International Journal of INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING Region-based Network for Yoga Pose Estimation with Discriminative Fine-Tuning Optimization

Shilpa Gite
Symbiosis International University
95 PUBLICATIONS **740** CITATIONS

Deepak T. Mane
Vishwakarma Institute of Technology
57 PUBLICATIONS **275** CITATIONS

Sunil Digamberrao Kale
Vishwakarma Institute of Information Technology, Pune.
25 PUBLICATIONS **81** CITATIONS

International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Region-based Network for Yoga Pose Estimation with Discriminative Fine-Tuning Optimization

Shilpa Gite[1,*], Deepak T. Mane[2], Vijay Mane[3], Sunil Kale[4], Prashant Dhotre[5]

Submitted: 26/05/2023 **Revised**: 07/07/2023 **Accepted**: 25/07/2023

Abstract: Pose estimation of human activity recognition has been a keen area of interest in augmented reality experiences, gaming and robotics, animations, behavioral analysis, and more. One such exciting variant of pose estimation in the field of health and science is yoga pose estimation. This paper explores yoga pose estimation using deep learning networks. The research aims to build a system for estimating 45 different complex yoga asanas from 11,000 images using deep learning algorithms. This system is built using a Region-based Convolutional Neural Network (RCNN) to estimate the joints in the body, followed by a Convolutional Neural Network (CNN) for classifying the poses. The model is trained using the Yoga-82 (hierarchically labeled) dataset, a new dataset with complex pose variations mainly designed for hierarchical labeling. Next, it highlights the pose estimation task through ResNet models followed by an optimization algorithm, which increases the accuracy by 10%. The resultant accuracy is 90.5% for the ResNet50 model. Finally, it provides a solution for overlapping yoga poses, multi-person, in-air, and non-conventional poses using a dense network of 17 critical points for analysis and prediction.

Keywords: *Yoga pose estimation, region-based network, CNN RCNN deep-learning; optimization*

1. Introduction

Pose estimation is an extensive application of computer vision that deals with analyzing the study of individual body parts that make up the posture through critical point data analysis [1] for different applications like fitness (professional trainer through artificial intelligence (AI) led instructor) [2], [3], physical therapy (posture correction based on mapping and correction of postures) [4], [5] video game or movie production with enriched visuals (based on mapping on avatars through infrared-IR sensors) and robotics (for flexible and smooth reflexes with minimal recalibration) [6], [7] Pose estimation application comprises of tracking changes in human posture and providing feedback in real-time [8].

Yoga pose estimation has been an extensive area of research in clinical applications [9], behavioral analysis, human pose co-estimation (PCE) and prototype pose characterization [10]. Many models like PoseNet [11], Open Pose [12] as well as OpenCV contour detection [13] have been curated and customized to build AI-based pose estimators for medical [14], [15] and fitness related applications [16]. Recent advances also involve pose estimation in 3D space using mediapipe [17]. Tensorflow MoveNet [18] has also paved the way for designing an animated AI pose trainer [19]. Despite the range of models that are available and proposed for pose estimation, the work on the variety of poses stays limited [20]. There are many instances like dog pose, cat pose where the key joint points are obscured in the pose image [21], in such a scenario, detection and prediction need to go hand in hand. The training of the dataset through the proposed Region Based Convolutional Neural Networks (RCNN) [22] model ensures that no such limitation is faced in the everyday yoga pose applications. Further optimization gives robustness to the proposed model [23].

Pose estimation from an image or video frame is a highly challenging task. It depends on the scale and resolution of the image and other aspects like lighting conditions, fluctuations, occultations, background conditions, and more [24].

The complexity increases when pose estimation is applied to fitness-related activities [25], it is mainly due to the wide variety and diversity of possible poses (e.g., thousands of yoga asanas), occlusions (e.g., obstruction of key-point locations due to varied poses), and different angles of appearance (front, back, side view)[26].

1Symbiosis Institute of Technology, SIT, Pune, Maharashtra, 412115; shilpa.gite@sitpune.edu.in
2,3Vishwakarma Institute of Technology, SIT, Pune, Maharashtra, 412115;
dtmane@gmail.com
vijay.mae@vit.edu
4Vishwakarma Institute of Information Technology, Pune-411048, Maharashtra, India
kalesunild@gmail.com
5MIT Art, Design and Technology University, Pune-412201,Maharashtra, India
prashantsdhotre@gmail.com
*Correspondence: Deepak Mane. Email: dtmane@gmail.com

Sometimes it also considers the color of the fitness gear to separate it from the environment or surroundings. Yoga pose estimation becomes an actual application in the study of pose estimation tasks for fitness.

Fig 1. Example body poses for commonly practiced yoga asanas.

Taking its origin from the Sanskrit word "yuji," yoga signifies the union of body and soul [27]. Yoga has been extensively studied as an art of healing for centuries[28]. If performing an incorrect stance reaps out all the medical benefits claimed in reputed health sciences studies [29], some commonly practiced yoga poses are shown in figure 1. To make this practice easier and accessible for all, computerized self-training systems are being developed with easy-to-follow animated tutorials. This, in turn, helps the user to improve simultaneously through guided feedback loops. This can be enabled through innovative AI based monitoring tools designed to map the needs of the user, along with easy-to-follow voice instructions and activity mapping.

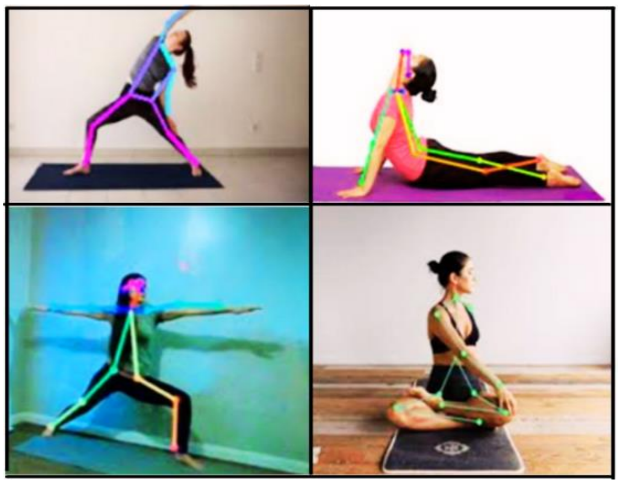


Fig 2. Implemented models on pose estimation for yoga activity.

As per the literature, it is found that some state-of-the-art techniques like posenet [30], Kinect [31], real-sense [32], [33] do not perform well when the posture is a horizontal body posture or when both the legs overlap each other. This creates the need for a better model, which works well on a generalized dataset. According to the position of relevant joints, the pose structure can be estimated using deep learning-based classification[34].The beginners can make comparisons with expert poses. The difference in angles of both poses can help beginners improve their posture by correcting it against the expert pose. This paper introduces the implementation of ResNet models with determined fine-tuning optimization for key-point detection and yoga pose estimation on one of the most challenging and diversified datasets, i.e., Yoga-82. This work has been done on a whole of 45 classes of yoga poses. The current limitation of yoga pose estimation research is the inability to work on extensive and complex datasets like Yoga-82 and provide a robust solution. As a result, this paper proposes an efficient solution to this problem with perfect accuracy. The proposed methodology in this paper uses deep learning algorithms to estimate the yoga asanas with a reasonable accuracy of 90.5%. The model first extracts the critical points required from the images/videos through hierarchical clustering and ResNet neural networks. The extracted key points are then joined as pairs according to the limbs present in the human body, thus creating a complete skeletal structure. These structures are then fed to the ResNet networks [ResNet40 and ResNet50] for final training and further optimization. Later, the results of this study are discussed.

2. Related Work

Yoga pose estimation has been a growing area of research in recent times for medical and fitness applications. Many proposed algorithms are being used to estimate complex pose angles and variations. This section gives an overview of the work done on the common standard objects in consideration (COCO) dataset for human pose estimation, followed by deep learning-based methods for pose estimation. It concludes with a brief discussion on AI-led tutor-based systems with key-point analysis for yoga pose recognition.

2.1 COCO dataset for pose estimation

Researchers have frequently used the COCO dataset to estimate human pose [35]. The *Pifpaf* network was used to train the COCO dataset in [36] to recognize relevant key points for postures and joints; the output was then trained using the ResNet 50 neural network. It was found that, even though the model worked well on the dataset, the error rate increased considerably when images were shot from a distance.

In, the model was tested on the COCO dataset, but training was done on the Cityscapes dataset. The video frames were trained using Mask RCNN architecture, followed by a residual network-feature pyramid network (ResNet50 FPN). This method suggested considering the contour edge information as an essential measure, for instance, segmentation and object detection tasks. It resulted in boosting gradient flow and increased sensitivity of the model towards boundary data points. The ResNet framework is also used in [37] to train for pose estimation and human joints data on the COCO dataset, giving a mean Intersection over Union (IOU) accuracy of 95.39%. Multi-person pose detection has been performed on the COCO dataset in [38] using dense heat map networks combined with Faster RCNN to estimate highly localized key points with an average precision of 0.685.

2.2 CNN-based methods for human pose-classification

In another work, [39] used the Frames Labeled in Cinema (FLIC) motion dataset, a set of human pose videos trained using CNN for 2D pose detection, using optical flow feature RGB images. The model's performance for the average precision degraded 3.9% from -10 pixels offset to -1 pixel offset respectively with increasing frame step.

In [40], the approach was broadly divided into two phases, first the key-point coordinates of the postures were detected, and then CNN model trained coordinates were. Further research in [41] included an estimation of 3D poses from monocular 2D images. This Multiview-Consistent Semi-Supervised Learning (MCSS) method focuses on utilizing information from different angles, correlating it through hard negative mining, and then training the data using the ResNet framework to improve the baseline result by 25%.

Subsequently, a new approach was proposed in [42], where the authors used heat maps of images for pose estimation using SpatialNet and deep regression networks. However, this approach suffered from localization problems and thus is not considered as the most suitable approach for the problem, although it provided a breakthrough in spatial fusion layers and learning from multiple frames using optical flow to create a combined confidence map from all the obtained heat maps.

2.3 Yoga Pose Classification

Researchers of Yog.ai [43] suggested some key insights to solve the localization problem; they suggested training the model on grayscale images of the RGB dataset and finding relevant key points to train the model. These approaches were able to gather an accuracy of around 97%.

Kothari in [44] used CNN, support vector machine (SVM), and a combination of CNN and long short-term memory (LSTM) to train the yoga pose datasets, which worked well on the dataset with an accuracy of 85 Approaches like [45]use Microsoft's 3D Unity system, called Kinect, as the base approach to detect critical points from images and pre-recorded video clips, with an accuracy of 94.78%. These models were trained on a limited amount of data. The analyzed frames were captured using sensor-based cameras with high-quality resolutions. This setup is industrially supported, hence cannot be made use of for generalized applications.

In another work [46], the researchers extracted the key-point coordinates of the joint from the images and then trained them in UNITY 3D using Microsoft Kinect. The model was built solely focused on mapping body coordinates using depth analysis to get pose estimation. A deviation up to 2.5 degrees in pose variation was considered acceptable with an accuracy of 97%.

2.4 Yoga Tutor-based Systems

Patil et al. [47] recommended a "Yoga Tutor" project to achieve the goal of designing a training system for personal fitness; it was based on the speeded-up robust features, commonly referred to as Speeded up Robust Features (SURF). However, this approach only considered contour and related information for posture estimation, which was insufficient for pose estimation.

Wu et al. [48] in 2010 proposed a model, which was an expert yoga system designed completely based on raw images and its text-based annotations; however, the model did not analyze user's posture as it was based only on the textual annotations and information, thus making it unsuitable for real-world datasets.

2.5 Key-point localization for yoga poses

Some widely used networks for key-point localization are OpenPose [49], DensePose [50], PoseNet [51]etc. The work proposed in [52] uses OpenPose for basic key-point extraction followed by CNN and LSTM (long short-term memory) hybrid model to get pose predictions. They achieved 99.38% accuracy, but this model was only created for six poses, thus limiting fewer data used for model creation.

BlazePose [53] is a lightweight CNN architectural model which analyzes 33 critical points for pose estimation and is robust for real-world applications. The limitations of the discussed review can be highlighted by the need for robust pose estimation systems that consider the contour, the image details (clothing, multi-person pose estimation), and provide a highly trained model for keypoint detection and estimation. Further, the ability of the system to overcome the limitations of lighting, occluded images, are listed. The prediction model should identify pose changes in pose angles and give accurate results.

The review above also suggests the need for work on a generalized model over a diversified dataset to create a self-training expert. This paper uses RCNN for creating the skeleton of the yoga poses in images using key point coordinates.

2.6 Dataset

Most datasets used for pose-estimation-based tasks lack variety and diversity of postures. This has served as motivation for researchers to study deeper into this task. Some examples of complex yoga poses are shown in the figure below.

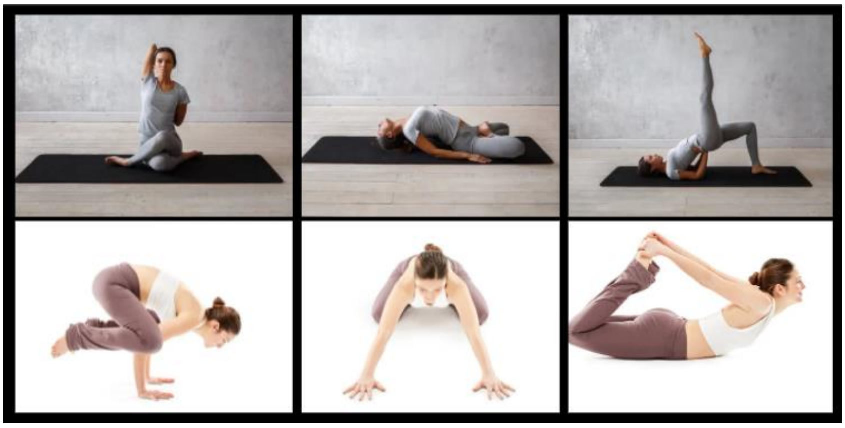


Fig 3. Some complex yoga poses

In figure 3, it can be observed that the postures demonstrated are complex and would be difficult to detect. The detection of such postures is challenging because not all body limbs are visible due to overlapping some limbs. The background and clothing also play a significant role in the pose detection process. The same type of clothing and occlusions can often lead to misconceptions and inaccurate predictions and thus the model's poor performance.

Non-standard parking behavior recognition using a vision-based system would be of great help in crowded areas. However, angle detection and camera occlusion are some challenges in real-time adoption of such systems. Robust image processing algorithms, features extraction techniques, 3D object/cars detection and parking events detection such as car in, car out, car leaving, car entering are some possible solutions to overcome these issues.

A new dataset was discussed in [54] to overcome this limitation. It consists of 82 yoga asanas for large-scale yoga pose recognition with the hierarchical label based on the body configuration of the poses instead of finer annotations. In this paper, we have used 45 classes of the dataset mentioned above.

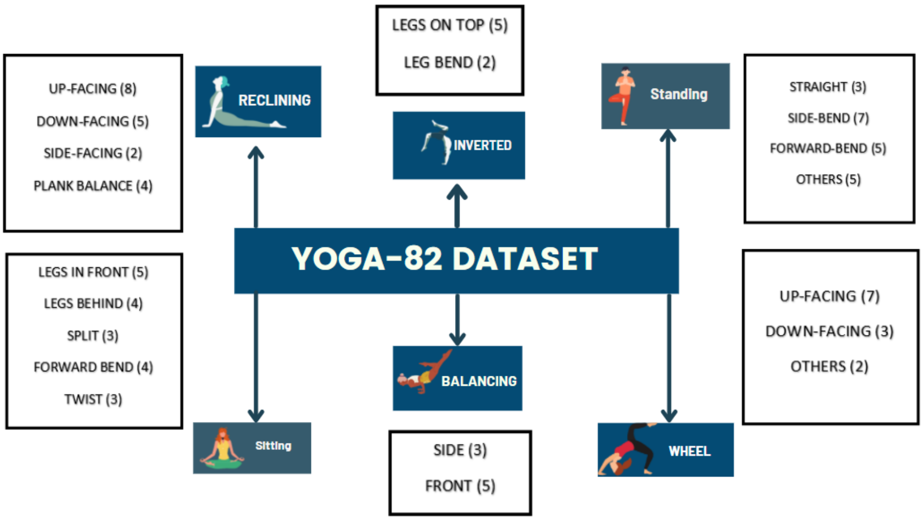


Fig 4. Yoga-82 dataset hierarchical labeling.

This dataset contains around 28,000 yoga pose images divided into 82 different classes, representing 82 different simple and complex yoga poses and class annotations based on their class hierarchy. This dataset comprises a three-level hierarchy that defines the variations in body positions in different postures. The three levels of hierarchy are described below:

- Basic description of the posture like standing, sitting, balancing, inverted and more at the first level.
- The second level has a bit more elaborated classification like straight, forward bend, side bend, twist, downward-facing, upwards-facing, etc.
- The final level consists of the actual English and Sanskrit names of the postures like Bound Angle Pose or Baddha Konasana, Camel Pose or Ustrasana, Cobra Pose or Bhujangasana, Eagle Pose or Garudasana, Handstand pose or Adho Mukha Vrksasana, and more.

This three-level hierarchy has proven helpful for classifying complex poses, as they can be distinguished more precisely based on their given levels of description.

Table I: Comparison of Yoga-82 dataset with other datasets.

Datasets	Total instances	Sources	Target poses
MPII	25,000	YouTube	Diverse
LSP-Ext	10,000	FB checker	Sports
SHPD	23,334	Surveillances	Pedestrians
Yoga-82	28,487	Bing	Yoga

3. Proposed Methodology

In the literature, it was revealed that Convolutional Neural Networks [55] [56]have always been the preferred architecture for all image classification and segmentation-related applications [57], [58].

It is a deep learning algorithm [59], [60] that focuses on taking input images, assigning their respective weights and biases according to their feature importance, and finally giving a classified output.

- CNN [61] performs feature extraction [62] using filters (also called kernels) which perform convolution between pixels to find the most relevant features from the images. These features are then reduced to a feature map consisting of all the relevant features of the image.
- These feature maps are then passed to the pooling [63] layer, which serves the purpose of dimensionality reduction [64]by finding the most relevant features from an image and removing the unwanted pixels. The most used type of pooling is max pooling, which finds the maximum pixels with relevant features.
- The pooled feature map is then passed on to the next layer, which has only one purpose, i.e., to change the dimensionality of the feature map into a 1-D array, i.e., the flatten layer [65]to send it further into the network.
- Lastly, the features are sent to the fully connected layer [66], [67], which connects the convolutional network to the Artificial Neural Network (ANN) to get the final prediction.

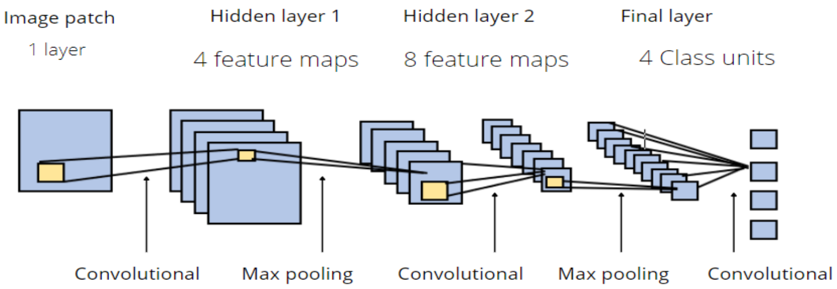


Fig 5. Convolutional neural network architect used in this work.

CNN architecture provides exceptional accuracy in pose classification tasks, thus making it a highly desirable choice for this application. They can be trained on key points of joint locations of the human skeleton or can be trained directly on the images [68]. [69]used CNN for 2D human pose estimation on human exercise images and achieved an accuracy of 83%.

Region-based CNN [70] network is used to boost the algorithm performance by reducing the computational burden by activating semantically meaningful regions, which helps to avoid localization problems for key-point extraction from the image. It was used to ensure that all image details containing sensitive information crucial for pose prediction are considered. Fig 6 shows the detailed flow of this research, along with the architecture of the stepwise followed methodology.

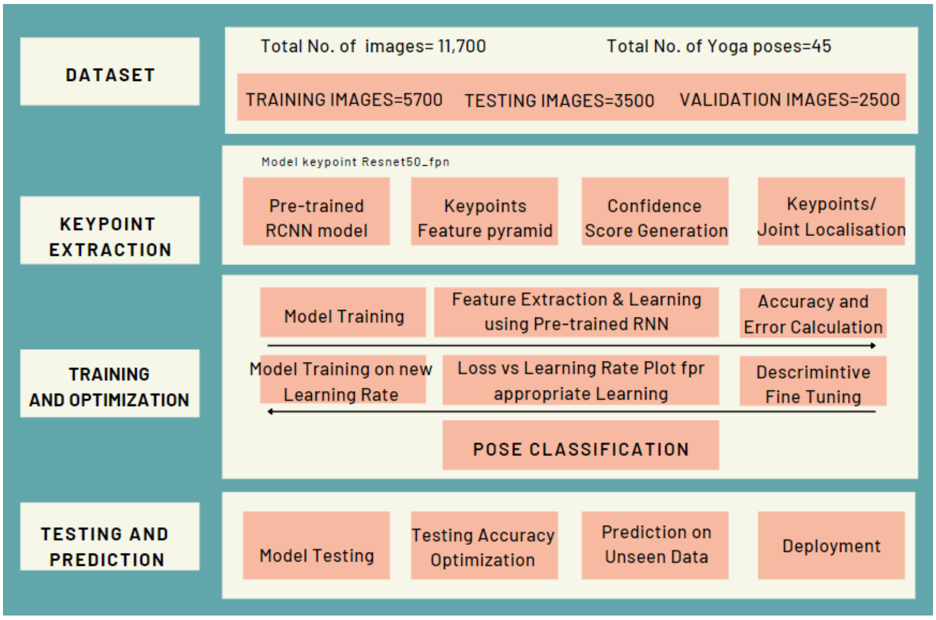

DATASET
Total No. of images= 11,700
Total No. of Yoga poses=45
TRAINING IMAGES=5700
TESTING IMAGES=3500
VALIDATION IMAGES=2500
KEYPOINT EXTRACTION
Model keypoint Resnet50_fpn
Pre-trained RCNN model
Keypoints Feature pyramid
Confidence Score Generation
Keypoints/ Joint Localisation
TRAINING AND OPTIMIZATION
Model Training
Feature Extraction & Learning using Pre-trained RNN
Accuracy and Error Calculation
Model Training on new Learning Rate
Loss vs Learning Rate Plot fpr appropriate Learning
Descrimintive Fine Tuning
POSE CLASSIFICATION
TESTING AND PREDICTION
Model Testing
Testing Accuracy Optimization
Prediction on Unseen Data
Deployment

Fig 6. The architecture of the proposed methodology

The various steps followed in the procedure are as follows:

3.1. Dataset:

The images are taken from the Yoga-82 dataset, which is designed specifically for yoga pose classification tasks. It aims to fulfill more complex and diverse data to develop a more accurate and generalized system. Out of the 82 available classes, 45 yoga pose classes are chosen for this paper based on their complexities.

The dataset [71]used in this paper is one of the latest and most challenging datasets, i.e., Yoga82 hierarchically labelled dataset which is being used for yoga pose estimation tasks with a wide range of complex pose variations and images that are collected from real-world settings. This paper introduces the work done on a wide range of 11,000 images spread across 45 classes.

3.2. Key-point Extraction:

The pre-processing is done using manual labelling of the images, followed by train-test validation and batch normalization for standardization of data. It has further been reshaped and enhanced to fit the requirements of the training model.

Region-based Convolutional Neural Network (R-CNN) [72]has proven to work well on various challenging benchmark datasets. It has also helped in producing exceptional state-of-art results for many objects detection and segmentation tasks. Previously used CNN models for object detection and segmentation have faced localization problems for large datasets. Previous deep learning models aimed to maintain a high spatial resolution of image resolution for different frames of inputs.

This type of RCNN network generates class-independent proposals of regions of the object (in our case, joints) to be located. After getting all the possible proposals, the Selective Search technique [73]is utilized to determine a vast set of possible joint locations from the images [74]. This is performed by forming clusters of image pixels into segments, followed by hierarchical clustering [75], [76], a supervised learning method for classification that further combines the formed segments into possible joint proposals.

RCNN is shown to work very well for Object detection tasks (limbs in this case). Instead of working on a massive number of regions, the RCNN algorithm proposes a bunch of boxes in the image and checks if any of these boxes contain any object using the selective search technique. Faster RCNN is preferred generally because it's faster than the RCNN model when it comes to making predictions for each new image. But this also uses regions to identify objects. The network does not look at the complete image in one go, instead it focuses on parts of the image sequentially. This further creates some complications like the algorithm still requires many passes through a single image to extract all the required regions from the input image. Also, the RPN is trained where all the anchors in the mini batch, which are of size 256, are extracted from a single image. Since all samples are from a single source, it is possible that they may correlate i.e., the features are similar so the network may take a lot of time to converge. Faster RCNN would have the convergence problem for this application as the features which are the keypoints are somewhat similar.

Instead of using segmentation as ground truth, the critical point uses heat maps [77], [78] of fixed points. A heat map is a vector with the input image height and width and contains mostly zero values. It was observed that the pixel turned positive when a point of interest was encountered. For example, the left hand of a human is present in the background. The image array has positive pixels, with the "hottest" pixel in the heat map being in the center. This positive pixel in the middle is called a critical point. Networks can then train to find common patterns around this crucial point and learn how to find them in an image [79].

3.3. Pre-Trained ResNet Model:

The key-point_resnet50_fpn network uses feature pyramid networks [80]for feature selection. The purpose of using feature pyramids [81] is to combine low-resolution solid features with high-resolution ones. This helps in attaining the most relevant features from the input frames. The key-point RCNN model used here takes an image tensor as inputs for detection. First, convert the input images into tensors using the PyTorch transform method. The output from the transformation is in the format [*batch size x number of channels x height x width]*. The output is obtained in the form of a dictionary list, which comprises the resulting tensors. The fields of the dictionary are as follows:

- Boxes: [x1, y1, x2, y2] format,
    - x = [0, W] and,
    - y = [0, H]
- Scores: The confidence scores for each prediction.
- Key points are the predicted interest points (i.e. joint locations) in [x, y, v] 3D Cartesian coordinate format.

3.4. Detected Key-points:

There are 17 critical points detected in this model, shown in table 2 and their respective numbering, which is shown further in the image afterward. The key points are numbered and paired according to the limbs they form in a human skeleton. Different limbs are formed by pairing these key points, giving the body's final skeletal structure for a particular pose.

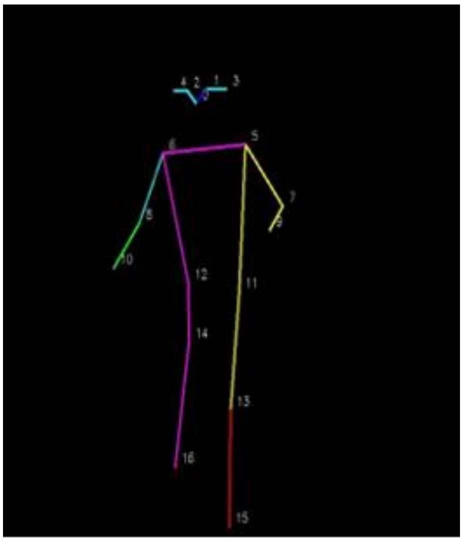


Fig 7. Key-points detected using RCNN.

3.5. Forming the skeletal structure:

The next focus is on the formation of the skeleton based on the pairs obtained. While joining the key points, a threshold needs to be set to get more accurate predictions. Here, the threshold taken is 0.9, as it was observed that the joints with a confidence score of more than 0.9 tend to give more false positives in the output. Following this, the pre-trained keypoint_rcnn_resnet50 model is used, a pre-trained Keypoint-RCNN model with ResNet50 backbone, to detect critical points.

According to Table 2, the following pairs have been chosen in order to form the edges of the skeletal structure of the pose; (0, 1), (0, 2), (2, 4), (1, 3), (6, 8), (8, 10), (5, 7), (7, 9), (5, 11), (11, 13), (13, 15), (6, 12), (12, 14), (14, 16), (5, 6).

Table II. Utilized key-points

Sr. No.	Joint/Key-point	Sr. No.	Joint/Key-point	Sr. No.	Joint/Key-point
0	Nose	6	Right shoulder	12	Right hip
1	Left eye	7	Left elbow	13	Left knee
2	Right eye	8	Right elbow	14	Right knee
3	Left ear	9	Left wrist	15	Left ankle
4	Right ear	10	Right wrist	16	Right ankle
5	Left shoulder	11	Left hip		

There is a confidence score associated with each key point, based on which the key points are located. The pre-trained model used here for key-point detection generates critical points for every limb separately. Further, it is joined based on the body's limbs and superimposes them on the actual image to get the final key points.

3.6. Pose Classification:

The detected vital points are then used for pose classification using Residual Networks (R.N.s). R.N.s are preferred for tasks like pose estimation because this network is pre-trained on an extensive set of data based on different types of activities that can be used for various segmentation and detection tasks and thus can work as a perfect backbone for creating a new model for a specific type of application, the knowledge from the trained model can help further enhance the training of the model which in return provides more accurate results.

Here, the detected key-points were then trained on two convolutional network structures, ResNet34 and ResNet50; as the name specifies, ResNet34 has 34 deep layers while the latter has 48 deep layers along with 1 MaxPool [82]and 1 Average Pool Layer [83], using fastai [84]. The idea used here in building the model is to combine fastai with transfer learning to create a pose classification model.

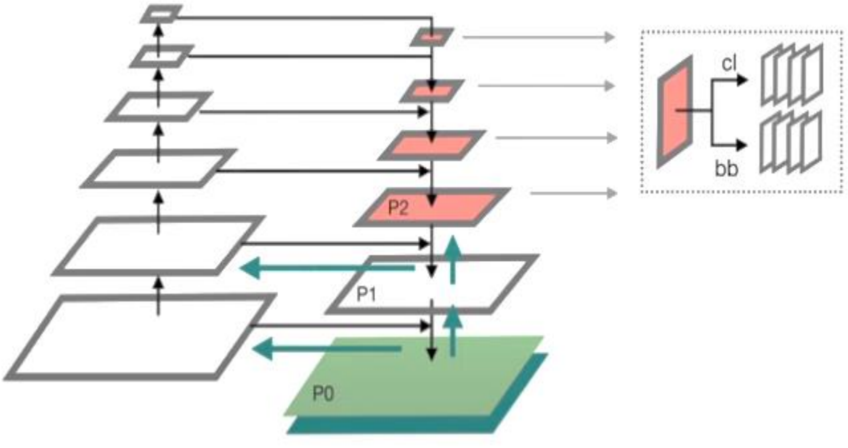


Fig 8. 50-layer ResNet network with feature pyramid.

Fastai [85], a deep learning library, provides complex components that give state-to-the-art results in standard deep learning areas. It provides accurate results in fewer computations without compromising on factors like flexibility and performance.

Transfer learning is a studying approach in which a model advanced for an undertaking formerly on a massive set of statistics corpus is reused as the start line for a version on a new task. It is a widespread technique in deep learning where pre-skilled models are used as the place to begin on computer vision and natural language processing tasks given the tremendous computing and time resources required to develop neural network fashions on these troubles and from the massive jumps in a skill that they offer on associated troubles.

In this approach, the CNN-learner from fastai is directly used to create the model. DataBunch is used to encapsulate the training and validation data. The models used as a base model to apply transfer learning are ResNet34 and ResNet50 models, R.N.s, a type of CNN. These models work on images convolved by the convolutional filters formed at the beginning of every CNN model to generate the feature maps.

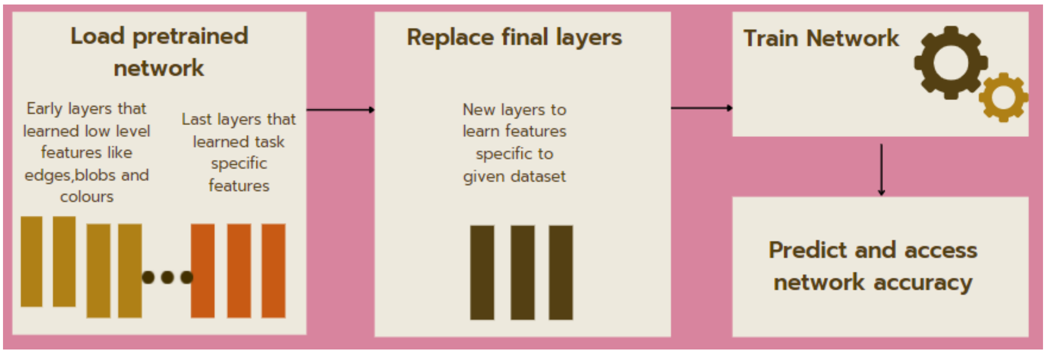

Load pretrained network
Early layers that learned low level features like edges,blobs and colours
Last layers that learned task specific features
Replace final layers
New layers to learn features specific to given dataset
Train Network
Predict and access network accuracy

Fig 9. Process of transfer learning.

Using transfer learning helps to incorporate previous knowledge of the model, trained on Imagenet dataset, which is extensive data collection used for image classification, object detection, and segmentation task helps to build a new model for our dataset and create a more generalized model with more dataset and more petite compilation.

3.7. Testing and Prediction:

After training the model with Residual backend on the detected vital points, the model is tested on the testing dataset, and predictions are made on the anonymous data to find the testing accuracy of the model. Followed by the final testing, the model can be deployed to develop a complete self-training system for Yoga Act.

4. Results and Discussion

Each phase of the training model is followed by testing. The results are then optimized to achieve good accuracy in the model. Hence, the section is discussed in three divisions, starting with initial training using ResNet models, then optimizing the ResNet models, followed by a comprehensive discussion of results for final training.

4.1 Initial Training

After the initial training with ResNet34 and ResNet50, respectively, the results from both the models are shown in the tables below:

Table III. Result of initial training on ResNet34.

Epoch	Train_loss	Valid_loss	Accuracy	Time
0	3.5589	1.9186	0.4926	22:33
1	2.3018	1.4089	0.6137	05:26
2	1.6377	1.0957	0.6870	05:21
3	1.2089	0.9206	0.7392	05:24
4	0.9653	0.8229	0.7673	05:24
5	0.8139	0.7485	0.7896	05:26
6	0.6433	0.7146	0.7965	05:27
7	0.5471	0.6652	0.8093	05:25

8	0.4540	0.6515	0.8104	05:28
9	0.4083	0.6503	0.8116	05:28

Table IV. Result of initial training on ResNet50.

Epoch	Train_loss	Valid_loss	Accuracy	Time
0	4.3934	2.4492	0.3726	06:22
1	2.6965	1.4727	0.5934	06:21
2	1.8465	1.2074	0.6635	06:31
3	1.4257	1.033	0.7064	06:36
4	1.1646	0.8977	0.7380	06:59
5	0.9618	0.8469	0.7554	07:00
6	0.8108	0.7924	0.7731	07:03
7	0.7311	0.7668	0.7838	07.08
8	0.6740	0.7579	0.7838	06:52
9	0.6239	0.7532	0.7861	06:52

The results of the table can be summarized as follows:

1. ResNet34 - 78.6%
2. ResNet50 - 81.16%

We observe that ResNet50 gives a better accuracy as compared to ResNet34 by 2.56%. The training loss decreases from 4.3934 to 0.6239, while the validation loss decreases from 2.4492 to 0.7532 in 10 epochs. Similarly, the training loss decreases from 3.5589 to 0.4083, and the average validation loss is 0.9596.

ResNet introduces the concept of “identity shortcut connection.” Increasing the number of layers in the neural network does not ideally guarantee a more suitable model or greater accuracy. Still, ResNet with the feedforward memory network ensures the decrease in vanishing gradient problems with increasing depth. Hence, the Resnet50 model gives a greater accuracy, as it involves more data training through an extensive network of residual blocks.

4.2 Optimization

Despite having an accuracy of 81.16%, the ResNet50 model can further be optimized to get better results. For optimization, *Discriminative Fine-Tuning* [86], [87] method is used. This method enables training on all different layers of the network at different learning rates. The main advantage lies that new layers formed after the creation of the model can be trained at the same learning rate, while the pre-trained layers from the base model can be trained at a comparatively lower rate; this ensures we focus on the new layers.

The principle here is to observe how the loss in the training model varies at a different range of learning rates. From that observation, the optimal range of learning rate for the model can be explicitly estimated.

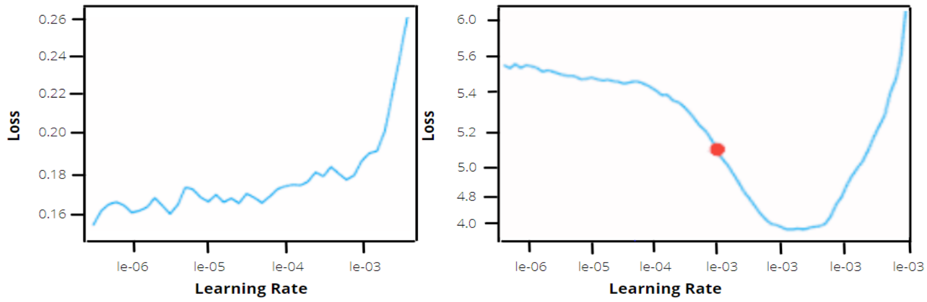


Fig 10. Learning vs Loss Curve for ResNet34 & ResNet50.

After optimization, the range of learning rates from figure 8 and 9 for both the models is as follows:

ResNet34 - 1e-06 to 1e-04

ResNet50 - 1e-04 to 1e-02

The model architecture leverages transfer learning by introducing fine tuning in the training model. The model has already been trained on the original model which benefits by enhancing the unfrozen layers and training on the new weights, thus improving optimization.

Data augmentation is a tested method to reduce over-fitting in the dataset. Yoga-82 dataset by design is diverse and variable hence the over-fitting of the model is done by image data generator to introduce a variety of images which are rotated, skewed, sheared, zoomed, cropped and so on. This adds generalization ability to the entire dataset as well as eliminates the problem of overfitting

4.3. Final Training

The plot of learning rate vs. loss gives the range of learning rates that should be used for both the models according to the change in losses for different learning rates.

The model has trained again, with the newly obtained range of learning rates. Tables 5 and 6 show the obtained results of the optimized models.

Table V. Final training results on ResNet34.

Epoch	Train_loss	Valid_loss	Accuracy	Time
0	0.6919	0.7425	0.7823	12:39
1	0.5991	0.7072	0.7936	05:31
2	0.5614	0.6726	0.7988	05:31
3	0.5100	0.6496	0.8102	05:31
4	0.4527	0.6329	0.8122	05:35
5	0.4206	0.6253	0.8157	05:39
6	0.4019	0.6192	0.8168	05:37
7	0.4197	0.6230	0.8157	05:37

Table VI. Final training results on ResNet50

Epoch	Train_loss	Valid_loss	Accuracy	Time
0	0.7676	1.4320	0.6780	14:00
1	1.3870	2.2410	0.5482	05:31
2	1.3047	1.1580	0.6689	05:33
3	0.9908	0.9102	0.7667	05:31
4	0.7304	0.6251	0.8333	05:36
5	0.5330	0.4585	0.8704	05:36
6	0.3362	0.3748	0.8962	05:37
7	0.2352	0.3568	0.9055	05:30

Final accuracies for both the models are:

1. ResNet34 - 81.6%
2. ResNet50 – 90.5%

The training loss reduces significantly to 0.235 from 0.7676, and the validation loss is 0.9445 for ResNet50, which is less than the initial training model.

We observe a significant change in the results after the optimization and final training; the resulting model provides greater accuracy from the initial model by almost an increase of 10%.

Table 7 shows an accuracy of 90.5% obtained on ResNet50. In the next section, the final predictions of the model on the testing data are shown in the form of a figure.

Table VIII. Comparison between ResNet34 and ResNet50.

Architecture	Depth (layers)	#Params	Model size	Learning rate range	Top Accurac y
ResNet34	34	21.28M	178.5mb	1e-06 to 1e-04	81.5
ResNet50	50	23.15M	190.4mb	1e-04 to 1e-02	90.5

The CNN architecture is proven to provide exceptional performance in pose estimation and other activity detection tasks, thus making it a highly desirable choice for this application. They can be trained on key points of joint locations of the human skeleton or can be trained directly on the images. RCNN network is used to boost the performance of CNN algorithm as it focuses on extracting all the relevant regions in the input image thus reducing the localization problem for the joint detection.

The efficiency of the model can be seen in two ways i.e., the computational efficiency which indicates how much time and/or space it will take on an input of size N to arrive at the output, and through the classification score i.e., the accuracy of the model. Model building is a time-consuming process but using Fastai helped to reduce the training time of the pretrained model.

Both the models resulted in a good accuracy score. Further optimization in the learning rates of the models resulted in the best possible results which can be seen through the graphs below.

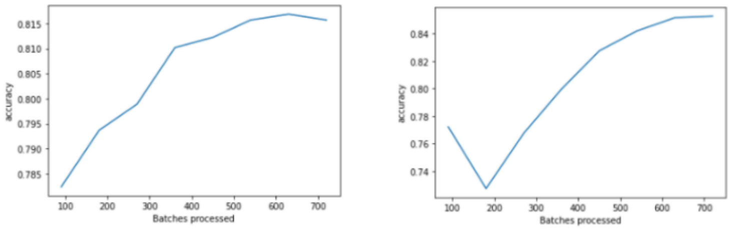


Fig 11. Accuracy graphs for ResNet34 and ResNet50 models

4.3 Discussion

Due to the difference in the datasets, the accuracies and performance of the previous models cannot be compared directly with this model. Similar experiments were performed in the previous work using Kinect, Star Skeleton [88] with accuracies as higher as 99.4%. However, the poses used for those models were different and much less complex than the ones used in this model. This model is tested on every type of asanas from reclining to wheel and has shown consistent results. The summarized results of the model proposed in this paper are given in table 8.

Table 8: Summarized results of ResNet34 and ResNet50.

Model	Before optimization	After optimization
ResNe34	81.1%	81.5%
ResNet50	78.6%	90.5%

The work done in [89]using the state-of-art PoseNet model has shown prominent results in detecting vertical poses like the standing or inverted ones but has a significantly poor performance for horizontal poses like the balancing poses. Our model has proven to be orientation agnostic as it has shown to work prominently for every type of pose, as shown in figure 11(a) and figure 11(d) thus making it a generalized approach.

Another work in [90] gives a self-learning system for pose estimation using Kinect sensors. This model was designed by taking three asanas into consideration i.e., tree pose, downward dog, and warrior III, with all images taken from a high-definition camera, while our model takes these three poses along with many such poses and still gives a good accuracy. In [33], the authors have used an SVM, CNN network for key point extraction, along with a combination of CNN and LSTM to train the model, where the CNN network helps to identify patterns from input frames while LSTM neurons examine the attributes of the different frames. They got an accuracy of 85% on their model, which is only trained on 6 poses. All those poses are present in our dataset and many more.

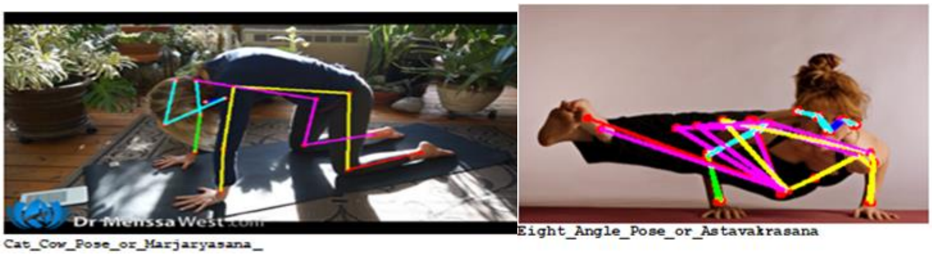
Table IX. Comparative study with state-of-the-art models.

Method	Dataset	Accuracy
MR-CNN [91]	MS COCO, PASCAL, VOC	89.3%
CNN-LSTM[52]	6 Poses, 12 People	98.92%
BLAZEPOSE[53]	1000 Pictures	97.2%
OPENPOSE[53]	AR Dataset	87.8%
SVM[92]	6 Poses, 15 People	98.58%
ResNe34	Yoga-82 dataset (11,000 images with 45 classes)	**81.5%**
ResNet50		**90.5%**

Table 9 summarizes the benchmark work done in yoga pose estimation with different customized datasets facing the limitation of the restricted range of yoga poses. The last two models, which are contributions of this paper, suggest a great accuracy for an extensive domain of 11,000 images spread over 45 classes in a hierarchical cluster.

The work in [93]uses body contour and skeletal information with the help of Kinect sensors to find the keypoints. They got an accuracy of 76.22 to 99.87% for all the keypoints. However, the methodology used here requires manual feature extraction for each pose using which a separate model is created for every pose based on the features extracted, which makes this a time-consuming process. Also, this procedure requires a new set of features every time a new pose is added to the dataset. In our approach, feature extraction is done using a region-based network wherein predictions of features are based on regional proposals thus no need for manual feature extraction or separate models for each pose. Also, adding a new pose to the dataset only requires adding one neuron to the final layer of the model.

In [52] discussed earlier, another state-of-art technique has been used i.e., the OpenPose architecture, which is the most frequently used model for pose detection and estimation tasks. The authors got an accuracy of 99.34% by using this architecture for real-time recognition. However, systems constrained by OpenPose pose recognition algorithms tend to sometimes fail in case of overlapping parts, where not every limb of the body is clearly visible, along with cases of false positives on animals, sometimes even statues. Our model has shown comparatively better results for overlapping limbs in complex poses, such as Eight Angle Pose and Child Pose, as shown in figure 11(b) and figure 11(f).


Dr Melissa West.com
Cat_Cow_Pose_or_Marjaryasana_
Eight_Angle_Pose_or_Astavakrasana

(a)

(b)

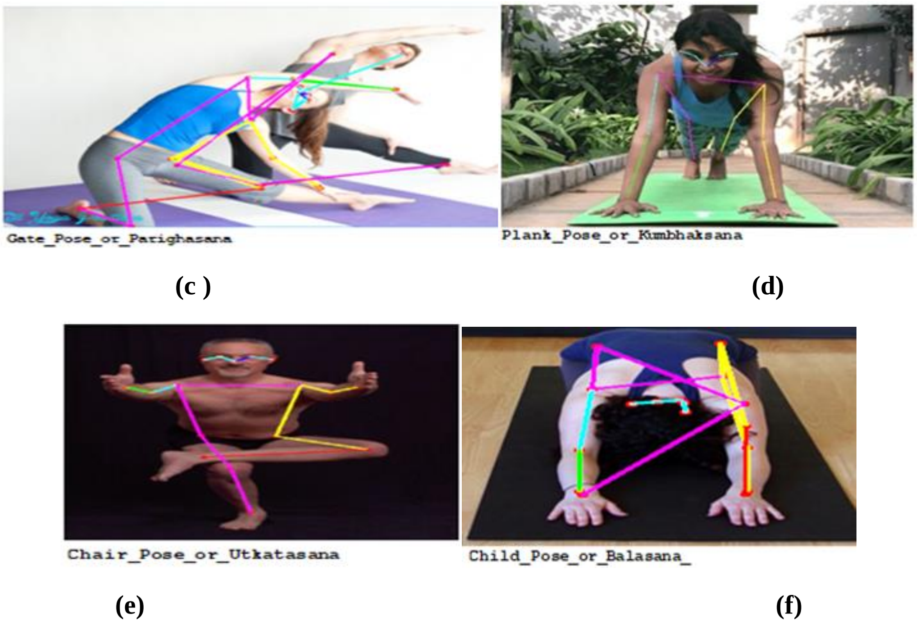


Fig 12: Depicting final predictions for yoga asanas:

(a) Cat_Cow_Pose_or_Marjaryasana,

(b) Eight_Angle_Pose_or_Astavakrasana,

(c) Gate_Pose _or_Parighasana,

(d) Plank_Pose_or_Phalakasana,

(e) Chair_pose_or_Utkasana, and

(f) Child_Pose_or_Balasana.

5. Conclusions

This paper proposes building a computer vision model for a yoga pose estimation task on a sophisticated dataset with complex pose variations. The literature suggested a lack of work done on diverse datasets for yoga pose estimation tasks. The existing state-of-the-art build models work well on a limited dataset scope, with almost significantly fewer variations in the proposed yoga asanas. The proposed methodology in this paper extracts the essential 17 key points (body joints) from an image and forms a skeletal structure to examine the posture. Later, these key points are trained by the ResNet50 model, which acts as a pose classification model. The result gives an accuracy of 90.5% for the proposed model with precise predictions on the testing and training data. This model further can be used to make a user-friendly application where the user can compare their yoga pose image with the same pose image of the instructor or an expert. The extensive–trained model will point out the flaws in the user's posture and thus prevent any mishaps. The yoga training model is based on key point identification and analysis. Hence, it can be used in healthcare applications to identify joint and alignment-related problems with the limbs and body.

The key point detection has been done uniquely by introducing the feature pyramid based RCNN model powered by ResNet50 as the backbone model. The key point extraction is carried out by hierarchical clustering by heatmap generation and later cluster analysis followed by tensor generation for each joint in the image of the human body.

The classification of the detected key points into their respective classes has been presented as a comparative study between ResNet34 and Resnet50 which gives accuracies of about 81.1 and 78.6% respectively, these models are later optimized using deterministic fine-tuning model to give accuracies of 81.5% and 90.5%.

As a result, the novelty of the method introduces the work done on newly published dataset taken from complex real-world settings, the model is built in a series of layers starting from key point localization using RCNN to later predicting and placing the occluded key point images using hierarchical clustering. The fine- tuning model used towards the end for optimization uses elimination of the overfitting of the model onto the given Yoga-82 dataset.

References

[1] R. M. Haralick, H. Joo, C. Lee, X. Zhuang, V. G. Vaidya, and M. B. Kim, "Pose estimation from corresponding point data," *IEEE Transactions Syst Man Cybern*, vol. 19, no. 6, pp. 1426–1446, 1989, doi: 10.1109/21.44063.

[2] P. Vyas, "Pose estimation and action recognition in sports and fitness," 2019, doi: 10.31979/etd.w8ug-4v5c.

[3] S. Chen and R. R. Yang, "Pose Trainer: Correcting Exercise Posture using Pose Estimation," *Arxiv*, 2020.

[4] Y. Li, C. Wang, Y. Cao, B. Liu, J. Tan, and Y. Luo, "Human pose estimation based in-home lower body rehabilitation system," *2020 Int Jt Conf Neural Networks Ijcnn*, vol. 00, pp. 1–8, 2020, doi: 10.1109/ijcnn48605.2020.9207296.

[5] S. R. Rick, S. Bhaskaran, Y. Sun, S. McEwen, and N. Weibel, "NeuroPose," *Proc 24th Int Conf Intelligent User Interfaces Companion*, pp. 105–106, 2019, doi: 10.1145/3308557.3308682.

[6] J. Segen and S. Kumar, "Shadow gestures: 3D hand pose estimation using a single camera," *Proc 1999 IEEE Comput Soc Conf Comput Vis Pattern Recognit Cat Pr00149*, vol. 1, pp. 479-485 Vol. 1, 1999, doi: 10.1109/cvpr.1999.786981.

[7] H. Kang, C. W. Lee, and K. Jung, "Recognition-based gesture spotting in video games," *Pattern Recogn Lett*, vol. 25, no. 15, pp. 1701–1714, 2004, doi: 10.1016/j.patrec.2004.06.016.

[8] H. Xie, A. Watatani, and K. Miyata, "Visual Feedback for Core Training with 3D Human Shape and Pose," *2019 Nicograph Int Nicoint*, vol. 00, pp. 49–56, 2019, doi: 10.1109/nicoint.2019.00017.

[9] G. S. Birdee, G. Y. Yeh, P. M. Wayne, R. S. Phillips, R. B. Davis, and P. Gardiner, "Clinical Applications of Yoga for the Pediatric Population: A Systematic Review," *Acad Pediatr*, vol. 9, no. 4, pp. 212-220.e9, 2009, doi: 10.1016/j.acap.2009.04.002.

[10] M. Eichner and V. Ferrari, "Human Pose Co-Estimation and Applications," *IEEE T Pattern Anal*, vol. 34, no. 11, pp. 2282–2288, 2012, doi: 10.1109/tpami.2012.85.

[11] Z. Yang, X. Yu, and Y. Yang, "DSC-PoseNet: Learning 6DoF Object Pose Estimation via Dual-scale Consistency," *2021 IEEE Cvf Conf Comput Vis Pattern Recognit Cvpr*, vol. 00, pp. 3906–3915, 2021, doi: 10.1109/cvpr46437.2021.00390.

[12] R. Divya and J. D. Peter, "Smart healthcare system-a brain-like computing approach for analyzing the performance of detectron2 and PoseNet models for anomalous action detection in aged people with movement impairments," *Complex Intelligent Syst*, pp. 1–20, 2021, doi: 10.1007/s40747-021-00319-8.

[13] Y. Wu *et al.*, "A Computer Vision-Based Yoga Pose Grading Approach Using Contrastive Skeleton Feature Representations," *Healthc*, vol. 10, no. 1, p. 36, 2021, doi: 10.3390/healthcare10010036.

[14] S. P, K. Manik, and S. K, "Role of yoga in attention, concentration, and memory of medical students," *National J Physiology Pharm Pharmacol*, vol. 8, no. 9, p. 1526, 2018, doi: 10.5455/njppp.2018.8.0723521082018.

[15] I. Stephens, "Case report: The Use of Medical Yoga for Adolescent Mental Health," *Complement Ther Med*, vol. 43, pp. 60–65, 2019, doi: 10.1016/j.ctim.2019.01.006.

[16] S. Goyal and A. Jain, "Yoga Pose Perfection using Deep Learning: An Algorithm to Estimate the Error in Yogic Poses," *J Student Res*, vol. 10, no. 3, 2021, doi: 10.47611/jsrhs.v10i3.2140.

[17] J. Palanimeera and K. Ponmozhi, "Classification of yoga pose using machine learning techniques," *Mater Today Proc*, vol. 37, pp. 2930–2933, 2021, doi: 10.1016/j.matpr.2020.08.700.

[18] G. Balakrishnan, A. Zhao, A. V. Dalca, F. Durand, and J. Guttag, "Synthesizing Images of Humans in Unseen Poses," *2018 IEEE Cvf Conf Comput Vis Pattern Recognit*, pp. 8340–8348, 2018, doi: 10.1109/cvpr.2018.00870.

[19] A. Rajšp and I. Fister, "A Systematic Literature Review of Intelligent Data Analysis Methods for Smart Sport Training," *Appl Sci*, vol. 10, no. 9, p. 3013, 2020, doi: 10.3390/app10093013.

[20] A. Weitz, L. Colucci, S. Primas, and B. Bent, "InfiniteForm: A synthetic, minimal bias dataset for fitness applications," *Arxiv*, 2021.

[21] A. Kortylewski, Q. Liu, H. Wang, Z. Zhang, and A. Yuille, "Combining Compositional Models and Deep Networks For Robust Object Classification under Occlusion," *2020 IEEE Winter Conf Appl Comput Vis Wacv*, vol. 00, pp. 1322–1330, 2020, doi: 10.1109/wacv45572.2020.9093560.

[22] V. Sharma, M. Gupta, A. Kumar, and D. Mishra, "Video Processing Using Deep Learning Techniques: A Systematic Literature Review," *IEEE Access*, vol. 9, pp. 139489–139507, 2021, doi: 10.1109/access.2021.3118541.

[23] Y.-H. Byeon, J.-Y. Lee, D.-H. Kim, and K.-C. Kwak, “Posture Recognition Using Ensemble Deep Models under Various Home Environments,” *Appl Sci*, vol. 10, no. 4, p. 1287, 2020, doi: 10.3390/app10041287.

[24] A. Badiola-Bengoa and A. Mendez-Zorrilla, “A Systematic Review of the Application of Camera-Based Human Pose Estimation in the Field of Sport and Physical Exercise,” *Sensors Basel Switz*, vol. 21, no. 18, p. 5996, 2021, doi: 10.3390/s21185996.

[25] A. Ross and S. Thomas, “The Health Benefits of Yoga and Exercise: A Review of Comparison Studies,” *J Altern Complementary Medicine*, vol. 16, no. 1, pp. 3–12, 2010, doi: 10.1089/acm.2009.0044.

[26] S. Jain, A. Rustagi, S. Saurav, R. Saini, and S. Singh, “Three-dimensional CNN-inspired deep learning architecture for Yoga pose recognition in the real-world environment,” *Neural Comput Appl*, vol. 33, no. 12, pp. 6427–6441, 2021, doi: 10.1007/s00521-020-05405-5.

[27] K. A. P. D. PF and N. P. E. Partini, “The Implementation of Yoga Teaching in Improving Elementary School Students’ Learning Concentration.”

[28] A. Büssing, A. Michalsen, S. B. S. Khalsa, S. Telles, and K. J. Sherman, “Effects of Yoga on Mental and Physical Health: A Short Summary of Reviews,” *Evidence-based Complementary Altern Medicine Ecam*, vol. 2012, p. 165410, 2012, doi: 10.1155/2012/165410.

[29] M. D. Tran, R. G. Holly, J. Lashbrook, and E. A. Amsterdam, “Effects of Hatha Yoga Practice on the Health-Related Aspects of Physical Fitness,” *Prev Cardiol*, vol. 4, no. 4, pp. 165–170, 2001, doi: 10.1111/j.1520-037x.2001.00542.x.

[30] G. G. Chiddarwar, A. Ranjane, M. Chindhe, R. Deodhar, and P. Gangamwar, “AI-Based Yoga Pose Estimation for Android Application,” *Int J Innovative Sci Res Technology*, vol. 5, no. 9, pp. 1070–1073, 2020, doi: 10.38124/ijisrt20sep704.

[31] P. Plantard, H. P. H. Shum, and F. Multon, “Filtered pose graph for efficient kinect pose reconstruction,” *Multimed Tools Appl*, vol. 76, no. 3, pp. 4291–4312, 2017, doi: 10.1007/s11042-016-3546-4.

[32] T. Ou, Y. Hoshino, H. Ohsuga, M. Yamada, and T. Miyamoto, “The Humanoid Robot/Camera System for Teaching YOGA Exercise Motions,” *2020 IEEE 9th Global Conf Consumer Electron Gcce*, vol. 00, pp. 826–830, 2020, doi: 10.1109/gcce50665.2020.9292075.

[33] C. Buizza and Y. Demiris, “Rotational Adjoint Methods for Learning-Free 3D Human Pose Estimation from IMU Data,” *2020 25th Int Conf Pattern Recognit Icpr*, vol. 00, pp. 7868–7875, 2021, doi: 10.1109/icpr48806.2021.9413050.

[34] Y. Chen, Y. Tian, and M. He, “Monocular human pose estimation: A survey of deep learning-based methods,” *Comput Vis Image Und*, vol. 192, p. 102897, 2020, doi: 10.1016/j.cviu.2019.102897.

[35] S. Jin *et al.*, “Whole-Body Human Pose Estimation in the Wild,” *Arxiv*, 2020.

[36] W. Deng, L. Bertoni, S. Kreiss, and A. Alahi, “Joint Human Pose Estimation and Stereo 3D Localization,” *2020 IEEE Int Conf Robotics Automation Icra*, vol. 00, pp. 2324–2330, 2020, doi: 10.1109/icra40945.2020.9197069.

[37] X. Chen, Z. Zhou, Y. Ying, and D. Qi, “Real-time Human Segmentation using Pose Skeleton Map,” *2019 Chin Control Conf Ccc*, vol. 00, pp. 8472–8477, 2019, doi: 10.23919/chicc.2019.8865151.

[38] G. Papandreou *et al.*, “Towards Accurate Multi-Person Pose Estimation in the Wild,” *2017 IEEE Conf Comput Vis Pattern Recognit Cvpr*, pp. 3711–3719, 2017, doi: 10.1109/cvpr.2017.395.

[39] A. Jain, J. Tompson, Y. LeCun, and C. Bregler, “MoDeep: A Deep Learning Framework Using Motion Features for Human Pose Estimation,” *Arxiv*, 2014.

[40] H. Altwaijry, A. Veit, and S. Belongie, “Learning to Detect and Match Keypoints with Deep Architectures,” *Procedings Br Mach Vis Conf 2016*, p. 49.1-49.12, 2016, doi: 10.5244/c.30.49.

[41] R. Mitra, N. B. Gundavarapu, A. Sharma, and A. Jain, “Multiview-Consistent Semi-Supervised Learning for 3D Human Pose Estimation,” *2020 IEEE Cvf Conf Comput Vis Pattern Recognit Cvpr*, vol. 00, pp. 6906–6915, 2020, doi: 10.1109/cvpr42600.2020.00694.

[42] K. Zhou, X. Han, N. Jiang, K. Jia, and J. Lu, “HEMlets PoSh: Learning Part-Centric Heatmap Triplets for 3D Human Pose and Shape Estimation,” *IEEE T Pattern Anal*, vol. PP, no. 99, pp. 1–1, 2021, doi: 10.1109/tpami.2021.3051173.

[43] A. Lai, B. Reddy, and and B. van Vlijmen, “Yog.ai: Deep Learning for Yoga,” *CS230:Deep Learning, Winter 2019. Stanford University, CA.(LateX template borrowed from NIPS 2017.). p. 6, 2019*, 2019.

[44] J. Jose and S. Shailesh, “Yoga Asana Identification: A Deep Learning Approach,” *Iop Conf Ser Mater Sci Eng*, vol. 1110, no. 1, p. 012002, 2021, doi: 10.1088/1757-899x/1110/1/012002.

[45] E. W. Trejo and P. Yuan, “Recognition of Yoga Poses Through an Interactive System with Kinect Device,” *2018 2nd Int Conf Robotics Automation Sci Icras*, vol. 00, pp. 1–5, 2018, doi: 10.1109/icras.2018.8443267.

[46] M. U. Islam, H. Mahmud, F. B. Ashraf, I. Hossain, and Md. K. Hasan, “Yoga Posture Recognition by Detecting Human Joint Points in Real Time Using Microsoft Kinect,” *2017 IEEE Region 10 Humanit Technology Conf R10-htc*, pp. 668–673, 2017, doi: 10.1109/r10-htc.2017.8289047.

[47] S. Patil, A. Pawar, A. Peshave, A. N. Ansari, and A. Navada, “Yoga Tutor Visualization and Analysis Using SURF Algorithm,” *2011 IEEE Control Syst Graduate Res Colloquium*, vol. 1, pp. 43–46, 2011, doi: 10.1109/icsgrc.2011.5991827.

[48] W. wu, W. Yin, and F. Guo, “Learning and Self-instruction Expert System For Yoga,” *2010 2nd Int Work Intelligent Syst Appl*, pp. 1–4, 2010, doi: 10.1109/iwisa.2010.5473592.

[49] R. Huang, J. Wang, H. Lou, H. Lu, and B. Wang, “Miss Yoga: A Yoga Assistant Mobile Application Based on Keypoint Detection,” *2020 Digital Image Comput Techniques Appl Dicta*, vol. 00, pp. 1–3, 2020, doi: 10.1109/dicta51227.2020.9363384.

[50] R. A. Güler, N. Neverova, and I. Kokkinos, “DensePose: Dense Human Pose Estimation in the Wild,” *2018 IEEE Cvf Conf Comput Vis Pattern Recognit*, pp. 7297–7306, 2018, doi: 10.1109/cvpr.2018.00762.

[51] Z. Wang, J. Chen, and S. C. H. Hoi, “Deep Learning for Image Super-resolution: A Survey,” *Arxiv*, 2019.

[52] S. K. Yadav, A. Singh, A. Gupta, and J. L. Raheja, “Real-time Yoga recognition using deep learning,” *Neural Comput Appl*, vol. 31, no. 12, pp. 9349–9361, 2019, doi: 10.1007/s00521-019-04232-7.

[53] V. Bazarevsky, I. Grishchenko, K. Raveendran, T. Zhu, F. Zhang, and M. Grundmann, “BlazePose: On-device Real-time Body Pose tracking,” *Arxiv*, 2020.

[54] M. Verma, S. Kumawat, Y. Nakashima, and S. Raman, “Yoga-82: A New Dataset for Fine-grained Classification of Human Poses,” *2020 IEEE Cvf Conf Comput Vis Pattern Recognit Work Cvprw*, vol. 00, pp. 4472–4479, 2020, doi: 10.1109/cvprw50498.2020.00527.

[55] X. Lin, C. Zhao, and W. Pan, “Towards Accurate Binary Convolutional Neural Network,” *Arxiv*, 2017.

[56] F. Cao, K. Yao, and J. Liang, “Deconvolutional neural network for image super-resolution,” *Neural Networks*, vol. 132, pp. 394–404, 2020, doi: 10.1016/j.neunet.2020.09.017.

[57] C. Affonso, A. L. D. Rossi, F. H. A. Vieira, and A. C. P. de L. F. de Carvalho, “Deep learning for biological image classification,” *Expert Syst Appl*, vol. 85, pp. 114–122, 2017, doi: 10.1016/j.eswa.2017.05.039.

[58] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson, “Deep Learning for Hyperspectral Image Classification: An Overview,” *IEEE T Geosci Remote*, vol. 57, no. 9, pp. 6690–6709, 2019, doi: 10.1109/tgrs.2019.2907932.

[59] X. Yang, Y. Ye, X. Li, R. Y. K. Lau, X. Zhang, and X. Huang, “Hyperspectral Image Classification With Deep Learning Models,” *IEEE T Geosci Remote*, vol. 56, no. 9, pp. 5408–5423, 2018, doi: 10.1109/tgrs.2018.2815613.

[60] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, “Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks,” *IEEE T Geosci Remote*, vol. 54, no. 10, pp. 6232–6251, 2016, doi: 10.1109/tgrs.2016.2584107.

[61] K. O’Shea and R. Nash, “An Introduction to Convolutional Neural Networks,” *Arxiv*, 2015.

[62] M. Jogin, Mohana, M. S. Madhulika, G. D. Divya, R. K. Meghana, and S. Apoorva, “Feature Extraction using Convolution Neural Networks (CNN) and Deep Learning,” *2018 3rd IEEE Int Conf Recent Trends Electron Information Commun Technology Rteict*, vol. 00, pp. 2319–2323, 2018, doi: 10.1109/rteict42901.2018.9012507.

[63] T. Otsuzuki, H. Hayashi, Y. Zheng, and S. Uchida, “Regularized Pooling,” *Arxiv*, 2020.

[64] H. Zhang and J. Ma, “Hartley Spectral Pooling for Deep Learning,” *Arxiv*, 2018, doi: 10.4208/csiam-am.2020.

[65] J. Jin and A. D. & E. Culurciello, “flattened convolutional neural networks for feed forward acceleration1412.5474.pdf,” *3rd Int. Conf. Learn. Represent. ICLR 2015 - Work. Track Proc., no. 2014, pp. 1–11*, 2015.

[66] A. Novikov, D. Podoprikhin, A. Osokin, and D. Vetrov, “Tensorizing Neural Networks,” *Arxiv*, 2015.

[67] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, “Convolutional Long Short-Term Memory, Fully Connected Deep Neural Networks,” *2015 IEEE Int Conf Acoust Speech Signal Process Icassp*, pp. 4580–4584, 2015, doi: 10.1109/icassp.2015.7178838.

[68] N. B. Nordsborg, H. G. Espinosa, and D. V. Thiel, “Estimating Energy Expenditure During Front Crawl Swimming Using Accelerometers,” *Procedia Engineer*, vol. 72, pp. 132–137, 2014, doi: 10.1016/j.proeng.2014.06.024.

[69] S. Haque, A. S. A. Rabby, M. A. Laboni, N. Neehal, and S. A. Hossain, “Recent Trends in Image Processing and Pattern Recognition, Second International Conference, RTIP2R 2018, Solapur, India, December 21–22, 2018, Revised Selected Papers, Part I,” *Comm Com Inf Sc*, pp. 186–193, 2019, doi: 10.1007/978-981-13-9181-1_17.

[70] J. Hosang, R. Benenson, P. Dollar, and B. Schiele, “What Makes for Effective Detection Proposals?,” *IEEE T Pattern Anal*, vol. 38, no. 4, pp. 814–830, 2015, doi: 10.1109/tpami.2015.2465908.

[71] A. Chaudhari, O. Dalvi, O. Ramade, and D. Ambawade, “Yog-Guru: Real-Time Yoga Pose Correction System Using Deep Learning Methods,” *2021 Int Conf Commun Information Comput Technology Iccict*, vol. 00, pp. 1–6, 2021, doi: 10.1109/iccict50803.2021.9509937.

[72] G. Gkioxari, B. Hariharan, R. Girshick, and J. Malik, “R-CNNs for Pose Estimation and Action Detection,” *Arxiv*, 2014.

[73] R. Girdhar, G. Gkioxari, L. Torresani, M. Paluri, and D. Tran, “Detect-and-Track: Efficient Pose Estimation in Videos,” *2018 IEEE Cvf Conf Comput Vis Pattern Recognit*, pp. 350–359, 2018, doi: 10.1109/cvpr.2018.00044.

[74] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, “Selective Search for Object Recognition,” *Int J Comput Vision*, vol. 104, no. 2, pp. 154–171, 2013, doi: 10.1007/s11263-013-0620-5.

[75] Y. Zhao, G. Karypis, and U. Fayyad, “Hierarchical Clustering Algorithms for Document Datasets,” *Data Min Knowl Disc*, vol. 10, no. 2, pp. 141–168, 2005, doi: 10.1007/s10618-005-0361-3.

[76] C. Briggs, Z. Fan, and P. Andras, “Federated learning with hierarchical clustering of local updates to improve training on non-IID data,” *2020 Int Jt Conf Neural Networks Ijcnn*, vol. 00, pp. 1–9, 2020, doi: 10.1109/ijcnn48605.2020.9207469.

[77] A. Nibali, Z. He, S. Morgan, and L. Prendergast, “3D Human Pose Estimation with 2D Marginal Heatmaps,” *Arxiv*, 2018.

[78] M. Oberweger, M. Rad, and V. Lepetit, “Making Deep Heatmaps Robust to Partial Occlusions for 3D Object Pose Estimation,” *Arxiv*, 2018.

[79] A. Eklund, “Cascade Mask R-CNN and Keypoint Detection used in Floorplan Parsing,” *Examensarbete 30 hp Juli 2020*, 2020.

[80] N. Liu, T. Celik, and H.-C. Li, “Gated Ladder-Shaped Feature Pyramid Network for Object Detection in Optical Remote Sensing Images,” *IEEE Geosci Remote S*, vol. 19, pp. 1–5, 2022, doi: 10.1109/lgrs.2020.3046137.

[81] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature Pyramid Networks for Object Detection,” *2017 IEEE Conf Comput Vis Pattern Recognit Cvpr*, pp. 936–944, 2017, doi: 10.1109/cvpr.2017.106.

[82] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *Arxiv*, 2014.

[83] D. Yu, H. Wang, P. Chen, and Z. Wei, “Rough Sets and Knowledge Technology, 9th International Conference, RSKT 2014, Shanghai, China, October 24-26, 2014, Proceedings,” *Lect Notes Comput Sc*, pp. 364–375, 2014, doi: 10.1007/978-3-319-11740-9_34.

[84] N. D. Reddy, “Classification of Dermoscopy Images using Deep Learning,” *Arxiv*, 2018.

[85] J. Howard and S. Gugger, “Fastai: A Layered API for Deep Learning,” *Information*, vol. 11, no. 2, p. 108, 2020, doi: 10.3390/info11020108.

[86] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *2014 IEEE Conf Comput Vis Pattern Recognit*, pp. 580–587, 2014, doi: 10.1109/cvpr.2014.81.

[87] J. Howard and S. Ruder, “Universal Language Model Fine-tuning for Text Classification,” *Arxiv*, 2018.

[88] H.-T. Chen, Y.-Z. He, C.-C. Hsu, C.-L. Chou, S.-Y. Lee, and B.-S. P. Lin, “MultiMedia Modeling, 20th Anniversary International Conference, MMM 2014, Dublin, Ireland, January 6-10, 2014, Proceedings, Part I,” *Lect Notes Comput Sc*, pp. 496–505, 2014, doi: 10.1007/978-3-319-04114-8_42.

[89] Hassan, Hussein Ayman. “Automatic Feedback For Physiotherapy Exercises Based On PoseNet.” (2020).

[90] H.-T. Chen, Y.-Z. He, C.-L. Chou, S.-Y. Lee, B.-S. P. Lin, and J.-Y. Yu, “Computer-assisted self-training system for sports exercise using kinects,” *2013 IEEE Int Conf Multimedia Expo Work Icmew*, pp. 1–4, 2013, doi: 10.1109/icmew.2013.6618307.

[91] H. Wang, “Neural Network-Oriented Big Data Model for Yoga Movement Recognition,” *Comput Intel Neurosc*, vol. 2021, p. 4334024, 2021, doi: 10.1155/2021/4334024.

[92] S. Kothari, “Yoga Pose Classification Using Deep Learning,” 2020, doi: 10.31979/etd.rkgu-pc9k.

[93] H.-T. Chen, Y.-Z. He, and C.-C. Hsu, “Computer-assisted yoga training system,” Multimed Tools Appl, vol. 77, no. 18, pp. 23969–23991, 2018, doi: 10.1007/s11042-018-5721-2.

[94] Mr. A. Kingsly Jabakumar. (2019). Enhanced QoS and QoE Support through Energy Efficient Handover Algorithm for UMTS Architectures. International Journal of New Practices in Management and Engineering, 8(01), 01 - 07. https://doi.org/10.17762/ijnpme.v8i01.73

[95] Omondi, P., Ji-hoon, P., Cohen, D., Silva, C., & Tanaka, A. Deep Learning-Based Object Detection for Autonomous Vehicles. Kuwait Journal of Machine Learning, 1(4). Retrieved from http://kuwaitjournals.com/index.php/kjml/article/view/149

[96] Kawale, S., Dhabliya, D., & Yenurkar, G. (2022). Analysis and simulation of sound classification system using machine learning techniques. Paper presented at the 2022 International Conference on Emerging Trends in Engineering and Medical Sciences, ICETEMS 2022, 407-412. doi:10.1109/ICETEMS56252.2022.10093281 Retrieved from www.scopus.com