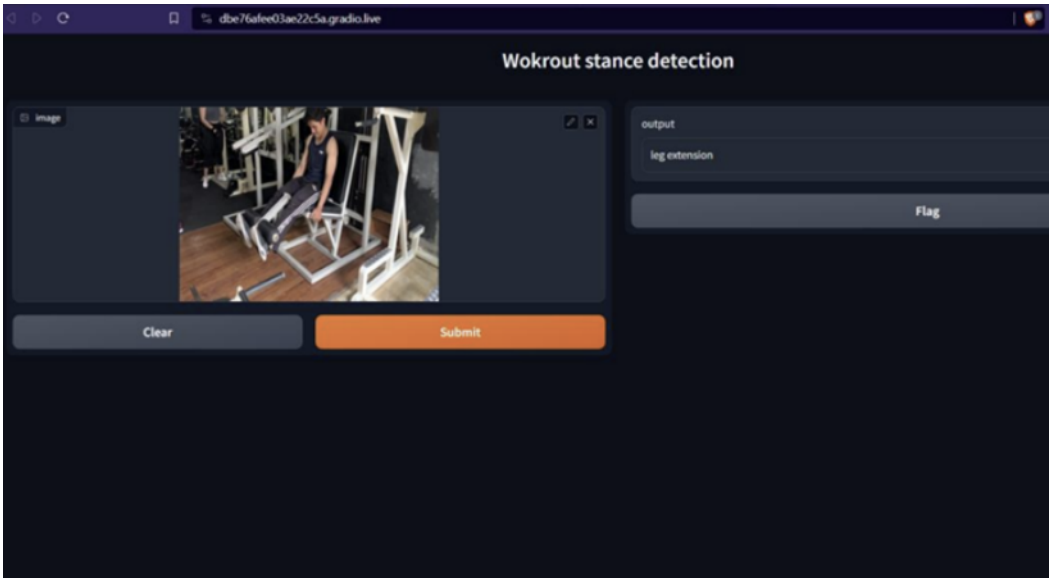


Figure 4.8: Gradio interface showing output for an image

4.2 Phase 2

The methodology followed in Phase 2 was described above. It involves data collection, followed by keypoint detection on the images in the dataset. The dataset chosen was the same as followed in Phase 1. On those images, 2 keypoint identification algorithms were used, namely YOLOv8 and MediaPipe, as mentioned before. Further, data augmentation using SMOTE technique has been performed to eliminate any class imbalance and ensure that all classes have equal representation. This takes care of preparing the dataset and also making sure that there is adequate diversity present in the dataset for the complex task of workout pose recognition. Then we begin training the model. The details of the models trained and their tuning has been discussed in detail later. After any model has been trained, testing is performed using the test set. It was previously mentioned that an 80:20 split was used between the train and test split of the data. After testing, evaluation is performed using multiple metrics. Using multiple metrics ensure that we can analyze and understand the depth of the model, its performance, check for inconsistencies and overfitting.

YOLOv8 predicted a total of 32 keypoints in this experiment while MediaPipe produced 33. Table 4.3 listed the best parameters found using Grid Search with respect to each machine learning and ensemble technique upon training on each skeletal data.

4.2.1 Hyperparameter tuning

For this phase, several hyperparameter tuning was done to optimize peroformance.

Machine learning hyperparameters are like the tuning knobs of a musical instrument, fine-tuning the performance of a model to achieve optimal results. These parameters, distinct from model parameters learned during training, dictate the behavior and performance of the learning algorithm itself.

Tuning hyperparameters is crucial because it directly influences a model's ability to learn and generalize patterns from data. Just as a symphony requires precise tuning for harmonious melodies, machine learning models require careful adjustment of hyperparameters to orchestrate optimal performance. The following Table 4.3 shows the models used and the best hyperparameters that were found. The analysis was performed fro both YOLOV8 models and MediaPipe based models.

Table 4.3: Best Hyperparameters Found Using GridSearchCV

Model	Parameter	Criterion	
		YOLOv8	MediaPipe
Decision Tree	max_depth	none	none
	min_samples_split	2	1
	min_samples_leaf	2	2
Linear SVM	splitter	best	best
	c	1	1
	loss	hinge	hinge
	penalty	l1	l2
Multi Layer Perceptron	activation	relu	relu
	alpha	0.0001	0.0001
KNN	n_neighbors	3	3
	weights	distance	distance
XGBoost	p	2	2
	learning_rate	0.001	0.001
	n_estimators	128	100
	max_depth	5	3
	gamma	0.1	0.2
Bagging	n_estimators	128	100
	learning_rate	0.003	0.001
LightGBM	n_estimators	100	100
	max_depth	3	5
Random Forest	n_estimators	128	100
	min_samples_leaf	1	1
	min_samples_split	2	5
	max_depth	none	5

4.3 Experimental results

According to Table 4.4, classifiers that use MediaPipe keypoints as opposed to YOLOv8 achieved higher accuracies. For instance, Linear SVM with MediaPipe shows accuracy increase from 69.88% to 90.90%, while GaussianNB's accuracy increases from 56.79% to 88.88% as well. LightGBM performed best with MediaPipe with 95.95%. Random forest closely follows it with 94.94%. Other ensemble techniques like Bagging also shows higher accuracy rates. Additionally, Multi-Layer Perceptron shows a notable improvement in accuracy employing MediaPipe keypoints, going from 81.89% to 91.41%.

For YOLOv8, Bagging showed the best performance of 82%. Further, in contrast to MediaPipe, Decision Trees and KNN show comparatively poorer accuracy rates when using YOLOv8. These results highlight how important feature extraction methods are to improving classifier performance.

Table 4.4: Test Accuracies of All Classifiers

Individual classifier	Accuracy (YOLOv8)	Accuracy (MediaPipe)
Decision Trees	50.00%	84.34%
Linear SVM	69.88%	90.90%
Multi-Layer Perceptron	81.89%	91.41%
KNN	66.68%	70.20%
GaussianNB	56.79%	88.88%
XGBoost	73.00%	91.91%
Bagging	82.00%	93.43%
LightGBM	80.00%	95.95%
Random Forest	78.00%	94.94%

Compared to using just one model, ensemble algorithms integrate several models to get more accurate predictions. They increase strengths and minimize weaknesses by utilizing the diversity of several models. This is reflected in the results with all ensembles showing better performance.

Overall it can be said that the combination of MediaPipe with ensemble algorithms show consistently good performance when compared with other combinations.

To obtain more fine-grained understanding of the performance, precision, recall and F1 score are shown in Table 4.5. Precision tells us out of all the positively predicted occurrences, what is the percentage of correctly positive predicted instances. Recall gives us the percentage of accurately predicted positive cases among all true positives. It tells us how well the model recognizes positive cases. F1 score gives us a single metric for model evaluation which is the just the harmonic mean of precision and recall.

Table 4.5: Precision, Recall and F1 score for Mediapipe models

Algorithm	Precision	Recall
Decision Tree	0.83	0.85
Linear SVM	0.89	0.87
Multi-Layer Perceptron	0.89	0.90
KNN	0.67	0.64
GaussianNB	0.87	0.88
XGBoost	0.96	0.95
Bagging	0.97	0.95
LightGBM	0.97	0.96
Random Forest	0.95	0.94

Accurate classification and a reduction in false positives using MediaPipe keypoints is demonstrated by most ensembles, which routinely attain precision and recall above 0.95. When it comes to reliably classifying images based on MediaPipe keypoints, Linear SVM and Multi-Layer Perceptrons also fare well, scoring around 0.89. Lower ratings for KNN, however, point to possible false positives and limitations in its classification. The findings highlight the significance of choosing the right technique with MediaPipebased features for classification tasks. It can be argued that YOLOv8 is a more popular model for object and keypoint detection than MediaPipe. However, it would be unwise to look at any algorithm in isolation and instead experiements such as the one conducted above should be performed. This would help determine the viability of any solution and allow more the

emergence of combination of models that can be further developed as frameworks. For instance, in our case MediaPipe+LightGBM may be good for action recognition.

The confusion matrix in Figure 4.9, provides a breakdown of exercise classifications. The accurate pose predictions are demonstrated via diagonal elements. The regions that are misclassified and hence need improvement are the offdiagonal elements. For every exercise, it displays the wrong class that the model predicts. From the confusion matrix, the system demonstrates 100% accuracy for a number of exercise categories, namely bench press, cobra, child's pose, downdog, deadlift, flutter kicks, leg extensions, lat pulldowns, plank, standing toe reach, shoulder press and triceps pushdown. This shows that the given combination of keypoint detection technique and model proves to be a good solution for the given task. It accurately detects many of the workout poses, even those with similar ranges of motion, for example flutter kicks and leg extensions. This is something that is desirable in such a context since many exercises go through similar range of motion with only slight tweaks. This completely changes the dynamics of the exercise and shifts focus towards different target muscle groups.

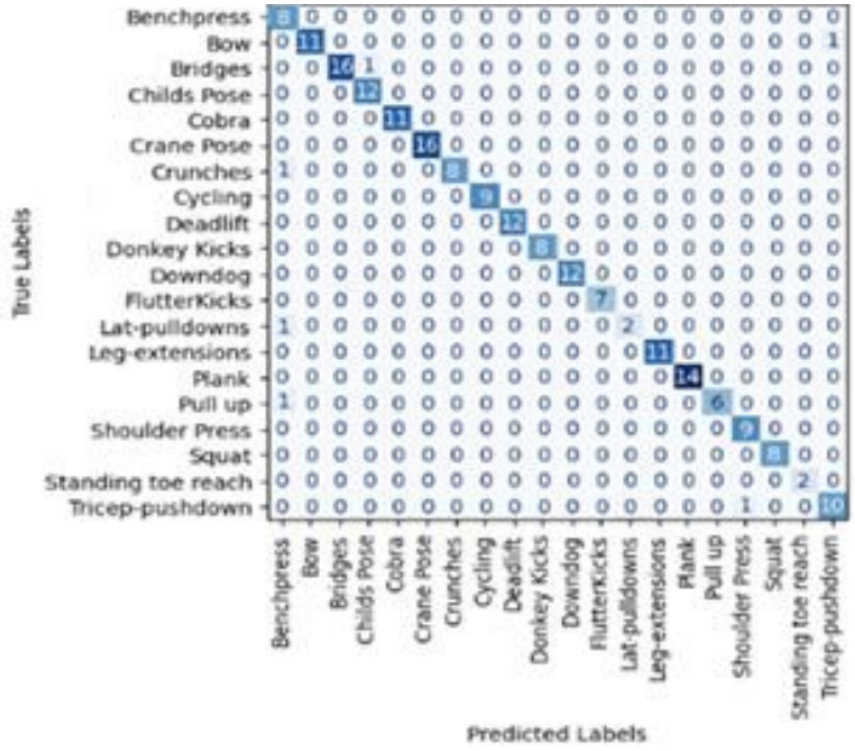


Figure 4.9: Confusion matrix of LightGBM model with MediaPipe landmarks

Table 4.6: Pose-wise accuracy for LightGBM+MediaPipe model

Pose	Accuracy
Bridges	94.11%
Bench Press	100.00%
Bow	91.66%
Cobra	100.00%
Child's Pose	100.00%
Crane Pose	93.75%
Cycling	88.88%
Crunches	88.88%
Downdog	100.00%
Donkey Kicks	87.50%
Deadlift	100.00%
Flutter Kicks	100.00%
Leg Extensions	100.00%
Lat Pulldowns	100.00%
Pullups	85.71%
Plank	100.00%
Squat	87.50%
Standing Toe Reach	100.00%
Shoulder Press	100.00%
Tricep Pushdown	100.00%

Table 4.6 shows the pose-wise accuracy for MediaPipe+LightGBM model. This is confirmed by its high precision and recall.

Figure 4.10 displays the learning curve for the LightGBM model, illustrating how the cross-validation score evolves as the number of training instances increases. On the x-axis, we observe the number of training samples, while the y-axis shows the cross-validation score. The green line shows the change in cross validation score. We observe an steep initial increase from around 0.6 to roughly 0.95. This fast increase indicates model's effective learning and becoming more proficient at extracting features from the data and leveraging them for classification. It also shows that the introduction of additional data into the model is enhancing the model's performance to a substantial degree. This is helping it to generalize better and make more accurate predictions.

However, upon the addition of more training samples, the curve doesn't show the steep increase that was evident in the beginning. Instead, it begins to plateau, and the score only slightly increasing to 0.96 from 0.95. This implies that there comes a point in training where the model's learning speed is fading. Additional data provides limited improvement. This is expected in model training, after a certain volume of data has been trained, the returns on investing more data into the model, is limited. This fact is important to understand, specially for practical and wide-use applications as it highlights the tradeoff of cost of acquiring more data and potential performance gains. It also shows the importance of realizing that focusing on other aspects for model improvement like hyperparameter tuning is essential. This is an aspect that we have focused on, and also on model architecture. The issue of model architecture is tackled to some extent using the variety of models we have used for our use cases. Thus, a learning curve not only depicts the efficiency of a model, in our case, LightGBM. Additionally, it provides valuable

insights for optimizing resources and efforts. Instead of focusing on gathering more and more data, wasting time and potentially money on it, efforts can be redirected towards enhancing data quality or alternative model enhancements. We know that gathering data is time-consuming and expensive. Consequently, this plateau phase helps in efficiently allocating resources and achieving a more robust and effective model.

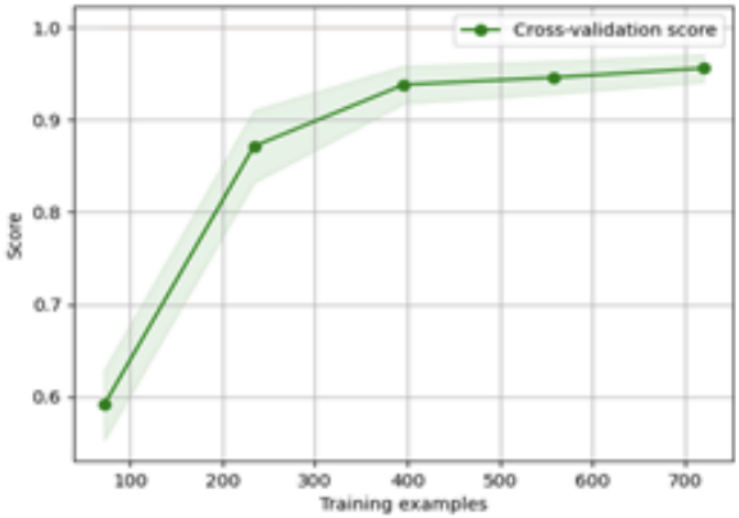


Figure 4.10: Cross-validation score vs number of samples

The Figure 4.11 shows the AUC-ROC curve for each class in the MediaPipe keypoints based LightGBM model. The ROC curve displays the true positive rate versus the false positive rate. The curve closest to the top-left corner represents the Benchpress class, indicating perfect classification performance. Classes like Bridges, Cobra and many others also have ROC curves hugging the top-left corner, suggesting excellent discrimination ability.

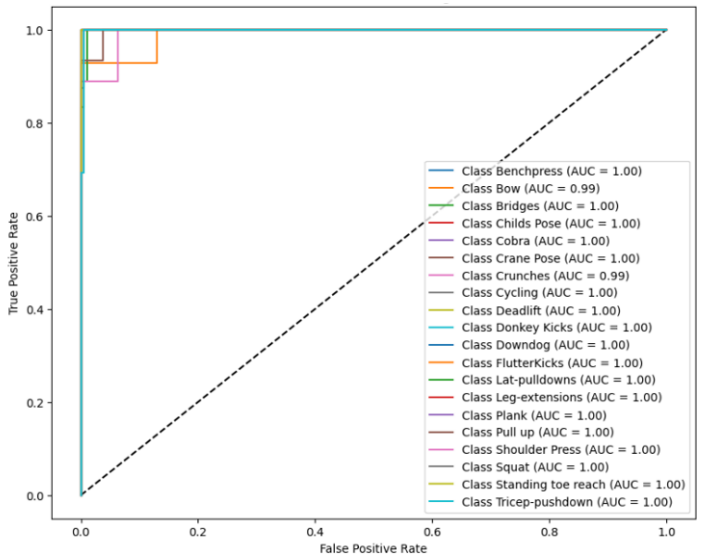


Figure 4.11: AUC-ROC curve for each class Mediapipe+LightGBM)

4.3.1 Outputs

In Figure 4.12, a series of images from the testing dataset are shown. It shows various exercise poses input to the model for testing purpose. Each image is listed with both the true labels and the predicted labels generated by the model. The true label tells us the exercise pose as found in the labeled dataset. Having this comparison visually allows quick assessment of the model's accuracy.

Furthermore, to showcase the performance of how accurate the keypoint algorithms are, the keypoints predicted using MediaPipe are overlaid on the poses. Because of this, we can see the algorithm's ability to accurately identify major keypoints. These include major joints such as shoulders, elbows, and knees, and limbs, etc which are crucial to identify the body pose, and then consequently identify exercise positions. The accuracy of keypoints predicted by the keypoint detection algorithms offers additional understanding into how effectively a model comprehends human poses.

This graphical display illustrates how well the model categorizes and accurately identifies poses, which can have practical uses such, as tracking fitness progress aiding in physical therapy and advancing sports science. By comparing the key points with the predicted labels it becomes simpler to pinpoint strengths and weaknesses in the models performance. This insight can guide enhancements like refining the model and enhancing its adaptability, in real life situations.

Figure 4.12: True vs Predicted pose of a few test samples

Chapter 5

CONCLUSION AND FUTURE SCOPE

The presented experimental results reveal insights into the performance of prominent CNN architectures—VGG16, VGG19, ResNet50, DenseNet121, DenseNet201, MobileNetV2, InceptionV3, Xception, and EfficientNetB0 on workout pose detection. The dataset provides a challenging and diverse set of visual data for training and evaluation. The observed variations in test accuracies across models and epochs underscore the influence of architectural differences. EfficientNetB0 has demonstrated the strongest performance, achieving top 1 accuracy of 91%. Architectures like DenseNet121 and DenseNet201 notably performed worse than EfficientNetB0. These findings highlight the importance of choosing configurations and architectures that balances model complexity with the complexity of the problem, taking into account the diversity of exercises and the context of application.

In this research, the domain of action recognition was explored by utilizing multiple techniques using multiple algorithms, using keypoint detection via YOLOv8 and MediaPipe. During training, data augmentation techniques were used to correct class imbalance and improve resilience. The classification step was carried out after data augmentation and keypoint extraction. MediaPipe with ensemble algorithms consistently showcased impressive performance. However YOLOv8 was found to be inferior to MediaPipe in all the tests, even though the number of keypoints that were found to be nearly the same for both. This also shows the capability of ensemble algorithms in how the different models combine their learned features to improve performance.

Several avenues for future research exist. Firstly, the exploration of ensemble methods holds promise for further improving the overall accuracy. Combining predictions from multiple models can enhance generalization. Secondly, the integration of transformer-based models for vision, which is the latest development in computer vision tasks also holds promise. Specifically, vision transformers[14] have shown to beat previous state-of-the-art set by CNN architectures on multiple tasks. Furthermore, attention to data augmentation techniques could enhance model generalization. Specifcally, investigating results through an augmentation technique called RandAugment[13]. We may also attempt to improve model performance by incorporating keypoints. Finally, an exploration of explainability and interpretability may contribute to the transparency of the system and facilitate user trust and acceptance. Overall, the findings from this study lay the foundation for further advancements in workout pose detection, encouraging the integration of novel architectures and methodologies to address the inherent challenges.

Bibliography

[1] Y. S. Jain, D. Chowdhury, and M. Chattopadhyay, "Machine learning based fitness tracker platform using mems accelerometer," *2017 International Conference on Computer, Electrical & Communication Engineering (ICCECE)*, pp. 1–5, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:53280514

[2] W. Zhang, C. Su, and C. He, "Rehabilitation exercise recognition and evaluation based on smart sensors with deep learning framework," *IEEE Access*, vol. 8, pp. 77 561–77 571, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:218565154

[3] R. Ji, "Research on basketball shooting action based on image feature extraction and machine learning," *IEEE Access*, vol. 8, pp. 138 743–138 751, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:221087665

[4] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Computer Vision - 14th European Conference, ECCV 2016, Proceedings*, ser. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Springer Verlag, 2016, publisher Copyright: © Springer International Publishing AG 2016.; 14th European Conference on Computer Vision, ECCV 2016 ; Conference date: 08-10-2016 Through 16-10-2016.

[5] A. Moran, B. Gebka, J. Goldshteyn, A. Beyer, N. Johnson, and A. Neuwirth, "Muscle vision: Real time keypoint based pose classification of physical exercises," 2022.

[6] S. R. Sreela and S. M. Idicula, "Action recognition in still images using residual neural network features," *Procedia Computer Science*, vol. 143, pp. 563–569, 2018, 8th International Conference on Advances in Computing and Communications (ICACC-2018). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050918321306

[7] W. Yang, Y. Wang, and G. Mori, "Recognizing human actions from still images with latent poses," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2030–2037.

[8] H. Kwon, Y. Kim, J. S. Lee, and M. Cho, "First person action recognition via two-stream convnet with long-term fusion pooling," *Pattern Recognition Letters*, vol. 112, pp. 161–167, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865518303027

[9] M. Singh, M. S. A. Ansari, and M. Govil, "Deeppose: An integrated deep learning model for posture detection using image and skeletal data," in *2023 14th In-*

ternational Conference on Computing Communication and Networking Technologies (ICCCNT), 07 2023, pp. 1–7.

[10] H. Rookba, b. hussein abdallah, A. Ahmed Abdallah, R. Osama Abdel-Aal, R. Reda Numan, A. Khaled Darwish, and W. El-Behaidy, "Automatic feedback for physiotherapy exercises based on posenet," *FCI-H Informatics Bulletin*, vol. 2, no. 2, pp. 10–14, 2020.

[11] J. Jose and S. Shailesh, "Yoga asana identification: A deep learning approach," *IOP Conference Series: Materials Science and Engineering*, vol. 1110, no. 1, p. 012002, mar 2021. [Online]. Available: https://dx.doi.org/10.1088/1757-899X/1110/1/012002

[12] V. Podgorelec, Pecnik, and G. Vrbancic, "Classification of similar sports images using convolutional neural network with hyper-parameter optimization," *Applied Sciences*, vol. 10, no. 23, 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/23/8494

[13] M. Y. Farhad, S. Hossain, M. R. K. Tanvir, and S. A. Chowdhury, "Sports-net18: Various sports classification using transfer learning," in *2020 2nd International Conference on Sustainable Technologies for Industry 4.0 (STI)*. IEEE Computer Society, 2020, pp. 1–4.

[14] K. Joshi, V. Tripathi, C. Bose, and C. Bhardwaj, "Robust sports image classification using inceptionv3 and neural networks," *Procedia Computer Science*, vol. 167, pp. 2374–2381, 2020, international Conference on Computational Intelligence and Data Science. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050920307560

[15] B. Dittakavi, D. Bavikadi, S. V. Desai, S. Chakraborty, N. Reddy, V. N. Balasubramanian, B. Callepalli, and A. Sharma, "Pose tutor: An explainable system for pose correction in the wild," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2022, pp. 3539–3548.

[16] D. Das, S. M. Busetty, V. Bharti, and P. K. Hegde, "Strength training: A fitness application for indoor based exercise recognition and comfort analysis," in *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2017, pp. 1126–1129.

[17] B. Zhou, M. Sundholm, J. Cheng, H. Cruz, and P. Lukowicz, "Never skip leg day: A novel wearable approach to monitoring gym leg exercises," in *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2016, pp. 1–9.

[18] Y. Yoshida and E. Yuda, "Workout detection by wearable device data using machine learning," *Applied Sciences*, vol. 13, p. 4280, 03 2023.

[19] V. Nunavath, S. Johansen, T. Johannessen, L. Jiao, B. Hansen, S. Berntsen, and M. Goodwin, "Deep learning for classifying physical activities from accelerometer data," *Sensors*, vol. 21, p. 5564, 08 2021.

[20] M. Capecci, M. Ceravolo, F. orazio, F. Ferracuti, S. Iarlori, G. Lazzaro, S. Longhi, L. Romeo, and F. Verdini, "A tool for home-based rehabilitation allowing for clinical evaluation in a visual markerless scenario," vol. 2015, 08 2015.

[21] M. Capecci, M. G. Ceravolo, F. Ferracuti, S. Iarlori, S. Longhi, L. Romeo, S. N. Russi, and F. Verdini, "Accuracy evaluation of the kinect v2 sensor during dynamic movements in a rehabilitation scenario," in *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2016, pp. 5409–5412.

[22] Q.-T. Pham, V.-A. Nguyen, T.-T. Nguyen, D.-A. Nguyen, D.-G. Nguyen, D.-T. Pham, H. Vu, and T.-L. Le, "Automatic recognition and assessment of physical exercises from rgb images," in *2022 IEEE Ninth International Conference on Communications and Electronics (ICCE)*, 2022, pp. 349–354.

[23] Y. Wu, Q. Lin, M. Yang, J. Liu, J. Tian, D. Kapil, and L. Vanderbloemen, "A computer vision-based yoga pose grading approach using contrastive skeleton feature representations," *Healthcare*, vol. 10, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:245515661

[24] T. Rangari, S. Kumar, P. P. Roy, D. P. Dogra, and B.-G. Kim, "Video based exercise recognition and correct pose detection," *Multimed. Tools Appl.*, vol. 81, no. 21, pp. 30 267–30 282, Sep. 2022.

[25] Q.-T. Pham, D.-A. Nguyen, T.-T. Nguyen, T. N. Nguyen, D.-T. Nguyen, D.-T. Pham, T. H. Tran, T.-L. Le, and H. Vu, "A study on skeleton-based action recognition and its application to physical exercise recognition," in *Proceedings of the 11th International Symposium on Information and Communication Technology*, ser. SoICT '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 239–246. [Online]. Available: https://doi.org/10.1145/3568562.3568639

[26] S. Haque, A. S. A. RABBY, M. A. Laboni, N. Neehal, and S. A. Hossain, "Exnet: Deep neural network for exercise pose detection," in *International Conference on Recent Trends in Image Processing and Pattern Recognition*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:199583316

[27] A. Dey, A. Dutta, and S. Biswas, "Workoutnet: A deep learning model for the recognition of workout actions from still images," in *2023 3rd International Conference on Intelligent Technologies (CONIT)*, 2023, pp. 1–8.

[28] K. M. W. Soekarjo, D. Orth, E. Warmerdam, and J. van der Kamp, "Automatic classification of strike techniques using limb trajectory data," in *Machine Learning and Data Mining for Sports Analytics*, U. Brefeld, J. Davis, J. Van Haaren, and A. Zimmermann, Eds. Cham: Springer International Publishing, 2019, pp. 131–141.

[29] M. O'Reilly, D. Whelan, T. Ward, E. Delahunt, and B. Caulfield, "Technology in rehabilitation: Comparing personalised and global classification methodologies in evaluating the squat exercise with wearable imus," *Methods of Information in Medicine*, vol. 56, 06 2017.

[30] N. Rao, P. M. Surana, R. Ragesh, and G. Srinivasa, "Analysis of joints for tracking fitness and monitoring progress in physiotherapy," in *2019 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, 2019, pp. 181–185.

[31] H. Sarwat, H. Sarwat, M. I. Awad, and S. A. Maged, "Assessment of post-stroke patients using smartphones and gradient boosting," in *2020 15th International Conference on Computer Engineering and Systems (ICCES)*, 2020, pp. 1–6.

[32] R. Kianifar, A. Lee, S. Raina, and D. Kulić, "Automated assessment of dynamic knee valgus and risk of knee injury during the single leg squat," *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 5, pp. 1–13, 2017.

Registration for conference 1: I2CT 2024

VIDUSHI RESEARCH INNOVATION & PUBLICATION

Invoicing and payments powered by

Payment Receipt Transaction Reference: pay_NZFOiljZ4RlXay

This is a payment receipt for your transaction on IEEE 9th I2CT 2024

AMOUNT PAID **₹ 9,500.00**

ISSUED TO
amanrehman18@gmail.com
+918800286070

PAID ON
10 Feb 2024

Name of Registred Author
Aman Rehman

DESCRIPTION	UNIT PRICE	QTY	AMOUNT
NON IEEE MEMBER FULL PAPER REGISTRATION	₹ 9,500.00	1	₹ 9,500.00
	Total		**₹ 9,500.00**
	Amount Paid		₹ 9,500.00

No Refund Policy

Page 1 of 1

Certificate for conference 1: I2CT 2024

9th International Conference for Convergence in Technology (I2CT)

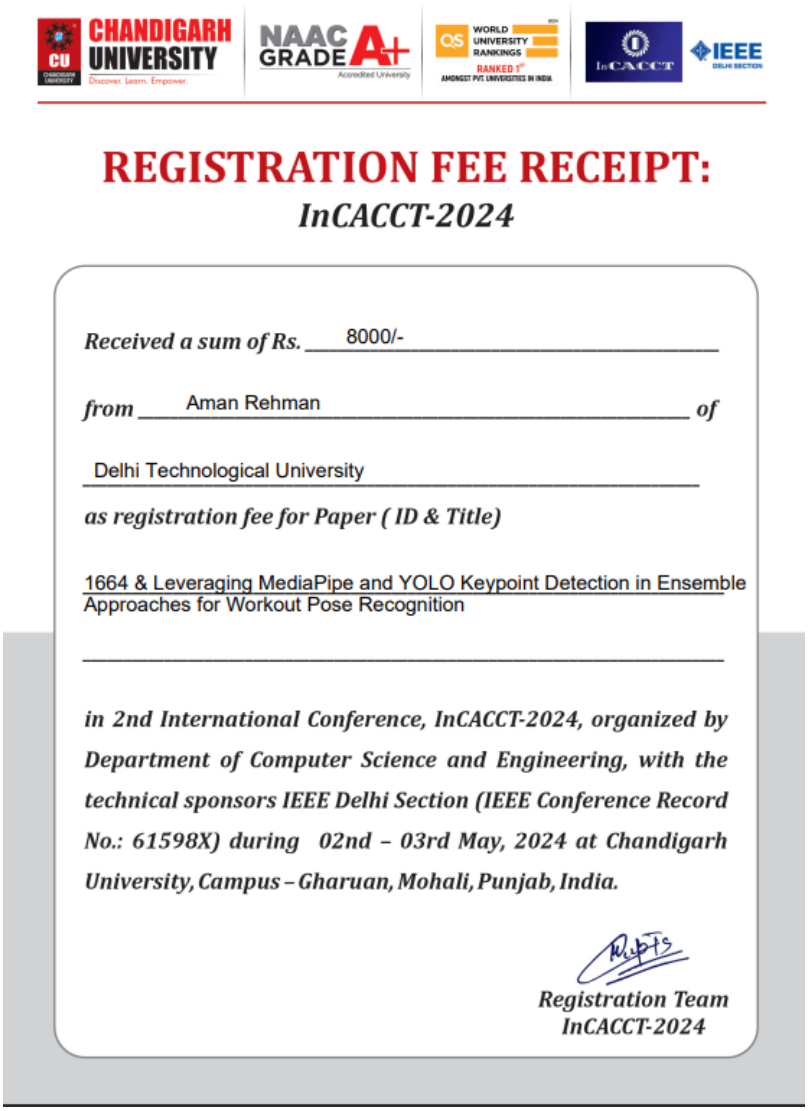
5th – 7th April 2024

Certificate

This is to certify that Dr./Prof./Mr./Ms. ***Aman Rehman*** *has presented paper entitled* ***A Comparison of Transfer Learning Inspired CNN Architectures for 2D Image Workout Recognition*** *in 9th International Conference for Convergence in Technology (I2CT) during 5th & 7th April 2024.*

Dr. Chanakya Kumar Jha
General Chair -I2CT

Registration for conference 2: InCACCT 2024



REGISTRATION FEE RECEIPT:

InCACCT-2024

Received a sum of Rs. 8000/-

from Aman Rehman *of*

Delhi Technological University

as registration fee for Paper (ID & Title)

1664 & Leveraging MediaPipe and YOLO Keypoint Detection in Ensemble Approaches for Workout Pose Recognition

in 2nd International Conference, InCACCT-2024, organized by Department of Computer Science and Engineering, with the technical sponsors IEEE Delhi Section (IEEE Conference Record No.: 61598X) during 02nd – 03rd May, 2024 at Chandigarh University, Campus – Gharuan, Mohali, Punjab, India.

Registration Team
InCACCT-2024

Certificate for conference 2: InCACCT 2024

Sr. No. _____

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

NAAC GRADE A+ Accredited University

QS WORLD UNIVERSITY RANKINGS 2024
RANKED 1ST AMONGST PVT. UNIVERSITIES IN INDIA

InCACCT

IEEE DELHI SECTION

Certificate of Participation

This is to certify that Prof./ Dr./ Mr./ Ms. Aman Rehman

of Delhi Technological University

participated/ presented a paper titled

Leveraging MediaPipe and YOLO Keypoint Detection in Ensemble Approaches for Workout Pose Recognition

2nd International Conference on **Advancement in Computation & Computer Technologies, 2024, (InCACCT-2024)**, organized by Department of Computer Science & Engineering, Technically sponsored by IEEE Delhi Section **(Record No.: 61598X)** during **02nd – 03rd May, 2024** at Chandigarh University, Gharuan, Mohali, Punjab, India.

Dr. Meenu Gupta
Convener & Conf. Organizing Chair
Chandigarh University, Punjab, India

Prof. (Dr.) Rakesh Kumar
Convener & Conf. Organizing Chair
AD-CSE, Chandigarh University, Punjab, India

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

PLAGIARISM VERIFICATION

Title of the Thesis: **Workout Pose Recognition**

Total Pages: **62**

Name of the Scholar: **Aman Rehman**

Supervisor: **Shailender Kumar**

Department: **Computer Science and Engineering**

This is to report that the above thesis was scanned for similarity detection. Process and outcome is given below:

Software used: **Turnitin** Similarity Index: **6%** Total Word Count: **15691**

Date: _____

Candidate's Signature **Signature of Supervisor**

PAPER NAME

Aman_Thesis2.pdf

WORD COUNT

15691 Words

CHARACTER COUNT

87538 Characters

PAGE COUNT

62 Pages

FILE SIZE

9.6MB

SUBMISSION DATE

May 23, 2024 8:26 PM GMT+5:30

REPORT DATE

May 23, 2024 8:27 PM GMT+5:30

6% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

- 5% Internet database
- 2% Publications database
- Crossref database
- Crossref Posted Content database
- 5% Submitted Works database

Excluded from Similarity Report

- Bibliographic material
- Quoted material
- Cited material
- Small Matches (Less then 8 words)