INFINITY YOGA TUTOR : YOGA POSTURE DETECTION AND CORRECTION SYSTEM

Fazil Rishan
Department of Software Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
it17098342@my.sliit.lk

Binali De Silva
Department of Software Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
it16029200@my.sliit.lk

Sasmini Alawathugoda
Department of Software Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
it17033442@my.sliit.lk

Shakeel Nijabdeen
Department of Software Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
it16174504@my.sliit.lk

Lakmal Rupasinghe
Department of Information Systems Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
lakmal.r@sliit.lk

Chethana Liyanapathirana
Department of Information Systems Engineering
Sri Lanka Institute of Information Technology
Malabe, Sri Lanka
chethana.l@sliit.lk

***Abstract*— Popularity of yoga is increasing daily. The reason for this is the physical, mental and spiritual benefits that could be obtained by practicing yoga. Many are following this trend and practicing yoga without the training of an expert practitioner. However, following yoga in an improper way or without a proper guidance will lead to bad health issues such as strokes, nerve damage etc. So, following proper yoga postures is an important factor to be considered. In this proposed system, the system is able to identify poses performed by the user and also guide the user visually. This process is required to be completed in real-time in order to be more interactive with the user. In this paper, the yoga posture detection was done in a vision-based approach. The Infinity Yoga Tutor application is able to capture user movements using the mobile camera, which is then streamed at a resolution of 1280 x 720 at 30 frames per second to the detection system. The system consists of two main modules, a pose estimation module which uses OpenPose to identify 25 keypoints in the human body, using the BODY_25 dataset, and a pose detection module which consists of a Deep Learning model, that uses time-distributed Convolutional Neural Networks, Long Short Term Memory and SoftMax regression in order to analyze and predict user pose or asana using a sequence of frames. This module was trained to classify 6 different asanas and the selected model which uses OpenPose for pose estimation has an accuracy of 99.91%. Finally, the system notifies the users on their performance visually in the user interface of the Mobile application.**

Keywords*— *Human Activity Recognition, Yoga Posture, OpenPose, Mask RCNN, LSTM, CNN

I. INTRODUCTION

In today's world, people are occupied with their schedules of hectic work. Due to those busy time schedules, most of the people are facing many health issues. According to the University of Rochester Medical Centre [1], top ten medical issues are physical activity and nutrition, overweight and obesity, tobacco substance abuse, HIV/AIDS, mental health, injury and violence, environmental quality, immunization and access to health care.

Therefore, people are in need of a healthy lifestyle. A healthy lifestyle consists of healthy food, healthy physical activities, weight management and stress management etc. It is feasible for people to maintain a healthy life easily by following healthy physical activities. Doing exercises are fallen under these healthy physical activities. There are several types of exercises such as aerobics, strength building, balance training, cardio and yoga. People can do exercises by going to an instructor or a studio or by watching videos on exercises or with their own knowledge. Due to the lack of free time in their daily routine, most people prefer to do exercises on their own with the use of an instruction manual or guides that could be found online.

Although there are many advantages from exercises, improper exercises could lead to a hazardous lifestyle. Therefore, it is mandatory to have good guidance for people who are doing exercises on their own. A proper guidance will lead to gain many benefits from exercises and improve the health of a person.

Yoga is a collection of disciplines or practices that originated in ancient India. Yoga is one of the six philosophical traditions of Hindu Orthodox schools. In the Western world, the word "yoga" also means a modern form of Hatha yoga, yoga as exercise, which consists largely of the postures called “asanas”. Currently Yoga has become popular with people in the western world trying to get fit. Proper yoga postures will assist to build awareness, harmony and strength in both the mind and body. However, improper yoga postures will lead to various kinds of serious injuries such as strokes, and nerve damage. So following proper yoga postures is mandatory.

Infinity Yoga Tutor, which is a yoga posture detection and correction system, uses a mobile-based approach for correcting improper yoga postures of the people who are doing yoga with the knowledge they have and doing yoga by watching yoga videos or using yoga applications. Although there are some systems on yoga posture detection, there are no significant amount of systems for correcting improper yoga postures. This mobile based approach consists of both yoga pose detection and yoga pose correction abilities. Moreover, this system consists of giving visual instructions to the user in real time, which would help the user to maintain a proper asana throughout the practice. In addition, the system also allows the user to select one of three difficulty levels which has different levels of accuracy threshold, to help beginners as well as experienced yogis to perform proper asanas. All these features help to guide user to do yoga in a safe way and achieve the best results from yoga. The current development in technology helps and makes this cause much easier to succeed with numerous research already completed on Human Activity Recognition (HAR). In this work we focused on how

978-1-6654-1475-3/20/$31.00 ©2020IEEE

to use the above models not only to detect but to guide the user to successfully practice poses much closer to perfection.

Detecting human postures is a complicated task. A higher degree of accuracy as well as real-time inference is expected in most of the actual world application of human pose estimation today. However, with the aid of the popular keypoints detection libraries, the keypoints detection task has become easier in this proposed system. Since the accuracy of this system also depends on the pose estimation or the keypoints detection module, two different popular keypoints detection libraries are used to compare, contrast and identify the optimum library for this system. One of the selected libraries is OpenPose [2] which detects keypoints in each frame using part affinity fields and part confidence maps by following a greedy algorithm. The other method used for yoga pose detection is detecting keypoints with the aid of Mask RCNN [3] which is also known as Mask Region Based Convolutional Neural Network in order to detect human in the video while detecting keypoints in each video frame by applying Fully Convolutional Network (FCN) and Region Proposal Network (RPN).

The output obtained from the above keypoints detection modules are normalized and modelled to fit to the prediction model, which consists of Time-distributed Convolutional Neural Networks (CNNs) and LSTM layers with a dense layer of activation Softmax. This gives an output probability for 6 classes. This prediction result is streamed to the mobile application which is visually represented to the user and guided with guide video related to the predicted output and the accuracy of the user performed pose or asana.

II. Background

There are several ways of detecting human activities including yoga postures. In [4], the authors have proposed a new approach for yoga postures detection using low-resolution infrared sensors. In that paper the authors have used a deep convolutional neural network (DCNN) and a low-resolution infrared sensor based wireless sensor network (WSN). But in order to facilitate a wide range of audience, in this paper a smartphone-based approach was taken which uses the mobile camera or the web camera. Moreover, there is another issue in that research, that is they have not used people with different ranges of Body Mass Index (BMI) to validate the system. Therefore, the current dataset consists of a sample of different ranges of BMI.

In [5], the authors have introduced a system using Microsoft Kinect and it has captured various keypoints of the human body in real time. It is also very expensive when comparing to a mobile phone camera. Microsoft Kinect is also having security concerns. So, it is not that much suitable for a yoga posture detection system. The authors have proposed this system to recognize the yoga postures, but it did not guide the user to correct the improper yoga posture. Therefore, it is also to be addressed.

Santosh, Amitojdeep, Abhishek and Jagdish have presented a real time yoga posture recognition method using deep learning [6]. They have used OpenPose for keypoints detection and proceeded those keypoints to convolutional neural network to extract features. Then LSTM has taken part to identify the changes of the postures over the time. Finally, it has predicted the yoga posture. In our research, we have used two approaches including OpenPose and Mask RCNN to choose the best approach in order to increase the accuracy of our system.

In [7], the authors have proposed a virtual personal trainer as the purpose of providing real time action assessment and action guide during fitness time of the users. They have used Microsoft Kinect Sensor for the proposed method. But it is very expensive when compared to a mobile phone camera and consists of security concerns. The user actions were captured by Kinect and compared to standard actions to provide a fitness score to represent the performance of users in real time. Users could see skeletons of their own action and standard action on the screen. The standard action and the user action were both captured by the Kinect sensor. But there are some physical limitations of Kinect for Windows. They have proposed a three-stage comparison method between the user action and the standard action including action classification, dynamic space time warping and fitness score. Action classification was done by the Kinect SDK which is an Adaboost trigger that can be used to detect a discrete gesture. In dynamic space time warping stage, they have used extended dynamic space time warping algorithm to warp the user action with the standard action to search the optimal path by dynamic programming. In the last stage they have computed the fitness score frame by using frame in 3D Euclidean space.

In our system it avoids the need for Kinect or any other external hardware to define Yoga posture recognition which can be implemented on input from an android device camera and most importantly unlike most of other researches this system aims to guide and correct the user in real time with aid of a skeleton. To the best of our knowledge, this is the first research that enables the user to select user levels. This component makes it possible to the user to set the accuracy levels which will adjust the posture detection preciseness level before the online yoga session. This system will calibrate the posture preciseness and provide real-time guidance based on the severity levels; Beginner, Intermediate and Expert which will facilitate all users in stages.

In [8], the authors have proposed a system called Match Pose which can be used to compare user's real time pose with a selected pose. They have used the PoseNet algorithm for real-time pose estimations of users. They have used pose comparison algorithms to compare and check whether user's real-time poses are imitated successfully. Proposed system allowed the user to select only an image which a user wishes to imitate. The real-time poses of the user were then captured through webcam and processed using human pose estimation algorithm. The selected image from the database was also processed using the same algorithm. Yoga poses involving finger positions cannot be compared in the system.

In [9], the authors have proposed a method that jointly models human pose estimation and tracking in a single formulation. They have presented body joint detections in a video by a spatio-temporal graph and solved an integer linear program to partition the graph into sub-graphs that correspond to plausible body pose trajectories for person.

The authors have solved the association of each person across the video and the pose estimation together. They have created recent methods for human pose estimation in images that build a spatial graph based on joint proposals to estimate the pose of a person in an image. They have casted the problem as an optimization of a densely connected spatiotemporal graph connecting body joint candidates spatially and temporally. They have proposed a dataset called PoseTrack, which provides annotations to quantitatively evaluate human pose estimation and tracking. The dataset provides detailed and dense annotations for each person in each video.

As stated by Patrik, Shrey and Hamed, real-world scenarios are often challenging because of occlusion, illumination variation, target deformation, background clutter and variation in proportions [10], [11]. A few past studies were done on single imagery recognition, where a CNN was used to predict or identify the object, but such studies would give very low accuracy levels on videos as the prediction has to depend on previous frames. For these types of problems Recurrent Neural Networks (RNN) were recommended [12]. In order to overcome occlusion problems researchers have also come up with recognizing occlusion and then changing results depending on the occlusion output which also uses a type of RNN known as Long Short-Term Memory (LSTM) [13].

The case of this research is such that it is required to receive and give output in real-time. So, the data obtained from the previous components should be evaluated with least possible delay. Furthermore, the above occlusion model is CPU intensive if a real-time feedback is required. Since this is a consumer-oriented product or a system, and to support that statement, our model or system requires the least usable resources that is readily available by a general consumer.

III. Methodology

The proposed Yoga Posture Detection and Correction System mainly consists of four components. They are,

- Keypoints Detection using OpenPose
- Keypoints Detection using Mask RCNN
- Higher Probability Prediction & Comparison
- Android Trainer Application

The overall workflow of the system is as follows. The user's movements are captured and streamed in real-time to the system using a media streaming server. Then the system detects the joints or the keypoints of the user with the use of a pose estimation library, either OpenPose or MaskRCNN. After keypoints are detected by the pose estimation methods, those are sent to the yoga pose detection module. Using the keypoints data obtained from the previous components it was possible to predict the user's yoga pose before the user reaches the final phase of the asana, using the movements that's required to be done to reach that stage, as the current model is able to predict users pose with only 30 frames of data, which is equivalent to a second. Then the result and accuracy values are sent to the android application using a data channel (Fig. 1).

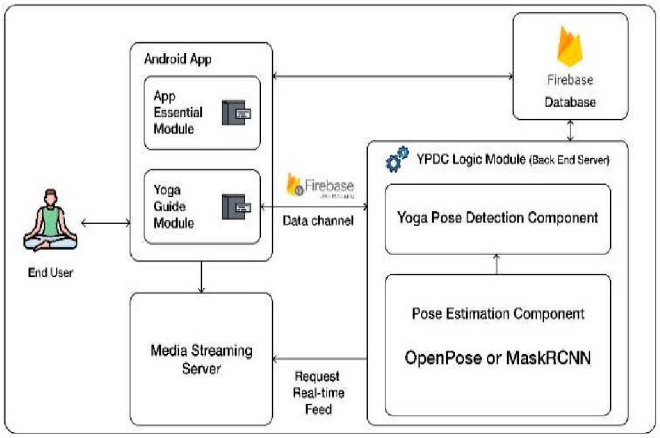

Fig. 1. System overview diagram.

A. Dataset

There is a publicly available dataset for yoga postures [14]. This dataset contains six yoga postures, which are Bhujangasana (Cobra Pose), Padmasana (Lotus Pose), Shavasana (Corpse Pose), Tadasana (Mountain Pose), Trikonasana (Traingle Pose) and Vrikshasana (Tree Pose). This dataset contains yoga postures as videos which are prepared using fifteen individuals; ten males, five females and all the videos are in 1366 x 768 resolution at 30 frames per second using a common web camera from a computer. The videos in this dataset has been recorded at a general lighting setting, with natural light illuminating from windows and top light on the performer, which could be a common setting in an everyday household. Furthermore, the videos were recorded at a distance of 4 meters from the camera. There are total of eighty-eight videos in the dataset. In order to efficiently stream, the videos were reduced to a resolution of 1280 x 720. Total duration of the dataset obtained is of 1 hour and 6 minutes which is approximately 118,950 frames. Each video has an average duration of 45 seconds.

B. Keypoints Detection using OpenPose

Initially, an incoming video stream was fed to the OpenPose through the media server. OpenPose library can detect 25 keypoints in the body including ankles, ears, elbows, eyes, hips, knees, nose, neck, shoulders, wrists. The incoming streams were sent to a baseline network to extract the feature maps. Then those feature maps were directed to part confidence maps and part affinity fields. After that, the output from both part confidence maps and part affinity fields were followed by a greedy algorithm in order to generate the output. The output generated for each frame of the stream was obtained in JSON format. This JSON output was contained body part locations of the persons detected in each frame. Finally, the output of was sent to the higher probability prediction and comparison component.

C. Keypoints Detection using Mask RCNN

At first an incoming video feed was added as the input in order to do human detection and keypoints detection using Mask RCNN. Next the input video stream was added to CNN for feature extraction to generate feature maps and these feature maps were added to a Regional Proposal Network (RPN) to generate region proposals and to predict existence of a human in that region. Then, RoI (Region of Interest) pooling layer was applied to convert all the regions into the same shape and to properly align extracted features with the input. Then these properly aligned regions were passed through a fully connected network (FCN), to generate bounding box for each region that contained a human. The human bounding boxes obtained by the human detection were fed into a stacked hourglass network with spatial transformer network to generate pose proposals. The generated pose proposals were refined by pose non-maximum suppression to obtain the estimated human poses. Finally, the output displaying keypoints was generated and the output results corresponding to each frame of a video was obtained in JSON format. Then output results of human detection and keypoints detection were passed to the pose detection module.

D. Higher Probability Prediction & Comparison

The foremost objective for this component will be to use the available "Joints JSON" data obtained from the pose estimation module to predict the pose done by the end-user. A

feedback of how close the user is to the original asana needs to be addressed to the user. This needs to be done with the minimum amount of delay to achieve a smooth user experience.

1) Models

There are two main types of deep learning models which are created; a model which supports all keypoints of the body (Single Model) and individual models created for 3 groups of keypoints in the body by splitting the joints/keypoints as Face (Eyes + Ears + Nose), Torso (Arms + Neck) and Waist Down (Hips + Legs) . The reason for segregation is to support abnormal body proportions of people with disabilities, therefore the human body is split into three to get a far more accurate result that has proven to output a closer to ground truth result [15].

Since this research is also used to compare and find the optimal keypoints detection module suitable for this system, a model is created for each keypoints detection algorithm. Both OpenPose and Mask RCNN were trained on the same dataset and has a similar architecture, but with different configurations. The optimum pose estimation is selected and used in the final system.

Each model uses a combination of time distributed CNNs and LSTM with dense layer of SoftMax activation as shown in Fig. 2. The above different types of models are compared with each other to find an efficient and accurate model which suits the current system.

The use of CNNs and LSTM both together is because, CNNs are proved to work well with pattern recognition [16] and are strongly capable of extracting spatial features which are shift and space invariant, therefore CNNs are used here to recognize patterns and extract features of keypoint positions, while LSTM is used here to understand a pattern between the change of frames. The prediction is obtained as a probability of the given classes by the softmax layer. The model is configured to output a probability prediction for 30 frames, which is a second of input.

Tensorflow's Keras Sequential API is used to program all the models. The model created for OpenPose has an input shape of 30 x 25 x 2, which is 30 time-steps or frames, 25 keypoints with X and Y coordinates. The model created for Mask RCNN has an input shape of 30 x 17 x 2, since Mask RCNN outputs only 17 keypoints. For feature extraction in both models 1D CNN layer with 32 filters of kernel size of, applied with ReLu activation which also helps to speed up the training process. The output is then sent through a dropout layer to prevent overfitting before flattening and passed through an LSTM layer. In summary, CNN is used here to identify spatial features and LSTM to identify both spatial and temporal relationships. The output from the final LSTM layer is passed to a dense layer with a SoftMax activation configured with 6 units to output a probability of corresponding yoga poses in terms of cross-entropy.

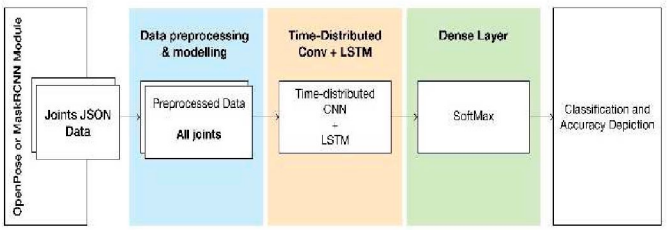

Fig. 2. High level system architecture for higher probability prediction and comparison component.

2) Training and Results

The data obtained after preprocessing was split into 3 groups, 70%, 10% and 20% for training, validation and testing respectively. Each model was being trained with 100 epochs on a system with AMD Ryzen 5 3550H, 16GB RAM and 4GB of Nvidia GeForce GTX 1650, which in average took 2 minutes for a single epoch during training. But most models converged early and stopped at an average epoch of 60. This can be seen in Fig.3 of OpenPose's single model which finished at 64 epochs on early stopping callback of patience for 10 consequent epochs which had no improvement in validation accuracy.

It can be observed from Table 1, that the model created for Mask RCNN keypoints has performed better than other models with an accuracy on train data of 99.85% and 99.96% accuracy on test data. Although the model trained with keypoints obtained from Mask RCNN performed well on test data, it was later observed that there were more false positives and true negatives when implemented on real-time feed while OpenPose's single model was able to perform efficiently with the real-time feed. Model created for the keypoints obtained from OpenPose for Tri-body split solution has an accuracy between 95% and 99.7% for test data, but the delay in predicting is much higher than the single model, therefore this is not an optimum solution. Single model trained with OpenPose's keypoints data was selected as the optimum solution and was integrated to the system. The selected model was able to achieve an accuracy of 99.87% on training data and 99.91% on test data or unseen data. The model has performed close to perfect in most of the poses, except for Tadasana and Vrikshana. This can be seen in the normalized confusion matrix (Fig. 4) which denotes that Vrikshana has only 99.5% accuracy and Tadasana with an accuracy of 99.94%. The output of pose detection on real-time feed can be seen in Fig. 5.

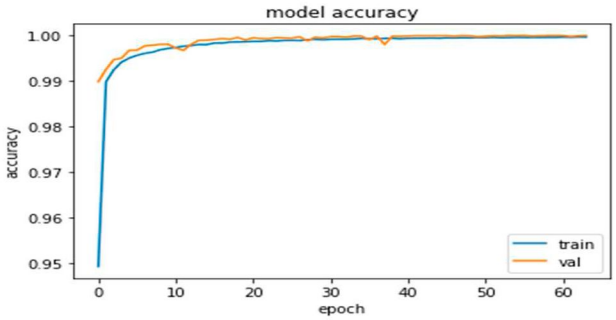

Fig. 3. Change in model accuracy (Accuracy vs Epoch), Blue : Train Data, Amber : Validation Data.

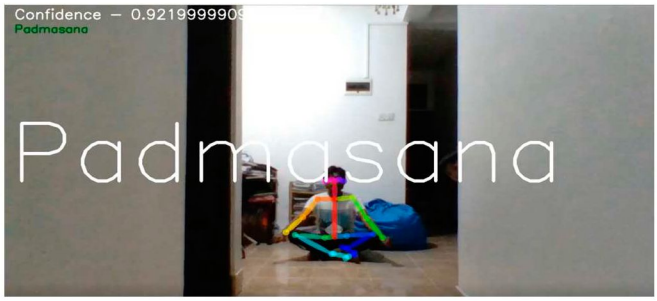

Fig. 4. Test output of yoga pose detection on real-time feed, detecting padmasana.

Fig. 5. Confusion matrix with normalization.

TABLE I. RESULT SUMMARY OF ALL MODELS

Pose Estimation Module	Model	Accuracy on Train Data	Accuracy on Test Data	More than 15% of False Positives and True Negatives
OpenPose	Controlled Model – Stacked LSTM	99.01%	98%	Yes
	Conv +LSTM Single Model	99.87%	99.91%	No
	Conv +LSTM Body Split Model	95 – 99.6%	95-99.7%	No
MaskRCNN	Conv +LSTM Single Model	99.85%	99.96%	Yes
	Conv +LSTM Body Split Model	93.2 – 94.5%	94.3-96.2%	Yes

E. *Android Trainer Application*

Reading user postures from end user device camera and streaming captured video frames to the pose prediction server is a vital area in this study. The efficiency of video streaming comes into the focus when evaluating the usability of the yoga posture detection system. It is because the usability affects streaming performance [17]. Choosing a streaming protocol is a difficult task, which depends on the type of information to transfer. Communication must be made using a protocol established by a set of rules describing how data is transmitted over the network and divided into various sections, such as headers, data, authentication and error handling. A streaming protocol can thus be regarded as a communication protocol where the data being transmitted are media data. The most important consideration for this purpose is the guarantee of low latency and reliable transmissions with occasional packet loss. Compression can be accomplished by reducing video redundancies [18]. A media streaming protocol is defined which takes into account the packet structure and the algorithms used to deliver real-time media to a network.

To achieve best outcome H.264 encoder is used for compression of video captured from the camera. H.264 is a video compression technology or codec developed jointly by the International Telecommunications Union (as H.264) and the Moving Picture Experts Group of the International Organization for Standardizations/International for Electrotechnical Commission. H.264 is a new video encoding standard that has become more evolved methods of compression other than the standard MPEG-2 compression and one of the advantages of H.264 is its high rate of compression, it is around 1.5 times more powerful than encoding with MPEG-2 [19].

1) RTSP library for streaming

The Infinity Yoga Tutor mobile application contains a video encoding library that compresses video into H.264 encoded bits. The transport protocol used to send the live streams is RTSP (Real Time Streaming Protocol) which establishes a streaming session between android device and media server for video transmission. The media server serves as a server-side streaming framework that receives incoming streams from an android device and launches it to the pose estimation module. Having dedicated media server for live streaming allows full customizability and provides features that makes it optimal for self-managed infrastructure for streaming and monitoring capabilities.

The concept in RTSP is that for multimedia servers, it functions as 'network remote control'. RTSP has been designed to be on top of RTP to manage and deliver content in real time [20]. To accomplish this, An RTSP library is imported into the android project. Media streaming server is utilized to transcode the incoming streams and transmit the video frames to the pose estimation module that resides in the backend server. Packaging of streams can be changed to a format that fits the requirement of the pose estimation module which gives flexibility.

The RTSP library that is used in the android project is libstreaming which is an open source API that allows to stream the camera of an android powered device using RTP (Real-time Transport Protocol) over UDP (User Datagram Protocol). Wowza media server serves as a streaming framework that will receive incoming streams from an android device and launches it to pose prediction model and a data channel is established to receive the posture guidance data back to the android device.

2) Data channel for communication

In order to receive the posture feedback in real-time, a data channel is established between the android trainer application and the pose detection module. Rendering the received response feedback back on the end user's device should be in sync with the real-time posture that the user is performing is another major concern in this study, because delay in feedback hinders the main purpose and the usability of the application which is to give feedback instantly for asanas as user is performing yoga session. To achieve this, Firebase Cloud Messaging (FCM) is implemented in the system to accomplish real-time bi-directional communication between android trainer application and the pose prediction model.

Initially upon user installation of the Infinity Yoga Tutor application, the device registers the user application instance with the firebase cloud messaging server which will assign a unique token to that particular instance and at the same time

when the new token is generated, it also registers the token along with the user ID in the backend server that host the pose prediction model in order to identify users device uniquely and communicate with the help of the token and user ID. The backend server stores users FCM token along with the user ID on Firestore.

When the server receives incoming live streams from the user, the server will identify the user and begin to process yoga live feed and send feedback messages that includes posture classification, and the accuracy through the FCM data channel using the particular user's token. In order to manifest a human body skeleton structure in the android device screen to indicate the user about the posture correctness, Tensorflow lite with PoseNet is imported into android project to achieve this.

IV. Results and Discussions

Multiple models were trained with keypoints data obtained from OpenPose and Mask RCNN pose estimation modules to find the optimum module for this system, and it was observed that the model trained with keypoints obtained from OpenPose was able to perform well overall with the least amount of delay when tested with real-time feed. It was found that the model had a difficulty differentiating Tadasana and Vrikshana at certain times, this could be because the movements leading to both the poses are almost similar. As mentioned above the model was able to predict with an accuracy of 99.87% for train data and 99.91% on unseen test data. Although individual models created for head, torso and legs had impressive accuracy values, the time taken to predict was higher than the other models.

V. Conclusion

In order to support a healthy lifestyle for the community of yoga practitioners, we have proposed a system which is able to guide them to practice yoga more accurately in real time. This proposed system is capable of identifying yoga postures using an android mobile. When the user practices yoga, a live mobile feed is streamed to the server which has multiple modules interconnected to predict and output the asana and the accuracy. A video guide of the predicted pose is shown to the user in real time helping the user reach the stance properly.

The dataset was trained on multiple models for keypoints obtained from two different pose estimation modules, OpenPose and Mask RCNN in order identify the most suitable pose estimation module for the current system, when OpenPose was finally selected as the appropriate pose estimation module which was later integrated to the system. Prediction module which consists of a time-distributed CNN layer that extracts spatial features and a LSTM layer which identifies spatial changes with temporal changes. The output from above layers are passed to a dense layer with an activation of SoftMax which gives a probability prediction for each class or pose.

The selected model which uses OpenPose to detect keypoints, achieved an accuracy of 99.87% for dataset used for training and 99.91% for unseen test data. The integrated model also performed exceptionally well when tested in real time for all poses except for Vrikshana which had a few false positives with Tadasana.

References

[1] University of Rochester Medical Center. (2020). *Top 10 Most Common Health Issues* [Online]. Available: https://www.urmc.rochester.edu/senior-health/common-issues/top-ten.aspx

[2] Z. Cao, T. Simon, S. Wei and Y. Sheikh, "Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 1302-1310.

[3] K. He, G. Gkioxari, P. Dollár and R. Girshick, "Mask R-CNN," 2017 IEEE International Conference on Computer Vision (ICCV), Venice, 2017, pp. 2980-2988.

[4] M. Gochoo *et al.*, "Novel IoT-Based Privacy-Preserving Yoga Posture Recognition System Using Low-Resolution Infrared Sensors and Deep Learning," in *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 7192-7200, Aug. 2019.

[5] M. U. Islam, H. Mahmud, F. B. Ashraf, I. Hossain and M. K. Hasan, "Yoga posture recognition by detecting human joint points in real time using microsoft kinect," 2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC), Dhaka, 2017, pp. 668-673. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8289047&isnumber=828 8891

[6] S.K. Yadav, A. Singh, A. Gupta, et al., "Real-time Yoga recognition using deep learning," *Neural Computing & Applications*, vol. 31, pp. 9349–9361, May 2019.

[7] M. C. Thar, K. Z. N. Winn, N. Funabiki, "A Proposal of Yoga Pose Assessment Method Using Pose Detection for Self-Learning, 2019 International Conference on Advanced Information Technologies (ICAIT), Yangon, Myanmar, 2019, pp. 137-142. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8920892&ampisnumber=892 0837

[8] P.K. Borkar, M.M. Pulinthitha, A. Pansare, 2019, "Match Pose – A System for Comparing Poses," International Journal of Engineering Research and Technology(IJERT), vol. 08, issue 10, Oct 2019.

[9] U. Iqbal, A. Milan, J. Gall, "PoseTrack: Joint Multi-person Pose Estimation and Tracking," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 4654-4663.

[10] P. Kulkarni, S. Mohan, S. Rogers, H. Tabkhi, "Key-Track: A Lightweight Scalable LSTM-based Pedestrian Tracker for Surveillance Systems," 2019, pp. 208–219.

[11] G. Chevalie, "LSTMs for Human Activity Recognition," 2016. [Online]. Available: https://github.com/guillaume-chevalier/LSTM-Human-Activity-Recognition

[12] H. Coskun, "Human Pose Estimation with CNNs and LSTMs," 2016.

[13] S.-H. Zhang et al., "Pose2Seg: Detection Free Human Instance Segmentation," 2018.

[14] Yoga - CEERI, "Yoga Vid Collected", March 2019. [Online].Available: https://archive.org/details/YogaVidCollected

[15] B. Gil, "Single Pose Comparison," Becoming Human: Artificial Intelligence Magazine, 2017. [Online]. Available:https://becominghuman.ai/single-pose-comparison-a-fun-application-using-human-pose-estimation-part-2-4fd16a8bf0d3

[16] S. E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2016-Decem, pp. 4724–4732, 2016.

[17] A. Hussain, E. Mkpojiogu and F. Kamal, Mobile Video Streaming Applications: A Systematic Review of Test Metrics in Usability Evaluation, vol. 8, no. 10, 2016.

[18] I. Ullah Khan and M. Ansari, "Performance Analysis of H.264 Video Coding Standard and H.263 Video Coding Standard", VSRD-TNTJ, vol. 2, no. 1, pp. 8-14, 2011.

[19] S. Choudhary and P. Varshney, "A Study of Digital Video Compression Techniques", vol. 5, no. 4, 2016.

[20] L. C Reddy and P. Hiremath, "RTSP Audio and Video Streaming for QoS in Wireless Mobile Devices", IJCSNS International Journal of Computer Science and Network Security, vol. 8, no. 1, 2008.