Yoga Pose Detection using Custom Convolutional Neural Network and Pre-Trained Models

MSc Research Project
Data Analytics

Maaz Ahmad
Student ID: x21134308

School of Computing
National College of Ireland

Supervisor: Michael Bradford

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Maaz Ahmad
Student ID:	x21134308
Programme:	Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Michael Bradford
Submission Due Date:	25/04/2023
Project Title:	Yoga Pose Detection using Custom Convolutional Neural Network and Pre-Trained Models
Word Count:	7189
Page Count:	22

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:	Maaz Ahmad
Date:	27th May 2023

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):	

Yoga Pose Detection using Custom Convolutional Neural Network and Pre-Trained Models

Maaz Ahmad
x21134308

Abstract

Yoga has been practiced for thousands of years and has become increasingly popular in the present day due to its numerous benefits. In today's fast-paced and stressful world, yoga provides a means to calm the mind and improve physical health, but it can be difficult to identify poses from pictures because of differences in lighting, camera angles, and body types. The aim of this research is to detect yoga poses using a custom Convolutional Neural Network (CNN) model and compare its performance with pre-trained CNN models such as Densenet and Resnet. The dataset used for this study comprises images of individuals performing different yoga poses. Moreover, to enhance the performance of the models, data augmentation techniques such as rotation, zooming, flipping, and shifting was employed. After data augmentation whatever the output is gathered will again proceed with the Custom CNN model to find out the performance of the model after data augmentation. Where the Custom CNN model is build using Keras Tuner with a specific hyper-parameter. The motivation of the research is to develop new technologies that help individuals recognize their yoga poses in real-time and assist trainers in guiding their clients.

1 Introduction

Yoga has become extremely popular in recent years due to its many advantages for enhancing both physical health (Gurrin et al.; 2014) & (Islam et al.; 2017). However, due to differences in lighting, camera angles, and body types, it can be difficult to recognize yoga poses from photographs. The creation of computer vision-based methods for spotting yoga poses can help practitioners recognize their own poses in the moment and help instructors direct their students. In this research paper, a custom Convolutional Neural Network (CNN) model is used to identify yoga poses, and its performance is compared to that of pre-trained CNN models like Densenet and Resnet. The images of people doing various yoga poses made up the dataset used for the study. Data augmentation methods like rotation, zooming, flipping, and shifting will be used to improve the performance of the models. The output will be fed back into the Custom CNN model after data augmentation has been completed in order to assess the model's performance and we intend to use the Keras tuner with particular hyperparameter to find the best model configuration for the custom CNN model. This method's main objective is to identify yoga poses from images as accurately as possible. To do this, we have established several objectives, which are listed in Table 1. We aim to find the best performing model that can outperform pre-trained CNN models like DensNet and Resnet in classifying yoga poses from images by utilizing the Keras Tuner and tuning the hyperparameter.

Motivation and Background

The aim of this research study, it is crucial to create computer vision-based methods for identifying yoga poses and assess how well custom CNN models perform against pre-trained CNN models. Custom layer CNN may improve the performance of the model Ogundokun et al. (2022). Additionally, it will show the potential of data augmentation methods that can improve the effectiveness of the models. (Rajendran and Sethuraman; 2023) this study could have a big impact on the creation of technologies that help people improve their physical health through yoga. Originating in ancient India, yoga is a well-known discipline for the body, mind, and spirit. Due to its many advantages, it has grown in popularity over time all over the world. Yoga's various poses, or asanas, have drawn attention from all over the world. These postures not only help to increase strength, flexibility, and balance, but they also have positive effects on mental health, such as lowering stress levels and elevating mood. Additionally, performing yoga poses can enhance digestion, breathing, and blood circulation, as well as general well-being. Yoga is therefore a valuable tool for promoting health and wellness all over the world because its significance transcends national boundaries and cultural boundaries.

1.1 Research Question and Research Objectives

As people are adopting yoga as an exercise for some physical activity but after post covid-19 a lot of people prefer to perform it through online classes or mobile applications. Where the main problem is to perform the correct pose and online classes make it difficult for Yoga teachers and students to do so.

Q1. *"Can a Convolutional Neural Network (CNN) model with user-defined hyperparameters outperform CNN models that have already been trained, such as DenseNet-121 and Resnet-50, at classifying yoga poses from images?"*

Q2.*"Can the use of data augmentation techniques on the training datasets improve the performance of a deep learning model in a high-end scenario?"*

The development and implementation of the research objectives listed in Table 1 was done in order to respond to the aforementioned research questions.

Table 1: Research Objective

Objective	Description
1	Literature Review on Yoga pose recognition(2010-2023)
2	Data gathering and preprocessing.
2.1	Splitting of data into two parts training and testing.
2.2	Image augmentation for Custom CNN model.
3	Result, Evolution and Implementation of Yoga Pose Recognition
3.1	Result, Evolution and Implementation of ResNet-50(Train and Test Data)
3.1.1	Result and Evolution of ResNet-50 with Train data.
3.1.2	Result and Evolution of ResNet-50 with Test data.
3.2	Result, Evolution and Implementation of DenseNet-121(Train and Test data).
3.2.1	Result and Evolution of DenseNet-121 with Train data.
3.2.2	Result and Evolution of DenseNet-121 with Test data.
3.3	Result, Evolution and Implementation of Custom CNN.
3.3.1	Result and Evolution of Custom CNN without augmentation with Train data.
3.3.2	Result and Evolution of Custom CNN without augmentation with Test data.
3.3.3	Result and Evolution of Custom CNN with augmentation with Train data.
3.3.4	Result and Evolution of Custom CNN with augmentation with Test data.
4.	Comparison of developed model

1.2 Project Contributions

The main contribution of this project is the training and comparison of two different pre-trained deep learning models, ResNet and Densenet. In contrast, building a custom CNN model with Keras Tuner and contrasting each model's performance to determine which of these three performs best. Also, the purpose of this study is to help yoga practitioners—both students and teachers—monitor their poses as they practice yoga. In order to find the right pose for each class, the developed models will recognize and categorize yoga postures.

2 Literature Review on Yoga pose recognition(2010-2023)

2.1 Introduction

Yoga poses classification is a difficult task that has attracted more attention recently because of its potential uses in monitoring health and wellness. To accurately recognize yoga poses, a variety of deep-learning techniques have been suggested. This review of the literature focuses on studies that suggested methods for a yoga pose. This literature review's main objective is to examine ways to improve the accuracy of yoga pose recognition. The following subsections make up this section's organisational structure 2.2 Investigation of model 2.3 Critique of Deep Learning in a yoga pose recognition 2.4 Comparing Pre-trained Models in Yoga Pose Recognition 2.5 Evaluation of Data Augmentation Techniques 2.6 Conclusion.

2.2 Investigation of Model

In the 2010s deep learning techniques where begun to explore by Researchers with the use of convolutional neural networks (CNNs) and other machine learning algorithms for pose estimation and recognition. Early research in this field concentrated on recognizing yoga poses using pre-trained CNN models like Densnet and ResNet, with accuracy levels of about 80–90% on small datasets. For pose recognition using CNN model can go up to 98% in accuracy as mentioned in (Tian et al.; 2022). KUTÁLEK (n.d.) used 22 different yoga poses and applied the different CNN models to the dataset which includes LeNet-5, Densnet, VGG-16, and ResNet these are some pre-trained CNN models. Which achieves an accuracy of 91% on yoga pose recognition for the hard data set and 95% for the easy dataset. In order to perform similar approach for yoga pose recognition Nguyen et al. (2022) used an RGB image using the BlazePose model. To pay more attention to the key points for each kind of yoga pose, They investigate different CNN models and choose VGG-16, VGG-19, and ReNet-101 which one can obtain the best trade-off between accuracy and parameter number to recognize yoga poses. Where they obtain 94.94% of accuracy for the VGG-16 model, 95.70% for VGG-19, and 94.54 using ResNet-101. A similar approach of custom layered CNN model which was made by three different layers like a custom CNN model to recognize the poses in which they took pose detection as a classification problem and try to detect results using CNN Based detection model. Where they achieve an accuracy of 96% for pose detection using CNN concluded in (Haochen et al.; 2017).

2.3 Critique of Deep Learning in a Yoga Pose Recognition

Jose and Shailesh (2021) suggested an approach for yoga pose recognition with the help of deep learning techniques like convolutional neural networks (CNN) and transfer learning. Where they took 10 different yoga pose asanas for predicting the accuracy. In a study using a dataset of 5 yoga poses, Chaudhari et al. (2021) proposed a deep learning model which used CNN for yoga poses which also concluded that CNN generates the best result for image recognition. Where if we talk about the pre-trained convolutional neural networks model, (Long et al.; 2022) also talk about some same idea about using the Pre-trained transfer learning CNN model for Yoga Posture recognition where they used 14 different yoga postures and applied TL-VGG16-DA, TL-VGG19-DA, TL-MobileNet-DA, TL MobileNetV2-DA, TL-InceptionV3-DA, and TL-DenseNet201-DA. Where they achieve an accuracy for TL-VGG16-DA, TLVGG19- DA, TL-MobileNetV2-DA, TL-IceptionV3-DA, and TL-DenseNet201-DA performances were 94.90%, 94.90%, 92.16%, 91.76%, and 98.04%, respectively. One more similar deep learning approach was made in (Shorten and Khoshgoftaar; 2019). Where use of an image data set on which CNN model was applied and data augmentation techniques. Where Shorten and Khoshgoftaar (2019) concluded that the future of data augmentation is very bright. Where data augmentation prevents over-fitting of the modifying limited dataset to process the characteristics of data. In (Pham et al.; 2018) also two primary contributions to a similar technique where they

discussed using Deep CNN Techniques for Image Classification. Secondly, they applied data augmentation to the dataset. Where they achieve an accuracy of 89%. They also mention that by using augmentation they achieve better results than the normal traditional method.
When deep learning techniques are taken into consideration, convolutional neural networks (CNNs) have demonstrated remarkable performance in image classification. The Conclusion of the study demonstrates the potential of CNN for yoga pose recognition. Kumar and Sinha (2020) looked into various ways through which yoga pose can be recognized like Repetitive Neural Network (RNN), Multilayer Perceptron (MLP), Long Short-Term Memory (LSTM) And Convolutional Neural Network (CNN) and then compare it with each other. After further studies some other paper also talks about yoga pose recognition. Where, they used an image dataset to identify yoga poses using various machine learning techniques as mention in (Agrawal et al.; 2020) they used 5500 images of the human skeleton and with the help of tf-pose which was used to applied various machine learning classifications, to explore the effectiveness of SVM CNN and RNN models in pose estimation, Kothari (2020) conducted experiments using a dataset of images. The dataset was processed and pre-processed to prepare, trained and evaluated on the dataset using various metrics such as accuracy and mean average precision. The results of the experiments showed that CNNs, SVMs, and RNNs can all be effective for pose estimation (Yadav et al.; 2019) all these paper shows yoga pose recognition. (Tian et al.; 2022) also mentions the use of a deep learning approach where they use a Convolutional Neural Network (CNN) layer to extract feature key points for the model.

2.4 Comparing Pre-trained Models in Yoga Pose Recognition

Where if we talk about the pre-trained convolutional neural networks model, (Long et al.; 2022) also mention some same idea about using the pre-trained transfer learning CNN model for Yoga Posture recognition. Where, they used 14 different yoga postures and applied TL-VGG16-DA, TL-VGG19-DA, TL-MobileNet-DA, TL MobileNetV2-DA, TL-InceptionV3-DA, and TL-DenseNet201-DA. Where they achieve an accuracy for TL-VGG16-DA, TLVGG19- DA, TL-MobileNetV2-DA, TL-IceptionV3-DA, and TL-DenseNet201-DA performances were 94.90%, 94.90%, 92.16%, 91.76%, and 98.04%, respectively. During this training the dataset is made by them. Where they used 14 different yoga poses and 1,120 Images. In Table 2 is all pre-trained model which was used in the (Long et al.; 2022).

Table 2: Comparing Pre-trained Models in Yoga Pose Recognition

Model	Percentage %
VGG 16	94.90
VGG 19	94.90
MobileNet	92.16
MobileNet V2	91.79
Denesnet 201	98.04

2.5 Evaluation of Data Augmentation Techniques

Data augmentation is a well-known technique in deep ML; it entails modifying existing images in a training data-set in order to improve the overall number of images in the dataset (Carranza-García et al.; 2019). The use of data augmentation in (Yu et al.; 2017) improved CNNs' performance in the classification of remote sensing scenes. The completeness and diversity of data can be significantly increased by data augmentation. The experimental outcomes of the model that is applied with the help of data augmentation to it when it was used to train deep learning models outperformed the model trained on the data-set with the model architecture without augmentation. Stivaktakis et al. (2019) randomly translated, rotated, and translated remote sensing images. These techniques were chosen to the information of the yoga images, which are both essential for a valid classification result.

2.6 Conclusion of Literature Review

According to the reviewed literature, CNN models are the best model to produce an output. However, a Custom Layered CNN may provide encouraging results for enhancing yoga pose recognition precision. According to these studies, CNN models built with various layers may be able to recognize yoga poses with higher accuracy rates than models that have already been trained. However, more investigation is required to fully examine the potential of customized CNN models for yoga pose recognition and to assess the utility of other machine-learning approaches. Therefore, custom CNN models have a great deal of potential to advance the science of yoga pose recognition, and they merit further research to ascertain how well they perform in this area.

3 Methodology Approach and Project Design

3.1 Introduction

This chapter will go over the scientific approach and structural layout used to create the Yoga Pose Recognition project. We will concentrate on the two-tier architecture used in this project as well as the altered scientific methodology in particular. The methodology and design of the project are thoroughly examined in this chapter.

3.2 Process Flow of Yoga Poses Recognition

The study used a methodology that started with data collection and progressed through pre-processing and augmentation of the data. For model training, two different strategies were used, one with data augmentation and the other without data augmentation. Three different CNN architectures Densnet, Resnet, and a customized CNN model with either 3 or 7 layers were applied to the two methods. The results of each model were then used to create the visualization. InFigure 1 depicted the study's various phases, including the gathering of data, pre-processing, training of the model, and creation of the visualizations.

3.3 Design Specification

In order to classify this research project's utilization of deep learning models to recognize yoga poses, a two-tier architecture was used. Figure 2 displays the primary implementation element of the project. The architecture's top tier is the client layer. The visualizations and outcomes of the classification models from this layer will be delivered

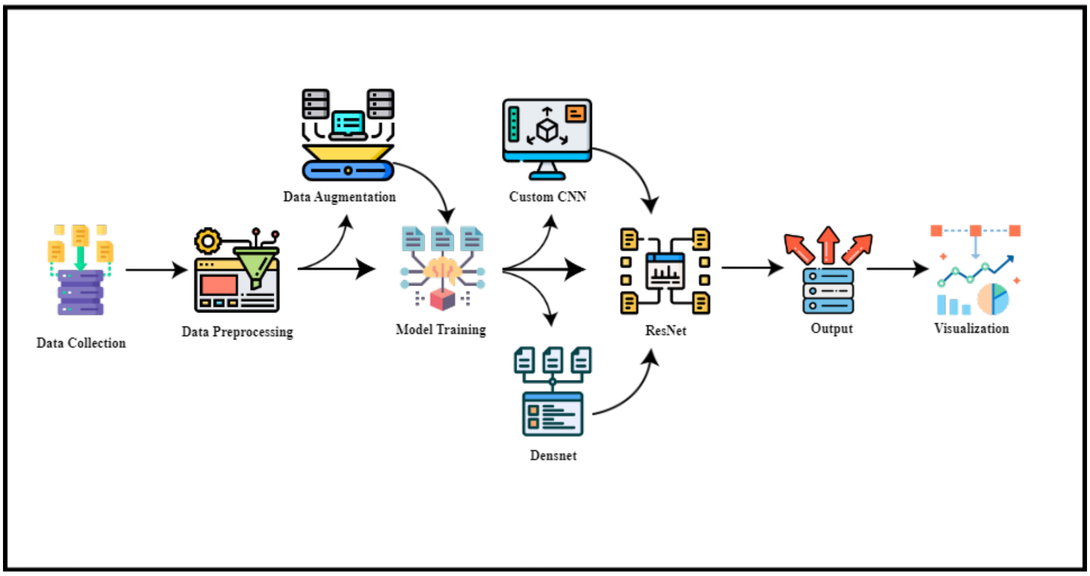

Figure 1: Process Flow of Yoga Poses Recognition

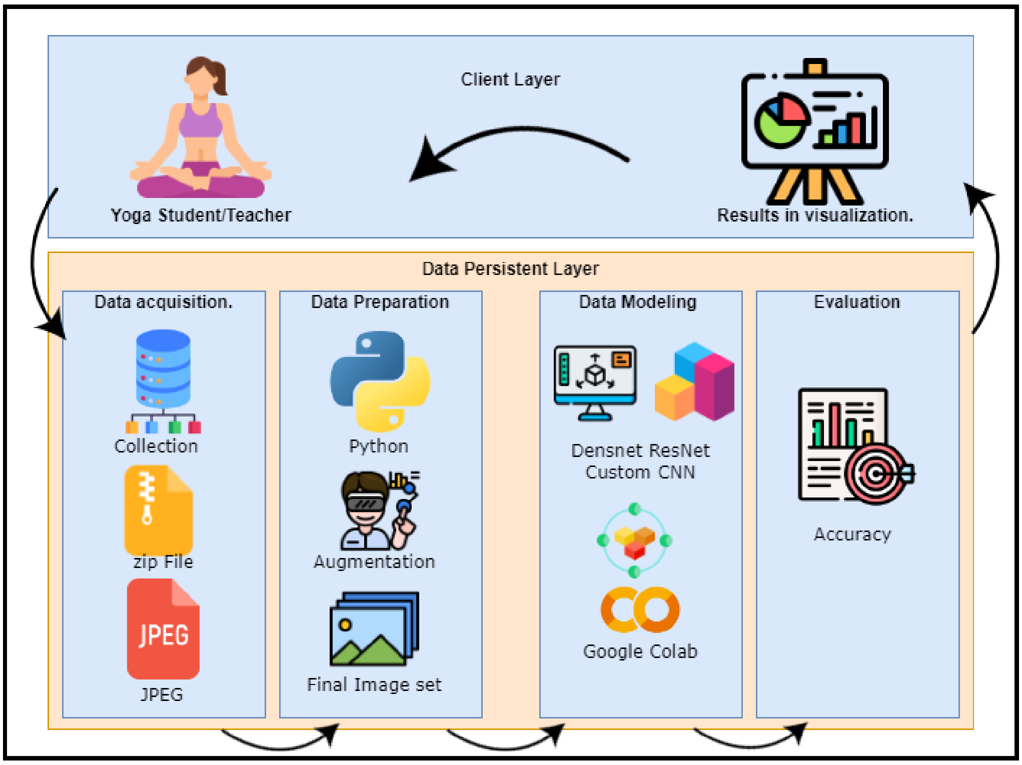

Figure 2: Design Specification

to the yoga student and teacher. The primary implementation component for preparation, data collection, and transformation, as well as for training classification models, was the Data Persistent Layer, which made up the second tier. The data-sets were obtained from Kaggle[1] and enhanced and formatted to make sure they were suitable for the model training. Following the training process, the models' accuracy were compared for evaluation.

3.4 Conclusion

In the implementation of the research project is based on a scientific methodology, and the data is rigorously tested and trained to meet the project's specifications. This approach has been modified to produce a process flow diagram that serves as a manual for designing a two-tier architecture that illustrates how the project will be carried out. The architecture's two tiers work together to provide a comprehensive remedy for Yoga Pose Recognition.

4 Implementation, Evaluation and Results for Yoga Pose Recognition

4.1 Introduction

This section covered the classification model for yoga pose recognition. We will be using data from Kaggle for this project. Python programming is used on Google Colab during implementation. When accessing the models, accuracy is used as a metric. The models used in this implementation will all be compared here, and the model with the best accuracy will be determined.

4.2 Creation and Pre-processing of Data

For this project, we used a yoga pose recognition dataset from Kaggle[2], which consists of 4,407 images in total and 107 different yoga poses. To guarantee accuracy and consistency in the pose labels, this dataset was meticulously curated and labeled. Different people performed the poses in the photos, which were taken from different perspectives and angles. We had plenty of data to train and test our deep learning models for yoga pose recognition because of the size and diversity of the dataset we used. To ensure that the data was prepared for training. However, preprocessing and cleaning the data were essential steps, which we will cover in more detail in this chapter. In the below Figure 3 you can visually understand the data and number of images.

After the images have been gathered, check to see if there are a variety of poses with various degrees of difficulty, as well as different lighting and background settings. To make sure that our deep learning models are strong and can correctly recognize the yoga poses under various circumstances, this variety in the images was necessary. This project's collection of images is crucial because it serves as the foundation for our training data and determines the efficacy and performance of our models.Figure 4&5 below show some of the image samples. And then the data is divided into two parts one for training

[1]https://www.kaggle.com/datasets/shrutisaxena/yoga-pose-image-classification-dataset
[2]https://www.kaggle.com/datasets/shrutisaxena/yoga-pose-image-classification-dataset

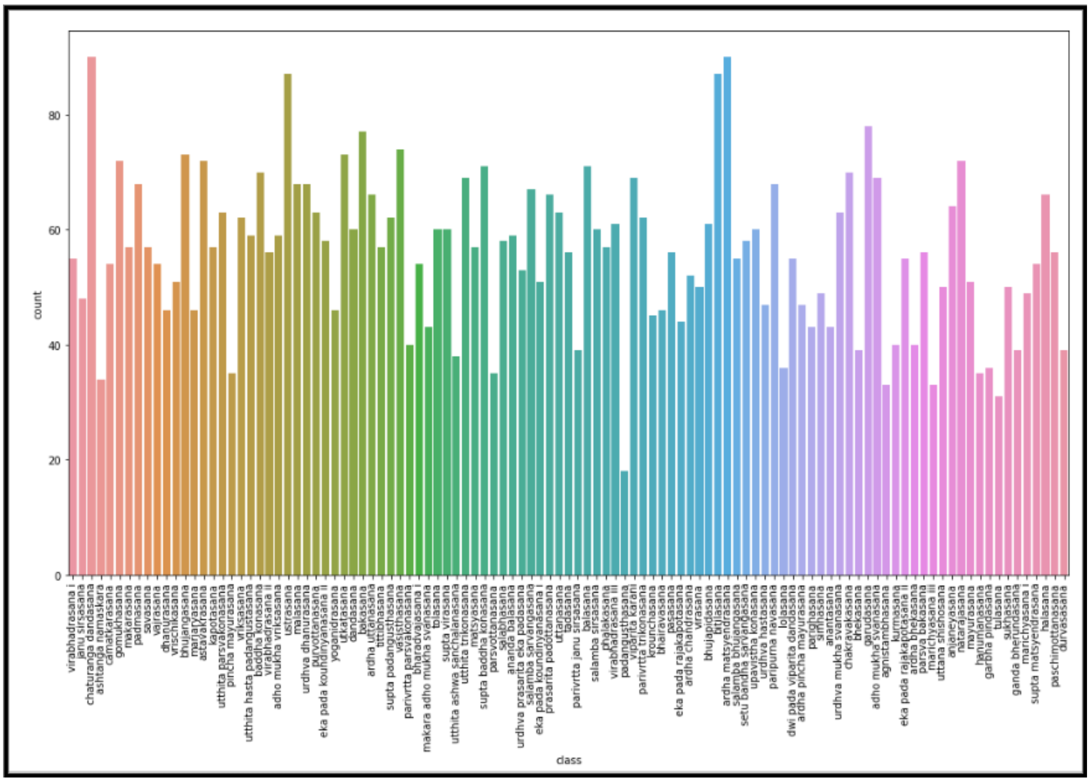

Figure 3: Data Presentation of 107 Yoga Poses

and one for testing which was in a ratio of 80:20. Where from main dataset 20% of the data was kept separately to test the mdoels.

4.3 Results, Evaluation and Implementation of ResNet-50

Deep neural network architecture known as ResNet, or "Residual Network," was first described by He et al. (2016). In their paper "Deep Residual Learning for Image Recognition" in 2015. In order to solve the issue of vanishing gradients that can arise in deep neural networks, particularly in very deep networks with many layers, ResNet was created. ResNet uses residual connections, also known as "skip connections," to allow data to be passed from one layer to another without being altered. The network is able to learn deeper and more complex features thanks to these residual connections, which also help to solve the vanishing gradient problem. With state-of-the-art performance on numerous benchmark datasets, ResNet has emerged as a popular option for many image recognition tasks.

$$Accuracy = \frac{TruePositive + TrueNegative}{TruePositive + FalsePositive + TrueNegative + FalseNegative} \tag{1}$$

Equation 1 displays the formula used to determine a model's accuracy. With its design, ResNet avoids the vanishing gradient issue that can arise in very deep networks by allowing information to flow directly from one layer to the next without being altered. This is accomplished by ResNet use of residual connections, also referred to as "skip connections," which combine the output of one or more earlier layers with the output of the current layer. Instead of attempting to learn the entire input-to-output mapping all at

Figure 4: Sample Image of Yoga pose of Janu Sirsasan

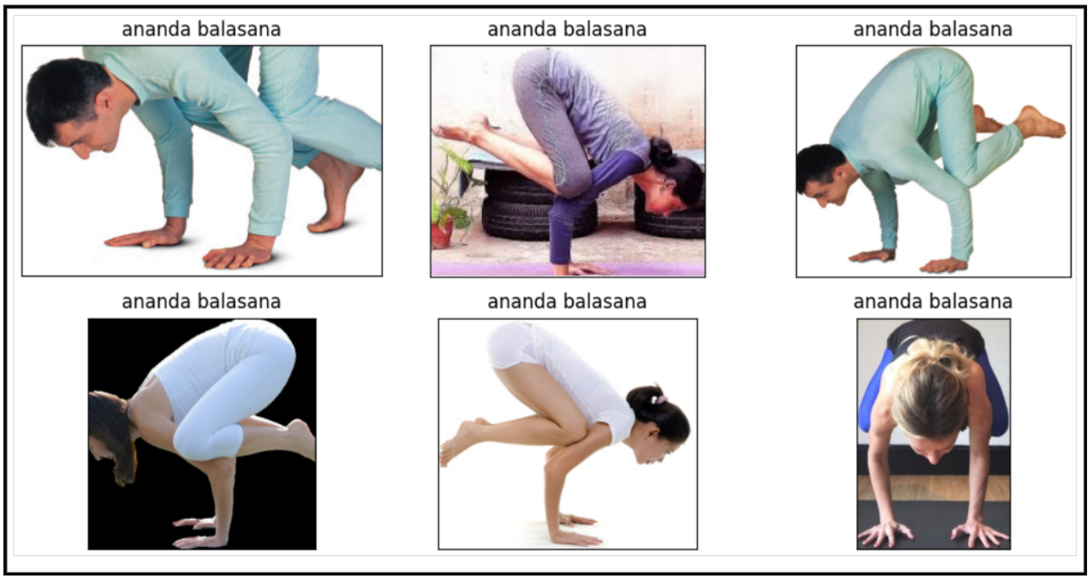

Figure 5: Sample Image of Yoga pose of Bakasana

once, the network can learn to make small adjustments to the output of earlier layers. ResNet is able to train very deep networks—up to hundreds of layers deep—while maintaining high accuracy and performance on image recognition tasks. This is accomplished by utilizing residual connections. Whereas, during this training the model was trained. The model is assessed based on its accuracy during both training and testing. The results and evaluation for training and testing are shown below.

Figure 6: Training and Validation Accuracy of ResNet-50 Over 75 Epochs

After training for 75 epochs the model archives the highest validation accuracy of 12 %. which is shown in Figure 6 .The graph displays a ResNet model's training and validation accuracy over the course of training. The blue line represents the training accuracy, and the orange line represents the validation accuracy. The training accuracy for both lines starts out at 1% and the validation accuracy at 2%, respectively. The model gets better at identifying the yoga poses in the training set as training goes on, and training accuracy steadily rises. As the model learns to generalize to new images that it has never seen before, the validation accuracy also rises at the same time, albeit more slowly.

The training accuracy peaks at 4% around the middle of the graph, while the validation accuracy hovers around 10%. This shows that the model is beginning to become too specialized in identifying the poses in the training set and is not generalizing well to new poses. Over-fitting to the training data is the process of a model becoming too similar to the training data. The growing distance between the training and validation accuracy lines illustrates this. While its reach the end of the training which is around 75 epochs the validation reached 12% at the end. After completion of the training of ResNet-51 model was then saved to test on the 20% of the data to see the performance of the trained model in which each image was different from the trained images. the accuracy which was achieved after the test was 4%.

4.4 Results, Evaluation and Implementation of DenesNet-121

DenseNet, or "Densely Connected Convolutional Networks," was presented by Huang et al. (2017) in their paper "Densely Connected Convolutional Networks" in 2017. Although it takes a different route, DenseNet addresses the issue of vanishing gradients in extremely deep networks, similar to ResNet. The main concept of DenseNet is to establish dense connections between the layers by connecting each layer to every other layer in a feed-forward manner. This implies that each layer's output is passed on to all layers after it as well as the one above it. In other words, each layer receives the feature maps from the layers that came before it as input, which is why the network is referred to as "dense." This strategy has a number of advantages, including lowering the number of parameters required to achieve a particular level of performance and enhancing gradient flow across the network. On a number of benchmark datasets, DenseNet has demonstrated state-of-the-art performance and has been applied to numerous image and video recognition tasks.

The manner in which Denesent was transmitted to all blocks' following layers. As a result, a highly connected structure is created, which can enhance feature reuse and gradient flow because each layer receives a rich feature map from all lower layers. Transition layers exist between each dense block, reducing the spatial dimensions of the feature maps while maintaining the number of feature maps. By reducing the number of parameters in the network and compressing the representation, information can still flow through the network. similar to ResNet-50 we trained the 80% of the data for 75 epochs where the results and evaluation for training was captured which can be seen below.

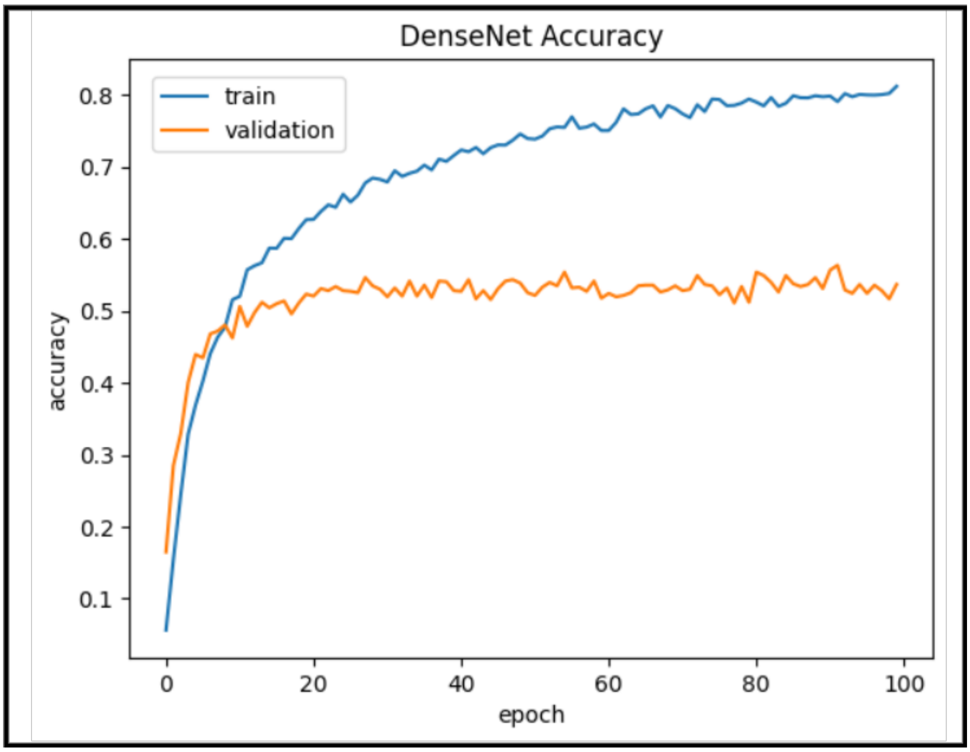

Figure 7: Training and Validation Accuracy of DenesNet-121 Over 75 Epochs

The training of the DenseNet-121 model's accuracy and validation accuracy trends are represented graphically in Figure 7. The number of training epochs is plotted on the x-axis, and the accuracy and validation accuracy are plotted on the y-axis. The training accuracy is represented by the blue line on the graph and indicates how well the model fits the training data over the course of training. The training accuracy gradually increases as the number of epochs rises, peaking at about 80% near the end of training. Whereas the validation accuracy, represented by the orange line on the graph, demonstrates how well the model generalizes to new data that it hasn't encountered during training. The validation accuracy initially has a low starting point of 18%, but as the model learns to generalize more effectively, it increases and reaches a peak of about 56% near the end of training. The model was successfully trained after training. 20% of the remaining data were used to test the model. The accuracy of 54% obtained from the test model evaluation indicated the model's ability to reasonably predict the classification of the correct yoga pose.

4.5 Results, Evaluation and Implementation of Custom CNN

Deep learning neural networks of the type known as convolutional networks are frequently used for image and video recognition tasks. It is intended to automatically learn and extract significant features from input images, like edges, textures, and shapes, and is inspired by the human visual cortex. This is accomplished by applying convolutional and pooling layers to the input data in order to perform convolutions and subsampling operations. CNN models can perform at the cutting edge on a variety of computer vision tasks, including object detection, facial recognition, and image segmentation. They are typically trained on large datasets using back-propagation algorithms. They have also been successfully used in other fields, like speech recognition and natural language processing. In this research paper we will discuss about Custom CNN model as opposed to using pre-trained models like ResNet or DenseNet, a custom CNN is created and trained by the user for a particular task. Convolutional, pooling, and fully connected layers, as well as their hyperparameters, such as the number of filters, kernel size, activation functions, and dropout rates, are designed and configured in custom CNN models. Custom CNN models provide more flexibility and control over the network's architecture and settings, enabling better personalization and fine-tuning for particular tasks. To design and train them, they need more knowledge and resources, and they might not always perform as well as pre-trained models.

However, Custom CNN model can me implemented using Keras Tuner also. An open-source library called Keras Tuner is used to fine-tune the hyperparameters of deep learning models created with the Keras API. To maximize the performance of a deep learning model, hyperparameter tuning involves determining the ideal combination of hyperparameters, such as learning rate, number of layers, and batch size. With the help of different search algorithms like Random Search, Hyperband, and Bayesian Optimization, Keras Tuner offers an easy and flexible way to define and explore the hyperparameter space. Deep learning experts can shorten training times and improve model performance by using Keras Tuner to expedite the process of identifying the ideal hyperparameters. While implementing the Keras Tuner the lower hyperparameters of 3 and higher hyperparameters of 7 the model was built. The results and evaluation for training and testing of the Custom CNN model are shown below in 4.5.1,4.5.2 and 4.5.3.

For the classification matrix, multi-class averaging will be used because in this data set, there are 107 classes. For both the binary and the multi-class case, classification metrics are implemented in Yardstick where both the truth and estimate columns are factors. Some metrics have their own unique multi-class implementations, and the multi-class implementations use micro, macro, and macro_weighted averaging when appropriate.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Equation 2 shows how multiclass predictions are broken down into several sets of binary predictions using macro averaging, which then calculates the corresponding metric for each of the binary cases and averages the outcomes. Think about precision in the binary case as an illustration.

$$Pr = \frac{Pr1 + Pr2 + ... + Prk}{k} = Pr1\frac{1}{k} + Pr2\frac{1}{k} + ... + Prk\frac{1}{k} \tag{3}$$

Equation 3 says if levels A, B, C, and D were present in the multiclass case, macro averaging reduces the issue to numerous one-vs-all comparisons. Precision is calculated using the recoded truth and estimate columns, with A serving as the "relevant" column, with the only two levels being A and other. To obtain a total of 4 precision values, this procedure is repeated for the remaining 3 levels. After that, the results are averaged. Where ***RR1*** is the precision calculated from recoding the multiclass predictions down to just class 1 and other mentioned in (Vaughan; 2023). In the page it also mention that Keep in mind that when contributing their respective portions of the precision value to the total (in this case, 1/4), each class receives equal weight in macro averaging. When there is a significant class imbalance, this calculation might not be accurate. In that situation, a weighted macro average that bases its weights on the frequency of that class in the truth column might be more appropriate represented in Equation 4.

$$Prweightedmacro = Pr1\frac{\#Obs1}{N} + Pr2\frac{\#Obs2}{N} + ... + Prk\frac{\#Obsk}{N} \tag{4}$$

Micro averaging computes 1 metric as opposed to k metrics that are averaged collectively, treating the entire set of data as an aggregate result. To achieve accuracy, the true positive results for each class are totaled and used as the numerator. The true positive and false positive results for each class are totaled and used as the denominator can be represented in the equation 5.

$$Prmicro = \frac{Tp1 + Tp2 + ... + Tpk}{(Tp1 + Tp2 + ... + Tpk) + (Fp1 + Fp2 + ...Fpk)} \tag{5}$$

4.5.1 Implementation and Evaluation of Keras Tuner For Custom CNN

Using Keras Tuner, a custom CNN model was created and optimized for the task. For important parameters like learning rate, number of layers, and batch size, the hyperparameter search space was defined to include values between 3 and 7. 80% of the dataset was used to train the model over 20 epochs, and the model with the highest accuracy was chosen as the best fit. The model that resulted demonstrated its potential for correctly identifying yoga poses from images by achieving an accuracy of 43%. After finding the best-fit model the dataset was again trained with two conditions one with augmentation and one without augmentation which are concluded in 4.5.2 and 4.5.3.

4.5.2 Evaluation and Results of Custom CNN without Augmentation

The best-fit CNN model was similarly trained on an 80-20 ratio of the reclassified image. After the model was run for 75 epochs a validation accuracy of 43% and 97% of accuracy was achieved, which is relatively low and may be caused by the more number of classes. On the other hand, the loss was around 0.03 and the validation loss was around 3.35.

Figure 8 and 9 display the graph for the loss and validation loss over epochs, as well as the training and validation accuracies over 75 epochs. Whereas, when the 20% of the test data was run on the trained model an accuracy of 41% was achieved and the loss was 3.53. And for classification matrix macro and micro averaging result is shown in Table 3 In the following section, the model is once again trained using augmented images.

Table 3: Macro and Micro averaging results for CNN without augmentation

	precision	recall	f1-score	support
accuracy			0.0114	1049
macro avg	0.0099	0.0092	0.0094	1049
weighted avg	0.0124	0.0114	0.0118	1049

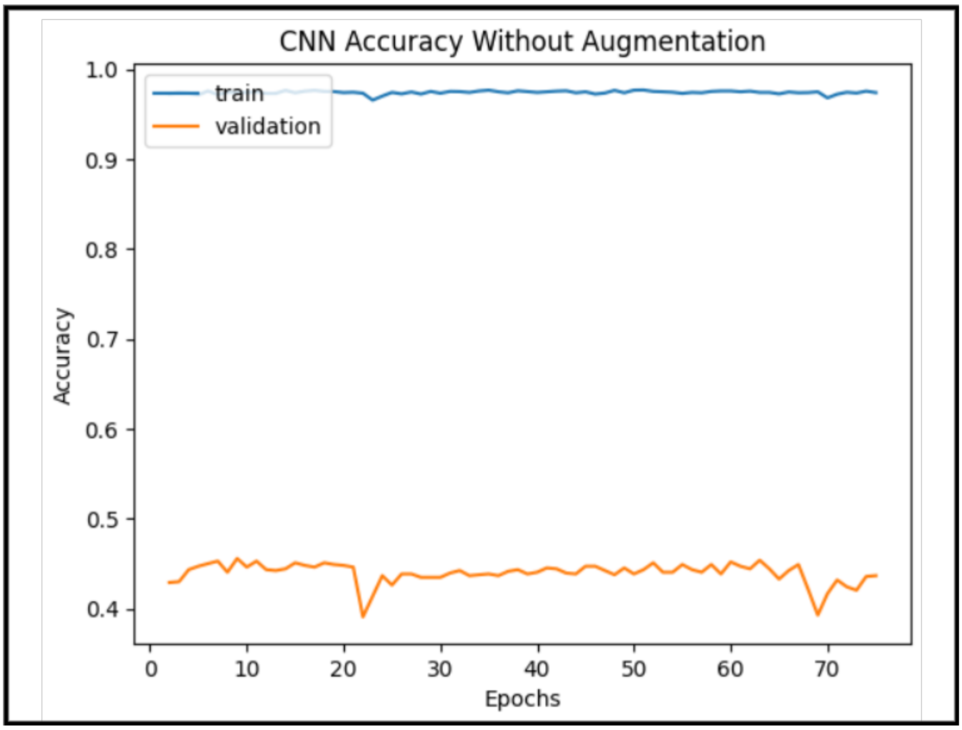

Figure 8: Training and Validation Accuracy of Custom CNN without Augmentation over 75 Epochs

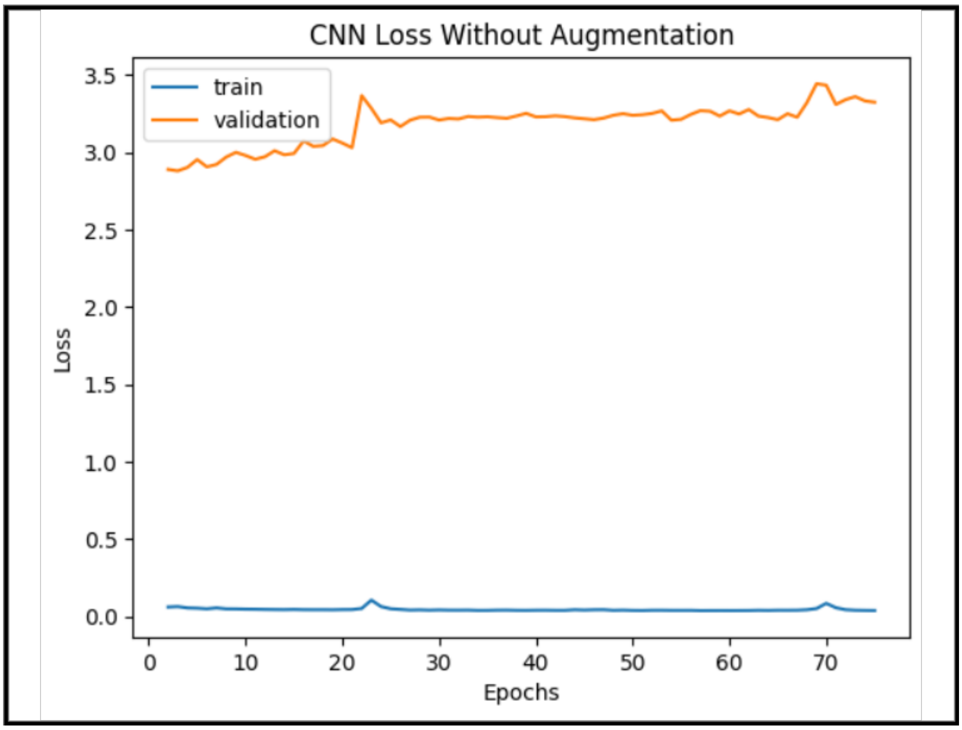

Figure 9: Training and Validation Loss of Custom CNN without Augmentation over 75 Epochs

4.5.3 Results and Evaluation of Custom CNN with Augmentation

Similarly to Custom CNN without augmentation, this model was also trained on an 80:20 split ratio where the model was trained for 75 epochs where it achieve a validation accuracy of 43% and an accuracy of 97%. Whereas in this model also validation loss is 3.51 and the loss is 0.03. When it comes to the test data set the accuracy was 41% but the loss was 3.37 with data augmentation as shown in Figure 10 and 11. And the results of the macro- and micro-averaging for the classification matrix are shown in Table 4.

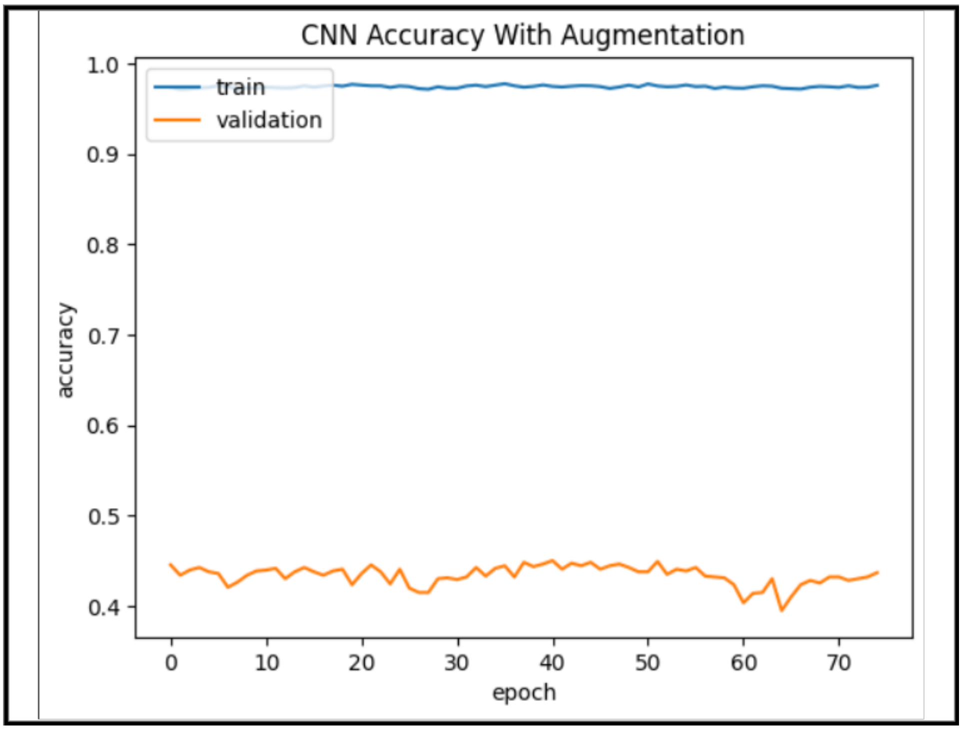

Figure 10: Training and Validation Accuracy of Custom CNN with Augmentation over 75 Epochs

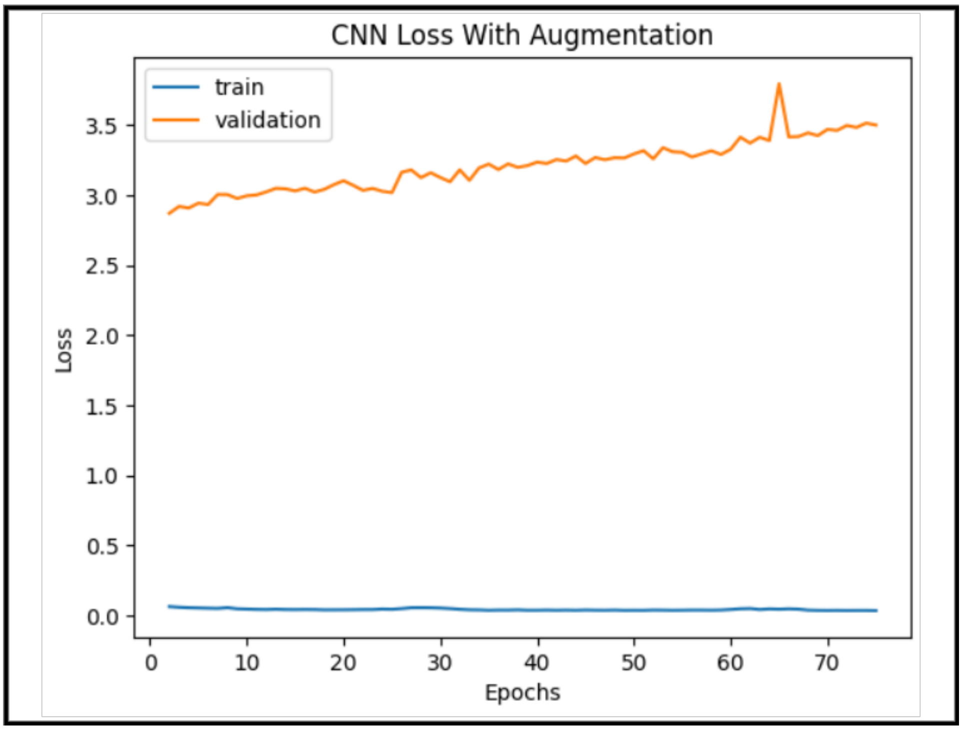

Figure 11: Training and Validation Loss of Custom CNN with Augmentation over 75 Epochs

Table 4: Macro and Micro averaging results for CNN with augmentation

	precision	recall	f1-score	support
accuracy			0.0105	1049
macro avg	0.0088	0.0089	0.0085	1049
weighted avg	0.0103	0.0105	0.0100	1049

4.6 Discussion and Comparison of Developed Models

4.6.1 Comparison of ResNet-50, DenesNet-121, Custom CNN without Augmentation and Custom CNN with Augmentation Models.

Once four different models had successfully undergone training and evaluation for identifying yoga poses. The accuracy of the ResNet-50 model was 69%, and its validation accuracy was 12%. With an accuracy of 81% and a validation accuracy of 56%, the DenseNet-121 model performed better in comparison. The two unique CNN models, one without augmentation and one with augmentation, were also created. Although the validation accuracy of the custom CNN without augmentation was only 43%, it still achieved an impressive accuracy of 97%. Although the validation accuracy was the same at 43%, the custom CNN with augmentation also achieved a high accuracy of 97% which can also be seen in Table 5.

Table 5: Comparison of training accuracy of ResNet-50, DenesNet-121, Custom CNN without Augmentation and Custom CNN with Augmentation.

	Accuracy(Percentge)	Validation(Percentage)
ResNet-50	69%	12%
DenesNet -121	81%	56%
Custom CNN without Augmentation	97%	43%
Custom CNN with Augmentation	97%	43%

Table 6 show all the model with their test accuracy result where ResNet-50 is the lowest with 4% of accuracy and DenseNet-121 is the highest with 54% of accuracy. Whereas, both the Custom CNN model with and without augmentation is at same accuracy of 41%.

Table 6: Test Accuracy of ResNet-50, DenseNet-121, Custom CNN without Augmentation and Custom CNN with Augmentation

Model	Accuracy(%)
ResNet-50	4
DenseNet-121	54
Custom CNN without Augmentation	41
Custom CNN with Augmentation	41

It is noteworthy that the DenseNet-121 model performed significantly better than the ResNet-50 model. This is probably because the DenseNet architecture enables better data flow and feature reuse, which enhances model performance. The accuracy of the two custom CNN models, both with and without augmentation, was comparable, but neither model's validation accuracy was as high as the DenseNet-121 model. This suggests that augmentation might not have a big effect on the model's accuracy but might make it better at generalizing to new data which will be further disscused in 4.6.2.

4.6.2 Comparison of Custom CNN without Augmentation and Custom CNN with Augmentation Models.

On the training set, both Custom CNN models achieved a remarkable accuracy of 97%. The Custom CNN with Augmentation performed slightly better than the Custom CNN without Augmentation in terms of validation accuracy, which was a significant difference. The validation accuracy line of the Custom CNN with Augmentation consistently outperforms that of the Custom CNN without Augmentation, especially during the 60th epoch, illustrating the performance difference further in the plotted graph which can be seen Figure 12. The fact that both models ultimately had the same level of validation accuracy suggests that augmentation may not be necessary in this particular situation. Nevertheless, the performance disparity suggests that augmentation might help the model perform better outside of the training set.

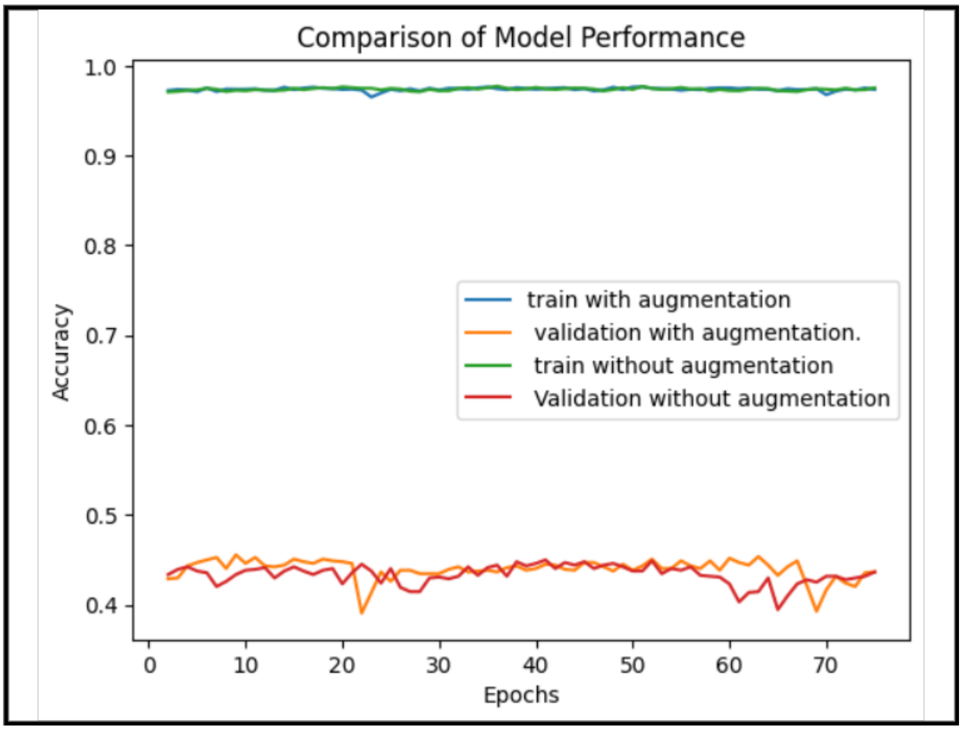

Figure 12: Comparison of CNN With Augmentation and Without Augmentation Over 75 Epochs

5 Conclusion and Future Work

After conducting a thorough literature review, this project formulates a methodology for Yoga Pose Recognition. The deep learning models DenseNet-121 and ResNet50 were chosen for additional research after a review of the literature. This study sought to determine how well deep learning models (DenseNet-121 & ResNet50) categorize images of yoga poses and whether image augmentation enhances model performance when trained on a dataset with 107 classes. Two sets of training datasets were created to find out. The model was trained on one set, and it was tested on the other set. We can draw the conclusion that the Denesnet-121 model performed the best among the other models based on the comparison and discussion of the various models, achieving an accuracy of 81% and a validation accuracy of 56%. Although the ResNet-50 model had a 69% accuracy rate, its validation accuracy was only 14%. The accuracy of the two customized CNN models, without and with augmentation, was the same at 97%, but the validation accuracy was considerably lower at 43%. It should be noted that during the training phase, the augmented data performed marginally better than the non-augmented data, but both models eventually attained the same accuracy.

In Conclusion, the Denesnet-121 model is the best model for the current image recognition task. But because of their high accuracy, custom CNN models—with or without augmentation—can also be taken into account. The ResNet-50 model's poor validation accuracy may make it less useful for this task. Overall, the model selection is influenced by the particular demands and limitations of the task at hand.

Future Work: In future employment opportunities come in a variety of forms. Whoever it may be to investigate the use of various augmentation methods to enhance the performance of the unique CNN models. The pre-trained ResNet and DenseNet models should also be examined to see if transfer learning can be used to modify them for the particular task of yoga image recognition. Another potential area for improvement is the dataset itself, where the size and variety of the images included could be increased to increase the models' robustness. The layer of hyperparameter can be increased in the future by the researcher. To improve overall accuracy and robustness, it may be worthwhile to investigate the use of ensembling techniques, which combine multiple models.

Acknowledgments

I want to sincerely thank Professor Michael Bradford for his constant and thorough direction throughout the semester and for his patience in always giving appropriate and thorough feedback. I also want to thank National College of Ireland's library for helping me with this technical report's writing and the research project's critical literature review, which I conducted using peer-reviewed articles from dependable sources. Without the encouragement, blessings, and trust of my parents, I would not have been able to complete this research.

References

Agrawal, Y., Shah, Y. and Sharma, A. (2020). Implementation of machine learning technique for identification of yoga poses, *2020 IEEE 9th international conference on communication systems and network technologies (CSNT)*, IEEE, pp. 40–43.

Carranza-García, M., García-Gutiérrez, J. and Riquelme, J. C. (2019). A framework for evaluating land use and land cover classification using convolutional neural networks, *Remote Sensing* **11**(3): 274.

Chaudhari, A., Dalvi, O., Ramade, O. and Ambawade, D. (2021). Yog-guru: Real-time yoga pose correction system using deep learning methods, *2021 International Conference on Communication information and Computing Technology (ICCICT)*, IEEE, pp. 1–6.

Gurrin, C., Hopfgartner, F., Hurst, W., Johansen, H., Lee, H. and O'Connor, N. (2014). *MultiMedia Modeling: 20th Anniversary International Conference, MMM 2014, Dublin, Ireland, January 6-10, 2014, Proceedings, Part I*, Vol. 8325, Springer.

Haochen, L., Bin, Z., Xiaoyong, S. and Yongting, Z. (2017). Cnn-based model for pose detection of industrial pcb, *2017 10th International Conference on Intelligent Computation Technology and Automation (ICICTA)*, IEEE, pp. 390–393.

He, K., Zhang, X., Ren, S. and Sun, J. (2016). Deep residual learning for image recognition, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Huang, G., Liu, Z., Van Der Maaten, L. and Weinberger, K. Q. (2017). Densely connected convolutional networks, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708.

Islam, M. U., Mahmud, H., Ashraf, F. B., Hossain, I. and Hasan, M. K. (2017). Yoga posture recognition by detecting human joint points in real time using microsoft kinect, *2017 IEEE Region 10 humanitarian technology conference (R10-HTC)*, IEEE, pp. 668–673.

Jose, J. and Shailesh, S. (2021). Yoga asana identification: A deep learning approach, *IOP Conference Series: Materials Science and Engineering*, Vol. 1110, IOP Publishing, p. 012002.

Kothari, S. (2020). Yoga pose classification using deep learning.

Kumar, D. and Sinha, A. (2020). *Yoga pose detection and classification using deep learning*, LAP LAMBERT Academic Publishing.

KUTÁLEK, J. (n.d.). Detection of a yoga poses in image.

Long, C., Jo, E. and Nam, Y. (2022). Development of a yoga posture coaching system using an interactive display based on transfer learning, *The Journal of Supercomputing* pp. 1–16.

Nguyen, T. N., Tran, T.-H. and Vu, H. (2022). An automatic tool for yoga pose grading using skeleton representation, *2022 9th NAFOSTED Conference on Information and Computer Science (NICS)*, IEEE, pp. 187–192.

Ogundokun, R. O., Maskeliūnas, R., Misra, S. and Damasevicius, R. (2022). A novel deep transfer learning approach based on depth-wise separable cnn for human posture detection, *Information* **13**(11): 520.

Pham, T.-C., Luong, C.-M., Visani, M. and Hoang, V.-D. (2018). Deep cnn and data augmentation for skin lesion classification, *Intelligent Information and Database Systems: 10th Asian Conference, ACIIDS 2018, Dong Hoi City, Vietnam, March 19-21, 2018, Proceedings, Part II 10*, Springer, pp. 573–582.

Rajendran, A. K. and Sethuraman, S. C. (2023). A survey on yogic posture recognition, *IEEE Access* **11**: 11183–11223.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning, *Journal of big data* **6**(1): 1–48.

Stivaktakis, R., Tsagkatakis, G. and Tsakalides, P. (2019). Deep learning for multilabel land cover scene categorization using data augmentation, *IEEE Geoscience and Remote Sensing Letters* **16**(7): 1031–1035.

Tian, D., Yamagiwa, S. and Wada, K. (2022). Heuristic method for minimizing model size of cnn by combining multiple pruning techniques, *Sensors* **22**(15): 5874.

Vaughan, D. (2023). Multiclass averaging, `https://cran.r-project.org/web/packages/yardstick/vignettes/multiclass.html`. Accessed on April 20, 2023.

Yadav, S. K., Singh, A., Gupta, A. and Raheja, J. L. (2019). Real-time yoga recognition using deep learning, *Neural Computing and Applications* **31**: 9349–9361.

Yu, X., Wu, X., Luo, C. and Ren, P. (2017). Deep learning in remote sensing scene classification: a data augmentation enhanced convolutional neural network framework, *GIScience & Remote Sensing* **54**(5): 741–758.