Yoga Poses Classification

Rosalie Pampolina
Student# 1006079226
rosalie.pampolina@mail.utoronto.ca

Marcus Hong
Student# 1009009984
marcus.hong@mail.utoronto.ca

George Wang
Student# 1009133876
georgez.wang@mail.utoronto.ca

Harry Nguyen
Student# 1008880025
har.nguyen@mail.utoronto.ca

—-Total Pages: 9

1 Introduction

The motivation for this project stems from our collective interest in applying deep learning to help physical activity; all of us either like working out at the gym or running. This project's goal is to use machine learning to accurately identify what yoga pose is represented in an image.

It is important for two reasons. First, this project has the potential to serve as a tool to help users independently learn and practice yoga by identifying how well they are doing a pose. Although this project cannot replace yoga lessons or instructional videos, we are hopeful that it could help yoga enthusiasts as an extra tool to improve their form, prevent injury, and track progress on a pose. As a similar project states, "Analyzing human posture can identify and rectify abnormal positions, improving well-being at home" (Talaat, 2023).

Secondly, this yoga pose classification model is a building block for other applications. For example, the deep learning model could be adapted to track posture in other physical activities we are interested in, such as paddling and skateboarding. Moreover, this model can be expanded with additional features, such as recognizing poses in videos similar to related projects that further identify yoga poses in videos (Singh et al., 2022), (Upadhyay et al., 2023) or send scores and feedback based on user's poses like an actual instructor.

Finally, deep learning is an appropriate approach because it excels at image classification problems, such as determining if an image is a goat. Through training, a deep learning model is flexible enough to identify the same pose performed by different people in different settings with different clothes.

2 Illustration/Figure Overview

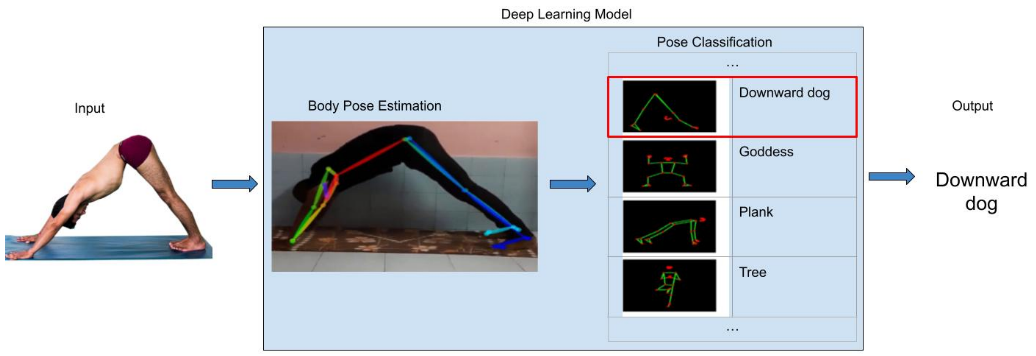

Figure 1: Yoga pose classification model.

3 Background and Related Work

In recent years, there has been a growing interest in classifying yoga poses automatically. Several projects have explored this intersection of yoga and machine learning, aiming to enhance yoga practice, monitor form, and provide personalized feedback. In this section, we will briefly overview five projects that have contributed to the field of yoga pose classification.

3.1 Yoga Pose Detection Using Deep Learning Techniques

The goal of this research paper (Narayanan et al., 2021) is to develop a software solution to offer the benefits of having a specialized guru to help instruct posture, which was common in older yoga methods, while also fitting into the user's schedule and modern day, which is often very busy. In order to do this, the developers have utilized OpenPose to deploy advanced machine learning models and effectively predict a human's pose and whether it properly adheres to the desired form of the given yoga pose.

3.2 Yoga Trainer App using Human Pose Detection

The goal of this research paper (Singh et al., 2022) was to develop a user-friendly yoga pose detection application that inspires users to pursue yoga further through the use of machine learning technologies and a modern mobile interface. The app attempts to learn from the various market solutions and improve on their weaknesses, such as improved stance detection in real time, through the use of modern machine learning models created in the Machine Learning Kit by Google.

3.3 Deep Learning-Based Yoga Posture Recognition Using the Y_PN-MSSD Model for Yoga Practitioners

This research paper (Upadhyay et al., 2023) outlines the needs for Yoga posture recognition software as there are minimal options for live tracking of poses to help reduce injuries and presents a capable solution. In the paper, the authors created a machine learning model by using a combination of Pose-Net and Mobile-Net SSD in order to handle the detection of the human and of feature points. This combination of models has allowed the authors to create a more effective model that outperforms the popular Pose-Net CNN model.

3.4 Yoga Pose Estimation Github Repo

The Yoga Pose Estimation repo (Kota, 2021) is an excellent example of a similar project in the field of yoga pose detection and analysis using machine learning. This application is well documented and made available for public use on the github repo. The app uses posenet and a KNN Classifier in order to detect the user's poses and has been trained with a custom dataset of three poses in the form of 3 videos. This is an excellent open source solution which allows for other users to interact with it and submit improvements via github.

3.5 Down Dog App

The Down Dog App (DownDogTeam) is a commercial solution through the form of a mobile app on the Apple and Google Play stores. This app uses machine learning to track the form of the user and analyze how accurately they are following the provided pose. This app is a good solution for the masses who are not technically savvy enough to use an app from a github repo and would rather use a fully developed mobile application that is easy and they can open anywhere. Although the techniques and models are not disclosed on the app's website, it is clear from their demonstration videos that machine learning is being used to detect the key points on a users body and predict how accurate they are in following the various poses.

4 Data Processing

The raw dataset created by Shruti Saxena (Saxena) is found on the website kaggle.com. It contains 107 yoga poses, with a total of 5994 images. This dataset is preferable for 3 main reasons:

1. it contains a high number of images
2. the number of images does not vary significantly across yoga poses
3. it contains diverse representations of people of different gender, age, and ethnicity

With these advantages, we hope to achieve a model that can be used by a larger group of users while having a higher classifying accuracy.

Next, this dataset requires the following cleaning processes:

- Standardizing image dimensions will fit all images into a standard-sized square background. This requires down-scaling or up-scaling images, as shown below.

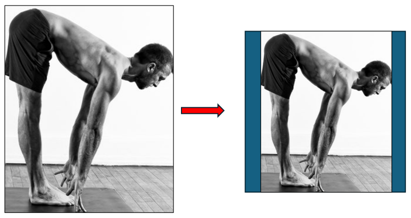

Figure 2: Scaling and fitting a rectangle image into a square background

- Converting image format from JPEG to PNG
- Pixel RGB values have to be adjust to ± 0.5 range, which is a requirement for OpenPose network
- Images that contain more than one human figure will be removed
- At this point, we are unsure whether dataset augmentation is required to train a good model. However, we will augment the dataset if necessary, by rotating, mirroring, scaling, etc.

5 Architecture

For our yoga pose classification project, we utilize **Convolutional Neural Networks** (CNNs) for feature extraction from images. This is a multiclass classification problem since the input yoga pose will be classified into one of 107 yoga poses. Our CNN model will receive input of a square image, and output a vector of probabilities of each yoga pose match. As an activity classification problem, we are inspired by the research work of Dong (Bearman & Dong). Figure 3 is the architecture used in Bearman's research, and we aim to use it as a reference. CNNs are suitable for this yoga classification project due to their ability to automatically learn relevant features from raw image data, reducing the need for manual feature engineering. Their convolutional layers can detect and recognize patterns such as edges, textures, and shapes that are critical for distinguishing between different yoga poses, allowing high accuracy.

A rough architecture of CNN should essentially have 5 types of layers:

- Input
- Convolution (Filters)
- Pooling
- Input Nodes
- Output

The architecture begins with an **input layer** that accepts images. Following the input layer, several **convolutional layers** apply filters to the images, detecting various features such as edges, textures, and shapes. These layers are vital to recognizing different parts of the body. **Pooling** layers are then used to reduce the spatial dimensions of the feature maps, converting into the **input nodes** while reducing the computational load. After the convolutional and pooling layers, those input

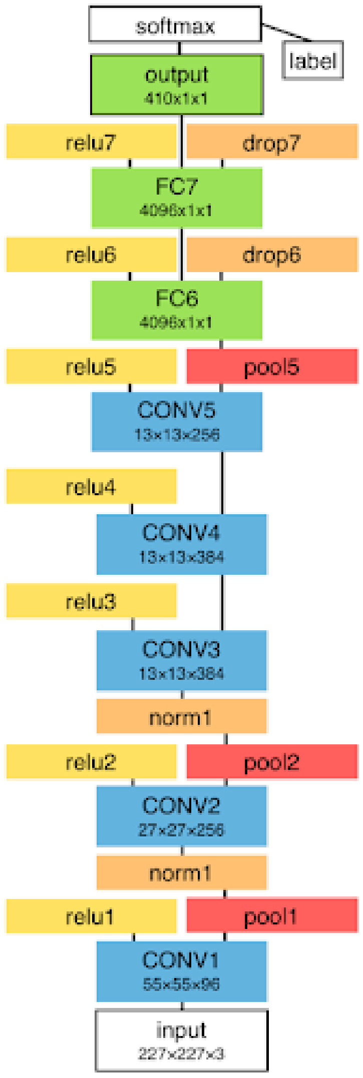

Figure 3: CNN architecture used for activity classification in Dong's research

nodes aggregate the features learned and make predictions about the yoga poses. The **output layer** provides the final predictions, either in the form of key points (speicific joint locations) or pose classes (the type of yoga pose).

6 Support Vector Machine (SVM) as a Baseline Model

The feature vectors derived from the handcrafted features train an SVM classifier (Bilogur) with a radial basis function (RBF) kernel (Bernstein). The SVM model hyperparameters include the kernel (RBF), the regularization parameter (C), and gamma, which defines how far the influence of a single training example reaches. The training process involves splitting the dataset into training and validation sets, training the SVM model on the training set, and evaluating its performance on the validation set using metrics like accuracy, precision, recall and F1-score(Kundu).

This approach provides a straightforward comparison comparison point for evaluating the performance of more complex neural network models. More particularly, SVMs are suitable due to their ability to handle high-dimensional feature spaces and their effectiveness in binary and multiclassification tasks. They can create decision boundaries and perform well even with smaller datasets, ensuring reliable pose classification.

7 Ethical Considerations

For our Yoga Pose Detection Model, there are a variety of ethical concerns that we must take into account. Firstly, there is the concern of how user data is retrieved, interacted with, and managed. We must ensure that user safety and privacy is maintained throughout the entirety of the development cycle and release of our model. User images should be anonymized as best as we can to help decouple user identities from data that is used to train and improve the model. Steps can be taken to anonymize the data, such as encryption, random identifiers that disregard any revealing information, and other common techniques. It is also important that we do not store any user data with identifying or personal information, such as user images, names corresponding to images, and more. This is important because in the case of a data breach, if we maintain these privacy strategies, the users will be far better protected and less likely to have critical information obtained by malicious actors.

In addition to maintaining privacy and ethical data management strategies, it is an important ethical consideration to ensure that the training data used for our model contains a wide variety of diverse body types and representations. This will help to ensure that our model is effective in extending to our wide variety of users who will have many different body types and want to receive a fair and equal experience. In doing this, it will also make our model less prone to overfitting a specific body type and have more accurate predictions when it comes time to being deployed for others to interact with.

Regarding the limitations of the model and data used for training it, it can be challenging to find a dataset that contains a large variety of yoga poses with a diverse range of body types in the data representing the poses. This poses a difficulty in ensuring that our model is accurate for our many diverse users. The amount of poses in the dataset will also constrict our model to only being able to accurately predict those poses and no other poses that we would want to incorporate into our model. One way to incorporate new poses into our model would be to find another dataset with the desired pose and format it to match the data format of our original dataset. Alternatively, we could create our own new dataset with these poses, which is time intensive and undesirable.

8 Project Plan

8.1 Project Plan Breakdown of Tasks with Deadlines

See *Table 2: Future Tasks* and *Table 1: Completed Tasks* for task assignments throughout the project.

Table 1: Completed Tasks

Task	Team member(s)	Deadline
Intro draft	George	Monday, June 3
Background	Rosalie	Monday, June 3
Architecture draft	Harry, Marcus	Monday, June 3
Finish Intro	George	Wednesday, June 5
Draft illustration	Marcus, George	Wednesday, June 5
Data process draft	Marcus	Wednesday, June 5
Architecture in paragraphs	Harry	Wednesday, June 5
Baseline model	Harry	Wednesday, June 5
Ethical considerations	Rosalie	Wednesday, June 5
Brainstorm project plan	Rosalie, George	Wednesday, June 5
Risk register draft	Rosalie	Wednesday, June 5

Table 2: Future Tasks

Task	Description	Team member(s)	Deadline
Move work to overleaf and finalize writing	Intro Illustration Project plan	George	Thursday June 5 before meeting
	Background Ethical Considerations Project plan writing Risk register	Rosalie	
	Data processing, architecture	Marcus	
	Baseline models architecture, GitHub link	Harry	
Data Collection	Collect a diverse dataset of yoga poses from different sources	Rosalie	Monday June 10th
Data Preprocessing	Go through the collected data and augment data, load dataset into model	Rosalie	Thursday June 13th
Primary model outline	Write outline of what functions DL model needs	George	Monday June 10th
Design Model Architecture	Use PyTorch to create the initial model architecture	George & Harry	Thursday June 13th
Baseline Model Training	Handcrafted features	Marcus	Monday June 10th
	Support Vector Machine (SVM)	Harry & Marcus	Thursday June 13th
Analyze our output	Analyze the initial results (plots) and two model's performance after training	Marcus & George	Saturday June 15th
Tune hyperparameters	Experiment with various hyperparameters to optimize the performance of our model	Rosalie & Harry	Tuesday June 18th
Evaluate Model	Evaluate the performance of our model based on its training error and accuracy, testing error and accuracy, etc	Harry & George	Tuesday June 18th
Write Progress Report Draft	Write a draft on the model's progress	Everyone (prepare the work they've done)	Tuesday June 25th
Progress Report Final Editing and Submission	Edit the final draft of the Progress Report and submit	Everyone (prepare the work they've done)	Wednesday July 3rd
Write Final Report	Write a draft on the model's final progress	Everyone (prepare the work they've done)	Thursday Aug 15th
Presentation Preparation	Prepare main presentation content: problem, data, data processing, model, demonstration, quantitative results, qualitative results, takeaways	Everyone (prepare the work they've done)	Saturday Aug 10th
Record Final Presentation	Have all presentation content prepared and ready to film	Everyone (prepare work they've done)	Sunday, August 12th
Presentation Final Editing and Submission	Have the Final Presentation Edited	Rosalie, George	Thursday, August 15
	Submit	Rosalie	
Final Report Final Editing and Submission	Edit the final draft of the Progress Report and submit	Marcus, Harry	Thursday, August 15

8.2 Communication and Collaboration

8.2.1 How Our Team Will Collaborate

Throughout the project, our team will make use of a variety of tools and techniques to ensure effective communication and collaboration. For collaborative document and code editing, we will mainly utilize Google Drive and Github. This will allow for team members to work concurrently and have our work saved in real time.

8.2.2 Meeting Schedule

We intend to schedule frequent meetings in order to have constant development and collaboration of our project. We have scheduled weekly Wednesday 9pm EST check-in meetings to ensure we meet our internal deadlines and goals. In addition, we plan to meet on weekday evenings when it is necessary to have more discussions and assist each other with larger tasks, such as writing reports.

8.2.3 Communication Tools

We will use a range of communication tools for various use cases:

Discord: This will serve as our primary channel for daily, asynchronous discussions and brief updates

Discord Call: When verbal contact is required we will utilize this application's feature for audio calls

Zoom: We'll utilize Zoom for formal meetings that involve screen sharing and video calls.

Google Drive: Our main repository for exchanging files, datasets, and other documents

8.2.4 Maintaining Code Integrity

Github will be used to maintain version control and prevent overwriting other's code. We plan on working on separate branches, with each member's code modifications requiring a pull request for others to review before merging the code into our main branch.

9 Risk Register

In order to be successful, recognizing and controlling risks is essential. This section includes a risk register with important, probable risks related to our project. Each risk is assessed by its likelihood, risk, and risk mitigation.

9.1 Team Member Dropping Course

Likelihood: Low. Not extremely likely based on our interest in the course, however still a possibility as sometimes external circumstances arise.

Risk: The project's overall development and team's workload distribution may get impacted, as every member has their own tasks to complete and different schedules. There also may be a skill gap in the team, which can stress the team

Mitigation:

- Meet with the team: conduct a meeting so that we can discuss and assess our current progress, then reassign responsibilities based on what is left to do of our project, leveraging each other's strengths and schedules.
- Document our work: keep thorough records of our work and project progress in our google drive, including meeting minutes, data processing procedures, commenting code, and drafts of our work

- Seek Help: If we are having difficulty navigating the loss of a team member we could possibly reach out to the course instructor or teaching assistants to ask for some guidance on how we can proceed as a group

9.2 Model Development and Training Taking Longer Than Expected

Likelihood: High. Model Development and training periods can take a longer time than expected especially for many iterations and modifications required.

Risk: Prolonged times needed for the model's development and training times can cause delays and setbacks for other project tasks such as evaluation, hyperparameter tuning, and report writing, which can shorten our timeline to work on these tasks.

Mitigation:

- Early Testing: test the model as early as possible so that we can find possible problems earlier on. We can also start building the model and train it incrementally with smaller portions of the dataset so that we can better understand and streamline the training procedure
- Optimize Code: Try our best to optimize our code where we can to reduce training time. For example, reducing code complexity, reducing the file size of data
- Include Buffer Time: anticipate delays by adding in a grace period for our project scheduling so that we can complete our tasks by our internal deadlines

9.3 Data Quality Issues

Likelihood: Medium. The dataset has been roughly looked over so the image quality should not be big issue.

Risk: For each class, i.e., yoga pose, there are about 60 images more or less, which poses a risk that these images are not enough to train a reliable/good model.

Mitigation:

- If the image quality turns out to be an issue, there is an alternative dataset that can be used: https://www.kaggle.com/datasets/tr1gg3rtrash/yoga-posture-dataset/data
- If the size of dataset is too small, we can apply augmentation methods by rotating, flipping, scaling the images.

9.4 Model Performs Below Expectations

Likelihood: Medium. There are challenges in accurately recognizing and classifying yoga poses due to participant body structure variations.

Risk: Models performing below expectations may include scenarios where the model fails to achieve accuracy, precision, or F1-score on a validation set. It can stem from inadequate training data, inappropriate model architecture, or ineffective hyperparameter tuning.

Mitigation:

- Implement ensemble methods (i.e combining predictions from multiple models (e.g., CNN and SVM) to improve overall accuracy.
- Use the dataset with more labelled examples to improve model training.
- Continuously tune the model's hyperparameters and architecture based on validation performance.

10 Github

Link to Github: https://github.com/Harry-Nguyen0248/APS360_Yoga_Poses_Classification.git

REFERENCES

Amy Bearman and Catherine Dong. Human pose estimation and activity classification using convolutional neural networks. URL https://cs231n.stanford.edu/reports/2015/pdfs/cdong-paper.pdf.

Matthew N. Bernstein. The radial basis function kernel. URL https://pages.cs.wisc.edu/~matthewb/pages/notes/pdf/svms/RBFKernel.pdf.

Aleksey Bilogur. Kernels and support vector machine regularization. URL https://www.kaggle.com/code/residentmario/kernels-and-support-vector-machine-regularization.

DownDogTeam. Down dog app. URL https://www.downdogapp.com/.

Sai Durga Kamesh Kota. Yoga-pose-estimation-app, 2021. URL https://github.com/Devtown-India/Yoga-Pose-Estimation-App.

Rohit Kundu. F1 score in machine learning: Intro calculation. URL https://www.v7labs.com/blog/f1-score-guide.

Sankara Narayanan, Devendra Kumar Misra, Kartik Arora, and Harsh Rai. Yoga pose detection using deep learning techniques. *SSRN*, 2021. URL https://papers.ssrn.com/abstract=3842656.

Shruti Saxena. Yoga pose image classification dataset. URL https://www.kaggle.com/datasets/shrutisaxena/yoga-pose-image-classification-dataset?select=dataset.

Thoudam Johnson Singh, Borish Kshetrimayum, Heman Budathoki, and Chelsea Dambe R Sangma. Yoga trainer app using human pose detection. *International Journal of Digital Technologies*, 2022.

A.S Talaat. Novel deep learning models for yoga pose estimator. *SN Applied Sciences*, 5, 2023. URL https://link-springer-com.myaccess.library.utoronto.ca/article/10.1007/s42452-023-05581-8#citeas.

Aman Upadhyay, Niha Kamal Basha, and Balasundaram Ananthakrishnan. Deep learning-based yoga posture recognition using the y pn-mssd model for yoga practitioners. *Healthcare*, 2023. URL https://www.mdpi.com/2227-9032/11/4/609.