National College of Ireland

A Deep Learning Framework to Classify Yoga Poses Hierarchically

MSc Research Project
MSc Data Analytics

Pooja Rakate
Student ID: x19241267

School of Computing
National College of Ireland

Supervisor: Dr Paul Stynes, Dr Pramod Pathak

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Pooja Rakate		
Student ID:	X19241267		
Programme:	MSc Data Analytics	**Year:**	2020 - 2021
Module:	Research Project		
Supervisor:	Dr Paul Stynes, Dr Pramod Pathak		
Submission Due Date:	16/08/2021		
Project Title:	A Deep Learning Framework to classify Yoga poses Hierarchically		
Word Count:	5829	**Page Count: 13**	

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Pooja Rakate

Date: 16/08/2021

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A Deep Learning Framework to Classify Yoga Poses Hierarchically

Pooja Rakate

X19241267

Abstract

Yoga is beneficial for both the physical and mental well-being of individuals of all ages. Currently, machine learning and deep learning techniques are employed to perform flat classification of precisely 12 yoga poses using the human pose estimation (HPE) model 'OpenPose' (OP) followed by CNN. However, building a deep learning framework considering hierarchical postures details and increasing the scope by including many yoga poses is a challenging region of research. Thus, this study aims to improve the hierarchical yoga pose classification performance by three methods. Firstly, by introducing the OP model in the deep learning framework. Secondly, by adding a connectivity pattern to build the DenseNet201 Hierarchical-concat architecture, and finally, it proposes to utilize the VGG19 network to classify a total of 82 yoga poses hierarchically. The performance of all these models was compared based on top-1, top-3, and top-5 accuracy rates. The said objective was achieved by DenseNet201 Hierarchical-concat proposed as the second approach. It was found to perform the best amongst all giving enhanced results than the state-of-the-art with 73%, 63.54%, and 52% top-1 accuracy on hierarchical levels 1, 2, and 3 respectively. While other models were found to be not performing satisfactorily. Passage of lower-level features directly to a higher level was driven by the connectivity pattern. This approach demonstrated a significant increase in the classification performance on all levels which would be able to classify a given pose well on the basic posture levels 1 and 2. Thus, with the success of this study, the scope of the yoga pose recognition system was broadened by considering a total of 82 poses. This facilitates a yoga pose correction system that can correct a wide variety of poses. However, there is a scope to obtain better results with all the models by fine-tuning and training them on the entire dataset.

1 Introduction

To get the best out of yoga, it is necessary to follow the correct steps and perform the pose correctly. As a result, it is vital to develop a system that can notify a yogi (yoga practitioner) whether they are correctly performing the yoga postures. The first phase of such a system is implemented in this research, which is to recognize the yoga pose that the yogi is doing. Only a modest number of common yoga poses have been classified to date using the Human Pose estimation (HPE) model such as the OpenPose (OP) model and a convolutional neural network classifier. However, none of these considered the yoga postures' hierarchical features and a variety of complex or similar-looking yoga poses. Considering the coarse to fine details can help to explain yoga positions more accurately and thus lead to building a better correction system. Also, the studies in this domain have not explored the potential of using pre-trained networks for yoga pose classification.

Thus, this research will try to fill the gap of considering only a limited number of poses by undertaking recognition of a total of 82 yoga poses. These poses have been labeled hierarchically on three levels based on their posture details. An example of hierarchical labeling of a pose is shown in Figure 1. The pose belongs to the "sitting" class on the coarse level 1, the "Normal 1 (legs in front)" class on the coarse level 2, and the "Bound Angle

pose" on the fine level. The poses will be classified hierarchically by modifying the architecture of DenseNet201 and VGG19 networks.

Hence, **the main goal of this research is to improve upon the existing hierarchical classification performance of the yoga pose recognition system**. Thus, the model would be trained to identify yoga poses on all three levels.

Following research objectives were derived to address the research question:

- Broadly investigate and implement the state of the art (SOTA) model built by Verma et al. (2020) which uses the DenseNet201 architecture for hierarchical classification.
- Designing and implementing a novel deep learning framework in which the OpenPose model would be used for pre-processing and the modified DenseNet201 architecture will be used as the classifier.
- Introducing a connectivity pattern in the modified DenseNet201 model to improve the classification accuracy.
- Modify the VGG19 architecture to perform hierarchical classification and compare it with all the models proposed above.
- Evaluate and discuss the results of the implemented framework by using evaluation metrics such as top-1, top-3, and top-5 prediction accuracy rates and find the approach giving the best results.

One of the contributions of this study is to verify how well a combination of the OpenPose model and a pre-trained classifier would perform when the data has very complex postures. The study also tests if the classification performance increases by introducing the connectivity pattern in the classifier. Overall, the enhanced classification performance for a total of 82 poses would lead to building a generic yoga pose correction system that covers a vast variety of complex and similar-looking poses. Owing to the system constraints, only half of the original "Yoga-82[1] " dataset was used for all the experiments.

Figure 1: Hierarchical representation of a yoga pose on coarse level 1, coarse level 2, and fine level in the hierarchy.

This paper discusses the deep learning and human pose estimation models used for classifying yoga posture images hierarchically in Section 2 related work. The research methodology is presented in section 3. The design specification and implementation are described in sections 4 and 5 respectively. Section 6 named Evaluation presents the experiments, results, and discussions. Section 7 concludes the research and discusses future work.

[1] https://sites.google.com/view/yoga-82/home

2 Related Work

A critical evaluation and discussion of the work done in the domains of yoga posture classification, Hierarchical classification, and the “Human Pose estimation model” is summarized in this section leading to understanding the need of investigating the performance of the OpenPose (OP) model concerning complex posture and the necessity of hierarchical classification in this domain.

Agrawal, Shah, and Sharma, (2020) and Palanimeera and Ponmozhi (2021) classified 10 different Yoga postures and the sun salutation poses respectively using the OP (Cao et al., 2017) model. The output from the OP model output was fed to several classical machine learning algorithms and the random forest was found to perform the best amongst all.
Using the OP model, Huang et al. (2020) created an application that could deliver real-time corrections to a user's posture. There was no adequate information provided about the model's training phase, and the information provided appeared to be insufficient to reproduce this work. Yadav et al., (2019) and Kothari (2020) used video yoga data. The former proposed a hybrid CNN-LSTM deep learning model to classify 6 yoga asanas and while the latter compared the CNN-LSTM model with an SVM and a CNN model and proved that the CNN-LSTM model is better. Yadav et al., (2019) achieved an accuracy of 99.38% on the video data. Images/videos of one or more participants practicing yoga in a closed room were captured for all studies. As a result, when the data lack a consistent background or illumination, these models are at risk of failing.

The flat classification of objects, also known as binary classification or multiclass classification, has been the focus of many studies. Hierarchical classification can, however, make it easier to differentiate the same entity at different hierarchical levels. Guo et al. (2018) implemented a CNN-RNN model where they used CNN to generate images and RNN to classify images hierarchically. With an accuracy rate of 90.69 percent, the CNN-RNN network exceeded all previous work results. A Hierarchical classification Convolutional Neural Network was built by Seo and Shin (2019) in the fashion apparel domain (H-CNN). They added a fully connected block each at the output of the 2^{nd}, 3^{rd}, and 5^{th} dense blocks of the VGG19 network to classify the coarse level 1, coarse level 2, and fine level. This model had 93.30% accuracy while the base model had 92.90%. A CNN called branch-net was built and trained by Inoue, Forster, and Santos (2020) one output branch corresponding to each hierarchical level. They proposed two connectivity patterns namely, Concat-net and Add-net. On Fashion MNIST and CIFAR-100 data, the Concat-net model outperformed the branch-net model, with an accuracy of 91.22% and 52.32% respectively. The fact that they evaluated the topologies on multiple levels of classifications, i.e., CIFAR-100 for three levels and Fashion-MNIST for two hierarchical levels, was an important aspect of this work. Zhang et al. (2021) proposed Hierarchical Bilinear CNN (HB-CNN). They introduce the bilinear module in addition to the connectivity pattern to transfer the feature maps from the lower level to the final prediction level intending to increase classification accuracy and to avoid the issue of vanishing gradient. As a result of their approach, they achieved 91.75 percent, 66.03 percent, and 91.10 percent accuracy on the CIFAR-10 dataset, CIFAR-100 dataset, and Orchid images dataset, respectively. "Yoga-82", a large-scale hierarchical dataset for yoga posture recognition, was introduced by Verma et al. (2020). Using a pre-trained DenseNet201 (Huang et al., 2017) and adding output branches for each level of the hierarchy, they classified yoga poses on three levels hierarchically, from coarse level to fine level. To improve classification accuracy on the first coarse level, they added a block of convolutional layers to the branch that classifies the first hierarchical level. This dataset inspired the current research project in a view to building a yoga pose correction system considering a higher number of diverse yoga poses.

Works by Wu et al. (2019) and Xin Jin et al. (2015) use either wearable devices or Microsoft's Kinect sensor for posture guidance. Indeed, such systems operate effectively, but a dedicated gadget is required to determine the body's key points, which is not user-friendly. Eliminating the need of using body sensors or devices various Human pose estimation models such as Mask RCNN, OpenPose (OP), PoseNet, etc. were built. Although Mask RCNN had better test accuracy than OP, Rishan et al. (2020) found that when evaluated on real-world data, Mask RCNN had more false-positive results and true negatives. Thus, A classroom-based sitting posture identification system was developed by Chen, (2019) using the OP model which was used to build skeleton pictures from the identified key joint points, which were then input to a 19-layer CNN. Park, Baek, and Kim (2020) designed a push-up counter which identified correctly performing pushups and counted those push-ups by using the OP model.

In conclusion, due to a lack of publicly available yoga pose datasets, researching the domain of yoga posture classification is a challenging task. Yet, a notable amount of work has been done on Yoga pose classification mostly using human pose estimation models. However, all of this work has focused on the flat classification of yoga postures and a maximum of ten different yoga poses were classified until now. The complexity of various yoga postures has not been touched upon yet. The Yoga-82 dataset has a total of 82 different yoga poses. Thus, it allows verifying if using the HPE model such as the OP model with CNN models can still fetch good results in pose recognition tasks concerning pose complexity. The use of the OP model has achieved good results in the flat classification of a few not so very complex yoga poses. Also, considering the hierarchical posture details while classifying the yoga poses is a relatively new experiment in the field of classification of yoga poses. So, a solution that addresses this problem is required. Hence, this research proposes to broaden the scope of the yoga recognition system by considering a total of 82 yoga poses while also improving their hierarchical classification performance. It aims to do so in three ways. Firstly by suggesting a novel deep learning framework by using OpenPose (Cao et al., 2017) as a pre-processing step to create skeleton (Chen, 2019) forms of the 82 poses. Secondly by adding a connectivity pattern (Inoue, Forster, and Santos, 2020) in the state-of-the-art architecture by Verma et al. (2020) and finally by proposing a modified VGG19 architecture. The yoga poses will be classified on a hierarchical level as it helps to understand the granular details of the image and also improves performance.

3 Research Methodology

To achieve the aim of improving the existing hierarchical yoga pose classification performance, three approaches were identified and the following research methodology comprising of five steps was followed.

In the first step, data were acquired from the first large-scale hierarchically labeled dataset for Yoga images developed recently. As opposed to the limited existing yoga datasets available, the Yoga-82 dataset is challenging and has diversity in its images concerning the number of yoga poses included and the complexity in the images. It includes images with a variety of backdrops, such as indoor, outdoor, etc., and some solely show the shadow, sketch, versions of the yoga poses. The data set provided three text files. The train, and test label files, and a file with image names and their URL to download them from the internet. The poses in this dataset have been hierarchically labeled on three levels that have 6, 20, and 82 classes, respectively. The three levels of hierarchy represent the basic posture of the body in coarse level 1 (6 classes), a detailed variation in the body posture in coarse level 2 (20 classes), and finally the actual pose names in the fine level (82 classes).

In the preprocessing step firstly, all the images excluding the Graphics Interchange Format (gif) images were downloaded and saved to the local machine using the JAVA programming language. Images not found were noted. Some downloaded images were not readable; hence, these images were noted and deleted using Python. Entries in the train and test label files matching with the not found / not readable images were removed to maintain consistency. This left us with a total of 14,401 training images and 4,935 test images. To properly train a model from scratch with such a huge number of images requires a system with high computational power. Hence, finally, this data was halved maintaining the original proportionality of images in each class and was used for all the experiments.

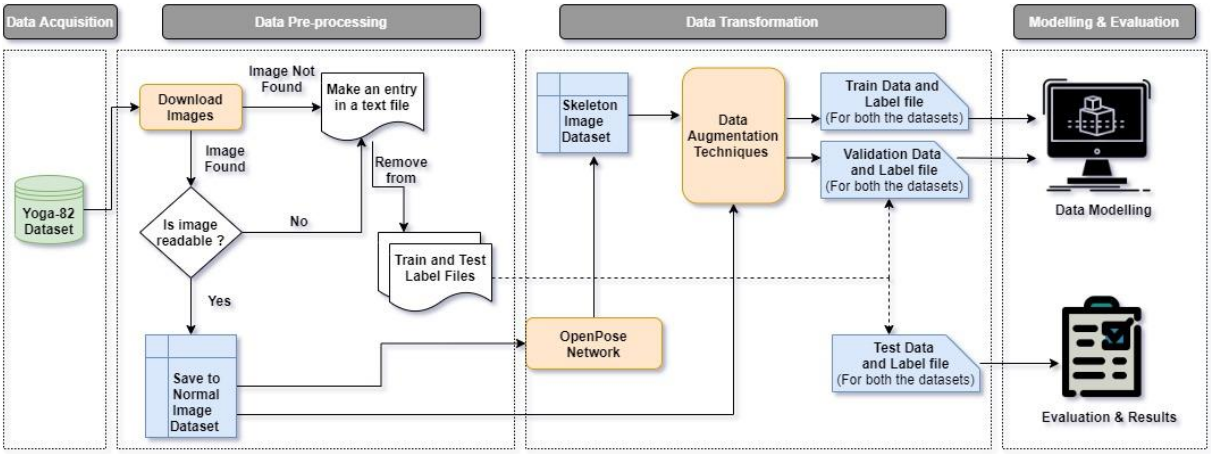


Figure 2: Process Flow Diagram

The data transformation step was divided into two parts. First was extracting the skeleton images from the OpenPose (OP) model data second was data augmentation resulting in a balanced dataset.

Figure 3: Example of how the skeleton form images created using OpenPose network look like

OP employs CNN-based architecture to detect the facial, hand, and foot key points such as elbow, neck, eyes, shoulder, nose, knees, hips, etc. of humans in an image. This would be a good way to capture only the poses and discard any other details in the images (Chen, 2019). The OP algorithm checks each grid in an image for a human joint location key point using the Part Confidence Maps and Part Affinity Fields. This grid size in this study is 256*256. The points with a probability greater than the specified thresholds were used to produce the individual's 2D skeleton by joining these key points. The OP added to the Python-based computer vision OpenCV library was used with the 18 points COCO Caffe model weights to obtain the skeleton images. Since, the images contain silhouette and drawing versions of the yoga poses too, the OP model did not detect the key points for them. Hence, images were dropped if no key points were detected or only a maximum of 5 key points were detected for them. The rest of the images were saved with a black background and containing the skeleton form of the yoga pose only as shown in Figure 3.

An alternative approach that would aid in increasing the hierarchical classification performance was also included. This was carrying out data augmentation to balance the data as both the skeleton and normal image datasets had an imbalance in the data. The classes in the training data had images ranging between a value of 13 to 230 per class. Such an imbalance can directly affect the model's prediction accuracy. Hence, to tackle this and not reduce the number of images further, each class was either down-sampled or up-sampled to have a total of 65 images in each class. The up sampling was done by flipping images and rotating the images at an angle of 45 degrees using the OpenCV library as shown in Figure 4. Only the training data was augmented. Subsequently, the entries of dropped images from both the transformation steps were removed from the train and test label files.

Figure 4: Different Augmentation Techniques applied on an image of Chair Pose

For model building and evaluation purposes, the provided test label file was divided maintaining original proportionality between the number of images in each class to create validation and test label files. In this phase, modified versions of pre-trained deep learning DenseNet201 and VGG19 architectures were fitted on the data. Details about all the models and their implementation are provided in section 4 (Design Specification and Implementation) of the report.

A total of 82 end classes makes this a large-scale classification challenge, thus, the **top-1, top-3, and top-5 accuracy** measures were used to evaluate the results of all the experiments. According to the top-1 accuracy, a model has classified a pose correctly if the model's top prediction was the actual true label while a model has a top-3 or top-5 accuracy when it classifies a pose, and its true label is among the top-three or five predictions respectively.

4 Design Specification

This research aims to improve upon the existing hierarchical classification performance of the yoga pose recognition system. Below is a brief description of the models designed to achieve this.

4.1 Modified State of the art (DenseNet201 Hierarchical) Architecture

DenseNet is a convolutional neural network that is designed such that each of its layers passes its feature maps to all its succeeding layers just like passing knowledge ahead. It typically consists of four dense blocks with 6, 12, 48, and 32 dense layers respectively. The top layers from the base DenseNet201 architecture were removed to customize it for hierarchical classification. Three output branches were added to this network to classify the three levels of hierarchical image annotations as shown in Figure 5. These three branches classify the coarse level 1, level 2, and the fine level (level 3) classes after the dense blocks 2, 3, and 4 respectively. To classify the coarse level 1 an additional dense block with 32 dense

layers is added to the first output branch. This was done to make sure that the classes at the coarse level 1 also are classified correctly. This dense block is the replica of the original dense block 4 of the DensetNet201 architecture. Thus, till the 2nd dense block, the model learns common features of the poses and then the further layers learn their specific details. This forms the structure of the state-of-the-art model built by Verma et al. (2020).

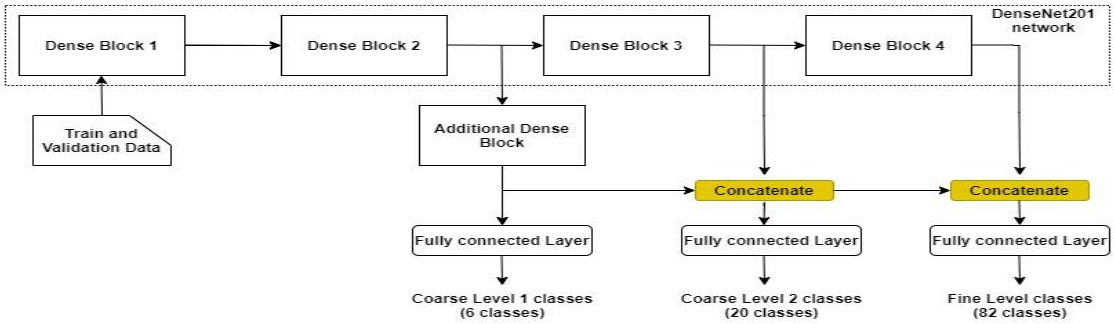


Figure 5: Architecture of DenseNet201 Hierarchical-concat

To improve the hierarchical classification performance, we proposed adding concatenation layers in the state-of-the-art architecture built by Verma et al. (2020) to the coarse level 2 and fine level output branches as shown in yellow blocks in Figure 5. This facilitated the direct passing of features specific to the coarser levels to higher levels for better hierarchical learning (Inoue, Forster, and Santos, 2020). For each output branch, the fully connected (FC) layer is preceded by a branch normalization layer, Relu activation, and a 2D global average pooling layer is applied. The FC layer of the current output branch is preceded by a concatenation layer. This concatenation layer concatenates the features from the pooling layer of the preceding output branch and the pooling layer of the current output branch. The FC layer has neurons equal to the number of classes in that level and the SoftMax activation function.

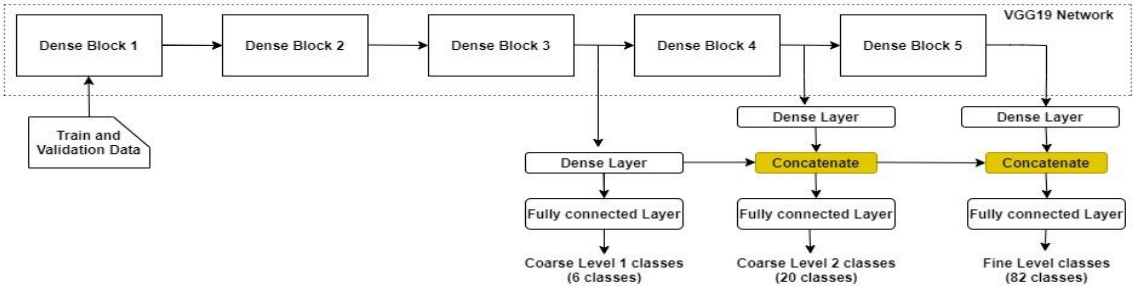


Figure 6: Architecture of VGG19 Hierarchical-concat model

4.2 Proposed Modified VGG19 Architecture

The VGG19 network typically has 5 dense blocks with only 19 layers of which 16 are convolutional layers, max-pooling layers, and three fully FC layers. Just like the model by Verma et al. (2020), branching has been done in the base model (VGG19) such that the three output branches classify the coarse level 1, level 2, and the fine level classes after the dense blocks 3, 4, and 5 respectively. As opposed to their (modified DenseNet201) architecture, no additional dense blocks were added to classify any hierarchical level. The output from the dense block is flattened and passed to each branch. A dense layer with 256 filters and Relu activation followed by a batch normalization layer for standardizing the inputs is added after the flatten layer for the coarse level 1 output branch. This is followed by a dropout layer for regularization and reducing overfitting with a factor of 0.5. A similar structure is designed for

the coarse level 2 and fine level output branches with only a difference in the number of filters (512) in the dense layer. Concatenation layers are added to the two higher-level branches between the dropout layer and the FC layer enabling the feature maps to directly pass from the previous level output branch to just before the current output level's FC layer. The FC layer has neurons equal to the number of classes in that level and the SoftMax activation function.

5 Implementation

Keras with TensorFlow backend was used to implement the novel deep learning framework and the proposed VGG19 network. The models and other pre-processing tasks were implemented in Jupyter Notebook using Python 3.7. Throughout all the experiments, the pre-processed images are resized to 224*224 before feeding them to the model. The Keras fit_generator function is used for fitting the model (training). A batch size of 32 was used for the experiments. The data generator function randomly fetches images equal to a number obtained by dividing the train samples by the batch size. Then it converts them to RGB color format, resizes them to 224*224, and then stores them into an array of size (batch size, 224, 224, 3). It encodes the true label using one-hot encoding for all three output levels and passes it to the generator function. For all the three output levels categorical cross-entropy is used as the loss function and loss weights are set to value 1 giving equal weightage to the loss for all three levels. The stochastic gradient descent (SGD) optimizer was chosen as the optimizer. The models were trained for 30 epochs owing to the system configuration limitations. 0.003 was set to be the starting learning rate and it was reduced by a factor of 10^{-1} when the validation loss did not improve. Since only the architecture of the DenseNet201 and VGG19 was used all the layers of the models were set to trainable so that the model could learn from scratch. There were a total of 5,350 images and 5,188 images in the imbalanced and balanced training datasets respectively. The validation data had 1453 images while the test data has 672 images. All the experiments were performed on Azure Windows Virtual Machine with 8 vCPU's and 112 GB RAM.

6 Evaluation

Table 1: a) Results from experiment 1; b) Results from experiment 2

Prediction Accuracy (in %)	*Without Augmentation*					
	Normal Image Dataset			Skeleton Image Dataset		
	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
Top - 1	**66.96**	**52.82**	**39.58**	57.58	39.58	28.12
Top - 3	89.73	73.66	56.54	84.37	70.23	46.72
Top - 5	97.76	84.07	65.62	96.57	80.95	57.88

a) Results for experiment 1

Prediction Accuracy (in %)	*With Augmentation*					
	Normal Image Dataset			Skeleton Image Dataset		
	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
Top - 1	**62.94**	**55.05**	**37.64**	57.88	49.7	34.52
Top - 3	88.69	76.33	59.97	85.56	74.85	55.95
Top - 5	98.06	84.22	68.6	97.47	84.07	66.51

b) Results for experiment 2

This study aims to increase the hierarchical classification performance of 82 yoga poses by the proposed three approaches. Apart from these, another approach of data augmentation was also considered. These experiments have been implemented and their results are discussed in this section. The model parameters described in implementation section 5 were maintained for all the experiments. Model weights and their structure were saved while training so that they could be loaded later. The results of the experiments carried out in this study are compared with the results obtained by implementing the hierarchical DenseNet201 model built by Verma, M. et al. (2020) as mentioned in section 2.2. For simplicity we will call this model as the base model. They trained their model for 50 epochs. In this research due to the system limitations, the based model was trained and tested on less than half of the

original dataset and for only 30 epochs. Hence, results cannot be directly compared with the results obtained by Verma, M. et al. (2020). In the results Table 1 and 2, the top-1,3, and 5 prediction accuracies are presented for the three hierarchical image annotation levels. Level 1, 2, and 3 represent the coarse level 1, coarse level 2, and fine level branches respectively.

6.1 Experiments and Results

Experiment 1 – The goal of this experiment was to replicate the state of the art model (base model). Both the normal and skeleton image dataset (created using the OpenPose model) were separately fed to the base model (Hierarchical DenseNet201) as shown in Figure 5 excluding the concatenate layers. No data augmentation techniques were applied to the datasets at this stage.

It is visible from the results of this experiment as shown in Table 1a that when the model is trained on a skeleton image dataset, it performs poorly than when trained with normal images. The model when trained with normal images has around 10% higher top-1 accuracy values than when trained with the skeleton image's dataset. In both the scenarios, there was not much difference in the top-3 and top-5 accuracies for all hierarchical levels. Thus, the first approach to achieve the aim of this research did not prove to work well. So, an alternate approach of data augmentation was tried in the next experiment. Also, the model seemed to overfit the data as the validation loss did not improve for a few epochs and hence necessary steps were taken in the next experiment.

Experiment 2 – The alternative approach of balancing the data (originally imbalanced) other than the originally proposed three approaches to increase the classification performance was tested in this experiment. Both the normal and skeleton image datasets obtained after doing data augmentation were fed to the base model for fitting. An early stopping callback was added while compiling the model that would stop model training if the validation loss did not improve after a specified number of epochs.

The results of this experiment can be seen in Table 1b. Near about the same scenario can be observed from the results of experiment 2 and the results in experiment 1. However, as opposed to the results of experiment 1, the top-1 accuracy is 5% higher for the model trained with the normal images than that trained with skeleton images.

A notable point from the first two experiments is that the prediction accuracy has improved only for coarse level 2 after doing data augmentation as compared to results without augmentation for the normal image dataset. However, the skeleton image dataset shows improvement in its performance after augmentation. Overall, we can conclude from these experimental results that complex yoga poses can be better classified using only the normal images. Thus, the next experiment was performed only on the original normal image dataset (without augmentation)

Table 2: Results from experiment 3

Prediction Accuracy ***(in %)***	**SOA DenseNet201 Hierarchical model (22.59M parameters)**			**DenseNet201 Hierarchical with concatenated layers (22.8M parameters)**			**VGG19 Hierarchical with concatenated layers (135.7M parameters)**		
	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3	Level 1	Level 2	Level 3
Top - 1	66.96	52.82	39.58	**73.06**	**63.54**	**51.93**	44.19	29.01	23.8
Top - 3	89.73	73.66	56.54	**93.75**	**84.97**	**69.94**	74.55	53.72	39.43
Top - 5	97.76	84.07	65.62	**98.8**	**91.51**	**76.19**	93.6	66.96	49.55

Experiment 3 – This experiment aimed implement the second and third approach defined to increase the classification performance. The two architectures as shown in Figures 5 and 6

were modeled in this experiment on the original normal image dataset. An early stopping callback was added to the model's callbacks to prevent the overfitting of models while training.

Table 2 presents the results obtained after performing this experiment. For comparison purposes, the results obtained from the base model have also been shown. The results demonstrate the fact that the proposed VGG19 Hierarchical-concat model has the worst classification performance amongst all and has the highest number of trainable parameters. While the proposed DenseNet201 Hierarchical-concat model shows significant improvement in the class prediction accuracy on all levels as compared with the state-of-the-art results. Using such a connectivity pattern to directly pass features from lower levels in the hierarchy to the upper levels has boosted the top-1, 3, and 5 accuracies for the upper-level classes greatly. Thus, a successful attempt at improving the classification performance was made by introducing the connectivity pattern in the DenseNet201 Hierarchical architecture. Also, a steady increase in the training and validation accuracy while model training was observed when the concatenation layers were added to the base DenseNet201 Hierarchical model.

6.2 Discussions

This section discusses the implications and interpretations of the experiments performed. As mentioned in the introduction section of this report, owing to system configurations all the experiments were performed by reducing the original data to half. Thus, firstly the model created by Verma, M. et al. (2020) was implemented using the same parameters and run on the same reduced dataset to have its results. These were compared with results from other experiments. Results from experiment 1 show that the hierarchical classification accuracy degrades when it is trained on the skeleton form images obtained from the OP model. It clearly shows that the first objective for improving classification performance using the OP model was not met. Unlike other studies that yield good flat classification results for a few yoga poses using the OP model, in this study a higher number of poses are considered. These poses had a lot of similarities between each other, and more complex postures as shown in Figure 8 which might lead to such results. Figure 7 shows an example of the similarity between classes at the third level. The first and second images are a "Headstand" yoga pose, while the third and fourth are the "Handstand" and the "Feathered peacock pose" respectively. The results also indicate that with the increase in the number of classes in the levels, the classification accuracy decreases. A probable reason for this again can be such closely similar poses.

Figure 7: Images of Headstand pose, Handstand pose, and Feathered Peacock pose

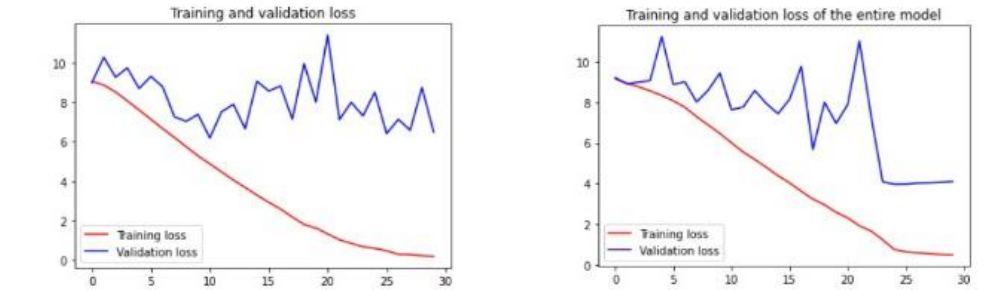
Experiment 2 shows that data augmentation does not improve classification accuracy. The reason for this can be that in this research the images in each class only on the third level were first counted. Then they were augmented such that each class on that level would have an equal number of images. The drawback is that this does not imply that there would be an equal number of images in all the classes of coarse level 1 and coarse level 2. Also, probably

including more augmentation techniques such as zoom, crop, scaling, changing the brightness or contrast of the images, and then modeling them can improve the results.

Building the DenseNet201 Hierarchical-concat by adding concatenation layers significantly improves the test accuracy as compared to the basic branched output model by Verma, M. et al. (2020). The results obtained from this architecture align with the results obtained by Inoue, Forster, and Santos (2020). Seo and Shin (2019) and Zhang et al. (2021) implemented branched models using VGG19 for hierarchical classification in domains other than posture recognition and obtained very good results as mentioned in section 2.2 of this report. However, VGG19 implemented in this study did not show similar kinds of results. This can be because the potential of this network was not explored fully due to system constraints. Only, one dense layer was added to each output branch before the fully connected layer and the dropout layer value was kept higher in this study. More dense layers can be added, and the value of dropout can be reduced which might lead to better results. Even though the parameters of the VGG19 network are more than that of DenseNet201 as shown in Table 2, the latter takes more time to run the models. In the original DenseNet201 (Huang, G. et al., 2017) architecture all following layers receive extra input from the preceding layers, and each layer passes on its feature maps to the next layers. The VGG19 does not have such a kind of passage of feature maps. Another question to look upon would be whether this difference in their original architectures can be attributed to the difference in the results obtained in this study.

Figure 8: Images of a few similar and complex yoga poses

While training the data it was observed that the validation loss did not improve for many epochs. This can be seen from Graph 1. The validation loss reduced but remained constant towards the end for the proposed DenseNet201 Hierarchical-concat as shown in Graph 1b, while as seen in Graph 1a it did not seem to improve for the state-of-the-art model. Hence, a good improvement was seen in the learning curve after adding the connectivity pattern. Probable reasons for the validation loss not reducing after a certain number of epochs would be that the dataset is small to learn from as the model architectures are quite complex. Hence, if all the experiments will be performed with the entire dataset and for more epochs, a better result could be expected for all the experiments. Overall, the aim of this research was achieved by the proposed DenseNet201 Hierarchical-concat model in this study, while the other approaches might still need more work to perform better.



a) Train and Validation Loss graph for state of the art model

b) Train and Validation Loss graph for the proposed DenseNet201 Hierarchical-Concat model

Graph 1: Training and Validation loss curves for the state-of-the-art and proposed models

7 Conclusion

In conclusion, this research aimed to improve the hierarchical yoga pose classification accuracy. Working towards this aim three approaches were defined. Firstly, building a novel deep learning framework comprising of the OpenPose model to create skeleton images as a pre-processing step followed by the DenseNet201 Hierarchical (Verma, M. et al., 2020) as a classifier. The second was building the DenseNet201 Hierarchical-concat architecture by adding a connectivity pattern (concatenation layers) between the output branches. The final one was exploring the potential of a modified VGG19 network concerning accuracy. On performing these experiments, the results demonstrated that the hierarchical classification accuracy improved with the second approach by almost 7%, 11%, and 12% on the coarse level 1, coarse level 2, and fine level respectively. As a result of this connectivity pattern, features were directly passed from the lower branches to the upper branches. While the other two approaches failed to perform well. This is because the dataset consists of a variety of complex postures for which OpenPose couldn't construct optimal skeletons capturing the exact pose. Due to the system constraints, this study performed the experiments on a comparatively smaller dataset and thereby limiting the experimentation scope. Overall, the aim of this research was achieved by the DenseNet201 Hierarchical-concat architecture, while the other approaches might still need more work to perform better.

In the future, studies can focus on improving the performance of the models implemented in this study. Fine-tuning the model parameters and training the model for more epochs by utilizing heavy cloud machines or GPU-loaded machines can lead to improved results. Also, a larger set of data can be used along with the fine-tuned models. An attempt can be made to further modify the VGG19 architecture to explore its potential to the fullest. It would be interesting to explore the performance of other pre-trained networks for such a classification task. Data augmentation can be tried by including more augmentation techniques.

References

Agrawal, Y., Shah, Y. and Sharma, A. (2020) 'Implementation of Machine Learning Technique for Identification of Yoga Poses', in *2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT)*, pp. 40–43. doi: 10.1109/CSNT48778.2020.9115758.

Cao, Z. et al. (2017) 'Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields', in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1302–1310. doi: 10.1109/CVPR.2017.143.

Chen, K. (2019) 'Sitting Posture Recognition Based on OpenPose', in. doi: 10.1088/1757-899x/677/3/032057.

Guo, Yanming et al. (2018) 'CNN-RNN: a large-scale hierarchical image classification framework', *Multimedia Tools and Applications, 77(8)*, pp. 10251–10271. doi: 10.1007/s11042-017-5443-x.

Huang, G. et al. (2017) 'Densely Connected Convolutional Networks', in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269. doi: 10.1109/CVPR.2017.243.

Huang, R. et al. (2020) 'Miss Yoga: A Yoga Assistant Mobile Application Based on Keypoint Detection', *in 2020 Digital Image Computing: Techniques and Applications (DICTA)*, pp. 1–3. doi: 10.1109/DICTA51227.2020.9363384.

Inoue, M., Forster, C. H. and Santos, A. C. dos (2020) 'Semantic Hierarchy-based Convolutional Neural Networks for Image Classification', in *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. doi: 10.1109/IJCNN48605.2020.9207246.

Kang, M. et al. (2018) 'The gesture recognition technology based on IMU sensor for personal active spinning', in *2018 20th International Conference on Advanced Communication Technology (ICACT)*, pp. 1–1. doi: 10.23919/ICACT.2018.8323825.

Kothari, S. (2020) 'Yoga Pose Classification Using Deep Learning', Master's Projects. doi: https://doi.org/10.31979/etd.rkgu-pc9k.

Palanimeera, J. and Ponmozhi, K. (2021) 'Classification of yoga pose using machine learning techniques', *Materials Today: Proceedings, 37*, pp. 2930–2933. doi: 10.1016/j.matpr.2020.08.700.

Park, H.-J., Baek, J.-W. and Kim, J.-H. (2020) 'Imagery based Parametric Classification of Correct and Incorrect Motion for Push-up Counter Using OpenPose', in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, pp. 1389–1394. doi: 10.1109/CASE48305.2020.9216833.

Seo, Y. and Shin, K. (2019) 'Hierarchical convolutional neural networks for fashion image classification', *Expert Systems with Applications*, 116, pp. 328–339. doi: 10.1016/j.eswa.2018.09.022.

Simonyan, K. and Zisserman, A. (2015) 'Very Deep Convolutional Networks for Large-Scale Image Recognition', undefined. Available at: /paper/Very-Deep-Convolutional-Networks-for-LargeScale-Simonyan-Zisserman/eb42cf88027de515750f230b23b1a057dc782108 (Accessed: 12 April 2021).

Verma, M. et al. (2020) 'Yoga-82: A New Dataset for Fine-grained Classification of Human Poses', in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 4472–4479. doi: 10.1109/CVPRW50498.2020.00527.

Yadav, S. et al. (2019) 'Real-time Yoga recognition using deep learning', *Neural Computing and Applications, 31*, p. https://link.springer.com/article/10.1007/s00521-019. doi: 10.1007/s00521-019-04232-7.

Zhang, X. et al. (2021) 'Hierarchical bilinear convolutional neural network for image classification', *IET Computer Vision, 15(3)*, pp. 197–207. doi: https://doi.org/10.1049/cvi2.12023.