Jurnal Ilmu Komputer dan Informasi (Journal of Computer Science and Information)
16/2 (2023), 115-124. DOI: http://dx.doi.org/10.21609/jiki.v16i2.1151

Yoga Pose Rating using Pose Estimation and Cosine Similarity

Ani Dwi Astuti, Tita Karlita, Rengga Asmara

Informatics and Computer Engineering Department, Politeknik Elektronika Negeri Surabaya, Surabaya, Jawa Timur 60111, Indonesia

E-mail: anidwi.ad1@gmail.com, tita@pens.ac.id, rengga@pens.ac.id

Abstract

One type of exercise that many people do today is yoga. However, doing yoga yourself without an instructor carries a risk of injury if not done correctly. This research proposes an application in the form of a website that can assess the accuracy of a person's yoga position, by using ResNet for pose estimation and cosine similarity for calculating the similarity of positions. The application will recognize a person's body pose and then compare it with the poses of professionals so that the accuracy of their position can be assessed. There are three types of datasets used, the first is the COCO dataset to train a pose estimation model so that it can recognize someone's pose, the second is a reference dataset that contains yoga poses performed by professionals, and the third is a dataset that contains pictures of yoga poses that are considered correct. There are 9 yoga poses used, namely Child's Pose, Swimmers, Downdog, Chair Pose, Crescent Lunge, Planks, Side Plank, Low Cobra, Namaste. The optimal pose estimation model has a precision value of 87% and a recall of 88.2%. The model was obtained using the Adam optimizer, 30 epochs, and a learning rate of 0.0001.

Keywords: *Yoga, Pose Estimation, Resnet, Cosine Similarity*

1. Introduction

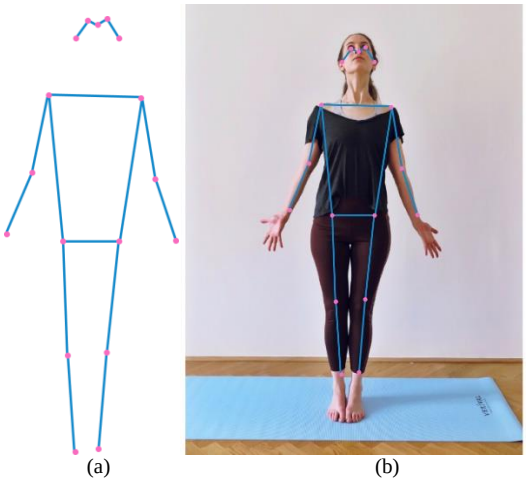
One type of exercise that more and more people are doing in recent years is yoga. Yoga is a practice that combines mind and body that has been practiced in the past 5,000 years ago in ancient Indian philosophy. Various styles of yoga incorporate physical postures, breathing techniques, and meditation or relaxation [1]. Yoga has become popular as a form of physical exercise based on poses that promotes control of the mind and body and promotes well-being in recent times. Yoga statistics show that there are 300 million people in the world who practice yoga. Interest in yoga also increases by 73% every year [2].

For a beginner in yoga, it is recommended to use a yoga instructor or teacher. Since the pandemic occurred, many yoga classes have been closed and people are advised to reduce physical contact as much as possible. And there are also some people who don't have the money or the right time to join a yoga class with an instructor. So that many people do yoga at home by watching yoga videos on various platforms. Even though doing yoga yourself without an instructor has a risk of injury if not done correctly.

Incorrect yoga poses are a serious problem for yoga participants. Recently, several studies have been conducted to address this problem, with different approaches or methods. There are several studies using classification methods to classify types of yoga poses using VGG[3], [4], MobileNet[3], Inception[3], DenseNet[3], CNN[5]–[8], and LSTM[5], [8]. Apart from that, there are also those who use pose estimation and similarity measurement. the pose estimation methods used are Surf algorithm[9], OpenPose[10], MoveNet[11], and PoseNet[12]. while the similarity measurement method used is cosine[10], [12], euclidean distance[9], new formula[11], and neural netwok[10].

2. Methodology

This research offers a solution in the form of an application that can assess the accuracy of a user's yoga position, by using ResNet for pose estimation and cosine similarity for position similarity calculations. This application has a feature to assess poses and movements that need to be corrected, and provides the information needed to avoid injury.



(a) (b)

Fig. 1. (a) Human pose skeleton (b)Pose estimation.

Pose estimation is done by identifying, finding, and tracking the body joints (keypoints) of the human skeleton pose in an image or video. Human pose skeleton is a set of coordinates to determine a person's pose. And a pair of coordinates is a limb. The depiction of a human pose skeleton as shown in Fig. 1(a), while Fig. 1(b) is a depiction for pose estimation. Pose estimation refers to computer vision techniques that detect human figures in pictures and videos, so that one can determine, for example, where a person's elbow appears in an image. Pose estimation only estimates where the main body joints are and does not recognize who is in the picture or video [13].

The architecture used to estimate poses is ResNet. Residual Neural Network (ResNet) is an Artificial Neural Network (ANN) of stacking residual blocks on top of one another to form a network. ResNet has many variants that run on the same concept but have a different number of layers, such as ResNet34, ResNet50, ResNet101, and ResNet152. Which means Resnet34 has 34 layers of neural networks, ResNet50 has 50 layers of neural networks, and so on. This research will use ResNet18 because it is lightweight. The ResNet18 architecture used can be seen in Figure 2.

Layer Name	Output Size	ResNet-18
conv1	112 × 112 × 64	7 × 7, 64, stride 2
conv2_x	56 × 56 × 64	3 × 3 max pool, stride 2 $\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 2$
conv3_x	28 × 28 × 128	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 2$
conv4_x	14 × 14 × 256	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 2$
conv5_x	7 × 7 × 512	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 2$
average pool	1 × 1 × 512	7 × 7 average pool
fully connected	1000	512 × 1000 fully connections
softmax	1000	

Fig. 2. ResNet18 architecture.

Cosine similarity, one of similarity measurement, is a metric used to measure two objects. Mathematically, this algorithm measures the cosine of the angle between two projected vectors in a multidimensional space. The smaller the angle, the higher the cosine similarity [14].

$$similarity(A,B) = \frac{A \cdot B}{||A|| \times ||B||} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}} \quad (1)$$

Equation (1) is the formula for calculating cosine similarity. Where Ai and Bi are components of vectors A and B respectively. The resulting values range between -1 and 1, where -1 is very different and 1 is very similar. In this study, cosine similarity was used to assess the similarity of used yoga poses with professional yoga poses by comparing the angles of each user's limbs with professional limbs. So that the difference in people's height will have no effect, because what is being compared is the angle.

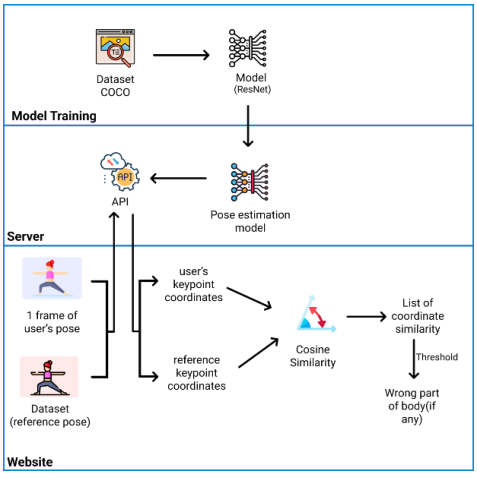


Fig. 3. Application System Design.

Figure 3 is the system design of the application to be made. The initial stage that must be done is to train a model so that it can recognize human body poses through keypoints. After the model is created, the model is run on the server using the API. Through the API model can be used in website applications. After that the website will send a picture of the user's pose and receive keypoint coordinates (collection of points) that represent the user's body pose. These coordinates will be compared with the coordinates of the reference pose and the similarity will be calculated using cosine similarity. After getting the values of similarity or resemblance for each member of the body, these values will be compared with the threshold value. If the similarity value of a limb is less than the threshold, then the limb is considered wrong. Then the website will display incorrect limbs to avoid injury due to incorrect limb movements or positions when doing yoga and to make it easier for users to correct their body position.

2.1. Datasets

The first dataset used is an opensource dataset called COCO. The dataset is in the form of images and dot annotations in the form of JSON files. The dataset has 118 thousand images for training data, 5 thousand for validation data, and 41 thousand for test data. The size and resolution of the images in this dataset vary widely. Overall, this dataset has 91 categories. In addition, there are also 250 thousand people in the image dataset which are annotated with keypoints. Each image has 17 keypoints, including 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.

Fig. 4. Referenced Pose Dataset

Meanwhile, the second dataset is a picture of an instructor doing the yoga pose in Figure 4. The image is used as a reference or the correct pose. There are 9 yoga poses used, namely Child's Pose, Swimmers, Downdog, Chair Pose, Crescent Lunge, Planks, Side Plank, Low Cobra, Namaste. The number of images to be used is one image for each yoga pose to be used. The image was taken by taking a screenshot of the video showing an instructor doing yoga movements. For the criteria for selecting a yoga video, the author chose a video where the video is taken with a straight view, the instructor's entire body can be seen from top to bottom, and with a simple background so that it only highlights the yoga poses being performed. The screenshot of the yoga pose video is a 1920x1080 image. After that, the image is cropped with a 3:2 ratio according to the size of the image on the website application mockup that was created, resulting in an image with dimensions of 1350x900 pixels with the .png image file format.

Fig. 5. Images for threshold determination.

The third dataset is images of poses that are considered correct. This image data will later be used to determine the threshold for determining whether the user's yoga pose is right or wrong. For each pose there are 12 images. In Figure 5 there are several examples of datasets used to determine thresholds. Images are obtained from combined open-source datasets and downloaded from the internet. The size and resolution of each image varies.

2.2. Models training

The initial stage that must be carried out is the model training stage. The model training flowchart for pose estimation is in Figure 6. The output of this process is a model that can recognize the pose of the human body from an image through keypoints. The model training process will be carried out using the Python programming language in the Google Colaboratory notebook environment, using an

Intel(R) Xeon(R) Processor, Nvidia Tesla K80 GPU, 12.68 GB of RAM, and 78.19 GB of storage.

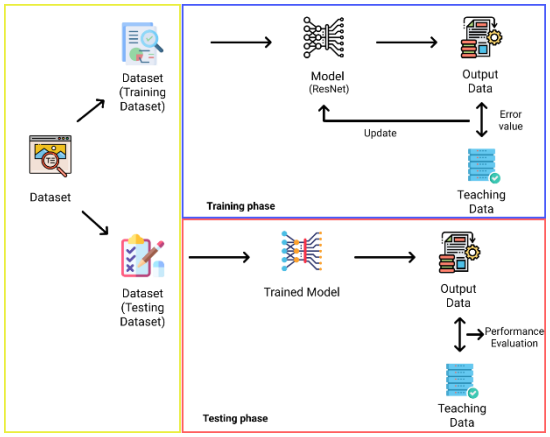


Fig. 6. Model training diagram.

The model needed to achieve the proposed solution is a model that can recognize a person's body pose through a keypoint. So that in the training process the model requires a dataset consisting of images and has labels or keypoint markers. The dataset used to train the model is an open source dataset named COCO [15]. The dataset consists of images and keypoint annotations. There are a total of 118 thousand images for training. As for the testing/validation dataset, there are 5 thousand images.

In the training or training phase, first the input from the dataset (image) will be trained by the ResNet model. Then the model will produce output in the form of keypoint annotations. The output is then compared with the teaching data (keypoint annotation dataset). This comparison will produce an error value which will then be used for model updating.

In the testing phase, the input from the dataset will be processed by the model that was previously trained in the training phase and will classify the input data. So as to produce output that will later be compared with existing annotations (teaching data). So that the model performance will be obtained. To test the performance of the model, two performance metrics will be used, namely precision and recall.

2.3. Server

After obtaining the expected model, then the model can be implemented. The model that has been created cannot be run directly on the website. But the model can be run on the server, which then results can be sent to the website via the API. The input received by the server is an image, after which the model will process the image and produce keypoint coordinates that will be used on the website.

2.4. Website

A previously trained pose estimation model will estimate poses from professional pose images and user pose images to produce coordinates of the body's joints. After that, each pair of coordinates (which form the limbs) will calculate the similarity value. As shown in Figure 7, there is an example of a body part whose similarity value is calculated, namely the right leg (calf) and right hand (forearm). After that, the similarity value of all members of the body will be obtained. For the overall value will be calculated the average of these values. Meanwhile, for each body part, if there is a value that is less than the threshold, the body part will be displayed.

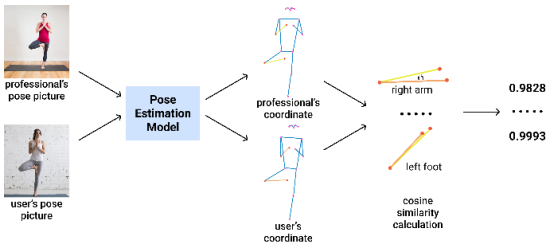


Fig. 7. Similarity Calculation

The threshold calculation flow is the same as the pose assessment flow on the website in Figure 7. However, in calculating this threshold, the pose assessment process is carried out on each image of the yoga pose dataset with professional yoga poses. After getting a set of pose similarity values, then the smallest value from the group will be used as the threshold value.

There is information on the preparation needed before doing yoga. After the user states that he is ready, the website will record in real-time. The recording will take several image frames per five seconds, and each image will be processed by the system. The first process is pose estimation, where the body posture is estimated from the image so that several keypoints are obtained. Furthermore, each of these keypoints is compared with the keypoints of the reference yoga pose image, by calculating their similarity using cosine similarity. After obtaining the similarity value for each keypoint, it will look for whether there is a value that is less than the specified threshold, if so, display it and convey it to the user.

3. Results and Analysis

3.1. Model Training

Table 1. Experiment parameter.

Experiment	Epoch	Learning Rate	Optimizer
1	5	0.001	SGD
2	5	0.001	Adam
3	10	0.01	Adam
4	10	0.001	Adam
5	10	0.0001	Adam

There are 5 experiments that have been done by the author (see Table 1). From these experiments, there were two parameters observed, namely learning rate and epoh. The first analysis is the optimizer, which aims to determine the effect of the optimizer on the performance of the resulting model. The experiments analyzed were experiment 1 and experiment 2 with SGD and Adam optimizer.

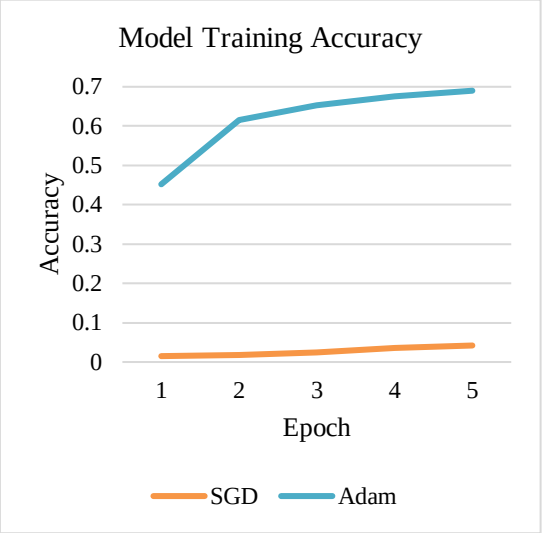


Fig. 8. Model Training Accuracy from Experiment 1 and 2.

Figure 8 is a graph of model accuracy in the training process from experiment 1 and experiment 2. The graph shows a fairly large difference between the two experiments. The model with the Adam optimizer has far greater accuracy than the model using the SGD optimizer.

Table 2. Model performance from experiment 1 and 2.

Optimizer	Precision	Recall
SGD	0.000	0.001
Adam	0.793	0.812

After the model is trained, the model needs to be tested to determine the performance of the resulting model. Table 2 is the result of model performance evaluation from experiment 1 and experiment 2. The resulting model performance has a comparison like the model training process, namely the model with the Adam optimizer has much better performance than the model using the SGD optimizer.

The second analysis is learning rate, which aims to determine the effect of learning rate on the performance of the resulting model. The experiments analyzed were experiment 3, experiment 4, and experiment 5 with learning rate values of 0.01, 0.001 and 0.0001.

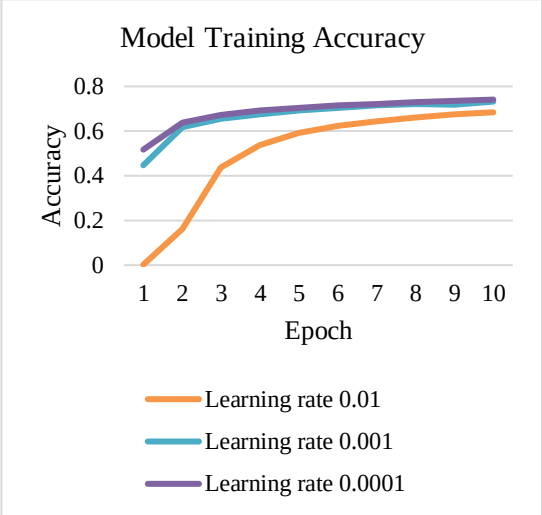


Fig. 9. Model Training Accuracy from Experiment 3, 4, and 5.

Figure 9 is a graph of model accuracy in the training process from experiment 3, experiment 4, and experiment 5. From the graph the smallest learning rate has the highest level of accuracy in the last epoch, namely the 10th. In addition, it can also be seen that the smaller the learning rate, the smaller the change in the accuracy of the resulting model.

Table 3. Model performance from experiment 3, 4, and 5.

Learning Rate	Precision	Recall
0.01	0.785	0.809
0.001	0.827	0.844
0.0001	0.837	0.859

Table 3 is the result of model performance evaluation from experiment 3, experiment 4, and experiment 5. From these results the model with a learning rate of 0.0001 seems to have the best performance in terms of precision and recall values. This is in line with the model training chart in Figure 8 before. From the training accuracy graph and the model performance graph in Figure 9 the smallest learning rate has the best performance, both during training and during testing.

After getting the best parameters, then the model is trained repeatedly using the same

parameters. Due to the limit of the tools (collaboratory), the training process can only be done with 10 epochs each time. And due to limited research time, the training process was only carried out with 30 epochs. So the first model training uses 10 epohs, then continues with 10 epohs 2 times, so that the total epohs used to train one model is 30.

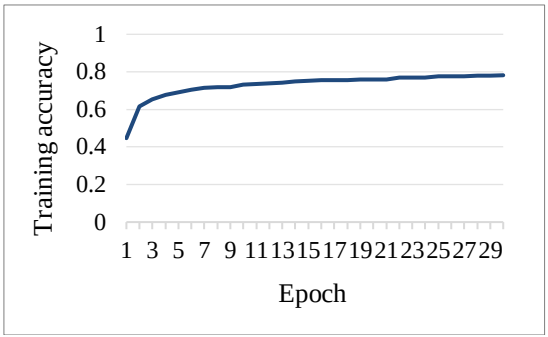


Fig. 10. Model Training Accuracy.

Model performance during training can be seen in Figure 10. Model accuracy in the last epoch was 78.2%. As for the results of performance testing, the resulting model has a precision value of 87% and a recall of 88.2%.

3.2. Model Testing on Yoga Pose Images

Testing is carried out using an image dataset that will be used to determine the threshold. Information of body parts that are still wrong is delivered through text that appears on the website, so users are required to see the information on the website. As a result, the position of the user's head will always face the device's camera (front). While there are several positions that have the head position facing the side. Therefore, out of the 17 existing keypoints, in this study only 12 keypoints were used from the whole body except for the head, including the nose, left eye, right eye, left ear and right ear. So that in this test the number of keypoints observed for all poses is 12. The testing process was carried out manually by observing each image visualizing the results of the predicted keypoints by the model that had been generated.

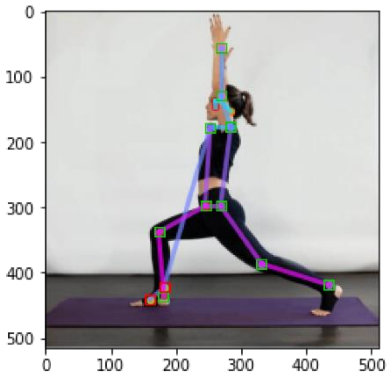


Fig. 11. Example of an image used for testing.

A keypoint is said to be correct if the point is still on the intended limb, while a point is said to be wrong if the point is outside the intended limb or refers to another limb. One example of the image used for testing in Figure 11, there are two red square annotation that have blue lines which are points to indicate the right elbow and wrist, but these points refer to the ankle, so that point is said to be wrong. While the other points are in accordance with the limbs, so these points can be said to be correct.

Table 4. Results of testing images of yoga poses.

Poses	Number of images	Correct keypoints	Percentage	Number of threshold images
Child's Pose	21	106	42%	0
Swimmer	12	85	59%	1
Downdog	23	129	47%	0
Chair Pose	31	350	94%	23
Crescent Lunge	20	224	93%	11
Plank	20	96	40%	0
Side Plank	26	266	85%	17
Low Cobra	24	162	56%	0
Namaste	27	313	97%	22

$$Percentage = \frac{Correct\ keypoints}{Number\ of\ keypoints}\ x\ 100\% \qquad (2)$$

Table 4 is the result of the tests carried out. Each type of pose has a different number of images, so that in order to make it easier to compare data for each pose, percentage calculations are made. The percentage in question is the percentage of the number of keypoints that

are correct with the formula as in equation (2). The number of keypoints is obtained from the number of images of each pose multiplied by the number of keypoints of each image, which is 12. And the number of threshold images is the number of images that can be used for threshold calculations. When the number of threshold image is 0, it means that the pose has no image that can be used for threshold calculations, so the pose cannot be used.

The percentage of correct predictions produced has different values. The pose that has the highest percentage of truth is Namaste at 97%, while the pose that has the lowest value is Plank at 40%. There are 3 poses that have a low percentage (below 50%) namely Child's Pose, Downdog, and Plank. Meanwhile, poses that have a high percentage (above 90%) are Chair Pose, Crescent Lunge, and Namaste.

Poses that have a high percentage tend to be simple poses, so the model can recognize them well. This is because the model is trained using a dataset that contains activities or activities that are generally carried out by people. So that yoga poses that resemble general activities will be well predicted by the resulting model. The pose that has the highest percentage is Namaste. This pose is a cross-legged sitting position that is very common for many people. Besides that, there are also many common activities that are usually done by sitting cross-legged, such as reading books, playing cellphones, operating laptops, relaxing, chatting, etc. Meanwhile, Chair Pose and Crescent Lunge are developments from a standing position viewed from the side. Besides that, the Side Plank pose also resembles a standing position seen from the front but with a different axis, namely horizontally.

On the other hand, for poses that tend to be complicated, it is still difficult to be detected by the model that has been made. Like the Swimmer pose which is very rarely found in daily activities. Meanwhile, the Plank and Low Cobra poses resemble the prone position. However, there are not many activities that require a prone position, so the model is not good at recognizing this position. In addition, there is also the Child's Pose which is a development from a prone position with legs bent inwards. This position is a special position that is used only for yoga and is not found in daily activities.

From the tests that have been carried out, the results are in the form of several images that can be detected by the pose of the body and can be used to determine the threshold. Images that can be used for threshold calculations are images where all 12 keypoints can be detected correctly. The number of images to be used can be seen in Table 4. column number of threshold images. There are several types of poses that do not have images that can be used for threshold calculations, meaning that no images can be detected by the model that has been made. So that on the website application, there are only 4 poses used, namely Chair Pose, Crescent Lunge, Side Plank, and Namaste.

3.3. Website Application

On the website application there are several views. At the beginning there will be a landing page, then after that it will be directed to the display for preparation. After that, the user can start practicing yoga.

Fig. 12. Landing Page.

Figure 12 is the landing page of the website application. This view is intended so that users know the purpose of this website. That this application is used for those who want to practice yoga independently.

Fig. 13. Preparation Page.

Figure 13 is a display of preparation information for doing yoga. This display contains several things that need to be prepared for doing yoga and things to pay attention to when doing yoga practice.

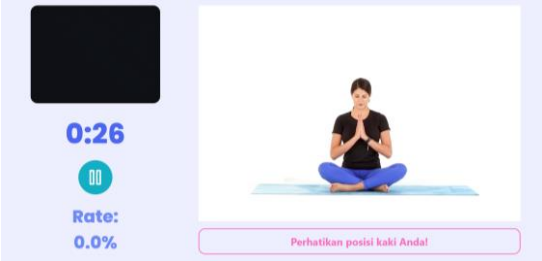


Fig. 14. Exercise Page.

Furthermore, Figure 14 is yoga exercise, the main menu of this application. In this display there are several components including pictures of instructor yoga poses, timers, pause or play buttons, value of poses performed by the user, and information if any part of the body is still wrong. And most importantly, there is the user's camera view.

3.4. Website Application Testing

The finished website application is tested to determine the performance of the application that has been made. Testing is done with several kinds of conditions. Some of the conditions used in this test can be seen in Figure 15.

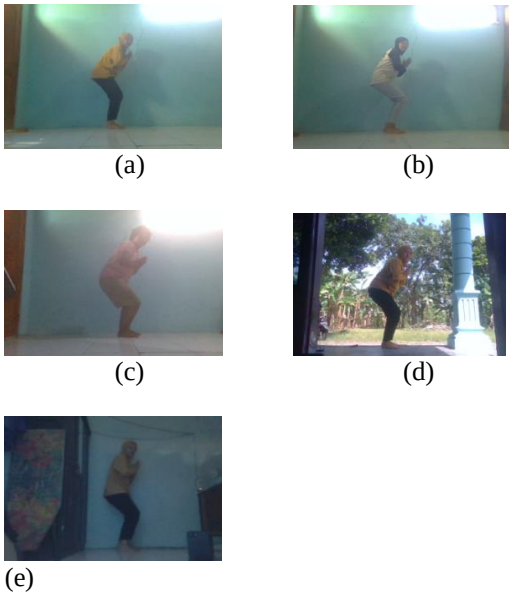


Fig. 15. Several tests condition. (a) Firt Condition. (b) Second condition. (c) Third condition. (d) Fourth condition. (e) Fifth condition.

The first condition will be considered an ideal or normal condition which will be compared with several other conditions. In the first condition the user uses a yellow hoodie with black pants. The background in the first condition is a plain wall with minimal objects around it. The first condition can be seen in Figure 15 (a). The second condition has the same background as the first condition, but the clothes the user wears are different. The clothes used in the second condition are a white shirt with navy blue sleeves, a black hood tucked into the shirt, and gray pants. Figure 15 (b) is an example of the second condition. The third condition also has the same background as the first and second conditions, but with different users and clothes. The clothing used in the third condition is a pink short-sleeved shirt with gray knee-length pants. The third condition can be seen in Figure 15 (c). The fourth condition uses the same people and clothes as the first condition, but the setting is different. The setting in the fourth condition is outdoors which consists of lots of trees. Figure 15 (d) is an example of the second condition. And finally, the fifth condition also uses the same people and clothes as the first condition. The fifth condition background uses a plain wall but there are several objects around that are included in the frame and have lower lighting. The fifth condition can be seen in Figure 15 (e).

The test is carried out by observing the predicted keypoint results from each image captured by the website. The number of images captured by the website in one yoga session is 40 images. So that the total keypoint is 480 points. The test results in several conditions are in table 5.

Table 5. Test results on the website.

Condition	Description	Undetected images	Correct keypoints
1	Normal	0	398
2	Different clothes	0	418
3	Different people and clothes	0	407
4	Outdoor	4	399
5	Lower lighting	4	312

At the time of testing there were several images where the keypoint was not detected at all, which means that the model did not provide output in the form of a keypoint. In that case, in one image it is considered to have detected all 12 keypoints incorrectly, because it is considered that the model failed to detect the user's body keypoint.

The number of correct keypoints in the second condition is slightly more than in the first condition. This means that the model can better recognize the user's pose in the second condition compared to the first condition. Even though

when viewed in terms of color, the first condition is easier to recognize because it has a clothing color that contrasts with the background color. However, the clothes worn in the second condition make the body parts clearer, especially the hands. Whereas in the first condition the clothes worn are slightly looser and have one color, so if you do a pose where the hands are close to the body it will be difficult to detect. From the comparison of the two conditions it can be seen that the difference in clothing has little effect on the results of the detection of the user's pose by the model, the looser the clothes will make it more difficult for the model to detect the user's pose.

The test results in the third condition are almost the same as the test results in the first and second conditions. This can be interpreted if the application can recognize the user's pose relatively well for different users.

In the fourth condition outdoors there are 4 images whose keypoint is not detected. This is because the model cannot recognize that there are people or humans in the image. The shape or color of the background that is not plain or varied may be one of the reasons for this case. However, overall the number of keypoints detected is almost the same as the first condition. So that overall the difference in patterned backgrounds as a whole does not affect the model's performance in terms of poses, but it is possible that there are some cases where body poses are not detected.

From the test results in Table 5 it can be seen that the model is the worst at recognizing poses in lower lighting conditions. In addition to low lighting, this fifth condition has several objects captured in the frame. This resulted in the model being unable to detect keypoints in some images. So it can be said that lower lighting has an effect on reducing the model's performance in recognizing the user's body pose.

In addition to several conditions that have been carried out for testing by observing the performance of the model, other tests have also been carried out. The test is a test by doing a movement that is different from the reference pose. Then the results will be observed to be able to see the performance of the website application that has been made. An example of one of the incorrect positions used for testing is shown in Figure 15.

Fig. 16. Testing with the wrong position.

From one excercise session or 40 frame captures of wrong movements as shown in Figure 16, there were 11 responses that considered true and did not display wrong limbs. Even though the introduction of user poses carried out by the model is good. Out of a total of 480 keypoints, 442 of them were predicted correctly. Meanwhile, the website application gets a response from the server on average about 10 seconds from when the website application sends the user's image. So that the assessment of user yoga poses on the website cannot be delivered in real time, and this results in a reduced accuracy of the yoga pose assessment carried out by the website application. Using another lighter model architecture might make the server response time faster.

4. Conclusion

The application can assess the user's yoga poses and displays the percentage of poses and displays body parts that are still wrong using cosine similarity calculations. The optimal model is obtained with a precision value of 60.4% and a recall of 64.5%. The model was obtained using the Adam optimizer, 30 epochs, and a learning rate of 0.0001. The model can predict well for Chair Pose, Crescent Lunge, and Namaste with a percentage of 90%. In addition, the model can also predict quite well for the Side Plank pose with a percentage of 85%. Meanwhile, the model prediction results are not good for Swimmer's pose with a percentage of 59%. And the prediction results of the model are bad for Child's Pose, Downdog, and Plank which have percentages below 50%.

Acknowledgement

The author would like to thank the supervisors for guiding in all aspects. The author also expresses many thanks to all who have contributed to this research.

References

[1] "Yoga." https://www.halodoc.com/kesehatan/yoga (accessed Jan. 15, 2022).

[2] "Yoga Statistics and Facts: 2021 Edition." https://moderngentlemen.net/yoga-statistics/ (accessed Jan. 15, 2022).

[3] C. Long, E. Jo, and Y. Nam, "Development of a yoga posture coaching system using an interactive display based on transfer learning," *J. Supercomput.*, no. 0123456789, 2021, doi: 10.1007/s11227-021-04076-w.

[4] 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.

[5] 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.

[6] S. S. Narayanan, D. K. Misra, K. Arora, and H. Rai, "Yoga Pose Detection Using Deep Learning Techniques," *SSRN Electron. J.*, pp. 1–8, 2021, doi: 10.2139/ssrn.3842656.

[7] A. Chaudhari, O. Dalvi, O. Ramade, and D. Ambawade, "Yog-Guru: Real-Time Yoga Pose Correction System Using Deep Learning Methods," pp. 1–6, 2021, doi: 10.1109/iccict50803.2021.9509937.

[8] F. Rishan *et al.*, "Infinity Yoga Tutor : Yoga Posture," 2020.

[9] S. Patil, A. Pawar, A. Peshave, A. N. Ansari, and A. Navada, "Yoga tutor: Visualization and analysis using SURF algorithm," *Proc. - 2011 IEEE Control Syst. Grad. Res. Colloquium, ICSGRC 2011*, pp. 43–46, 2011, doi: 10.1109/ICSGRC.2011.5991827.

[10] A. Suryanto, J. Swee, N. Hiong Chiang, and R. Lim, "Pose Comparison for Correct Yoga Posture Measurement."

[11] S. Goyal and A. Jain, "Yoga Pose Perfection using Deep Learning," *J. Student Res.*, vol. 10, no. 3, pp. 1–10, 2021, doi: 10.47611/jsrhs.v10i3.2140.

[12] S. Agarwal *et al.*, "FitMe: A Fitness Application for Accurate Pose Estimation Using Deep Learning," *ICSCCC 2021 - Int. Conf. Secur. Cyber Comput. Commun.*, pp. 232–237, 2021, doi: 10.1109/ICSCCC51823.2021.9478168.

[13] "Pose estimation | TensorFlow Lite." https://www.tensorflow.org/lite/examples/pose_estimation/overview (accessed Jan. 16, 2022).

[14] "Top 5 Distance Similarity Measures implementation in Machine Learning | by Shriya Gupta | Medium." https://medium.com/@gshriya195/top-5-distance-similarity-measures-implementation-in-machine-learning-1f68b9ecb0a3 (accessed Jan. 16, 2022).

[15] "COCO - Common Objects in Context." https://cocodataset.org/#home (accessed Jul. 14, 2022).