YOGA POSE CLASSIFICATION PROGRESS REPORT

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

ABSTRACT

In this project, we use a convolutional neural network (CNN) to classify a yoga pose given an input image. We processed the dataset Saxena (2020) by removing images and augmenting the data. We further explain our preliminary baseline model and primary CNN. Finally, we detail individual group member responsibilities. —-Total Pages: 9

1 BRIEF PROJECT DESCRIPTION

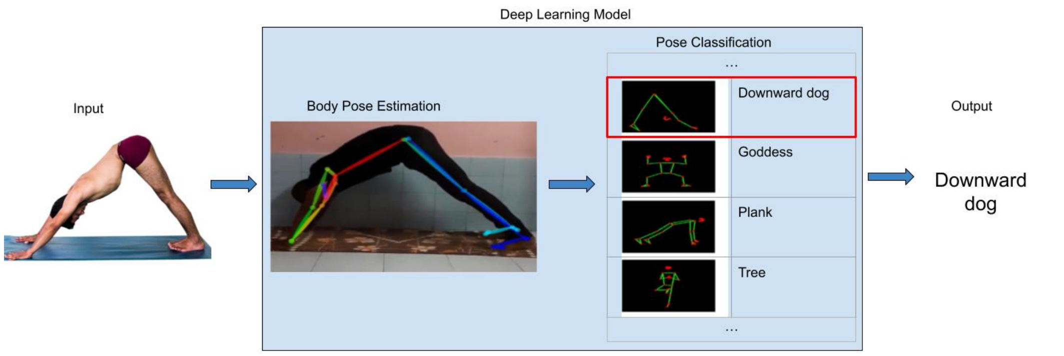


Figure 1: Yoga pose classification model.

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. The goal of this project is to use deep learning (and specifically a convolutional neural network) to accurately name/classify what yoga pose is represented in an image (see 1). Note that the body pose estimation in our deep learning model (figure 1) has not worked out as hoped, but our CNN could still predict a pose name given an input 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 (see figure 1). 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. For example, it could send scores and feedback based on user’s poses like an actual instructor or recognize yoga poses in videos like several related projects (Singh et al., 2022), (Upadhyay et al., 2023)

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 Individual Contributions and Responsibilities

2.1 Team Collaboration Tools and Methods

Our group is dedicated to promoting efficient communication and teamwork by utilizing a range of project management tools and methods.

2.1.1 Google Drive

Google Drive is our main collaborative platform for document editing, sharing, and file storage. It allows us to seamlessly share project related files, have real-time editing amongst multiple team members, and leave feedback using the comments and suggest edits features. Our team uses Google Drive as a project management tool as we create and manage task tables like **??** and 1 in Google Docs to track our progress. These tables are regularly updated to reflect our tasks statuses, task descriptions and criteria, which member is assigned to each task, and our internal deadlines. In addition, we store our meeting minutes, research notes, and working drafts in our shared folder.

2.1.2 Github

Github is used to manage our code and to maintain version control, preventing us from overwriting each other's code. We work on separate branches and create pull requests for other members to review the code before it is merged into the main branch. Each pull request corresponds to a different task and states what changes were made, its status, and what the intended result is. Github helps our team ensure code integrity and quality.

2.1.3 Discord

Discord is our primary communication channel for day-to-day, informal communication, quick updates, and planning. Direct messages are used for private conversations and one-on-one discussions. The discord call feature is used when we want to have quick update calls to answer or ask any questions, ensure that all of our tasks are clear, and provide updates on the current status of each assigned task. The ability of voice and video calls makes it easier to switch from text to verbal communication when necessary.

2.1.4 Meeting Schedule

Our team met on Wednesdays at 9pm to ensure consistent progress and address any emerging issues. These meetings acted as checkpoints for us to assess our progress, establish objectives, talk about any challenges, and help each other with our tasks. Holding these meetings were essential to keeping everyone in sync and in agreement with the direction of our project. We also had impromptu meetings that were scheduled as needed towards deliverable due dates. These extra meetings are especially helpful when dealing with more difficult tasks and to ensure we meet our deliverable deadlines.

2.2 Individual Contributions and Tasks

Our individual contributions and responsibilities are summarized in two tables. The first one shows our focus for this progress report **??**, while the second one shows our future deadlines and tasks 1. Finally, we add details about individual responsibilities in a section for each team member.

2.2.1 George's responsibilities

In the next three paragraphs, I detail my work for the project proposal, work for the progress report, and future expectations.

Table 1: Overall future tasks and deadlines

Task	Description	Assigned To	Due Date
Baseline Model Training	Try handcrafted features	Marcus	Thursday, July 11
	Try Support Vector Machine (SVM)	Harry & Marcus	
Collect testing data	Take manual pictures of yoga poses	Everyone	Monday, July 15
	Try to find and reformat dataset with unseen human photos	Rosalie	Thursday, July 25
Analyze our output	Analyze the initial results (plots) and two model's performance after training	Marcus & George	Thursday, July 11
Tune hyperparameters	Experiment with various hyperparameters to optimize the performance of our model	Rosalie & Harry	Thursday, July 18
Evaluate Model	Evaluate the performance of our model based on its training error and accuracy, testing error and accuracy, etc	Harry & George	Thursday, July 25
Write Final Report	Write a draft on the model's final progress and include: Introduction, Illustration/Figure, Background & Related Work, Data Processing, Architecture, Baseline Model, Quantitative and Qualitative Results, Evaluate Model on New Data	Everyone (prepare the work they've done)	Tuesday Aug 13
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 August 10
Record Final Presentation	Have all presentation content prepared and ready to film	Everyone (prepare the 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

For the project proposal, I completed four responsibilities. First, I drafted and later finished the introduction. Secondly, I built on Marcus's illustration draft to create the final illustration. Thirdly, I brainstormed future tasks for the project plan and moved our project plan tables from google docs to overleaf. Finally, I helped the team edit our project proposal.

For our progress report, I've been primarily responsible for building off my previous work and cleaning our model data. For example, I wrote the brief project description building off our past introduction. I also continued my role formatting citations and tables in overleaf. When cleaning the data, I failed to meet several deliverables/goals; I couldn't remove cartoon images and neglected to resize or load the data into our model (Rosalie did it instead). However, I did remove unwanted images from our data and write about that.

In the future (see 1), I'm responsible for analyzing and plotting performance of both baseline and primary models. For the final report and presentation, I plan to contribute to sections about project goals, data cleaning, and quantitative model results.

2.2.2 Marcus's Responsibilities

For the project proposal, I conducted research on datasets and the types of data processing required for our chosen datasets. I also researched conventional methods used for pose estimation and pose classification, specifically selecting the OpenPose library as our tool for pose estimation. Additionally, I investigated the types of architectures relevant to our project and wrote report sections corresponding to these research areas.

During the project proposal phase, I implemented OpenPose but encountered issues with the Windows Portable Version, which led me to pivot to other tasks temporarily. I also implemented a Convolutional Neural Network (CNN) as our primary model.

For future tasks, I will focus on tuning hyperparameters to optimize our model's performance.

2.2.3 Rosalie's Responsibilities

For the project proposal, I conducted research on five relevant works for the Background and Related Work section, summarizing their key elements and relevance to our project. I also examined ethical issues, including model limitations and biases in the training data, and suggested ways to mitigate these concerns in the Ethical Considerations section. Additionally, I assisted in writing the task descriptions for the Project Plan section and contributed to the Risk Register by analyzing significant project risks, their probabilities, and mitigation strategies.

For the project report, I developed data augmentation code to enhance dataset diversity and robustness by rotating, scaling, and altering images. I ensured the code's functionality and provided visual examples of the cleaned, augmented data. I outlined a testing strategy for the model on unseen data to ensure reliability and co-authored the section on team collaboration methods and tools.

In the future, I will be responsible for finding and reformatting datasets with unseen human images for testing our model, as well as taking photos of diverse individuals performing yoga poses in various settings and lighting conditions. I will tune the hyperparameters to optimize model performance by adjusting variables such as learning rate, batch size, and the number of epochs. Additionally, I will edit and assemble the final presentation video, ensuring it is coherent and engaging. Lastly, I will help write and edit the written deliverables to meet guidelines and clearly convey the project's status and results.

2.2.4 Harry's Responsibilities

For the project proposal, I focused on developing a comprehensive plan for using CNNs to classify yoga poses. I researched related works, summarized their findings, and addressed ethical issues for the use of the primary model.

In the project report, I implemented a Random Forest classifier as a baseline model for yoga pose classification. This involved preprocessing the image data, structuring it effectively, and optimizing the model through hyperparameter tuning. I managed memory usage challenges to ensure a robust and reliable model.

Moving forward, I will refine the Random Forest classifier and prepare for the CNN implementation. This includes optimizing parameters, expanding the dataset with diverse images, and enriching the training data with new photos. Once the baseline model is fully optimized, I will develop the CNN, tune its hyperparameters, and ensure it surpasses the baseline. Finally, I will help with the final presentation and written deliverables to communicate the projects' progress and results.

3 Data Processing

Our overall deep learning process can be summarized by these steps:

1. Remove unwanted images
   (a) Duplicates, outliers, blurry or dark
2. Augment data
   (a) Resize image to 680x440 pixels, and create 3 augmented images per each original image that are rotated, flipped, and transformed
3. Run data through openpose
4. Run data through CNN classifier

Our original data all came from one dataset on Kaggle (Saxena, 2020), since we determined that it had the most images (5994) and a great variety of poses (107). To import the data to a Google Colab notebook, we used the opendataset package (geeksforgeeks, 2023).

3.1 Removing unwanted data

We first removed misleading images to the best of our ability. The fastdup Python package (fastdup), (Neoh, 2023) allowed us to visualize, analyze, and remove unwanted data that might confuse our model when training. Specifically, we detected duplicates, blurry/dark images, and outliers to remove with the tool. It also confirmed that none of the dataset images were corrupted or broken. Note we incorporated code from a main fastdup tutorial (fastdup) and two other sources that helped find duplicates (Neoh, 2023), (dnth, 2024).

We visually checked that the "bright" images detected by fastdup could be easily classified into poses by a human. However, other categories of images appeared unusable. The dark and blurry images are pitch black, while outlier images do not look like yoga poses 3.

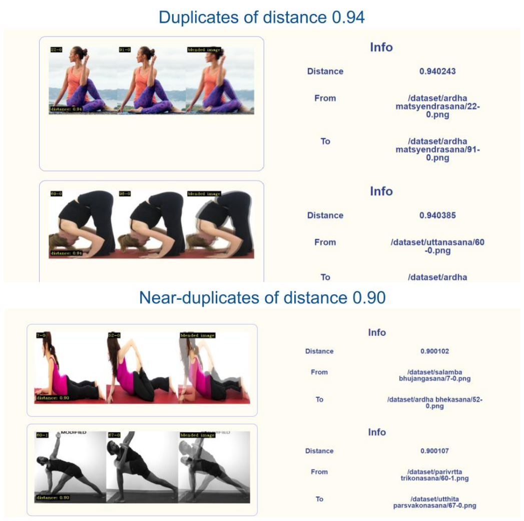
Note that fastdup determined which images are duplicates or outliers based on their distance from other images. A score of 1 indicates that two images are the same, while a score of 0 shows that they are totally different. By visualizing images of various distances (see figure 2), we decided to classify images with distance above 0.94 as duplicates. We removed one of the duplicates to avoid misleading the model. Finally, we used the fastdup tutorial default of classifying images with distance below 0.68 as outliers (fastdup).

Ultimately, we found 306 unique problematic images out of our initial 5994. This included 279 removed duplicates, 19 outliers, 8 dark images, and 7 blurry images. We stored the file paths in a python list and removed them with *os.remove(image_path)* (geeksforgeeks, 2022). Finally, we zipped the new dataset in Google Colab (Ahx, 2022) and stored the remaining 5688 image files on Kaggle.

3.2 Augmented Data

In order to augment our dataset, we created a custom coding program that automates the augmentation process, guaranteeing a methodical and repeatable procedure. The first thing we did was download and unzip the dataset provided by Kaggle with the various pose images. This data was structured by storing all of the images of a given pose in a subfolder titled by that pose's name. In order to augment this data, we iterate through each image file in each subfolder with our code. We then augment that image using our augment image function.

This augment image function loads the image file into an array and uses the keras ImageDataGenerator object to apply various augmentations to the data, such as flipping or rotating the image. The



Duplicates of distance 0.94

Info

Distance 0.940243

From /dataset/ardha matsyendrasana/22-0.png

To /dataset/ardha matsyendrasana/91-0.png

Info

Distance 0.940385

From /dataset/uttanasana/60-0.png

To /dataset/ardha

Near-duplicates of distance 0.90

Info

Distance 0.900102

From /dataset/salamba bhujangasana/7-0.png

To /dataset/ardha bhekasana/52-0.png

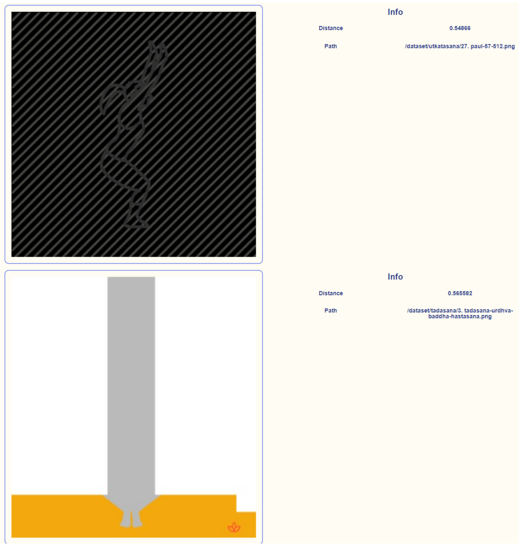
Info

Distance 0.900107

From /dataset/parivrtta trikonasana/60-1.png

To /dataset/utthita parsvakonasana/67-0.png

Figure 2: fastdup duplicate images



Info

Distance 0.54866

Path /dataset/utkatasana/27. paul-57-512.png

Info

Distance 0.565582

Path /dataset/tadasana/3. tadasana-urdhva-baddha-hastasana.png

Figure 3: fastdup outlier images

augment image function generates *num_images* augmented images and saves them under a subfolder title the name of the original image, which is contained in the subfolder for the image, along with the original image. In addition to these augmentations, we ensure that all images are within a 680x440 image for processing later on before saving them. In order to do this, we create an all white "frame" image that is 680x440 then paste the image in the center of this frame. For example, if *num_images* equals 3 and we are augmenting the image *1. 1* for the pose downward dog, the data augmentation process will generate the following folders/files:

augmented_dataset:

- downward dog
  - 1. 1
    - 1. 1_aug_0.png
    - 1. 1_aug_1.png
    - 1. 1_aug_2.png
    - 1. 1_original.png

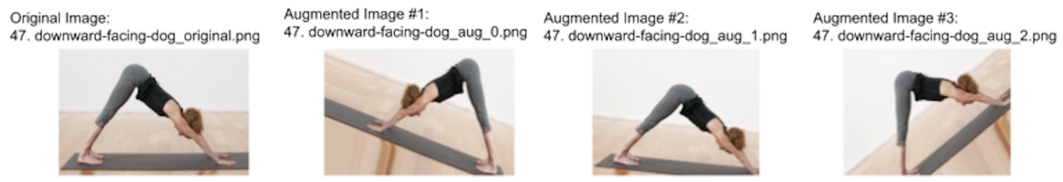

Original Image:
47. downward-facing-dog_original.png
Augmented Image #1:
47. downward-facing-dog_aug_0.png
Augmented Image #2:
47. downward-facing-dog_aug_1.png
Augmented Image #3:
47. downward-facing-dog_aug_2.png

Figure 4: Example of Augmenting an Image

3.3 FINAL TESTING DATA

For final testing, we plan to both take images of yoga poses ourselves and try finding a different unseen dataset. This unseen data is important since our model may have an advantage at identifying poses in our main training and validation dataset (Saxena, 2020). As shown in figure 2, many poses in that dataset are performed by the same person.

Self-Collection: We plan on taking our own images of us and others doing different yoga poses, under different lighting conditions, backgrounds, and angles in order to mimic real-world scenarios. Images will be taken in a broad range of settings, such as both indoors and outdoors with various lighting conditions. Additionally we will try to have many people do the yoga poses to add diversity, ensuring that there is a range of body types, attires, and styles represented.

Using A Different Dataset: We also intend to use a different dataset that was not used during the training or validation stages if gathering new data for certain yoga poses is not possible within our team's ability to physically do those yoga poses or if it is not within our time frame. The same yoga positions will be depicted in this dataset. If images in this dataset do not depict all of the yoga poses we had, we intend on collecting images from the internet that correspond to each yoga pose.

Evaluation of Unseen Data: In order to evaluate our model's performance using this unseen data, we plan on calculating the percentage of accurately predicted labels among all of the predictions. The model's precision and recall in identifying each class will also be calculated.

3.4 CHALLENGES

One challenge we faced during data processing was the huge variation of yoga photos in our dataset. They include cartoon, anatomical, and heavily annotated/labelled images as shown in figure 5.

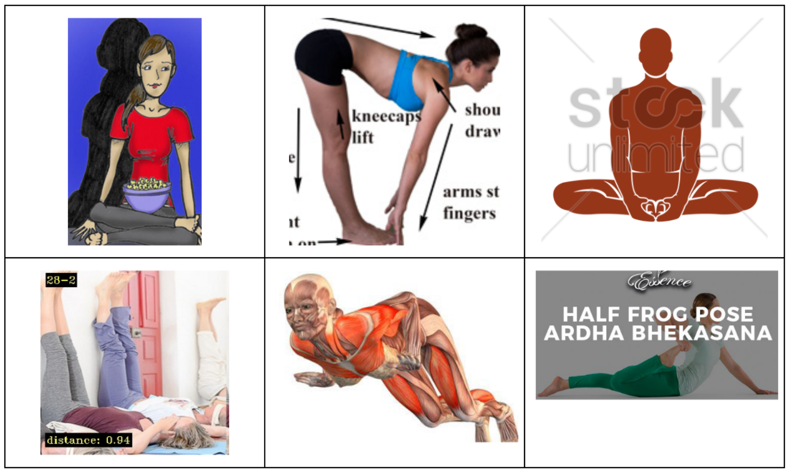


Figure 5: Variety of yoga image types

However, we planned to only classify yoga photos of a real person, since model users are expected to classify poses they perform. Thus, we worried that our model accuracy would be reduced since it learns from such a crazy variety of images.

Although we could not remove cartoon images as we planned, passing our data through OpenPose could extract the relevant body positions from any style of yoga image. This would resolve the challenge of using a huge variety of pose image styles.

4 BASELINE MODEL

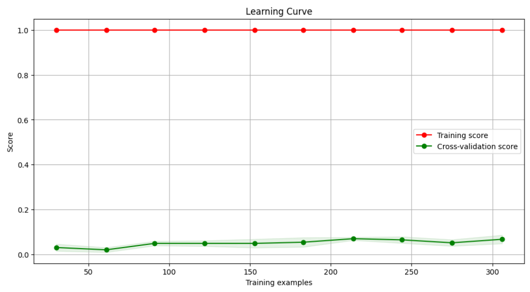


Figure 6: Learning Curve

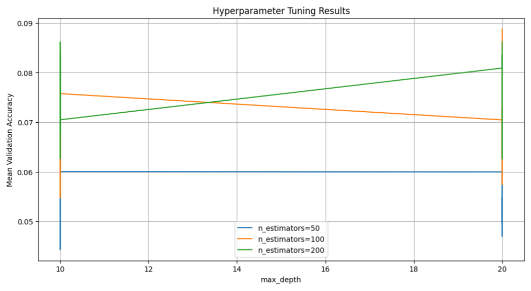


Figure 7: Hypertuning Results

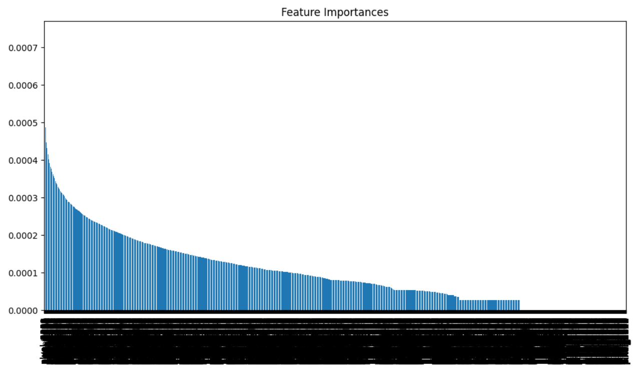


Figure 8: Feature Importance

4.1 Steps and Approaches

For our baseline model, we chose the Random Forest (RF) classifier due to its ability to handle large datasets. with high dimensionality. RF build multiple decision trees during training and outputs the mode of the classes for classification tasks. This method helps reduce overfitting and improves generalization compared to single decision trees. 1. Data Preprocessing: The image data was all uniformly resized to 640x480 pixels, normalized, and structured into training, validation, and test sets. This ensures uniformity and helps the model learn effectively.
2. Feature Selection: Using SelectKBest, we reduced the feature size to 10,000 most relevant features to manage computational load and improve performance.
3. Hyperparameter Tuning: We used GridSearchCV to find the optimal parameters for the Random Forest classifier. The parameters tuned included the number of trees (n$_{e}stimators), maximumtreedepth(max_{d}epth), minimumsamplestosplitanode(min_{s}amples_{s}plit), andminimumsam$
4. Model Training and Evaluation: The best model from GridSearchCV was selected and evaluated on the validation and test sets to measure its performance.

4.2 Quantitative Result

The hyperparameter tuning results are shown in image 7. The plot illustrates the mean validation accuracy for different combinations of ”n$_{e}stimators”and”max_{d}epth”$.
The image 8 shows the feature importance derived from the Random Forest classifier. The plot indicates which features contributed most to the models’ decisions.
The validation accuracy achieved was low (2.3 percent), indicating that the model struggled to generalize from the training data. However, this issue has been identified as primarily due to the high dimensionality of the data and the potential mismatch between feature extraction and model complexity. This problem is feasible to fix moving forward by further optimizing feature selection and exploring more suitable hyperparameters.

4.3 Qualitative Result

The learning curve is depicted in the image 6. This curve plots the training score and cross-validation score as functions of the number of training examples.
The learning curve shows that the training score is consistently high, while the cross-validation score is significantly lower. This indicates overfitting, where the model performs well on the training data

but poorly on unseen validation data. This observation confirms that the model is not generalizing well, which aligns with the low validation accuracy.
To address this, we will: 1. Improve Feature Selection: Use more advanced techniques to ensure the features fed into the model are more representative of the underlying data. 2. Regularization: Introduce regularization techniques to prevent overfitting. 3. Additional Data: Expand the dataset with more diverse images of yoga poses to improve model robustness.

5 Primary Model

The current model serves as a starting point for building our future CNN model. It consists of 2 encoder units (i.e., convolution + Relu + Pooling) as well as 2 fully connected layers. In actuality, our CNN model is meant to be combined with OpenPose, which preprocesses our image into figures of skeletons. As a result, our model should have much better predictive ability than the current one once OpenPose comes into play. However, at the time being, this CNN model is still being trained with raw dataset (i.e. images that have not been processed by OpenPose) so that we are sure the dimensions of each architecture components are being calibrated correctly. Figure 6 is an illustration of our CNN.

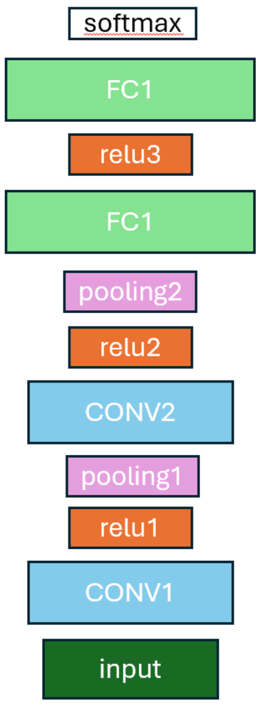


Figure 9: CNN Architecture

Our CNN model is being trained with the following hyperparameters:

- batch size: 32
- loss function: Cross Entropy
- learning rate: 0.001
- Number of epochs: 10

Below is the loss calculated at the 100th batch during each epoch:

- 1st epoch: 4.677
- 2nd epoch: 3.809
- 3rd epoch: 2.582
- 4th epoch: 1.535
- 5th epoch: 0.533
- 6th epoch: 0.131
- 7th epoch: 0.068
- 8th epoch: 0.043
- 9th epoch: 0.016
- 10th epoch: 0.021

The final test accuracy of the trained model is 36.03%. Despite the current test accuracy is unsatisfactory, we know that a significant reason is that OpenPose has yet to work together with this model. The future training of the model requires first successfully implementing OpenPose.

References

Ahx. Download folder from google-colab, 2022. URL https://www.youtube.com/watch?v=Ann21Y0kmVg.

dnth. Finding and removing duplicates, 2024. URL https://github.com/visual-layer/fastdup/blob/main/examples/finding-removing-duplicates.ipynb.

fastdup. Cleaning image dataset, 2023. URL https://visual-layer.readme.io/docs/cleaning-image-dataset.

geeksforgeeks. Delete a directory or file using python, 2022. URL https://www.geeksforgeeks.org/delete-a-directory-or-file-using-python/.

geeksforgeeks. How to import kaggle datasets directly into google colab, 2023. URL https://www.geeksforgeeks.org/how-to-import-kaggle-datasets-directly-into-google-colab/.

Dickson Neoh. fastdup: A powerful tool to manage, clean & curate visual data at scale on your cpu — for free., 2023. URL https://medium.com/visual-layer/fastdup-a-powerful-tool-to-manage-clean-curate-visual-data-at-scale-on-your-cpu-fo

Shruti Saxena. Yoga pose image classification dataset, 2020. 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.