

---

# Self-learning Yoga System

---

**Deepak Pandey**  
deepak01@stanford.edu

## 1 Introduction

The self-learning yoga system can be used by people to learn and improve their yoga practice. With the pandemic, there has been a change in the world and things have shifted towards the online mode and this is an effort to achieve the same for yoga training, by building a system that allows people to learn yoga anywhere and anytime. The input to the system is the user performing the yoga pose. The output of the system is the classification of the yoga pose and the evaluation of the user pose.

## 2 Background/Literature review

The traditional approach to pose classification or exercise classification is using the video/image based methods.

CNN's are extensively used to process the input and classify the pose user pose:

- Three-dimensional CNN-inspired deep learning architecture for Yoga pose recognition in the real-world environment paper [5] uses CNN but doesn't account for the sequential relationship of different frames
- Yoga pose classification: a CNN and MediaPipe inspired approach [6] uses mediapipe blazepose over open pose and other state of the art models to get performance boost
- Yoga classification deep learning approach[8] uses CNN's and transfer learning to achieve a similar goal

There also have been practices using coordinates of different parts or joints to classify the user pose

- Real-time Yoga recognition using deep learning paper[7] takes a similar approach by using CNN's to classify the yoga pose by using the coordinated of the joints
- Yog-Guru: Yoga Pose Correction System [9] also uses coordinated to measure the accuracy of the pose as well as provide feedback to user

All of these approaches have certain shortcoming and challenges when deployed in real world.

The challenges can be mainly divided into two broad categories as follows:

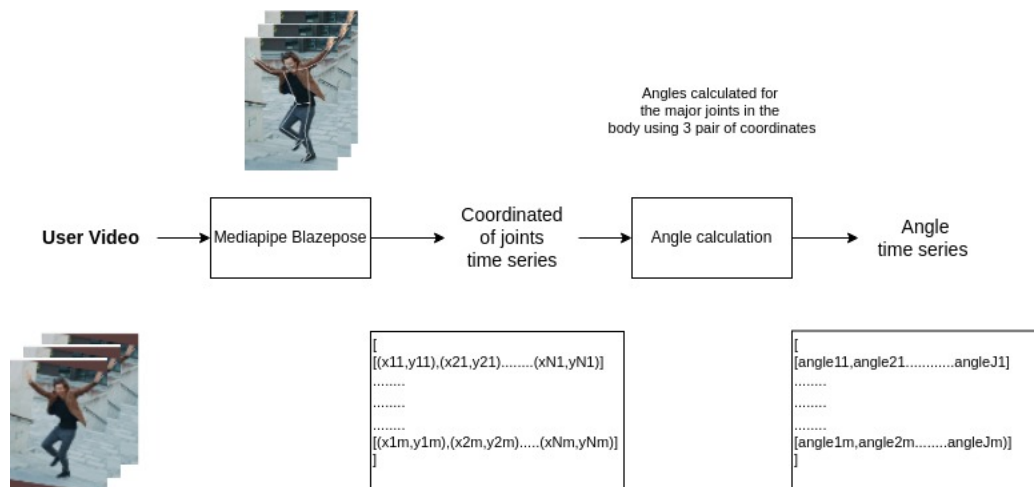
- Challenges related to Input  
The user is not expected to have multiple cameras to provide the video input, hence single-view video is the basic assumption of the system and that creates the following challenges:
  - Multiple poses look similar from a certain angle hence completely relying on coordinates frame by frame (images) might not be the most effective practice
  - Some of the parts of the pose get obstructed and are not visible clearly
- Challenges related to Algorithm  
Using CNN models with images as input doesn't perform as expected for the following reasons

- Final poses for multiple yoga poses are quite similar to each other based on the camera's angle
- Training CNNs to classify a certain pose requires large dataset with a variety of examples, as the dimensions of a user, distance from camera, point of view, and many other factors needs to be addressed in training data

### 3 Dataset and Preprocessing

Video data from across the web and self-recorded videos is used to create the input dataset. Authors of Real-time Yoga recognition using deep learning paper[7] have made the dataset they used publically available (<https://archive.org/details/YogaVidCollected>) , which is included with the dataset used in this project. The video data is preprocessed to create time series data consisting of angles at the body joints frame by frame while performing the yoga. The video input is passed through Mediapipe's BlazePose[1] to get the coordinates(33 points on the body) of which the coordinates of wrist, elbow, shoulder, hip, knee, ankle for both right and left are used to calculate 8 angle(including left and right both side) for the following groups:

- Wrist, Elbow, Shoulder
- Elbow, Shoulder, Hip
- Shoulder, Hip, Knee
- Hip, Knee, Ankle









Data Preprocessing pipeline

The angles are calculated for 30 equidistant data points through out the user pose.

Training data consists of 6 different poses:

- Padamasana (Lotus Pose)
- Bhujangasana (Snake Pose)
- Shavasana (Corpse Pose)
- Tadasana (Mountain Pose)
- Trikonasana (Triangle Pose)
- Vrikshasana (Tree Pose)

Sr. No.	Asana Name	Posture
1	Bhujangasana (Cobra Pose)	
2	Padmasana (Lotus Pose)	
3	Shavasana (Corpse Pose)	
4	Tadasana (Mountain Pose)	
5	Trikonasana (Triangle Pose)	
6	Vrikshasana (Tree Pose)	

Yoga poses (credit:Real-time Yoga recognition using deep learning paper[7])

## 4 Methods

The system is proposed to use the following methods for classification and evaluation:

- Time Series based classification  
Time series-based classification[3] is proposed over the classical CNN-based approaches on image input [2] due to the following advantages:
  - This approach takes into consideration the steps taken to reach the final pose rather than a single step or final step in isolation.
  - Change in angle act as better input as it doesn't vary much with the height of the person, the distance from the camera, the pace at which the yoga is performed, etc.
- Angle-based evaluation using DTW(Dynamic Time Warping )[4]  
To compare the user pose with the classified instructor pose, the proposed method is to use the angle time series by fitting them using DTW to calculate the differences in different angles due to the following advantages:
  - This provides reliable time alignment between the user and instructor pose.
  - This allows us to compare the time series of different length, hence taking in account the speed variation of performing the pose.

### Baseline

The baseline used to evaluate the approach is a MLP with the following configuration:

input(240)->Dense(512,relu)->Dense(256,relu)->Dense(64,relu)->Dense(30,relu)->Dense(6,softmax)->output

Each time series of the pose of dimensions 30x8 is converted into a tensor of shape [240,1] and given as input to the model. Four hidden layers with relu activation functions are used. An output layer with softmax activation is used to do the multiclass classification following are the other attributes :

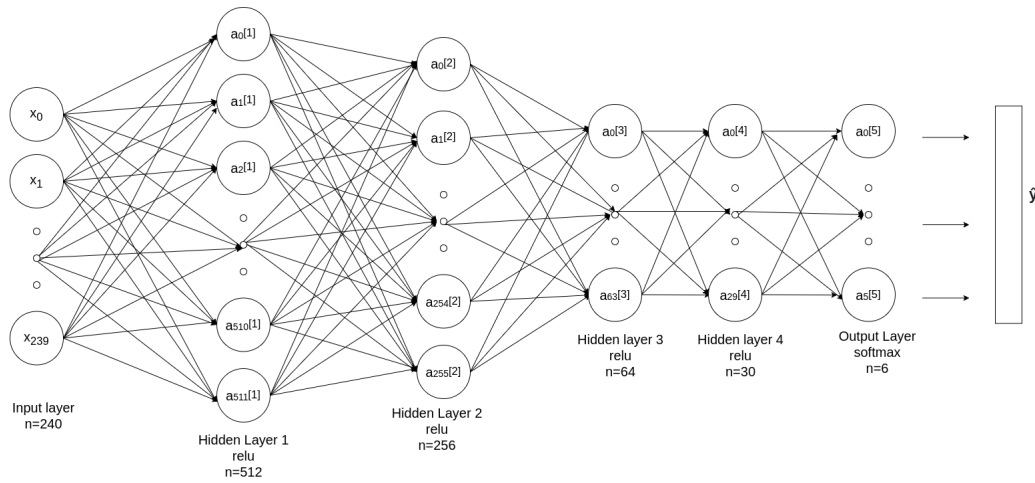
Loss=Categorical Crossentropy

Optimizer= Adam

Metrics=Accuracy

This model gives around 91% accuracy for classification

This verified that the angle time series is a viable technique to classify the yoga poses as it capture the distinction between poses



### Final Approach

As the input is a time series of angle and RNN's are great with the time series data, the next choice of mode was using an LSTM model with the series and input. This approach better capture the sequential behavior of data and the relation that one intermittent pose has with the next while performing the yoga pose.

LSTM with the following parameters is settled to be the final model:

Time series of the pose of dimensions 30x8 is given as input to the model.

Two LSTM layers with 30 units each are used with relu activation function accompanied by a dropout of 0.5

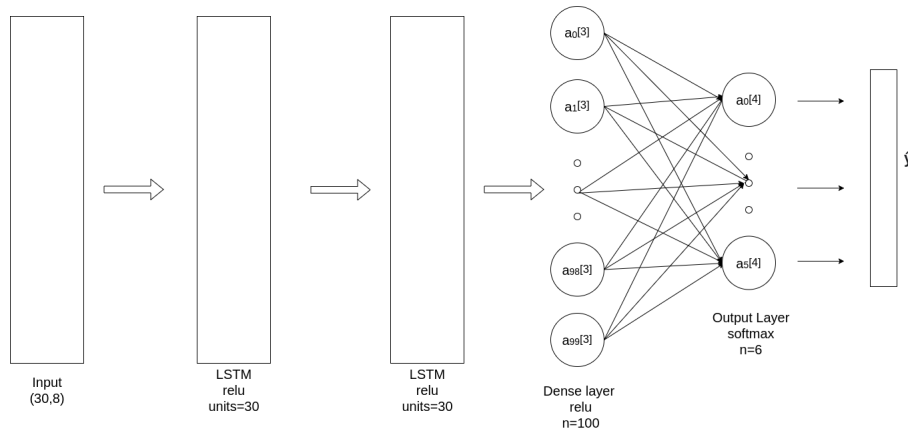
An output layer with softmax activation is used to do the multiclass classification

Loss=Categorical Crossentropy

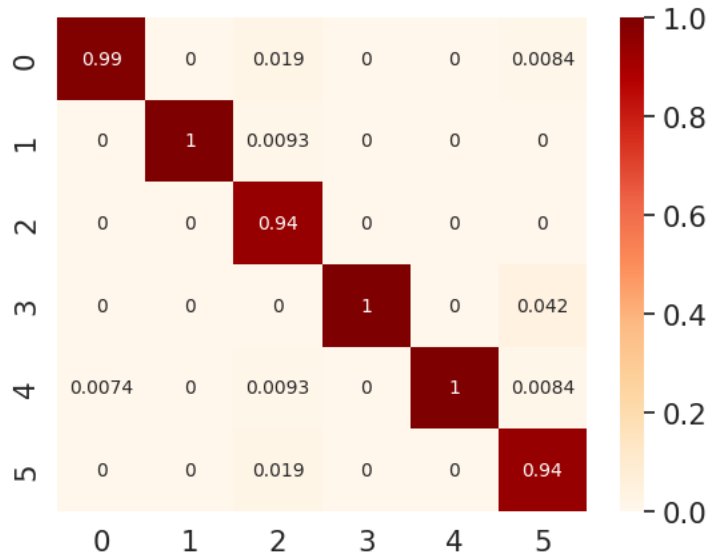
Optimizer= Adam

Metrics=Accuracy

This model gives around 97.91% accuracy for classification of yoga pose



Confusion metrics for the above LSTM model is shown below with Snake Pose:0 ,Lotus Pose:1, Corpse Pose:2 , Mountain Pose:3 , Triangle Pose:4 ,Tree Pose:5



### Evaluation

Once the User pose is classified as one the 6 poses, the instructor pose time series for the same yoga pose is used to evaluate the user pose.

The difference between the user angle timeseries and instructor angle time series is calculated by using Dynamic Time Warping. DTW find the closest resembling data point in instructor pose for each data point in user pose and calculated their difference. Doing so for each data points give a time series of angle difference.

The angle difference are classied as follow:

Angle Difference	Classification	Color
Less than 25 degrees	Good	Green
Between 25 to 50 degrees	Average	Blue
More than 50 degrees	Bad	Red

## 5 Experimentation

The following are the system components that were evaluated by research and experimentation:

- **input size**

The input size was decided to be a time series of 30 data points considering the average length of yoga poses and taking 30 equidistant data points

- **Hyperparameters**

The hyperparameters were searched using the scikit learn's random search and grid search with scikeras to incorporate kerasclassifier. Following are the major hyperparameters that were experimented for:

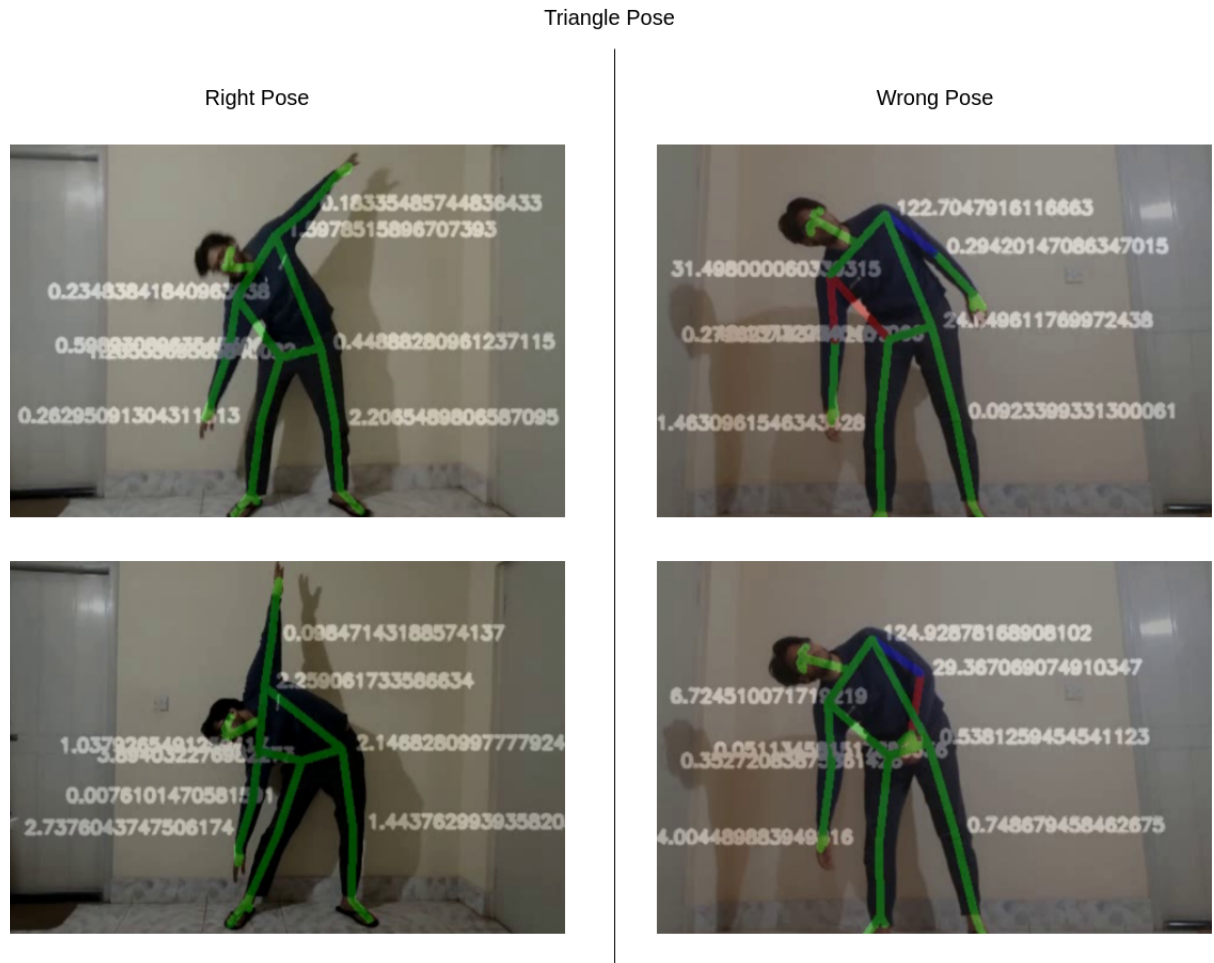
Hyperparameter	Experiment Values	Final Value
batch size	[32,64,128]	64
epochs	[50,100,200]	100
LSTM layers	[1,2,3]	2
nodes in the LSTM layer	[30,60,90]	30
dropout	[0.2,0.5,0.7]	0.5
units in the dense layer	[5,10,50,100]	100

## 6 Analysis and Result

By looking at the results of the experiment we can evaluate following:

- The LSTM as expected performs better than the MLP as it does a better job at capturing the learning based on the sequence of the angles.
- LSTM also overcomes the shortcomings of image based approaches as it doesn't rely on the isolated images, instead tried to find the relation between one frame to next and classifies based on that.
- Also the poses that look really similar and contain subtle difference or subtle moments that might not be captured by the image based CNN approach is taken care by angle based time series approach because even the minute changes cause change in angles and hence creating a distinction.
- The LSTM also successfully captures the idea that the yoga practice is not a stationary pose but a series of poses that lead to that pose.

The final result after comparing the user pose with instructor pose looks as follows:



## 7 Code Repo

URL - <https://github.com/deepak80199/cs230project>

## References

- [1] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang & Matthias Grundmann (2020) BlazePose: On-device Real-time Body Pose tracking. In CVPR Workshop on Computer Vision for Augmented and Virtual Reality, Seattle, WA, USA, 2020
- [2] M. C. Thar, K. Z. N. Winn and N. Funabiki(2019) A Proposal of Yoga Pose Assessment Method Using Pose Detection for Self-Learning.In 2019 International Conference on Advanced Information Technologies (ICAIT), 2019, pp. 137-142, doi: 10.1109/AITC.2019.8920892.
- [3] Ismail Fawaz, H., Forestier, G., Weber, J. (2019) Deep learning for time series classification: a review. Data Mining and Knowledge Discovery volume 33, 917–963 (2019).
- [4] Yu, Yuncong Mayer, Thomas Knoch, Eva-Maria Frey, Michael Gauterin, Frank. (2019). Time Series Comparison with Dynamic Time Warping, Convolutional Neural Network and Regression.

- [5] Jain, S., Rustagi, A., Saurav, S. et al. Three-dimensional CNN-inspired deep learning architecture for Yoga pose recognition in the real-world environment. *Neural Comput Applic* 33, 6427–6441 (2021). <https://doi.org/10.1007/s00521-020-05405-5>
- [6] Garg, S., Saxena, A. Gupta, R. Yoga pose classification: a CNN and MediaPipe inspired deep learning approach for real-world application. *J Ambient Intell Human Comput* (2022). <https://doi.org/10.1007/s12652-022-03910-0>
- [7] Yadav, S.K., Singh, A., Gupta, A. et al. Real-time Yoga recognition using deep learning. *Neural Comput Applic* 31, 9349–9361 (2019). <https://doi.org/10.1007/s00521-019-04232-7>
- [8] Josvin Jose and S Shailesh 2021 IOP Conf. Ser.: Mater. Sci. Eng. 1110 012002, Yoga Asana Identification: A Deep Learning Approach
- [9] A. Chaudhari, O. Dalvi, O. Ramade and D. Ambawade, "Yog-Guru: Real-Time Yoga Pose Correction System Using Deep Learning Methods," 2021 International Conference on Communication information and Computing Technology (ICCICT), 2021, pp. 1-6, doi: 10.1109/ICCICT50803.2021.9509937.