A Project Report

on

Yoga Script & InstructionsJet

Submitted in partial fulfillment of the requirement for the award of the degree of

B.Tech CSE

GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of
Mr. V. Arul
Assistant Professor

Submitted By

Isha Gupta - 19SCSE1010499

Nikhil Kumar Jha - 19SCSE1010895

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA, INDIA

December, 2021

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

We hereby certify that the work which is being presented in the project, entitled **"Yoga Script & InstructionsJet"** in partial fulfillment of the requirements for the award of the Bachelor Of Technology submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of August, 2021 to December and 2021, under the supervision of Mr. V. Arul... Assistant Professor, Department of Computer Science and Engineering, School of Computing Science and Engineering , Galgotias University, Greater Noida.

The matter presented in the project has not been submitted by us for the award of any other degree of this or any other places.

Isha Gupta, 19SCSE1010499
Nikhil Kumar Jha, 19SCSE1010895

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Mr. V. Arul
Assistant Professor

CERTIFICATE

The Final Project Viva-Voce examination of Isha Gupta: 19SCSE1010499, Nikhil Kumar Jha: 19SCSE1010895 has been held on 24th December, 2021 and his/her work is recommended for the award of Bachelor Of Technology.

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Project Coordinator **Signature of Dean**

Date: 24 December, 2021
Place: Greater Noida

Acknowledgement

Place: Greater Noida

Date: 24 December, 2021

On this great occasion of accomplishment of our project on Yoga Script & InstructionsJet, we would like to sincerely express our gratitude to Mr. V. Arul, who has been supported through the completion of this project.

We would also be thankful to our reviewer Mrs. Indra Kumari of Galgotias University for providing all the required facilities in completion of this project.

Finally, as one of the team members, I would like to appreciate all my group members for their support and coordination. I hope we will achieve more in our future endeavors.

Isha Gupta
Nikhil Kumar Jha

Abstract

Pandemic has taken a toll on the health and lifestyle of the people. They are not motivated enough to go to the gym or yoga classes. Exercising is boring and tiresome for beginners so they quit in most cases. Hence, we wish to solve this problem with our web app 'Yogic'. It deals with the localization of human joints in an image to form a skeletal representation.

To automatically identify a person’s pose in an image is a difficult task as it depends on several viewpoints such as scale and resolution of the image, illumination shift, background noise, apparel variations, environment, and interaction of humans with the environment.

This project lays the foundation for building such a system by discussing various machine learning and deep learning approaches to accurately classify yoga poses in real-time. The project also discusses various pose estimation and keypoint detection methods in detail and explains different deep learning models used for pose classification.

Yogic is a Yoga Pose/Exercise Estimation App that will detect humans and their yoga pose/exercise in real-time. It will have guided Gym workouts as well as yoga poses that’ll be developed with the help of ML.

It will be easily accessible from anywhere. It will keep track of your previous workouts and will show real-time statistics in the dashboard.

Index

List of Table

Chapter-1
Introduction

1.1 Introduction

Pose estimation is a domain of computer vision that involves identifying individual components that build up a human body. Human pose estimation is a tricky problem in the discipline of computer vision. It deals with the localization of human joints in an image to form a skeletal representation. To automatically identify a person's pose in an image is a difficult task as it depends on several viewpoints such as scale and resolution of the image, illumination shift, background noise, apparel variations, environment, and interaction of humans with the environment. An application of pose estimation which has intrigued many researchers in this department is exercise, physical activity and fitness.

A class of exercise with elaborate postures is Yoga which is an age-old exercise that started in India but is now globally famous for many of the perks in spiritual, physical and mental domains. The problem with yoga however is that, just like any other exercise, it is of utmost concern to practice it correctly as any incorrect posture during a yoga session can be unproductive and perhaps harmful. This requires the need for an instructor to supervise and guide the session and correct the individual's posture. Since not all users have access to an instructor or resources, an AI-based application might come in handy to identify yoga poses and provide personalized feedback to help people enhance their form and profile.

In recent years, human pose estimation has profited greatly from deep learning and huge gains in performance have been achieved. Deep learning propositions provide a more straightforward way of mapping the structure instead of having to deal with the dependencies between structures manually.

This project focuses on exploring the different approaches for yoga pose classification and seeks to attain insight into applying deep learning to yoga pose classification in real-time. This project focuses on exploring the different approaches for yoga pose classification and seeks to attain insight into the following: What is pose estimation? What is deep learning? How can deep learning be applied to yoga pose classification in real-time? This project uses references from conference proceedings, published papers, technical reports and journals. The first section of the project talks about the history and importance of yoga. The second section talks about pose estimation and explains different types of pose estimation methods in detail and goes one level deeper to explain discriminative methods – learning based (deep learning) and exemplar. Different pose extraction methods are then discussed along with deep learning based models - Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).

The application of pose estimation for yoga is challenging as it involves complex configuration of postures. Furthermore, some state-of-the-art methods fail to perform well when the asana involves horizontal body posture or when both the legs overlap each other. Hence, the need to develop a robust model which can help popularize self-instructed yoga systems arises.

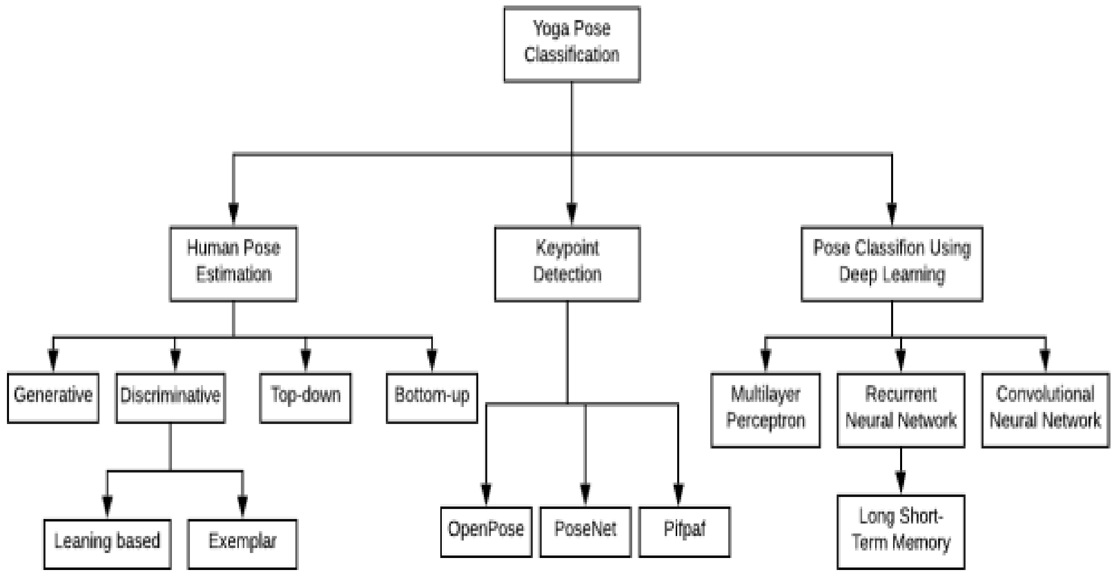


Fig. 1. Conceptual outline of topics

This project focuses on exploring the different approaches for yoga pose classification and seeks to attain insight into the following: What is pose estimation? What is deep learning? How can deep learning be applied to yoga pose classification in real-time? This project uses references from conference proceedings, published papers, technical reports and journals. Fig. 1 gives a graphical overview of topics this paper covers.

1.2 History

Humans are prone to musculoskeletal disorders with aging and accidents. In order to prevent this some form of physical exercise is needed. Yoga, which is a physical and spiritual exercise, has gained tremendous significance in the community of medical researchers. Yoga has the ability to completely cure diseases without any medicines and improve physical and mental health. A vast body of literature on the medical applications of yoga has been generated which includes positive body image intervention, cardiac rehabilitation, mental illness etc. Yoga comprises various asanas which represent physical static postures. The application of pose estimation for yoga is challenging as it involves complex configuration of postures. Furthermore, some state-of-the-art methods fail to perform well when the asana involves horizontal body posture or when both the legs overlap each other. Hence, the need to develop a robust model which can help popularize self-instructed yoga systems arises.

1.3 Human Pose Estimation

Human posture recognition has made huge advancements in the past years. It has evolved from 2D to 3D pose estimation and from single person to multi person pose estimation. uses pose estimation to build a machine learning application that helps detect shoplifters whereas uses a single RGB camera to capture 3D poses of multiple people in real-time. Human pose estimation algorithms can be widely organized in two ways. Algorithms prototyping estimation of human poses as a geometric calculation are classified as generative methods while algorithms modeling human pose estimation as an image processing problem are classified as discriminative methods. Another way of classifying these algorithms is based on their method of working. Algorithms starting from a higher-level generalization and moving down are called top-down methods, whereas algorithms that start with pixels and move upwards are called bottom-up methods.

A. Generative

Generative procedures provide a technique to predict the features from a given pose hypothesis. They start with initializing the posture of the human body and project it to the image plane. Adjustments are made to make the projected image and current image observations compliant. Generative based approaches offer easy generalization due to less constraint of a training pose dataset. However, due to the high dimensional projection space search, this method is not considered computationally feasible, and is thus slower as compared to discriminative methods. describes a generative Bayesian method to track 3D segmented human body figures in videos. This is a probabilistic method which consists of a generative model for image appearance, an initial probability distribution over joint angles and pose that represents movement of humans and a robust likelihood function. Even though the method is able to track humans in unknown complicated backgrounds, it faces the risk of eventually losing track of the object.

B. Generative

Contrary to generative methods, discriminative methods start with the evidence of the image and learn a technique to model the relationship between the human poses and evidence on the basis of training data. Model testing in discriminative methods is a lot faster as opposed to generative methods due to the search in a constrained space as opposed to a high dimensional feature space [7]. [11] explores a discriminative based learning method to obtain 3D human pose from silhouettes. This approach does not require a body model explicitly nor any prior labeled parts of the body in the image. It restores the pose using non-linear regression based on the shape descriptor vectors fetched automatically from silhouettes of images. It uses Relevance Vector Machine (RVM) regressors and damped least squares for regression [11]. The method, though increasing the accuracy by three times, is not accurate enough, as there are some instances of incorrect poses and results showing significant temporal jitter. Discriminative methods are further categorized into learning methods and exemplar methods.

1.2 Problem Formulation

Problem Statement is in the covid time most of the users have to use multiple apps to track their fitness activity, do workout, exercises & meal planning. People lose interest after a while as they find it very cumbersome to use different apps and keep track of it. Solution to the above is a web application that will help users develop the habit of practicing daily to track their progress (activity, meal, nutrition intake, workout. etc.) and stay motivated especially when they are slacking.

1.3 Tools & Technologies Used

The web application uses html, css and css framework for the front-end part of the application. For the backend and server side development we are using Nodejs. To train data models for several poses python is used. Open CV for detection of body movement. Tensorflow for developing ML models in Nodejs. For deploying and testing projects we are using Heroku. For code collaboration across team members we are using Git & Github.

Chapter-2
Literature Survey/Project Design

2.1 Literature Survey

Human Posture recognition has made huge progress in recent years. It has evolved from 2D to 3D pose estimation and also from a single person to a multi-person pose estimation. Some projects use pose estimation to develop a machine learning application that helps catch shoplifters and other projects use a single RGB camera to capture 3D poses of various people in real-time.

Human pose estimation algorithms can be broadly classified in two ways. Generative methods revolve around algorithms prototyping estimation of human poses as a geometric calculation whereas Discriminative methods involve algorithms modelling human pose estimation as an image processing problem.

Method of working is another basis to classify these algorithms. Top-down and bottom-up are included in this classification. Top-down methods include algorithms that start from a higher-level generalization and move down eventually, whereas bottom-up methods include algorithms that start with pixels and move upwards.

Key point detection involves concurrently identifying people and localizing their key points. Key points are equivalent to interest points. They are spatial locations or points in the image that define what stands out in the image and hence what point is interesting. They are invariant to image rotation, interpretation, shrinkage, distortion, illumination and so on. Key point detection methods include OpenPose, PoseNet and Pifpaf.

OpenPose is a deep learning framework that involves multi-person real-time key point detection. It began a revolution in the field of pose estimation. OpenPose applies CNN based structure to distinguish facial, hand and foot key points of a human body from individual images. RGB cameras are utilized to identify human body joints.

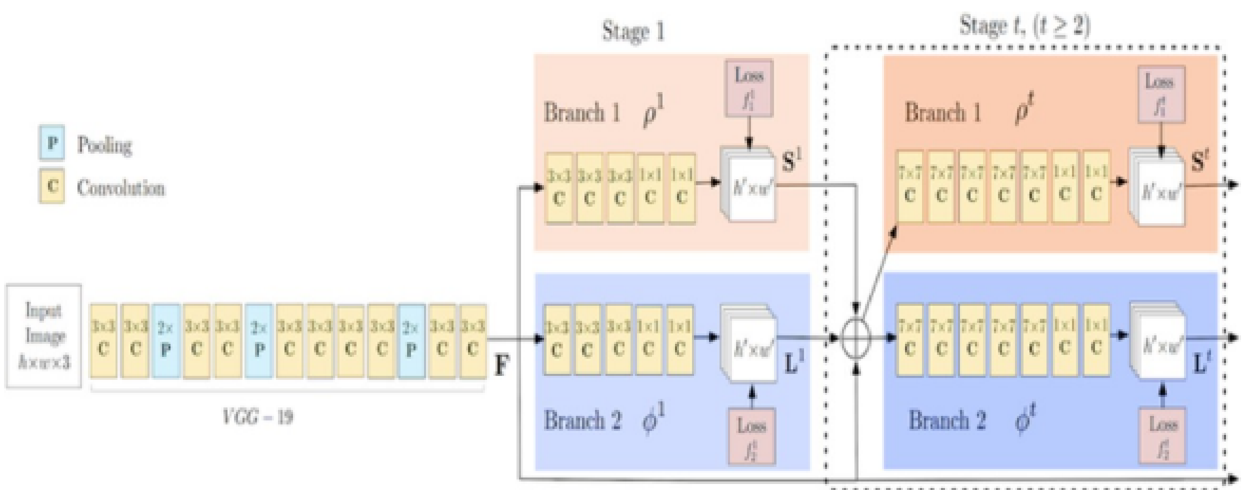
OpenPose key points include eyes, ears, nose, neck, shoulders, elbows, wrists, hips, knees, and ankles. Because of this characteristic, it finds its utilization in a diversity of applications ranging from sports, inspection, motion detection to yoga pose recognition.

PoseNet is another deep learning framework that is similar to OpenPose. It is utilized in the identification of human poses in images or video sequences by identifying joint locations in a human body. These joint locations or key points are indexed by a confidence score called "Part ID" the value of which lies in the range of 0.0 and 1.0. The PoseNet model's execution modifies depending on the design and output stride. The PoseNet model is independent of the dimension of the image.

In addition, it offers incredible intelligence and the ability to easily alter graphics using Qt illustration devices. No exit into the field of estimating the human pose has yet been created. Estimating human posture is a confusing point in the Computer Vision realm, as it involves the formation of a human skeletal figure based on a fully characterized structure of the human body. exercises certainly play an important role in human life.

However, improper performance of the exercises can lead to injuries, which require adequate training and support during the performance of the action. But waiting for an instructor at all times while doing an exercise is too much to ask. This requires a self-learning model of yoga that accurately informs the user when exercise is being performed correctly.

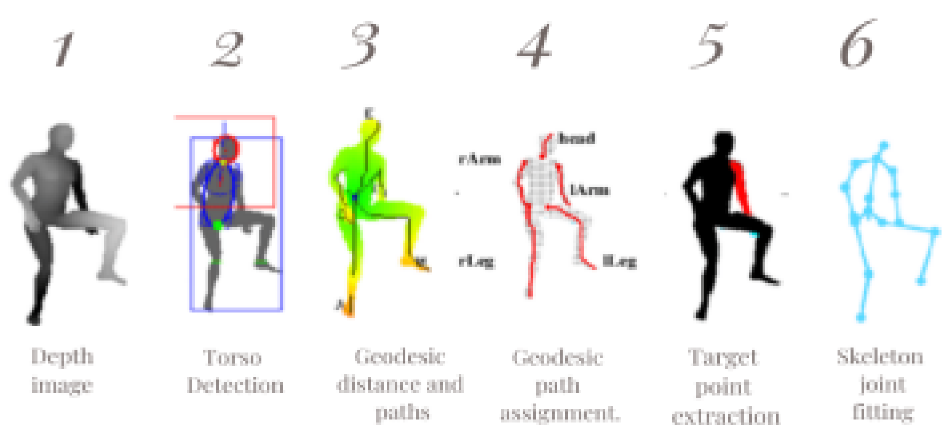
So when building a model for the yoga posture data set, we found that adding more features to the data set improved the overall accuracy of the model. Furthermore, the use of OpenPose, PyQt, and a neural network model in the dataset with the 3D values is considered more effective than the 2D values.



2.2 Project Design

The depth of any object is shown in the depth image. This means that we can figure out the approximate distance of any object by looking at its depth image. The farther objects are shown in a darker shade and the objects near the camera are shown in a lighter shade. After the depth image, we will identify the torso in the human figure and try to locate a centre point on the image.

Now we find the geodesic distance and path in the human figure. Geodesic distance is the shortest distance between two vertices of any figure. Here, the geodesic distance will show the shortest distance between the head, the palm of the right hand, left hand and right and left toes. The values of the geodesic distance will now help to identify the geodesic paths which in turn will help us to identify the head, hands and legs of the human figure.



Flowchart for
Human Pose Estimation

We can now label them as head, rArm, rLeg, lArm and lLeg respectively, for the head, right arm, right leg, left arm and left leg.

We can use this information to find our target point in the human figure. By joining the arms, legs, head and torso with single straight lines, we can create a skeletal joint fitting image of the person performing yoga.

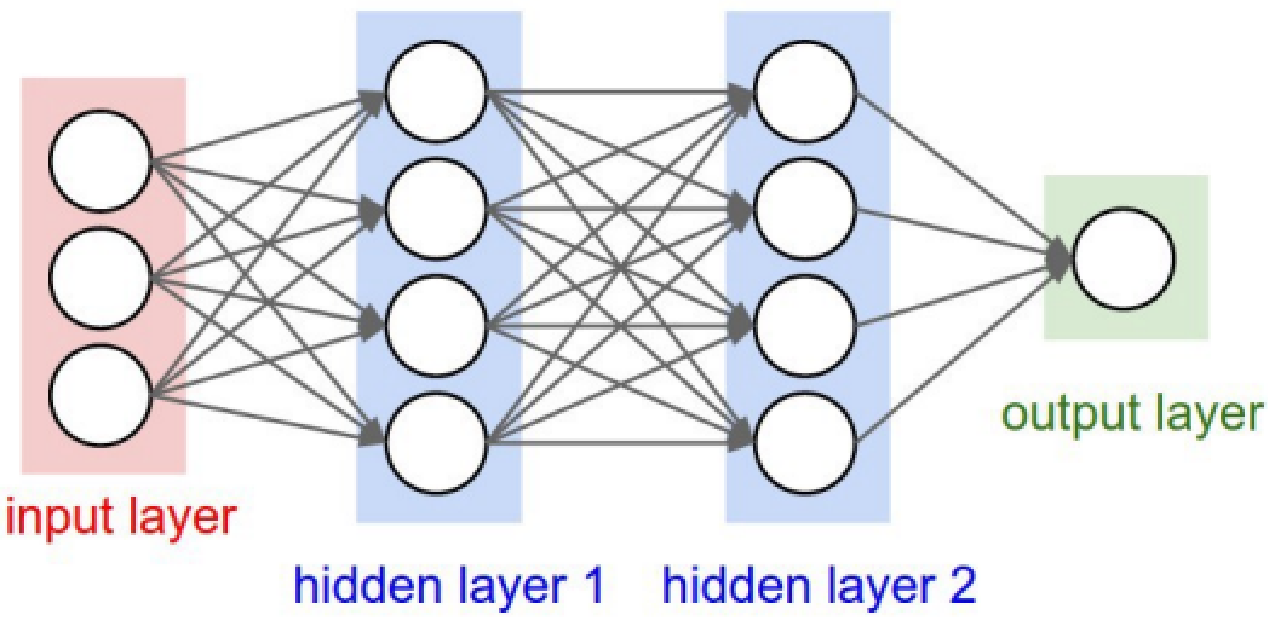
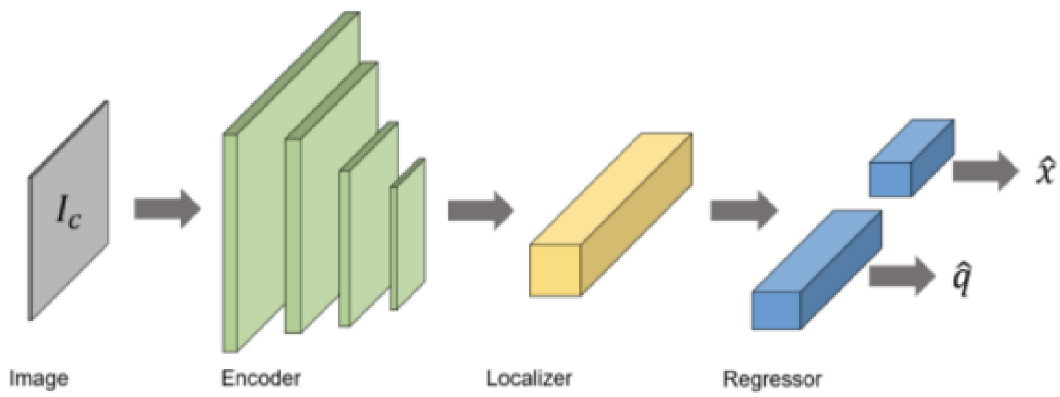


Fig. 5. Schematic diagram of multilayer perceptron

2.3 System Architecture Of PoseNet:



In PoseNet, the softmax layer is replaced by a sequence of fully connected layers.

A high level architecture of PoseNet is shown in Fig. 3 [31]. The first component in the architecture is an encoder which is responsible for generating the encoding vector v, a 1024-dimensional vector that is an encoded representation of the features of the input image. The second component is the localizer which generates vector u which denotes localization features. The last component is a regressor which consists of two connected layers that are used to regress the final pose.

In the case of keypoints, CNN extracts features from 2D coordinates of the OpenPose keypoints using the same convolutional filter technique explained above. Based on the filter size, the convolutional filter slides to the next set of input. After the convolution, an activation function Rectified Linear Unit (ReLU) is generally applied to add nonlinearity in the CNN, as the real world data is mostly nonlinear and the convolution operation by itself is linear. Tanh and sigmoid are other activation functions, but ReLU is mostly used because of its better performance.

CNNs consist of a minimum of one convolutional layer which is the first layer and is responsible for feature extraction from the image. CNNs perform feature extraction using convolutional filters on the input and analyzing some parts of the input at a given time before sending the output to the subsequent layer. The convolutional layer, through the use of convolutional filters, generates what is called a feature map. With the help of a pooling layer, the dimensionality is reduced, which reduces the training time and prevents overfitting. The most common pooling layer used is max pooling, which takes the maximum value in the pooling window.

Chapter-3
Functionality/Working Of Project

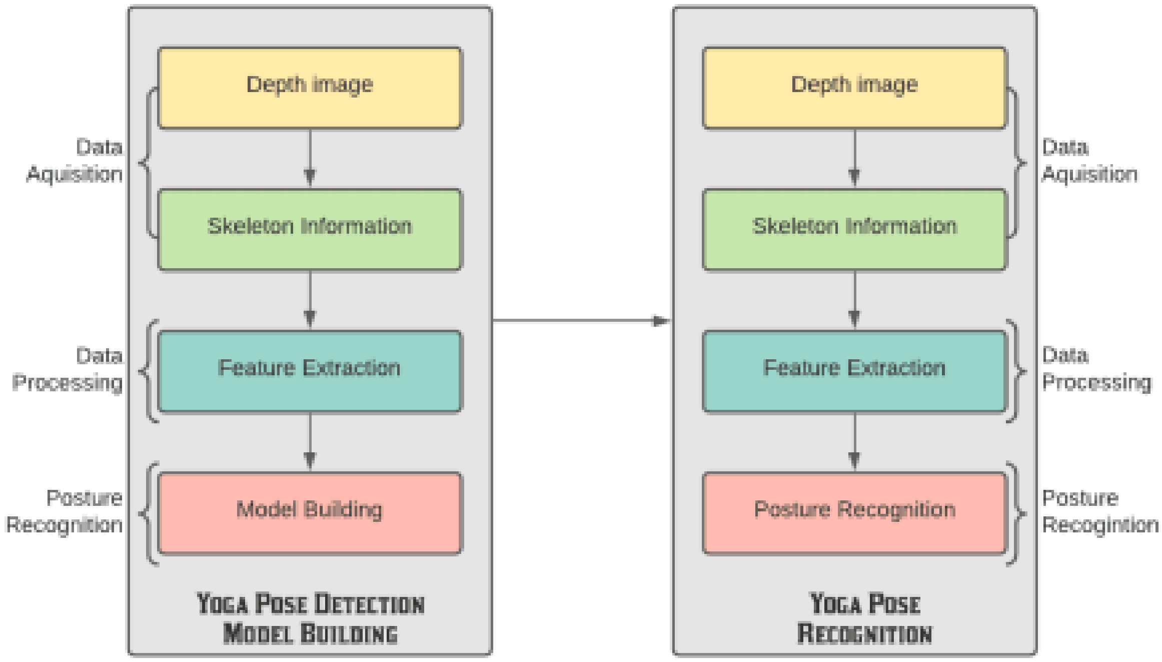
3.1 Module Description

There are two main components to take care of while making a yoga pose detector, namely:

1) Model building for yoga pose detection.

2) Yoga pose recognition. The model building consists of 2 parts: training data and testing data. The data is acquired from the depth image and skeleton image. Then feature extraction is done for data processing and the model is built with the help of training data provided.

Then it is tested on testing data. If we get the desired result, we can deploy the model, else we have to train the data again till the desired result is achieved.

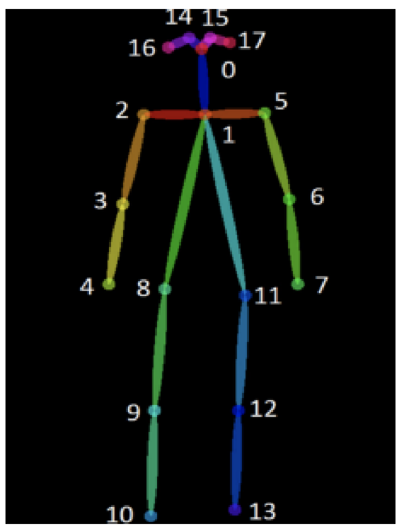


We will collect a dataset on different yoga poses and train our model using deep learning to identify the correct yoga posture and/or find what the user is doing wrong in the yoga pose and show how that can be corrected. The video of the person performing yoga is captured by the webcam using OpenCV.

Then the depth image is obtained which helps in finding the geodesic path which in turn aids us to find the skeleton fitting image. We use this image to identify the key points in the figure. We can now compare these key points to the ones in our sample dataset to see if they are matching or not. In case, the pose does not match, we will compare the key points where it is showing a mismatch and guide the user to correct his/her pose accordingly.

On the dashboard, the users can see their active time, calories burnt and their streak number, which is the number of consecutive days a user is active on the platform. The streaks are displayed only after a minimum of 3 days of active time and badges will be given to users on achieving milestones like 1 month, 3 months, 6 months, etc. which they can share on their social media handles.

As we probably know, Python's strength lies in exploratory data science and visualization using various devices, e.g. To reach all these Python devices directly from the application, PyQt is used. PyQt allows us to create complex information based applications and smart dashboards. Although it is possible to put matplotlib plots in PyQt, the experience does not feel native. With this in mind, PyQt is used instead of matplotlib for simple and deeply intuitive graphs. This is based on PyQ5's local QGraphicsScene and provides better drawing execution, especially for a live feed.

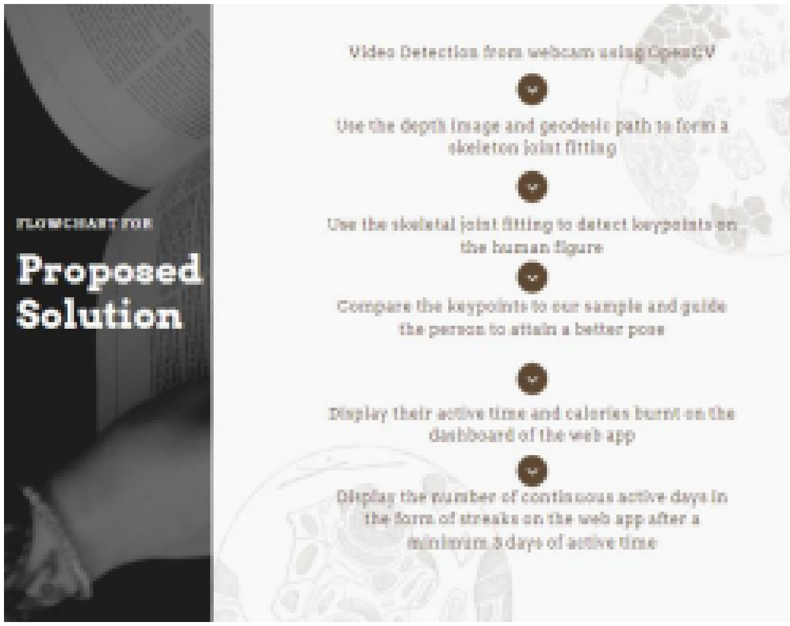


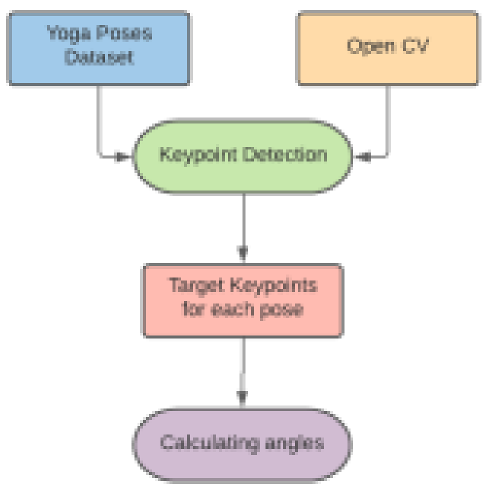
A default SVM has been trained on the training data with the radial basis function (rbf) kernel. Rbf is the default and most popular kernel which is a gaussian radial basis function. It provides more flexibility as compared to other kernels, linear and polynomial. The value of the soft margin parameter C is 1 and the decision function is one-vs-rest. The key points captured using OpenPose are used as features to SVM. These 18 keypoints are represented by X and Y coordinates which makes the total number of features as 36 (18 * 2). The data is reshaped to make the number of samples equal.

The first step in preprocessing the data is extracting key points of poses in video frames using the OpenPose library. For recorded videos, pose extraction is done offline whereas for realtime, it is done online wherein key points identified from the inputs to the camera are supplied to the model. OpenPose is run on each frame of the video and the corresponding output of each frame is stored in JSON format. This JSON data includes the locations of body parts of each person identified in the video frame. Default setting of OpenPose has been used for extracting pose key points for ideal performance.

The number of frames varied from 60,20,20 split at the video level. This was because of the difference in duration of videos.

3.2 Flowchart of Proposed Solution





Pre-processing part of the system

A deep learning model for classifying yoga poses can be built where the initial keypoint extraction of the human joint locations is done using OpenPose. The model can incorporate feature extraction capabilities of CNN along with context retention abilities of LSTM to effectively classify yoga poses in prerecorded videos and also in real time [2].

This model can be thought of as a hybrid model. We also plan to experiment with basic CNN networks and compare the performance with the hybrid model. Kinect sensors could be a way to perform human pose estimation, but it accounts for additional equipment and specialized hardware and the performance is not always good in different surroundings.

Machine learning models, although not widely used for human pose estimation, will be explored for comparison with the deep learning models. The evaluation of the yoga pose classification system will be done by using classification scores, confusion matrix and evaluations by people. The system will predict the yoga pose sequence being performed by the user in real-time and we can examine if the prediction made.

A lot of work has been done in the past in building systems that are automated or semiautomated which help to analyze exercise and sports activities such as swimming [24], basketball [25] etc. Patil et al. [26], proposed a system for identifying yoga posture differences between an expert and a practitioner using speeded up robust features (SURF) which uses information of image contours. However, describing and comparing the postures almost by using only the contour information is not sufficient. A system for yoga training has been proposed by Luo et al. [27] which consists of inertial measurement units (IMUs) and tactors. But this can be uncomfortable to the user and at the same time affect the natural yoga pose. [28] presented a system for yoga pose detection for six poses using Adaboost classifier and Kinect sensors and achieved an accuracy of 94.8%. However, they have used a depth sensor based camera that may not be always accessible to users. Another system for yoga pose correction using Kinect has been presented by [29] which takes into account three yoga poses, warrior III, downward dog and tree pose. However, their results are not very impressive, and their accuracy score is only 82.84%. The traditional method of skeletonization has now been replaced by deep learning-based methods. Deep learning is a promising domain where a lot of research is being done, enabling us to analyze tremendous data in a scalable manner.

As compared to traditional machine learning models where feature extraction and engineering is a must, deep learning eliminates the necessity to do so by understanding complex patterns in the data and extracting features on its own. The system is correct. The results will also be compared to existing methods. The model loss curve represents a minimizing score (loss), which means that a lower score results in better model performance.

The model accuracy curve represents a maximizing score (accuracy), which means that a higher score denotes better performance of the model. A good fitting model loss curve is one in which the training and validation loss decrease and reach a point of stability and have minimal gap between the final loss values. On the other hand, a good fitting model accuracy curve is one in which the training and validation accuracy increase and become stable and there is a minimum gap between the final accuracy values.

Chapter-4

Result & Discussion

4.1 Training Setup:

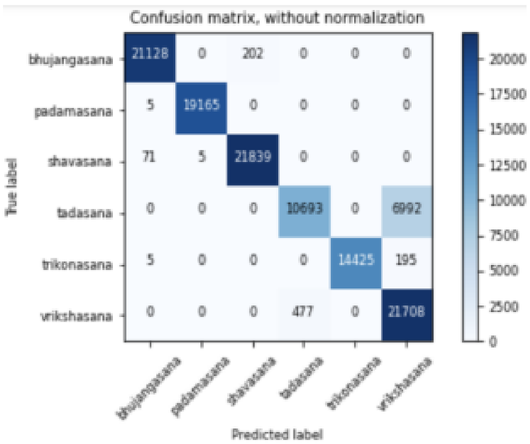
The models are built using Python libraries such as TensorFlow - Keras (Theano backend), OpenPose, NumPy, Scikit Learn on a system with NVIDIA Tesla 1080 GPU having 4 GB memory.

4.2 Results:

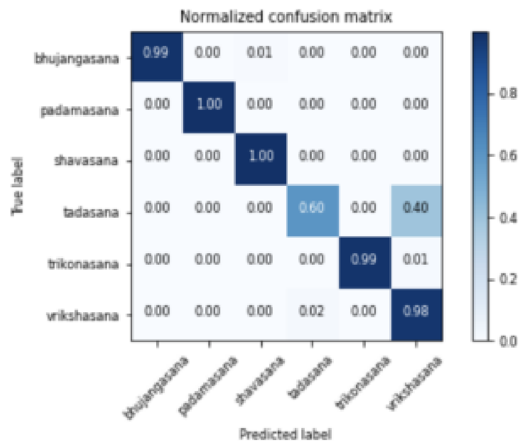
Train accuracy: 0.9953

Validation accuracy: 0.9762

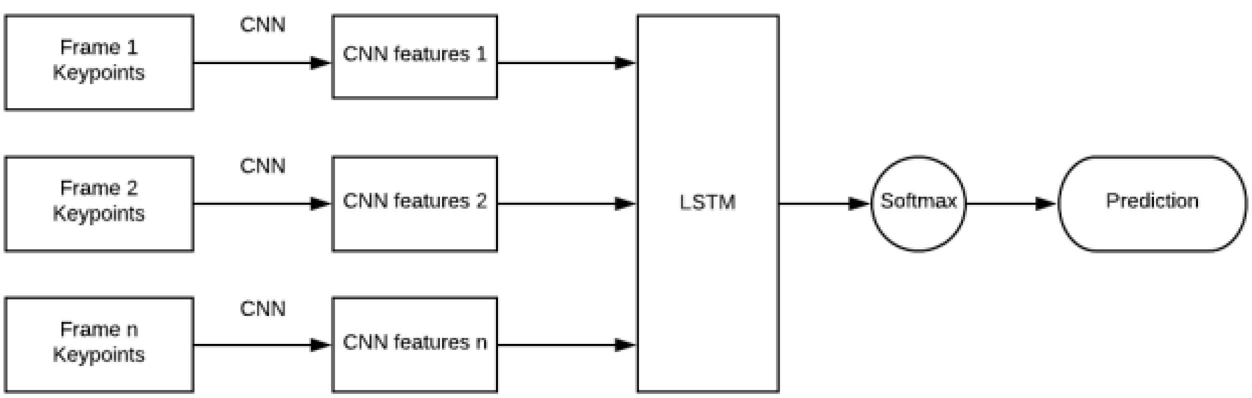
Test accuracy: 0.9319



Confusion matrix, without normalization
True label
Predicted label
bhujangasana
padamasana
shavasana
tadasana
trikonasana
vrikshasana
21128 0 202 0 0 0
5 19165 0 0 0 0
71 5 21839 0 0 0
0 0 0 10693 0 6992
5 0 0 0 14425 195
0 0 0 477 0 21708
20000
17500
15000
12500
10000
7500
5000
2500
0



Normalized confusion matrix
True label
Predicted label
bhujangasana
padamasana
shavasana
tadasana
trikonasana
vrikshasana
0.99 0.00 0.01 0.00 0.00 0.00
0.00 1.00 0.00 0.00 0.00 0.00
0.00 0.00 1.00 0.00 0.00 0.00
0.00 0.00 0.00 0.60 0.00 0.40
0.00 0.00 0.00 0.00 0.99 0.01
0.00 0.00 0.00 0.02 0.00 0.98
0.8
0.6
0.4
0.2
0.0



Frame 1 Keypoints
Frame 2 Keypoints
Frame n Keypoints
CNN
CNN
CNN
CNN features 1
CNN features 2
CNN features n
LSTM
Softmax
Prediction

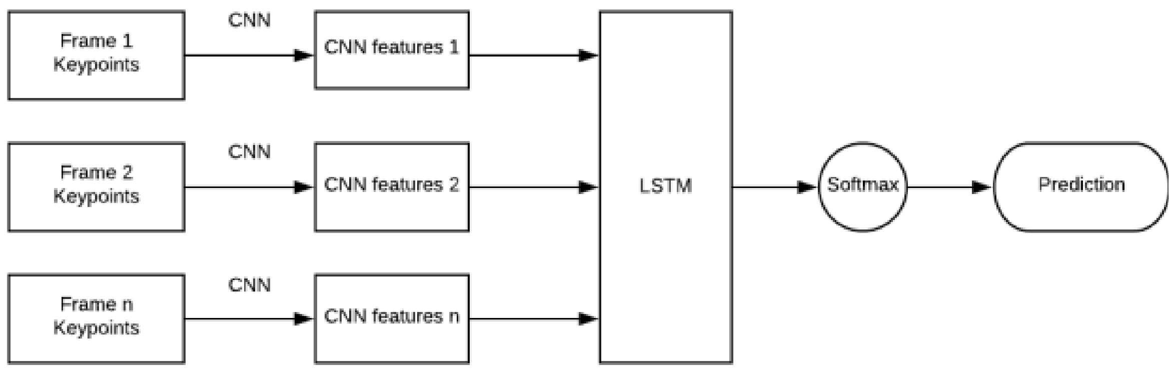
4.3 Analysis

The training accuracy of the model is pretty high at 0.99. There is a slight decrease in the validation and test accuracy, but the results are still good. We can see in the confusion matrix that most classes are classified correctly except for tadasana (mountain pose). Out of 17,685 frames for tadasana, 6992 have been misclassified as vrikshasana (tree pose) and similarly there is some incorrect classification for vrikshasana. This could be because of the similarity in the poses as both of them require a standing position and also the initial pose formation is similar.

Model Architecture Summary:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
time_distributed_1 (TimeDist (None, 45, 18, 16)        112
_________________________________________________________________
time_distributed_2 (TimeDist (None, 45, 18, 16)        64
_________________________________________________________________
time_distributed_3 (TimeDist (None, 45, 18, 16)        0
_________________________________________________________________
batch_normalization_2 (Batch (None, 45, 18, 16)        64
_________________________________________________________________
time_distributed_4 (TimeDist (None, 45, 288)           0
_________________________________________________________________
lstm_1 (LSTM)                (None, 45, 20)            24720
_________________________________________________________________
time_distributed_5 (TimeDist (None, 45, 6)             126
=================================================================
Total params: 25,086
Trainable params: 25,022
Non-trainable params: 64
_________________________________________________________________
```

4.4 Model Architectural Diagram:



SVM is a supervised machine learning model that is inherently a two-class classifier. However, as most problems involve multiple classes, a multiclass SVM is often used. A multiclass SVM forms multiple two class classifiers and differentiates the classifiers on the basis of the distinct label vs. the rest (one-vs-rest or one-vs-all) or between each pair of classes (one-vs-one). SVM performs the classification by creating a hyperplane in such a way that separation between classes is as wide as possible.

Chapter-5
Conclusion & Future Scope

5.1 Conclusion

Human pose estimation has been studied extensively over the past years. As compared to other computer vision problems, human pose estimation is different as it has to localize and assemble human body parts on the basis of an already defined structure of the human body. Application of pose estimation in fitness and sports can help prevent injuries and improve the performance of people's workout which suggests, yoga self-instruction systems carry the potential to make yoga popular along with making sure it is performed in the right manner. Deep learning methods are promising because of the vast research being done in this field. The use of hybrid CNN and LSTM models on OpenPose data is seen to be highly effective and classifies all the 6 yoga poses perfectly. A basic CNN and SVM also perform well beyond our expectations. Performance of SVM proves that ML algorithms can also be used for pose estimation or activity recognition problems. Also, SVM is much lighter and less complex when compared to a neural network and requires less training time.

5.2 Future Works

The proposed models currently classify only 6 yoga asanas. There are a number of yoga asanas, and hence creating a pose estimation model that can be successful for all the asanas is a challenging problem. The dataset can be expanded by adding more yoga poses performed by individuals not only in indoor settings but also outdoors. The performance of the models depends upon the quality of OpenPose pose estimation which may not perform well in cases of overlap between people or overlap between body parts. A portable device for self-training and real-time predictions can be implemented for this system. This work demonstrates activity recognition for practical applications. An approach comparable to this can be utilized for pose recognition in tasks such as sports, surveillance, healthcare etc. Multi-person pose estimation is a whole new problem in itself and has a lot of scope for research. There are a lot of scenarios where single person pose estimation would not suffice, for example pose estimation in crowded scenarios would have multiple persons which will involve tracking and identifying pose of each individual. A lot of factors such as background, lighting, overlapping figures etc. which have been discussed earlier in this survey would further make multi-person pose estimation challenging.

Chapter-6

References

6.1 References

[1] L. Sigal. “Human pose estimation”, Ency. of Comput. Vision, Springer 2011.

[2] S. Yadav, A. Singh, A. Gupta, and J. Raheja, “Real-time yoga recognition using deep learning”, Neural Comput. and Appl., May 2019. [Online]. Available: https://doi.org/10.1007/s00521-019-04232-7

[3] U. Rafi, B. Leibe, J.Gall, and I. Kostrikov, “An efficient convolutional network for human pose estimation”, British Mach. Vision Conf., 2016.

[4] S. Haque, A. Rabby, M. Laboni, N. Neehal, and S. Hossain, “ExNET: deep neural network for exercise pose detection”, Recent Trends in Image Process. and Pattern Recog., 2019.

[5] M. Islam, H. Mahmud, F. Ashraf, I. Hossain and M. Hasan, "Yoga posture recognition by detecting human joint points in real time using microsoft kinect", IEEE Region 10 Humanit. Tech. Conf., pp. 668-67, 2017.

[6] S. Patil, A. Pawar, and A. Peshave, “Yoga tutor: visualization and analysis using SURF algorithm”,Proc. IEEE Control Syst. Graduate Research Colloq.,pp. 43-46, 2011.

[7] W. Gong, X. Zhang, J. Gonzàlez, A. Sobral, T. Bouwmans, C. Tu, and H. Zahzah, “Human pose estimation from monocular images: a comprehensive survey”, Sensors, Basel, Switzerland, vol. 16, 2016.

[8] G. Ning, P. Liu, X. Fan and C. Zhan, “A top-down approach to articulated human pose estimation and tracking”, ECCV Workshops, 2018.

[9] A. Gupta, T. Chen, F. Chen, and D. Kimber, “Systems and methods for human body pose estimation”, U.S. patent, 7,925,081 B2, 2011. YOGA POSE CLASSIFICATION USING DEEP LEARNING 38

[10] H. Sidenbladh, M. Black, and D. Fleet, “Stochastic tracking of 3D human figures using 2D image motion”, Proc 6th European Conf. Computer Vision, 2000.

[11] A. Agarwal and B. Triggs, “3D human pose from silhouettes by relevance vector regression”, Intl Conf. on Computer Vision & Pattern Recogn.pp.882–888, 2004.

[12] M. Li, Z. Zhou, J. Li and X. Liu, “Bottom-up pose estimation of multiple person with bounding box constraint”, 24th Intl. Conf. Pattern Recogn.,2018.

[13] Z. Cao, T. Simon, S. Wei, and Y. Sheikh, “OpenPose: realtime multi-person 2D pose estimation using part affinity fields”, Proc. 30th IEEE Conf. Computer Vision and Pattern Recogn,2017.

[14] A. Kendall, M. Grimes, R. Cipolla, “PoseNet: a convolutional network for real-time 6- DOF camera relocalization”, IEEE Intl. Conf. Computer Vision, 2015.

[15] S. Kreiss, L. Bertoni, and A. Alahi, “PifPaf: composite fields for human pose estimation”, IEEE Conf. Computer Vision and Pattern Recogn, 2019.

[16] P. Dar, “AI guardman – a machine learning application that uses pose estimation to detect shoplifters”. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/06/ai-guardman-machine-learningapplication-estimates-poses-detect-shoplifters/

[17] D. Mehta, O. Sotnychenko, F. Mueller and W. Xu, “XNect: real-time multi-person 3D human pose estimation with a single RGB camera”, ECCV, 2019.

[18] A. Lai, B. Reddy and B. Vlijmen, “Yog.ai: deep learning for yoga”. [Online]htttp://cs230.stanford.edu/projects_winter_2019/reports/15813480.pdf

[19] M. Dantone, J. Gall, C. Leistner, “Human pose estimation using body parts dependent joint regressors”, Proc. IEEE Conf. Computer Vision Pattern Recogn., 2013.

[20] A. Mohanty, A. Ahmed, T. Goswami, “Robust pose recognition using deep learning”, Adv. in Intelligent Syst. and Comput, Singapore, pp 93-105, 2017.

[21] P. Szczuko, “Deep neural networks for human pose estimation from a very low resolution depth image”, Multimedia Tools and Appl, 2019.

[22] M. Chen, M. Low, “Recurrent human pose estimation”, [Online]. Available: https://web.stanford.edu/class/cs231a/prev_projects_2016/final%20(1).pdf

[23] K. Pothanaicker, “Human action recognition using CNN and LSTM-RNN with attention model”, Intl Journal of Innovative Tech. and Exploring Eng, 2019.

[24] N. Nordsborg, H. Espinosa, “Estimating energy expenditure during front crawl swimming using accelerometrics”, Procedia Eng., 2014.

[25] P. Pai, L. Changliao, K. Lin, “Analyzing basketball games by support vector machines with decision tree model”, Neural Comput. Appl., 2017.

[26] S. Patil, A. Pawar, A. Peshave, “Yoga tutor: visualization and analysis using SURF algorithm”, Proc. IEEE Control Syst. Grad. Research Colloquium, 2011.

[27] W. Wu, W. Yin, F. Guo, “Learning and self-instruction expert system for yoga”, Proc. Intl. Work Intelligent Syst. Appl, 2010.

[28] E. Trejo, P. Yuan, “Recognition of yoga poses through an interactive system with kinect device”, Intl. Conf. Robotics and Automation Science, 2018.

[29] H. Chen, Y. He, C. Chou, “Computer assisted self-training system for sports exercise using kinetics”, IEEE Intl. Conf. Multimedia and Expo Work, 2013.