Grenze International Journal of Engineering and Technology, June Issue

A Deep Learning Approach for Standing Yoga Pose Recognition using YOLOv11-based Keypoint Detection

Pavithrakumar L[1] and Jayashree N V[2]
[1-2]Department of Computer Applications, JSS Science and Technology University, Mysuru, Karnataka, 570006, India
Email: pkl@jssstuniv.in

***Abstract*— The current research focuses on increasing the need for automated systems that enable real-time recognition of standing yoga poses because of the growing interest in yoga and smart fitness devices. Traditional machine learning algorithms rely on manually extracted features, whereas current state-of-the-art deep learning methods tend to use two different models – one for keypoint detection and another for classification of yoga poses. In order to overcome these problems, we propose a YOLOv11-based framework to detect and classify standing yoga poses in real-time. We collected a custom dataset with 2,250 images of nine different standing yoga poses using several people and various camera angles. Our algorithm applies YOLOv11 backbone, FPN-PAN neck, and specific pose head in order to perform effective keypoint detection and pose classification. We have experimentally shown that our method is better than YOLOv5 and YOLOv8. For a split of 70:20 train/validation, the achievements are 98.6% accuracy, 99.0% precision, 98.2% recall, 98.0% F1 score, and 98.6% mAP, whereas for a split of 80:10, they are 98.4% accuracy and 98.0% mAP.**

***Keywords*— Computer Vision, Human Pose Estimation, Keypoint Detection, FPN-PAN, Skeleton-Based Analysis, MediaPipe, YOLO, OpenPose.**

I. INTRODUCTION

With the advent of smart digital health devices and intelligent fitness equipment, the process of participating in physical activities such as yoga has considerably changed. Yoga is an ancient exercise, originally from India, and it is renowned for the improvement of body flexibility, mind relaxation, and general well-being. Statistics provided in international health surveys show that over 300 million people practice yoga globally, with the steady average growth rate equal to about 6–8% annually [4]. Additionally, the yoga industry's market value is forecasted to exceed USD 200 billion by 2030 [2]. In India, yoga popularity has increased considerably since declaring International Yoga Day, and there are already 30 million practitioners who regularly take part in yoga gatherings [7]. However, inappropriate pose alignment can reduce the efficiency of the exercise and increase the likelihood of injuries. As a result, the need for automated systems capable of recognizing and analyzing the poses arises due to the absence of real-time feedback and expert supervision.

The process of human pose estimation and keypoint detection is one of the essential computer vision tasks associated with the analysis of human body movements. Human pose estimation implies estimating the location of the body through the detection of critical joints like shoulders, elbows, hips, knees, and ankles. Keypoint detection, in its turn, refers to the identification of the selected anatomical points on the image or video.

Grenze ID: 01.GIJET.12.2.25_25

After extracting the keypoints, pose classification techniques can be applied to classify the detected poses into different predefined categories, such as yoga poses. This methodology allows applying the obtained results in the field of fitness, rehabilitation, sport analytics, and human-computer interaction. Some of the early studies include works on DeepPose [1] and OpenPose [2], while other models like stacked hourglass networks [3] and baselines [4] were introduced later.

Various deep learning and computer vision studies have recently emerged in the field of yoga pose detection and classification. These include techniques involving the use of convolutional neural networks (CNN), OpenPose, and MediaPipe models for extracting the skeletal key-points and then using machine learning-based pose classification [21], [23]. Some popular yoga pose datasets include Yoga-82, Yoga-10, and COCO Keypoints dataset. However, recent techniques have shifted towards adapting the principles of object detection models to pose estimation problems with improved accuracy and in real-time speeds using YOLOv5 and YOLOv8 architectures [12], [13]. YOLO-Pose [8] and transformer-based networks [6], [7] have been explored to boost pose estimation capabilities and pose representation learning. Additionally, novel hybrid networks combining CNNs, transformers, and graph neural networks to increase classification accuracy have also been explored [5], [7]. However, in many of these models, a multi-stage pipeline has been adopted with pose estimation and classification being performed using separate models.

There remain various issues with the development of an effective yoga pose recognition model. One of these concerns the limited availability of real-time yoga datasets, specifically Indian yoga poses, covering a wide range of body types, different lighting conditions, and poses. Many existing datasets lack diversity from the yoga perspective or were captured in artificial settings, reducing their real-life applicability. In addition, some pose recognition models require the use of several individual models where pose estimation could be achieved using MoveNet or PoseNet, followed by classifiers for pose classification, leading to increased computational costs and latency [10], [21]. This makes it difficult to achieve real-time performance. There also remains no universal model which can perform pose estimation and pose classification tasks.

In this paper, we present an efficient deep learning approach for standing yoga pose recognition based on YOLOv11-based keypoint detection. Unlike conventional multi-stage models, our model adopts a one-model framework that performs both pose estimation and classification. Specifically, this architecture involves using the fast keypoint detection capability of YOLOv11 for detecting the 17 human body keypoints, making real-time yoga pose recognition possible [11].

II. Related Work

The area of human pose estimation has developed quickly alongside advances in deep learning and computer vision. First works in this direction included the use of deep learning methods for estimating human joint coordinates, replacing traditional manual techniques based on hand-crafted features [1]. Next steps were related to the development of algorithms for detecting multiple body keypoints using part affinity fields like OpenPose [2]. Stacked hourglass networks [3], simple baselines [4], and Mask R-CNN [5] contributed to higher estimation accuracy with multistage feature extraction and better localization approaches. Although most of these models provided excellent results, they required powerful computers and long execution times, preventing their usage for real-time applications.

More recently, there were attempts to leverage transformer architectures for enhancing feature representation. The HumanPoseNet [6] used transformer structures to capture long-range dependencies, whereas hybrid models that combined transformer and Graph Convolutional Network (GCN) architectures led to improved performance in 3D pose estimation tasks [7]. While these architectures provide a very accurate output, they need powerful hardware, large amounts of data, and longer training periods, which makes them impractical for real-time applications in the field of fitness.

A number of works related to machine learning and deep learning applied to yoga pose recognition have been published. Conventional approaches based on manually engineered skeletal features and classifiers such as SVM, random forest, and KNN had issues with recognizing more complex body poses and estimating their orientations [14], [15]. Approaches that utilized Convolution Neural Networks (CNN) for automatic learning of features from images showed higher results [16], [17]. Systems such as DeepYoga [18] and some other AI-based posture correction systems also achieved good results in yoga pose recognition. However, a number of those approaches implemented discrete stages of keypoint detection and classification, leading to more complex algorithms.

Limitations in available datasets remain one of the critical issues in yoga pose recognition. Popular public databases include Yoga-82, Yoga-10, and COCO Keypoints [21], [22]. The main problem with those datasets is

that they contain relatively few yoga poses in relatively static settings with no variety in clothing, lighting conditions, body type, etc. Moreover, they do not include examples of Indian yoga practice environment. That may reduce generalization abilities in the field of yoga recognition.

In order to improve the speed of real-time processing, recent developments tend to use YOLO-type network structures. Faster inference compared to the existing multi-step methods can be seen from the implementations of YOLO-Pose [8], YOLOv5 [12], and YOLOv8 [13]. However, the earlier YOLO models faced some problems in recognizing the fine-grained body joint points. In this regard, the current work uses YOLOv11 for the recognition of yoga poses while standing. By combining both the steps of pose estimation and classification in one architecture, YOLOv11 allows for a faster and more precise detection.

III. Data Description

A major contribution made by this research effort includes the development of a unique set of data for recognizing standing yoga poses in real time. The fact that most publicly available databases have few variations in viewpoints as well as standing poses being combined with non-standing poses is one of the reasons for which this particular dataset was created. For better quality of images, this dataset is obtained by using a mobile camera indoors with proper illumination. In order to make the dataset more generalized, several participants, including the researchers themselves along with their colleagues, participated in data collection, ensuring variation in terms of body size and flexibility among others. Each pose was recorded from three different viewpoints, including front, left, and right viewpoints. This dataset includes nine standing yoga poses – Namaskara Pose, Tree Pose, Chair Pose, Goddess Pose, Raised Arm Pose, Side Bend Pose, Triangle Pose, Warrior I Pose, and Warrior II Pose. For each pose, around 250 images were collected, making up a total of 2,250 images. Unlike other publicly available databases like Yoga-82, the current database is specifically focused on standing yoga poses.

Fig 1: Dataset Sample Frames

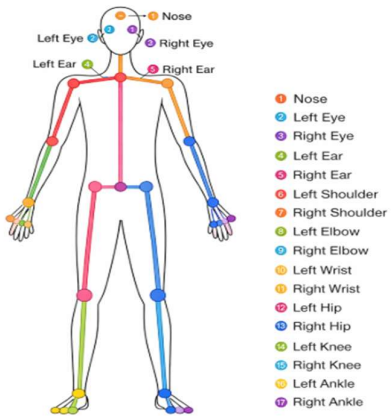

Fig 2: Human Skeleton with 17 Keypoints

Once all datasets have been collected, all images were manually annotated to enable the use of the proposed model. Manual annotations were done using the Roboflow platform by annotating all images with the help of bounding boxes and key points, with the total number of keypoints being 17 per image, as is required in the COCO Keypoint Format. The keypoints included the following human body parts: Nose, left eye, right eye, left ear, right ear, left shoulder, right shoulder, left elbow, right elbow, left wrist, right wrist, left hip, right hip, left knee, right knee, left ankle, and right ankle. By specifying these body joints, a skeletal structure of the human body can be formed, which would allow the model to analyze any changes in joint position and pose during yoga. All annotations were done in the format required for the proposed YOLOv11 algorithm, i.e., bounding box coordinate values, keypoint coordinate values (x, y), visibility values, and pose class values.

III. Methodology

A. Proposed YOLOv11 Pose Estimation Architecture

The design architecture of the system proposed here has been based on the YOLOv11 human pose estimation algorithm, with an aim to facilitate real-time human pose detection and classification. Unlike conventional

approaches that adopt a multi-step strategy for human pose detection and classification, the current approach aims to integrate all steps within one unified architecture.

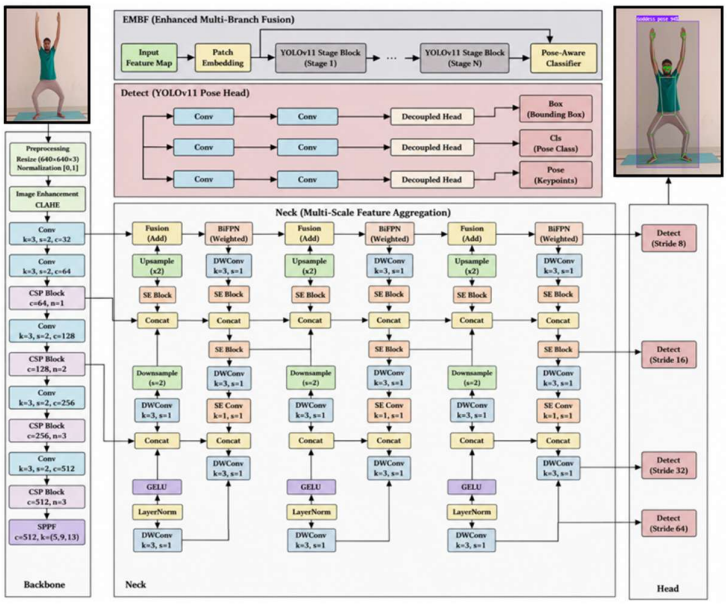

Fig 3. Proposed YOLOv11 Pose Estimation Architecture

Pre-processing and Image Enhancement

The process involved in the proposed pipeline begins with the use of images obtained using mobile phones, where problems like illumination variations, shadows, and varying backgrounds within indoor scenes may affect the detectability of the body joints used for detecting yoga poses. Contrast Limited Adaptive Histogram Equalization (CLAHE) is used as part of preprocessing before training the YOLOv11 pose model to address this problem. CLAHE makes use of local contrast improvement by dividing an image into smaller sections and processing these separately, along with application of a clipping limit that prevents any increase in noise level. After clipping, pixels are re-distributed into histogram bins, and then histogram equalization and bilinear interpolation take place. As a result, the visibility of key anatomical areas, especially the arms, legs, and body joints, is significantly improved.

$$g(p,q) = \frac{CDF\,(f(p,q)) - CDF_{min}}{(T_p \times T_q) - CDF_{min}} \times (L\text{-}1) \qquad (1)$$

$$H'_{(i)} = \min\left(H_{(i)},\, C\right) + \frac{F}{L} \qquad (2)$$

where the Cumulative Distribution Function (CDF) is computed from the clipped histogram $H'_{(i)}$ *and* $H_{(i)}$ represents the original histogram. C denotes the clip limit used to restrict contrast enhancement, i represents the pixel intensity. The excess pixels E generated after clipping are redistributed uniformly across all histogram bins. L denotes the total number of intensity levels, $T_p \times T_q$ represents the size of the local tile., and Cdf_{min} is the minimum non-zero value of the cumulative distribution function within the tile.

YOLOv11 Backbone for Feature Extraction

The backbone network aims to abstract spatial and semantic characteristics of the input yoga image for accurate pose estimation. In this research, a backbone model based on YOLOv11 architecture is utilized, which includes convolutional layers, cross stage partial (CSP) blocks, skip connections, and a spatial pyramid pooling fast (SPPF) layer. The input image with a resolution of 640 × 640 × 3 undergoes initial processing using convolutional layers, which generate both low-level and high-level features such as body shape, joint positions, and postures.

Convolution can be represented mathematically as:

$$F(x,y) = \sum_{i=0}^{k} \sum_{j=0}^{k} I(x+i, y+j) \cdot W(i,j) \quad (3)$$

where I(x,y) represents the input image, W(i,j) is the convolution kernel, and k is the kernel size.
Batch normalization ensures stability during training and effective feature extraction:

$$\hat{x} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}} \quad (4)$$

where μ and σ represent the batch mean and variance.
The CSP blocks simplify computations and facilitate gradient transfer through feature map segmentation and fusion:

$$F_{CSP} = Concat(F_1, F(F_2)) \quad (5)$$

where F1 represents the shortcut connection, while F(F2) symbolizes the altered feature map.
Lastly, the SPPF layer integrates multi-scale context information to provide better body feature representations:

$$F_{SPPF} = Concat(P_1(F), P_2(F), P_3(F), F) \quad (6)$$

where Pi(F) represents pooling at different levels.The extracted feature maps are then forwarded to the neck for further processing and pose prediction.

Neck for Feature Aggregation
Neck module acts as a middle layer between the backbone and the prediction head. It performs feature map aggregation, integration, and refinement based on multi-scale feature maps obtained from the backbone. Since the joints in yoga poses occur at various resolutions and locations, detecting body parts from one feature map would result in wrong detections. As a way of overcoming the above problem, the neck layer uses a technique of multi-scale feature fusion, where it integrates semantic and spatial features, hence improving the localization of body joints. The fusion of features can be mathematically presented as follows: where F1 and F2 are two different backbone feature maps.

$$F_{fusion} = Conv(Concat(F_1, F_2)) \quad (7)$$

where F1 and F2 represent feature maps from different backbone layers.
The Neck layer commonly uses the Feature Pyramid Network (FPN) and Path Aggregation Network (PAN) in performing feature map aggregation. Feature Pyramid Network propagates features using a top-down approach. This involves propagating features that have strong semantics from deeper networks to shallower ones. The propagation of the features allows the network to detect smaller joints and preserve body structure understanding. Upsampling is carried out as follows:

$$F_{up} = Upsample(F_{high}) \quad (8)$$

The propagated feature maps are combined with lower features using the following equation:

$$F_{fpn} = Concat(F_{up}, F_{low}) \quad (9)$$

Where Fhigh is a high-level feature map and Flow refers to low-level features.
Path Aggregation Network works together with FPN by adding bottom-up feature propagation that enhances localization using spatial information flow from lower to higher levels. This makes the network able to detect key points from complex poses of yoga better than before. Refinement using Path Aggregation Network is performed as follows:

$$F_{pan} = Conv(F_{fpn}) \quad (10)$$

The output from the neck layer is the fusion of multi-scale feature maps {P3, P4, P5}. The output maps are then passed to the prediction head.

Head for Pose Estimation and Recognition
The head block makes up the last element of the proposed YOLOv11-based framework for pose estimation, where multi-scale feature maps, generated from the neck block, are converted into meaningful predictions. It is

designed to simultaneously detect, classify, and estimate key points of the subject at once during a forward pass through the network.
Multi-scale feature maps {P3, P4, P5} form an input to the head block and then are processed with a series of convolution layers with convolution, normalization, and SiLU activations. This way, the features become predictable.
As mentioned earlier, the head block consists of three parallel paths responsible for specific tasks:
Bounding Box Regression: Firstly, the network predicts a position of the human object on the picture by calculating its center coordinates, width, and height:

$$B = (x, y, w, h) \qquad (11)$$

Here in (x, y) are the center coordinates of the bounding box, whereas w and h are the width and height of it, respectively. Thus, the network is able to determine accurately the location of the person making the pose.
Pose Classification: Secondly, the model assigns the detected object to one of predefined yoga poses. To do that, it uses a softmax activation to get classification probabilities:

$$P(c|x) = \frac{e^{z_c}}{\sum_{k=1}^{C} e^{z_k}} \qquad (12)$$

whereby C is the total number of classes and z_c – score value of the class c.
Keypoint Estimation: Finally, the network calculates coordinates of 17 body keypoints corresponding to such joints as shoulders, elbows, hips, knees, and ankles. Keypoints are calculated according to the following formula:

$$K_i = (x_i, y_i, v_i) \qquad (13)$$

where x_i, y_i, are coordinate values, and v_i is a visibility score of the i-th keypoint.
An array of 17 keypoints looks as follows:

$$K = \{K1, K2, \ldots, K17\} \qquad (14)$$

Key points can be used for constructing a skeleton of the pose.
Final Prediction Output
All results from three branches are then aggregated to form the final prediction:

$$O = \{B, c, K, S\} \qquad (15)$$

where B stands for a bounding box, c is a predicted pose class, K denotes key points set, and S – overall score value.

IV. Results and Discussion

A. Experimental Setup

This section discusses the experimental setup used to evaluate the developed standing yoga pose recognition system using YOLOv11. A dataset containing 2,250 images from nine different yoga poses was used to train and test the system. The images were taken from several participants under indoor controlled conditions and from three different views (frontal, left, and right). Contrast enhancement was achieved through CLAHE.
The implementation used Python, PyTorch, Ultralytics YOLO, OpenCV, and Roboflow libraries. The training process took place in Google Colab using NVIDIA T4 GPU. The images fed into the model had a constant resolution of 640 × 640 pixels.

Dataset Splitting and Model Training Strategy
A number of data splitting strategies, including 70:30, 80:20, and 60:40 ratios, were tested for the customized yoga data set to determine the generalization power of the developed system under different training scenarios. For each ratio, a proportion of the train data set was reserved for validation purposes during the training stage. Equal proportions of all pose categories were included in each data partition to avoid any possible class imbalance problem. The YOLOv11 pose detector model was then trained for 50 epochs using the Adam optimizer with a fixed learning rate and proper batch size. Multi-task loss function including bounding box loss, classification loss, and keypoint loss was used to optimize both keypoint detection and pose classification. Convergence performance was monitored regularly to prevent overfitting.

B. Performance Analysis of Proposed YOLO Base Model

For evaluating the efficiency of the proposed methodology, a comparison has been done among various YOLO-based models under varied dataset split conditions. The focus of the analysis has been on accuracy, precision, recall, F1-score, and mean Average Precision (mAP). All the above-mentioned parameters help us to measure the efficacy of the model for detecting, classifying, and localizing human poses accurately. In this process, it is our aim not only to identify the most accurate model but also to measure the level of generality and consistency of the model. These results have been elaborated upon in Table 1.

TABLE I: PERFORMANCE COMPARISON OF YOLO MODELS

Model	Train V/S Test	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	mAP
YOLOv5 Pose	70:20	0.932	0.934	0.929	0.923	0.931
	80:10	0.930	0.927	0.923	0.922	0.929
YOLOv8 Pose	70:20	0.951	0.955	0.950	0.950	0.951
	80:10	0.950	0.953	0.950	0.944	0.950
YOLOv11(Proposed)	70:20	0.986	0.990	0.982	0.980	0.986
	80:10	0.984	0.988	0.978	0.970	0.980

As per Table 1, the proposed YOLOv11 model outperforms both YOLOv5 ,YOLOv8 models for all the performance metrics. YOLOv11 achieves more accuracy, precision, recall, F1-score, and mAP as compared to other models. While the performance of YOLOv8 is better than that of YOLOv5, it fails to perform at par with YOLOv11 in terms of detecting and localizing the fine-grained body joints of humans. The reason behind better performance of YOLOv11 is due to the enhanced architecture of its backbone, efficient feature fusion, and accurate keypoints detection technique.

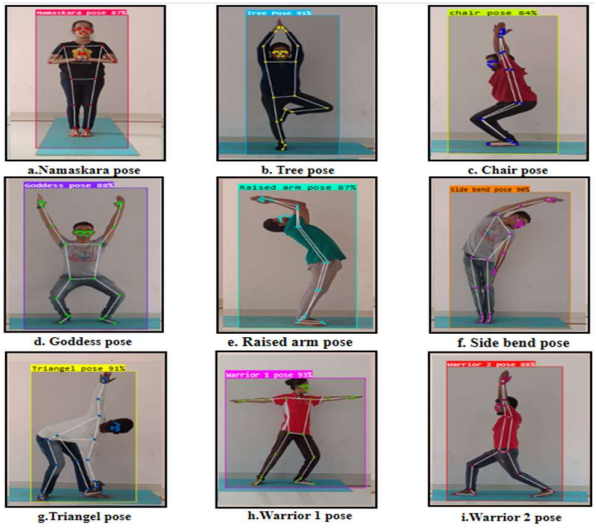

Figure 4: Sample Output of Proposed YOLOv11 Pose Model

The qualitative results shown in Figure 4 illustrate the effectiveness of the proposed YOLOv11 pose model in real-world scenarios. The model accurately detects human body keypoints and constructs a clear skeletal representation of the pose. It performs well under different conditions, including variations in background, lighting, and pose orientation. The predictions are consistent and reliable, demonstrating the model's strong generalization capability. Additionally, the system operates efficiently in real time, making it suitable for practical applications. These results confirm that the proposed model is robust, accurate, and capable of delivering dependable pose recognition performance.

C. Comparative Analysis with Traditional Machine Learning Models

Further, to evaluate the proposed YOLOv11 model, a comparative study was done against traditional machine learning algorithms using keypoints extracted through MediaPipe as input features for classification of yoga poses. Models used in the experiment were SVM, KNN, Decision Tree, Random Forest, and Gradient Boosting. Performance results of the above models are provided in Table 2.

TABLE II: PERFORMANCE OF MACHINE LEARNING MODELS

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Medianet + SVM (Linear)	0.8904	0.89	0.91	0.90
Medianet + SVM (RBF)	0.6549	0.71	088	079
Medianet + SVM (Polynomial)	0.9904	0.99	0.99	0.98
Medianet + K-Nearest Neighbors	0.7980	0.80	0.80	0.80
Medianet + Decision Tree	0.5536	0.55	0.55	0.55
Medianet + Random Forest	0.8096	0.82	0.81	0.81
Medianet + Gradient Boosting	0.8464	0.85	0.85	0.85

Even though some models achieved decent results, however, conventional machine learning techniques are highly dependent upon handcrafted key-points and face problems while dealing with complex spatial relations between body joints. Also, a few models were suffering from overfitting issues and lacked generalization capacity. In contrast to these models, YOLOv11 achieves better accuracy and provides end-to-end yoga pose detection.

D. Comparative Analysis with Deep Learning Models

The deep learning models were tested in order to solve the limitations of traditional machine learning methods using autonomous learning of features from MediaPipe keypoints. The models analyzed include MLP, 1-D CNN, LSTM, ResNet-MLP, and MobileNet, which are discussed here along with their performance results in Table 3.

TABLE III: PERFORMANCE OF DEEP LEARNING MODELS

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Medianet + MLP	0.916	0.89	0.91	0.90
Medianet +1-D CNN	0.907	0.90	0.91	0.92
Medianet + LSTM	0.955	0.93	0.95	0.96
Medianet + ResNet-MLPs	0.945	0.91	0.90	0.93
Medianet + MobileNet	0.899	0.87	0.84	0.88

From all of them, the LSTM network model showed better results by efficiently identifying relations between different body joints. However, these models still use different keypoint detection algorithms like MediaPipe, resulting in the creation of a two-step pipeline and hence making the computation complex. On the other hand, the proposed YOLOv11 model provides an end-to-end solution for the real-time identification of yoga poses.

E. Training Performance Analysis

Training performance of the proposed YOLOv11-based model for pose estimation is investigated in order to understand its learning behavior and convergence in respect of training epochs. The model is trained with carefully chosen parameters to make sure that the learning process is stable and the detection performance of the model is improved. During the training process, several performance measures, along with different types of loss functions, are being analyzed.

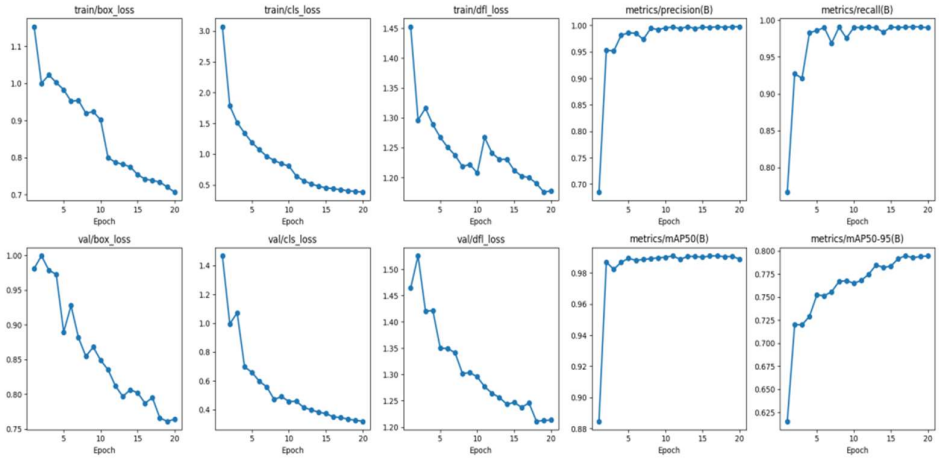

Figure 5: YOLOv11 Model Learning Curves (Loss, Precision, Recall, mAP)

It can be seen that all the values of loss functions tend to converge and decrease as the number of training epochs increases. In addition, the curves of precision and recall increase monotonically and reach nearly optimal values. Therefore, it can be concluded that the learning of the model occurs effectively and efficiently while converging properly. It should be noted that the value of mean average precision (mAP) increases as well and reaches near-optimal levels. Based on the analysis above, the proposed YOLOv11 model can be considered stable and well-optimized.

VI. CONCLUSION

The research work develops a deep learning system based on YOLOv11 for recognising yoga postures in standing position, combining the capabilities of object detection, keypoint localization, and posture classification. The system is an alternative to existing machine learning algorithms which require handcrafted features, and classical deep learning techniques that demand separate steps of processing. Experimental results prove the effectiveness of the framework, which achieves accuracy of 98.6% and mAP of 98.6% on the custom yoga standing posture dataset. The method shows high stability with respect to changes in the pose angles, viewpoint, background, and lighting. It is able to detect human keypoints correctly, building accurate human skeleton models. Moreover, due to its computational efficiency, the system can be implemented for practical purposes such as health and fitness tracking, posture correcting, yoga posture training, and intelligent healthcare. Overall, the system presented in the research is highly reliable and efficient in recognising yoga postures and has multiple real-life applications.

REFERENCES

[1] A. Toshev and C. Szegedy, "DeepPose: Human pose estimation via deep neural networks," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2014. https://doi.org/10.1109/CVPR.2014.214

[2] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2D pose estimation using part affinity fields," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2017. https://doi.org/10.1109/CVPR.2017.143

[3] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in Proc. European Conf. Computer Vision (ECCV), 2016.https://doi.org/10.1007/978-3-319-46484-8_29

[4] B. Xiao, H. Wu, and Y. Wei, "Simple baselines for human pose estimation and tracking," in Proc. European Conf. Computer Vision (ECCV), 2018.https://doi.org/10.1007/978-3-030-01231-1_29

[5] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in Proc. IEEE Int. Conf. Computer Vision (ICCV), 2017.https://doi.org/10.1109/ICCV.2017.322

[6] V. Gupta, A. Yadav, and D. K. Vishwakarma, "HumanPoseNet: An all-transformer architecture for pose estimation with efficient patch expansion and attentional feature refinement," Expert Systems with Applications, 2023.https://doi.org/10.1016/j.eswa.2023.122894

[7] X. Pan, G. Li, N. Zhang, and J. Li, "3D human pose estimation based on a hybrid approach of transformer and GCN-Former," Journal of Visual Communication and Image Representation, 2025. https://doi.org/10.1016/j.jvcir.2025.104696

[8] J. Ding, S. Niu, Z. Nie, and W. Zhu, "Research on human posture estimation algorithm based on YOLO-Pose," Sensors, vol. 24, no. 10, 2024.https://doi.org/10.3390/s24103036

[9] S. Cai, H. Xu, W. Cai, Y. Mo, and L. Wei, "A human pose estimation network based on YOLOv8 framework with efficient multi-scale receptive field and expanded feature pyramid network," Scientific Reports, 2025.https://doi.org/10.1038/s41598-025-00259-0

[10] D. Maji, S. Nagori, M. Mathew, and D. Poddar, "YOLO-Pose: Enhancing YOLO for multi-person pose estimation using object keypoint similarity loss," 2021. https://github.com/TexasInstruments/edgeaiyolov5

[11] Y. Zhou, S. Dong, H. Sheng, and W. Ke, "YED-Net: Yoga exercise dynamics monitoring with YOLOv11-ECA-enhanced detection and DeepSORT tracking," Applied Sciences, vol. 15, no. 13, 2025.https://doi.org/10.3390/app15137354

[12] A. Maqbullah, A. N. Handayani, and W. C. Kurniawan, "Yoga posture recognition and classification using YOLOv5," International Journal of Data and Software Engineering (IJODAS), vol. 6, no. 2, 2024.https://doi.org/10.56705/ijodas.v6i2.228

[13] P. Jaronde, C. Khapekar, D. Bisen, G. Ghate, and A. Shinde, "Yoga posture detection and correction using YOLOv8," International Journal for Research in Applied Science and Engineering Technology (IJRASET), 2025.https://doi.org/10.22214/ijraset.2025.68275

[14] P. S. Patil, "Yoga pose estimation using YOLO model," International Journal of Scientific Research in Engineering and Management (IJSREM), vol. 7, no. 10, pp. 1–11, Oct. 2023. https://doi.org/10.55041/IJSREM26536

[15] P. U. Pawar and H. H. Kulkarni, "Real-time yoga pose detection using AI," International Journal of Latest Technology in Engineering, Management & Applied Science (IJLTEMAS), vol. 14, no. 10, Oct. 2025.https://doi.org/10.51583/IJLTEMAS.2025.1410000020

[16] X. Meng and Z. Liu, “Yoga pose recognition using dual structure convolutional neural network,” PeerJ Computer Science, vol. 11, 2025.https://doi.org/10.7717/peerj-cs.2907
[17] S. Sangeetha and H. Kumar, “Recognizing yoga pose using deep learning,” International Journal of Innovative Science and Research Technology (IJISRT), vol. 9, no. 7, Jul. 2024. https://doi.org/10.38124/ijisrt/IJISRT24JUL1267
[18] R. Bansal, R. Sharma, P. Jain, R. Arora, S. Pal, and V. Vishal, “DeepYoga: Enhancing practice with a real-time yoga pose recognition system,” Engineering, Technology & Applied Science Research (ETASR), vol. 14, no. 6, pp. 17704–17710, Dec. 2024.https://doi.org/10.48084/etasr.8643
[19] P. Kulkarni, S. Gawai, S. Bhabad, and A. Patil, “Yoga pose recognition using deep learning,” in Proc. IEEE Int. Conf. Emerging Smart Computing & Informatics (ESCI), 2024. https://doi.org/10.1109/ESCI59607.2024.10497433
[20] S.-C. Yeh and C.-K. Yang, “Yoga pose recognition and motion analysis for a home-based fitness monitoring and health management system,” Signal, Image and Video Processing, vol. 19, 2025.
[21] D. Debalaxmi, D. K. Vishwakarma, and V. Ranga, “Analyzing yoga pose recognition: A comparison of MediaPipe and YOLO keypoint detection with ensemble techniques,” in Proc. 3rd Int. Conf. Applied Artificial Intelligence and Computing (ICAAIC), 2024. https://doi.org/10.1109/ICAAIC60222.2024.10574984
[22] M. J. H., C. H. K., K. S. Santhosh Kumar, and S. P. Shiva Prakash, “Comprehensive analysis of pose estimation and machine learning classifiers for precise yoga pose detection and classification,” Procedia Computer Science, 2025.https://doi.org/10.1016/j.procs.2025.04.592
[23] S. Mathur, “Machine learning approach to real-time yoga pose detection and validation using MediaPipe and PoseNet,” 2024, pp. 1015–1022.https://doi.org/10.48084/etasr.8643
[24] M. A. Pradhan, V. Tendolkar, S. Rengade, Y. Pazade, and V. Yeole, “Yoga pose detection and feedback generation,” Journal of Novel Research and Innovative Development (JNRID), vol. 3, no. 6, Jun. 2025.https://doi.org/10.55041/IJSREM26536
[25] D. M. Kishore, S. Bindu, and N. K. Manjunath, “Estimation of yoga postures using machine learning techniques,” 2022..https://doi.org/10.56705/ijodas.v6i2.228