MCAST

Using Machine Learning for Pose Detection in Kickboxing.

Logan Camilleri

Supervisor: Kassandra Calleja

June - 2024

A dissertation submitted to the Institute of Information and Communication Technology in partial fulfilment of the requirements for the degree of BSc (Hons) Software Development

Authorship Statement

This dissertation is based on the results of research carried out by myself, is my own composition, and has not been previously presented for any other certified or uncertified qualification.

The research was carried out under the supervision of Ms Kassandra Calleja

10/06/2024
.....
Date


.....
Signature

Copyright Statement

In submitting this dissertation to the MCAST Institute of Information and Communication Technology, I understand that I am giving permission for it to be made available for use in accordance with the regulations of MCAST and the Library and Learning Resource Centre. I accept that my dissertation may be made publicly available at MCAST's discretion.

10/06/2024
.....
Date


.....
Signature

Acknowledgements

I would first and foremost like to thank my supervisor Ms Kassandra Calleja for her patience, dedication and guidance. I would like to thank my parents for their continuous support during my journey so far and I hope to have made them proud. I would also like to thank two wonderful friends who have been there for me during my studies at MCAST, Leandros Galea and Aiden Lee Briffa. A special thanks goes to all my friends at The New Kickboxing Club who, not only were my inspiration for this dissertation but also contributed to the dataset. Finally, I would like to thank my colleagues at ICON who have pushed me to become a better developer. This dissertation would not have been possible without the contribution of these people and I offer them my sincerest gratitude.

Abstract

Pose detection is a vast field in the world of technology with various applications ranging from motion capture acting to crowd monitoring for security systems. One of the applications involves sports and while considerable research has been done to further advance in this area most of the research is conducted on yoga data and other sports are neglected. This research aims to focus on kickboxing by constructing a model dedicated to processing dynamic poses. Due to the lack of kickboxing pose datasets, one was compiled for this research. The artifact construction was split into 2 phases, the pose detection algorithm and the pose prediction model. The pose detection algorithm iterates through the annotated dataset and utilizes PoseNet to detect the specified keypoints, eventually, the keypoint data is formatted and saved in a CSV file. The pose prediction model first reads the keypoint data from the CSV and transforms it into 6 arrays: training, validation, testing and an array of respective labels for each. These arrays were then formatted to be read and processed by the model. The model chosen for the prediction is a CNN model utilising hyperparameters such as Conv1D and dropout while implementing technologies such as Adam optimisation. The trained model is then used to make predictions using the test set and the output metrics were then displayed in the form of a confusion matrix and classification report. Results indicate the model is performing well with an overall accuracy of 84% and an overall F1 score of 83.6. The model however is experiencing some overfitting depicted by the training and validation accuracy of 83.0% and 60.9% respectively. Time was the limiting factor and due to the nature of the model fine-tuning it further requires heavy architectural changes however, the results are still promising and will aid in furthering the research on the use of machine learning in pose detection for dynamic sports.

Keywords: Pose Detection, PoseNet, Machine Learning, Kickboxing, CNN.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ANN Artificial Neural Network
ML Machine Learning
DL Deep Learning
KNN k-nearest neighbor
CNN Convolutional Neural Network

Chapter 1: Introduction

Machine learning is a vast research area which has been applied to several domains. Its ability to detect poses in sports [2], provide assistance with physiotherapy [4] as well as its use in motion capture systems [5], has been noted. In the domain of sports, pose detection has been thoroughly investigated in relation to yoga; research in the field focused mainly on comprehending and improving the detection methods for yoga postures, as demonstrated by [6] and [7]. While pose detection systems, designed specifically to detect yoga poses, achieved good results, other sports – particularly those characterized by a higher degree of movement, did not received the same attention in terms of research; this resulted in a research gap for pose detection across a range of sports. The lack of datasets, designed for pose detection in other sports, also contributed to the above-mentioned research gap. One such sport, which could benefit from further research in pose detection, was kickboxing. Pose detection in kickboxing has been briefly investigated by [8], however, their research displayed shortcomings and faults that incentivised further research. The research by [8] used 3 poses which were "Jab", "Slip" and "Front kick". The example images in the research demonstrate poor technique implying the researcher was not a trained practitioner. [8] never mentioned the dataset size nor its origin. If the dataset was created by the researcher what were the variables considered? What was train:validation:test ratio of the dataset? Is the environment like background going to affect the results? While the model by [8] gave an accuracy of 89%, accuracy is too broad of a metric

to be used on its own. There is no detailed analysis of the model's performance and how it did on an individual pose basis. The Experiments were mainly targeted at gauging the participant's learning rate over time using the system as a trainer. This study was motivated by the impact of COVID-19 on practitioners of contact sports, where due to restrictions placed at the time, such training was not possible for a prolonged period of time. As a practitioner of kickboxing, who was negatively impacted during such times, the motivation for this research, also held a personal element. In view of this, the investigations undertaken for the purposes of this study, aimed to identify the optimal set of machine learning techniques that could identify and detect poses pertaining to kickboxing – a sport which is highly characterized by dynamic movement, as accurately as possible. Benefits stemming from this research include creating virtual trainers to still be able to train in cases where a human trainer can't be present. Like kickboxing regular human movement is dynamic in nature, the more data on dynamic movement the better motion capture systems become and this data provides a deeper understanding when trying to simulate realistic human movement. To test the above hypothesis, the following research questions needed to be investigated:

1. What is the optimal model for pose detection in kickboxing, considering factors such as accuracy and performance?

2. How effective is the optimal model in classifying poses for other sports when comparing its accuracy to pose detection in kickboxing?

3. How does an increase in the number of poses being detected affect the optimal model in terms of accuracy and performance?

To address these questions, and fulfil the aim of the study, a machine-learning

kickboxing pose detection model was created. The techniques incorporated in the developed model were based on current literature relating to pose detection and key point detection, such literature was mainly related to pose detection in yoga as this was the sport with the highest degree of published research available. The creation of a dataset containing different kickboxing poses was also required. The dataset was compiled using photos of several kickboxing practitioners, performing 3 different kickboxing poses, with variations in equipment and backgrounds for each pose. All photos included in the dataset were taken by the researcher. For each research question, an experiment outline was created to gather the necessary results. The model's effectiveness in detecting poses for kickboxing, its effectiveness in detecting poses for other sports as well as, its effectiveness in pose detection as the dataset scaled up, was determined using Accuracy and F1 score metrics for (1) detections across all dataset classes and (2) detections for each dataset class. This study is structured into several chapters, each addressing an aspect of the research conducted. In 2, the evaluation of popular pose detection technologies between 2017 and 2022 was undertaken. Furthermore, comparisons between some of these technologies were reviewed to identify the ideal technologies for this research. The methodology section 3, a breakdown of dataset construction detailing the procedures taken to ensure adequate coverage was done. The two subsections for the construction of the proposed model, the pose detection algorithm and the pose prediction model, were structured to include the process of creating and fine-tuning the code to create the proposed artefact. This section also highlighted and defined an Experiment based on each of the research questions posed and continued to be explored in the results section 4. The re-

sults of the 3 Experiments were analysed and compared with the work of other researchers and the Experiments themselves.

Chapter 2: Literature Review

2.1 Introduction

Machine Learning has been used for pose detection in several fields such as physiotherapy [4] and yoga [6], [7]. Several machine learning techniques have been used for this purpose including, convolutional neural networks (CNN) [9], k-nearest neighbors (KNN) [10], and Media Pipe [11].

2.2 Selection of Prominent Techniques in Pose Detection

A review of the most prominently used techniques in pose detection, between 2018 and 2022, was conducted by [12]. To transparently shortlist research papers of suitable quality, the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) guidelines [13] were adopted [12]. Scopus and Web of Science were queried, on 27 July 2022, to obtain research papers matching the following query: (“pose estimation” OR “pose tracking”) AND (“exercise” OR “sport” OR “rehabilitation” OR “physical education” OR “motor” OR “movement” OR “athlete*”) AND (“assistance” OR “correction” OR “guidance” OR “feedback” OR “learning” OR “coach” OR evaluate* OR assess* OR performance*) [12]. The search considered the title, abstract, and keywords (i.e., “topic” in Web of Science) and was limited to papers published in English between 2017 – 2022 [12]. For Scopus, articles with the document type “Review” were excluded. CADIMA was utilised for data collection and selection, with the following inclusion criteria: (1) human ‘body’ pose estimation; (2) user movement

Year	Article	Count
2018	Tennis	1
2019	Dance, Ski, Tai Chi, Exercise, Rehabilitation	5
2020	Rehabilitation	1
2021	Ballet, Yoga, Tai Chi, Kickboxing, Baseball, Exercise	11
2022	Baseball, Exercise	2

Table 2.1: *Types and quantity of papers found categorised by year.*

assessment and communication. Articles that classified movement or counted the number of correct repetitions of a specific movement, without giving feedback to users, were excluded from the review [12].

The resulting 104 articles (81 from Scopus and 23 from Web of Science) were collated and 18 duplicated records were removed. The title and abstract of the remaining papers were screened and a further 66 articles were excluded as they failed to meet the criteria set by PRISMA guidelines [13] or did not provide access to the full article text. The result of this rigorous screening resulted in a final set of 20 papers which were categorized by year and technique; articles were sub-sectioned, 2 dimensionally (2D) or 3 dimensionally (3D), based on the technique applied. Table 2.1 provides the final count of papers for each year of publication. These 20 papers were further categorized according to the technique used; Table 2.2 shows the count of papers for each technique utilized. OpenPose and PoseNet proved to be the most widely used technologies, with OpenPose having been the technology of choice for several years, with most papers having been published in 2021; PoseNet on the other hand only had papers published in 2021 [12].

Technology	Count
OpenPose	12
PoseNet	3
Other CNN	4
Others	1

Table 2.2: *Number of papers that used technologies mentioned.*

2.3 Pose Estimation using OpenPose

OpenPose is a real-time multi-person pose estimation library, based on a multi-stage CNN, which is capable of detecting 135 key points in the human body, for either 2D or 3D pose detection [14]. A bottom-up approach is utilized whereby individual keypoints are detected first and then associated to form the complete pose. This approach was reported to handle scenes with multiple people well, as it detects keypoints independently. However, the associations were observed to be computationally expensive, especially in crowded scenes, where false associations might occur, leading to less accurate poses.

Research by [3] aimed to compare and analyse four popular pose estimation models, namely, OpenPose, PoseNet, MoveNet Lightning, and MoveNet Thunder, using pre-classified images. The same dataset was utilized for all 4 models and their performance was compared. For OpenPose, [3] used non-maximum suppression (NMS) to obtain candidates for the body, which were identified in the confidence map. After identifying an object, bounding boxes were created to contain the object, and the probability of the object being identified was set as a score. The scores were then sorted in descending order, and redundant bounding boxes were removed based on whether the intersection of union (IoU) was greater than a set threshold - this was determined as situations where two bounding boxes identified the same object. Based on this, OpenPose was used to create part

affinity fields (PAFs), a set of flow fields representing the relationships between parts of many persons. Finally, bipartite matching was performed on the candidates using confidence maps and PAFs, producing full-body poses. In the research by [3], OpenPose successfully identified multiple persons and accurately estimated the poses within each image and had an average accuracy of 86.2% second to PoseNet's 97.6%. Another shortfall highlighted by [3] was that the average time taken for OpenPose to process 1000 images was significantly higher than the rest with it being 643.536 seconds.

The approach adopted by [9], to detect yoga pose for self-learning, compared the performance of two different people by comparing the similarity of angles of specific body parts. The angles for a pose made by a yoga instructor were compared to those made by a yoga practitioner (learner) for the same pose. OpenPose was utilized for feature extraction; the model simultaneously predicted a set of confidence maps of body part locations which were then fed to a multi-CNN architecture. A total of 24 joints across the whole body were considered and pairs of coordinated points for each joint were then used to calculate the similarity of angles. The similarity of angles (or angle difference value) was calculated by subtracting the learner's angle from the instructor's angle. By using the angles instead of the position of body parts, [9] reportedly eliminated the discrepancies that would appear when comparing 2 different body shapes. To display the accuracy of the learner's pose compared to that of the instructor's, a joint map of the learner was outputted and different colours were used to indicate the correctness of the angle of the joint. If a joint was displayed with a red gradient, it indicated a significant angle difference between the learner's pose and the in-

structor's. A joint displayed with a green gradient indicated an angle difference which was within an acceptable range to deem the pose as correct. Poses were classified as "perfect", "good", "not good", and "bad" based on the average angle difference for all joints. The resultant value was then mapped with a range function to classify the performance. Figure 2.1 shows the joint map comparing joint angles for poses made by learners to those made by instructors.

OpenPose was also utilized by [2] for pose estimation within a video feed. [2]'s aim was to create a yoga posture detection and correction system that uses a mobile-based approach for correcting improper yoga postures. Although there are some systems on yoga posture detection, there are no significant amount of systems for correcting improper yoga postures. The dataset contained six yoga postures, which were Bhujangasana (Cobra Pose), Padmasana (Lotus Pose), Shavasana (Corpse Pose), Tadasana (Mountain Pose), Trikonasana (Triangle Pose) and Vrikshasana (Tree Pose). a media server detecting 25 main keypoints was used; Table 3 below provides a list of said keypoints. Incoming video streams were sent to a baseline network for feature map extraction; each frame was obtained in JSON format. Feature maps were then directed to part confidence maps and part affinity fields; the output from these was then fed to a greedy algorithm. ReLu activation was used to help speed up the training process for the required model. A dropout layer was also utilized to prevent overfitting. The output of the drop-out layer was flattened, by transforming the multi-dimensional tensor into a one-dimensional array or vector, by collapsing all dimensions except the batch dimension. The flattened vector was then passed through an LSTM layer; the output from the final LSTM layer was passed to a dense layer, with SoftMax

Nose	Neck
Right Eye	Left Eye
Right Ear	Left Ear
Right Shoulder	Left Shoulder
Right Elbow	Left Elbow
Right Wrist	Left Wrist
Right Hip	Left Hip
Right Knee	Left Knee
Right Ankle	Left Ankle
Right Big Toe	Left Big Toe
Right Small Toe	Left Small Toe
Right Heel	Left Heel
Mid-Hip	

Table 2.3: *keypoints for pose estimation by [2]*

activation, configured with 6 units. This outputted a probability of corresponding yoga poses in terms of cross-entropy. [2] opted for 70%, 10% and 20% for training, validation and testing respectively the model's scope was to identify spatial features and LSTM to identify both spatial and temporal relationships.

Figure 2.1: *Colour-coded joint map indicating the correctness of a yoga pose.*

OpenPose was observed to perform efficiently when the dataset was changed to a real-time feed from a webcam [2]. The keypoints obtained had an accuracy ranging between 95% and 99.7% for test data, but the delay in predicting such poses was much higher than anticipated [2]. Therefore, [2] concluded that it was not the optimal solution for the final model. Using the single model trained with

OpenPose's keypoints data, [2] was able to achieve an accuracy of 99.87% on training data and 99.91% on test data or unseen data.

Similarly to [2], OpenPose was utilized for keypoint extraction from video recordings by [14]. [14]'s aim of the research was to create a possible virtual yoga instructor which could provide feedback to help users rehearse their yoga poses without a human instructor present. The extraction system was applied to the recordings and ran at 3 frames per second (FPS). A combination of CNN and LSTM was used, and a SoftMax layer with thresholding was applied to distinguish outlines. The model also applied ReLu activation, trailed by a dropout layer, which haphazardly dropped a small amount of loads to forestall overfitting. At the video level, the data was split into 60:20:20 for training, validation and testing respectively. After preprocessing, around 8000, 2500, and 2300 casings were obtained and prepared as testing and approval cases, separately. Since the split was done at video level, this resulted in straying from the proposed 60:20:20 to roughly 62.5:19.53:17.97 due to the varying lengths of the recordings. The model achieved a train exactness of 0.9992, a validation precision of 0.9987 and a test exactness of 0.9938 [14].

2.4 Pose Estimation using PoseNet

PoseNet is used for human pose estimation in real-time [15]. Single-person pose estimation is based on four input elements: (1) an image, (2) an image scale factor, (3) a horizontal flip, and (4) an output stride [3]. For multi-person pose detection, PoseNet requires an additional three elements: (1) the maximum number of detected poses, (2) the threshold for pose confidence score, and (3) the

NMS radius [3]. It was noted that the single-person detection algorithm was faster and simpler than the multi-person detection algorithm [3].

PoseNet was utilized by [8] to create a system aimed at teaching people kick-boxing at home without the need of a human instructor. The proposed system aimed to classify kickboxing poses and also recognise "good techniques" and "common mistakes". An Artificial Neural Network (ANN), based on the ml5 neural network helper module, was used; this was designed around 3 layers: an input layer, a hidden layer, and an output layer. The ANN required 34 inputs in the form of key points for the poses being considered. The model produced 6 outputs in the form of classified poses which had 3 correct techniques and 3 common mistakes. On startup the system prompted the user to perform the three techniques recognized by the model; pose detection would then be carried out and compared to the poses the ANN was trained on. If the system detected the wrong pose being performed or a mistake being made, it prompted the user by displaying how to perform the correct pose.

Two types of experiments were carried out by [8] to evaluate the developed system. Experiment 1 required, 5 target participants, all with different environmental parameters such as lighting, computer specs and background, to use the system for 3 consecutive days. The progress of the users was tracked to determine if there was any progress or learning. Experiment 2 required target participants to use the system for 3 consecutive sessions, with 3 30-minute breaks between each session, during a single day, to track their progress in performing the three techniques. This experiment involved 16 target participants and was conducted in a controlled environment, with the same lighting conditions, computer

processing power, and camera quality, with a black spot to indicate where the participant should stand. For both experiments, the system prompted the user to perform a technique and move on, however, if the system detected a wrong pose or mistake, it directed the user to improve the technique by displaying messages such as" Protect your face." and "Get lower". At the end, the system recorded the time taken to perform all 3 techniques, to gauge the learning process. The time recorded showed that, with each use of the system, the time to carry out the techniques, would decrease, indicating that the user was learning [8]. The system performed with an accuracy of 0.89; out of 144 predictions, 129 were correct. Hence, it was concluded that the system could be used to teach kick-boxing however some improvements needed to be made [8]. According to [8], capturing more advanced techniques would be the way forward, this would require the implementation of Dynamic Time Warping and body motion to produce more accurate outcomes [8].

Research by [5] proposed a system whereby poses, which included postures from a single frame, as well as movement from multiple frames, would be obtained via PoseNet. The 17 keypoints detected by PoseNet and their coordinates were extracted to output a 2D skeleton of the pose. The keypoint coordinates were determined using a decoding algorithm, which calculated the area around the key point, by first estimating its position using a heat map, and then exacting the estimated coordinates, using the key points indicated by an offset vector. Due to the limitations of Raspberry Pi, the model had to be as lightweight as possible. An RGB image was encoded using MobileNetV1, which was specifically designed for mobile and embedded applications, to reduce calculation costs.

To verify the performance of the proposed system, [5] employed 10 kinds of poses for 6 target participants. The camera module was fixed at a height of 50 centimetres and 250 centimetres away from the participant. Evaluations of the proposed system, concluded that semi-real-time pose recognition with an accuracy of 70% could be achieved for 3 postures and 2 movements without using depth information [5].

In their research, [10] opted for a system that combined PoseNet's deep learning framework, with a K-Nearest Neighbors (KNN) Classifier, to identify critical points in the human body. Movements from yoga Asanas were analysed, using both video and images, to check the correctness of yoga poses. The Proposed Model was trained using the output video which consisted of images of different poses where a KNN classifier was used to detect the pose by detecting the key-points of the limbs of a human and the model was saved. As for testing real-time Yoga Poses of the user were sent to the KNN classifier which identified significant points and classified them into one of the Yoga Pose then, a video of the instructor performing the detected Yoga Pose was used to compare and the correctness of the pose obtained. The algorithm stored all different cases and classified poses based on similarities, making it a pattern recognition technique. The values of new data points were predicted based on feature similarity; this implied that new data points would be assigned a value depending on how closely they matched points in the training set. This approach reportedly achieved an accuracy of 98.51% [10].

2.5 Pose Estimation using CNN-based Models.

Squats were described as an exercise, performed by both athletes and non-athletes, to reduce pain [16]. According to [17], when inexperienced individuals perform squats without professional coaching, the risk of spinal and/or knee injuries is increased. Due to office workers, and other non-athletes, struggling to find sufficient time to receive professional coaching, the development of a self-coaching system, was perceived to help individuals evaluate their exercise performance without professional assistance. In their approach, [16] opted for a CNN–LSTM model, for the recognition of human activity, using wearable sensor data. Their model comprised three convolutional layers: (1) a recurrent layer, (2) a dense layer, and (3) a softmax layer. The model's SoftMax layer generated the probability distribution over the prediction of squat postures. Each convolutional layer had 3 x 3 x N kernels with stride 1, where N doubled each layer from 8 to 32; ReLUs were used for activation functions. Max-pooling was also implemented at the end of every convolutional layer. The recurrent layer employed the long short-term memory (LSTM) units with 64 cells. A drop-out was also implemented on each convolutional and dense layer, to reduce overfitting. The model was trained using TensorFlow with Adam optimizer for 500 iterations at a learning rate of 0.001.

Data from 39 target participants was divided equally into ten groups; one of these groups was randomly selected as the test data while the rest were used as training data. The classification results were assessed based on averaged accuracy, sensitivity, and specificity. For five inertial measurement units (IMUs), the classification accuracy of the deep learning (DL) model was 91.7%. In the case of combinations using two IMUs, the combination with IMUs on the right thigh

and right calf exhibited the highest performance with an accuracy of 88.7% for DL. In the case of a single IMU, the best result was obtained from the IMU on the right thigh, with an accuracy of 80.9% for DL.

In research by [18], the action classification problem was modelled as a multi-class classification problem; the aim of was to determine the effectiveness of a classification model when applied to karate. This also aimed to close the research gap in pose estimation in technical sports. A CNN with a 200 by 200 image input, was used, together with a vector of numbers representing the probabilities of each of the activity labels. This setup was used to classify 8 categories containing roughly 100 images per category [16]. The network architecture was designed around the Keras API in Python, which is designed to run on top of TensorFlow. This configuration allowed for building networks and adding new modules easily. A sequential model was used to build the network, as the desired layers could be added one by one. [18] opted for a fully connected dense layer to build a feed-forward network so that all neurons from one layer were connected to the neurons in the previous layer. Similarly to [16], [2] and [14], ReLU activation functions were utilized to give non-linearity to the model to learn non-linear decision boundaries. Since the problem was considered a multiclass classification problem, the final layer was constructed as a SoftMax layer.

The Stochastic Gradient Descent (SGD) optimizer was used to configure the network and a dropout layer was implemented to prevent overfitting; the use of dropout layers was also observed in approaches adopted by [6], [16] and [1]. In this case, the dropout layer randomly turned off a few neurons during the training phase, so that the dependency on the training set would be reduced. The number

of epochs for training was set to 300 and a VGG16 model with weights pre-trained on ImageNet was used; to save the bottleneck features from the VGG16 model, the function save bottleneck features method was added. The 768 image dataset, used by [18] was split into 700 images for training and 68 images for validation. Initially, an accuracy of 54% was obtained when the dataset was trained for the first time; this was later increased to 68% with further training but the small number of images in the dataset was observed to be insufficient for training the network properly [18].

2.6 Pose Estimation using other technologies.

A system that utilized OpenCV as part of a hybrid approach, combining machine learning, deep learning classifiers and a support vector machine (SVM), was proposed by [1]. The purpose of this research was to create an AI-based learning application to identify yoga poses and provide personalised feedback using an Android platform [1]. A dataset comprising images of the following poses: Padmasana (Lotus Pose), Shavasana (Corpse Pose), Tadasana (Mountain Pose), Trikonasana (Triangle pose), and Vrikshasana (Tree pose) was created. The dataset contained a series of videos recorded at 30 frames per second which were between 45 and 60 seconds long; a train:test:val split of 60:20:20 was applied. Using an image of the desired pose and an image of the user performing the said pose, the coordinates of keypoints, detected via OpenPose, were recorded. A comparator module determined the similarity of the user's pose to the desired one, based on the similarity of keypoint coordinates in the 2 images. To achieve this, a SVM, with a radial foundation characteristic (RBF) kernel, was utilized;

this offered additional flexibility when compared to other kernels [1]. Figure 2.2 illustrates the architecture for the model discussed by [1]. A training accuracy of 0.9841, a validation accuracy of 0.9910 and a test accuracy of 0.9868 were achieved [1]. It was noted that while the accuracies of the model proved promising, expanding the dataset could increase the effectiveness of the model to be used as a coach [1]. Introducing a multi-person aspect was considered as a future improvement to the model, however, various parameters, such as background, lighting, and overlapping figures, were perceived as challenging factors for multi-character pose estimation [1].

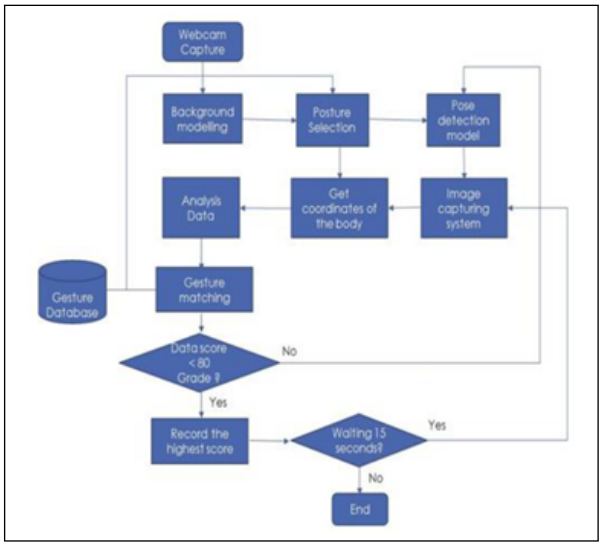

Figure 2.2: *Architecture of model developed by [1]*

The high performance of traditional CNN architectures was recognized by [19], however, occlusions were described as a major challenge for pose detection. The identification of occluded parts, by analysing full 3D poses, were seen to help, in some ways, to resolve this problem [19]. However, such approaches also introduced new problem areas; to analyse full body 3D poses, data available only through Motion Capture (MoCap) systems is required. Although accurate pose estimation was seen to be possible in controlled environments, such approaches did

not generalize well to real images [19]. According to [19], regression methods could only predict one pose for a given image and failed to model multi-modal outputs for ambiguous cases. Therefore, [19] proposed combining the discriminative power of classification networks, with the smoothness of regression methods through a simple modification within the learning procedure; this was similar in spirit to Faster R-CNN. A Region Proposal Network (RPN) was designed to generate high-quality region proposals where object bounds and scores were predicted. This network was proposed so that instead of classifying objects, human poses could be classified; an end-to-end Localization-Classification-Regression architecture, named LCR-Net, was proposed, to detect 2D and 3D poses in natural images [19]. According to [19], a 2D human pose could be defined as the pixel coordinates of each joint in an image, while a 3D pose could be defined based on the location of each joint relative to the body centre. The LCR-Net architecture, given an image, first computed the convolutional features then computed the localization component, also called Pose Proposals Network, in the context of pose detection. The Classification branch was tasked with estimating the probability of anchor-poses being correct at each location. This enabled it to learn to localize humans, as well as to estimate which anchor pose were more probable. The Regression branch computed an anchor-pose-specific regression to estimate the difference between the true human pose and the pose proposal. As a result, a list of pose proposals could be outputted, this consisted of a set of candidate locations, where the anchor poses were hypothesized. A region of interest (RoI) pooling layer was used to aggregate the features inside each candidate region. This approach achieved 47.9% mean average precision, at 0.5 confidence thresh-

old (mAP@0.5), on the test set [19]. It was observed that the more crowded the image was, and the higher the number of instances where there was an overlap between people, the poorer the model performed. [19] specified that the model detected multiple people even if they did overlap and that the model was also able to identify unusual poses and truncations. These conclusions were arrived to by testing the model with a specific test set, comprising images from the scenarios mentioned [19]. It was noted that a considerable boost to the model could be obtained by adequately scoring the pose proposals, the first attempt at rescoring poses showed encouraging results [19]. However, [19] emphasized the importance of having good quality training data as the main improvement required by the model. Initially, a solution to automatically annotate 2D images with "pseudo" ground-truth 3D poses was aimed for but research by [19] indicated that better 3D and 2D performances could be obtained with LCR-Net if more accurate real-world training data was available, e.g., through manual curation.

2.7 Comparison between OpenPose and PoseNet

The popularity of OpenPose and PoseNet for pose detection has been highlighted by [12]. Good results in pose detection using these techniques have been achieved by [8], [2], [10] in various domains such as yoga, karate, and general human posture. In a comparative analysis of OpenPose, PoseNet, and MoveNet, [3] concluded that MoveNet Lightning was the fastest, while OpenPose was the slowest. PoseNet was however observed to be the only model capable of estimating poses for multiple persons. The accuracies of OpenPose, PoseNet, MoveNet Lightning, and MoveNet Thunder, as reported by [3], are depicted in Table 2.4. The dif-

Technique	Accuracy
OpenPose	86.2%
PoseNet	97.6%
MoveNet Lightning	75.1%
MoveNet Thunder	80.6%

Table 2.4: *Accuracies of models tested by [3]*

ference in accuracy between OpenPose and PoseNet was reported to be 11.4%, in favour of PoseNet, this may be regarded as significant. The speed at which PoseNet processed 1000 images was approximately 78 seconds while OpenPose took approximately 650 seconds to process the same images; hence PoseNet was reportedly approximately 88% faster than OpenPose [3]. The data provided by [3] also indicated that PoseNet would provide a greater benefit with small-scale datasets while providing high accuracy; tests to obtain the above-reported results were performed with 1000 images. Therefore, based on the above, PoseNet was deemed to be the most ideal technique to be used as part of the research detailed in forthcoming chapters. This research aims to further close the research gap in pose detection for technical sports by detecting kickboxing poses.

Chapter 3: Research Methodology

3.1 Research Aim and Approach

This researched aimed to create a machine-learning pose detection model that could accurately identify kickboxing poses; a research gap was noted as a very limited number of studies focused on pose detection in kickboxing [8], this was also true for other sports with dynamic movement [18]. The majority of research in the area focused on pose detection in yoga [6], [7], [1], [9], [2], [14], [10]. To attempt to bridge the research gap, the following research questions were posed, along with the following hypothesis:

Hypothesis: Machine Learning Models can be used for accurate pose detection in Kickboxing.

Research Questions: (1) What is the optimal model for pose detection in kickboxing, considering factors such as accuracy and performance?

(2) How effective is the optimal model in classifying poses for other sports when comparing its accuracy to pose detection in kickboxing?

(3) How does an increase in the number of poses being detected affect the optimal model in terms of accuracy and performance? The above research questions were used as guidelines to answer the hypothesis. Based on findings from Chapter 2 an artefact was constructed based on the most ideal machine learning model based on research by [16], [4], [6], [14] and key point extraction framework according to existing research by [3], [12]; the model was trained on a dataset specifically compiled for the purposes of this research. For each research ques-

tion, an experiment was outlined to assess the implemented model's capability, based on metrics such as overall accuracy, overall F1-Score and, individual class F1-score. During the course of the above-mentioned experiments, the model was refined to achieve more optimal results; Multiple iterations were made to improve results prior to conducting the next experiment in order to ensure a high performance increasing the accuracy of the artifact.

3.2 Dataset Compilation

The availability of public datasets suitable for the purposes of this study was found to be limited due to the research gap that this study aimed to address. The data set utilized for the purposes of this study was hence compiled by the researcher using images captured with an 18MP Canon EOS 550D using an 18 - 55mm lens. The resulting dataset comprised of 426 images, for 3 widely used kick-boxing poses, namely the low kick (aiming at the thigh area), the knee strike (aiming at the chest area) and high roundhouse kick (aiming at the head or upper torso). The number of images for each pose is illustrated in Figure 3.1 below:

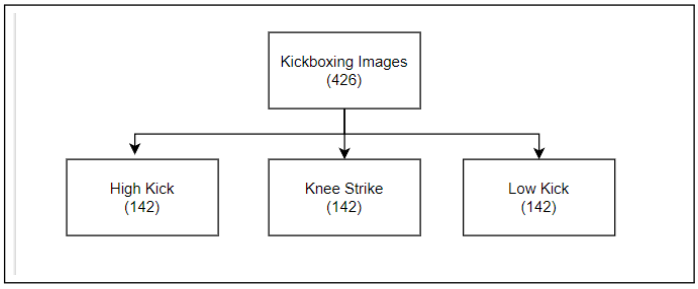

Figure 3.1: *The number of images taken and their distribution*

These poses were chosen due to being easily distinguishable from each other and other kick-boxing poses, but still requiring a certain level of technique and

flexibility to perform properly.

The images for each pose were taken using 9 different participants; all participants were trained kickboxing practitioners with varying levels of expertise in the sport as well as varying body types. Each practitioner was asked to perform, each of the required poses, with and without equipment, such as gloves, shin guards and a head guard. Practitioners were also asked to perform the poses in front of varying backgrounds within the dojo (gym); backgrounds varied from a blank wall to more complex backgrounds which had shelves, equipment, etc. Photos 3.2 and 3.3 are some examples of the variations mentioned. Images of the poses, with varying kinds of equipment and backgrounds, were included, as these were considered as 2 factors that may hinder the correct detection of a pose in kickboxing. Papers [18], [1], [4], [19] and, [20]all mentioned that the background was an important factor to consider as it affected the accuracy. Paper [12] determined that the equipment (snowboard) was sometimes blending with the background. [21] proposed background replacement or blurring to reduce noise.

Figure 3.2: *Left stance High-Kick with some equipment on busy background*

Figure 3.3: *Right stance Knee-Strike with no equipment on blank background*

Figure 3.4 below, details variations of poses and backgrounds for each pose together with the number of images included in each class. Table A.1 in the appendices details all combinations of backgrounds and equipment considered for each pose.

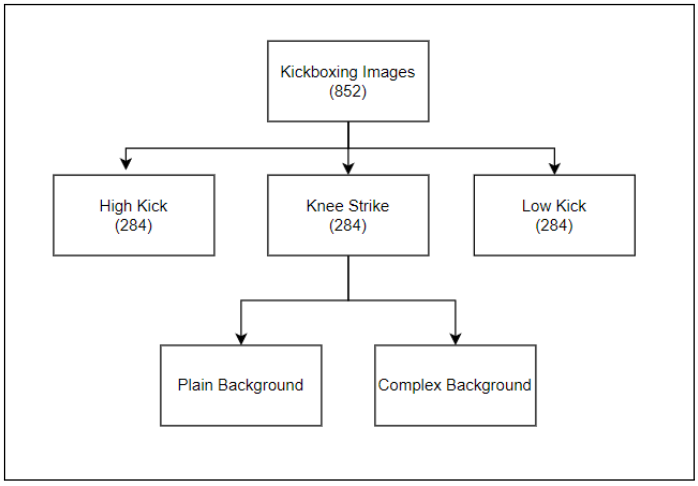

Figure 3.4: *Dataset including background subsection*

The images originally captured by the researcher, had all participants performing the poses using a left stance. To avoid any bias in pose detection, due to the stance, all images were mirrored along the vertical axis to mimic the pose

being performed from a right stance similar to [19]. This doubled the size of the original dataset to 852 images. Images were then annotated using Roboflow to identify the pose being performed and compiled, the images were split into training:testing:validation ratio of 70:20:10. The final structure of the dataset and image count of the train, test, valid split can be seen in figure 3.5 below.

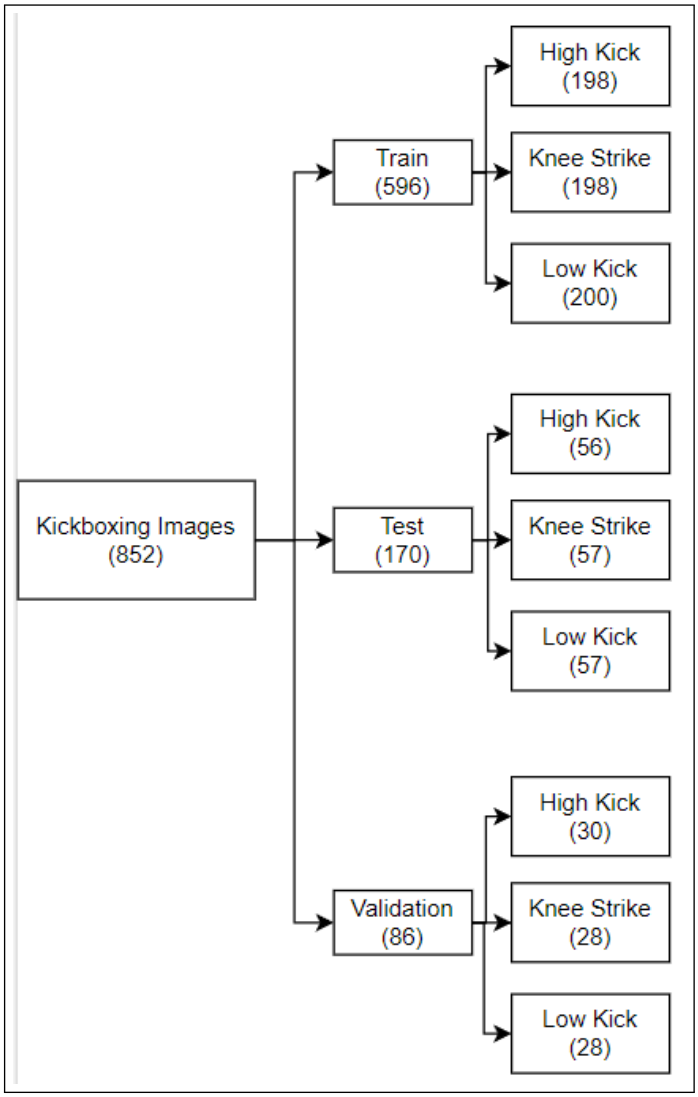

Figure 3.5: *Final dataset count including Train, Test and Validation*

3.3 Environment Setup

Python was deemed to be the best choice in terms of programming language for the construction of the artefact due to its utility in machine learning thanks to the

support offered by several of its libraries. For the purposes of this study, Python v 3.10.12 was utilized along with the following libraries: Tensorflow, NumPy, PIL, Matplotlib, os, cv2 and pandas. Google Colab was also used as the environment for training, testing and validating the model constructed; the extraction of keypoints and the resources required by the model to train and predict poses were deemed computationally expensive for a local machine; Google Colab offers T4 GPU's which reduce runtime.

3.4 Artifact Construction

As illustrated in Figure 3.6, the artefact constructed comprised of 2 parts, set up to run independently of each other. The first part carried out key point extraction using PoseNet. This was selected as the most ideal keypoint detection algorithm based on research by [3], [5] and [10]. Images of kickboxing poses from the dataset were fed to PoseNet and coordinates of keypoints were calculated to define relationships between keypoints. Keypoint were defined as the X and Y coordinates of a specified point of interest e.g. left elbow, right elbow, left wrist, right wrist, left hip and right hip; a full list of such points is provided in Table A.2 of the appendices. Relationships between keypoints were defined as edges; a full list of edges considered for the purposes of this studies defined in Table A.3 of the appendices.

Using keypoints and edges, a skeleton was superimposed on the image based on PoseNet detections; this allowed for human verification to ensure correct keypoint and edge detections. Keypoint and edge data was formatted using a data frame and saved as a CSV file.; a sample of the CSV file is provided in Figure

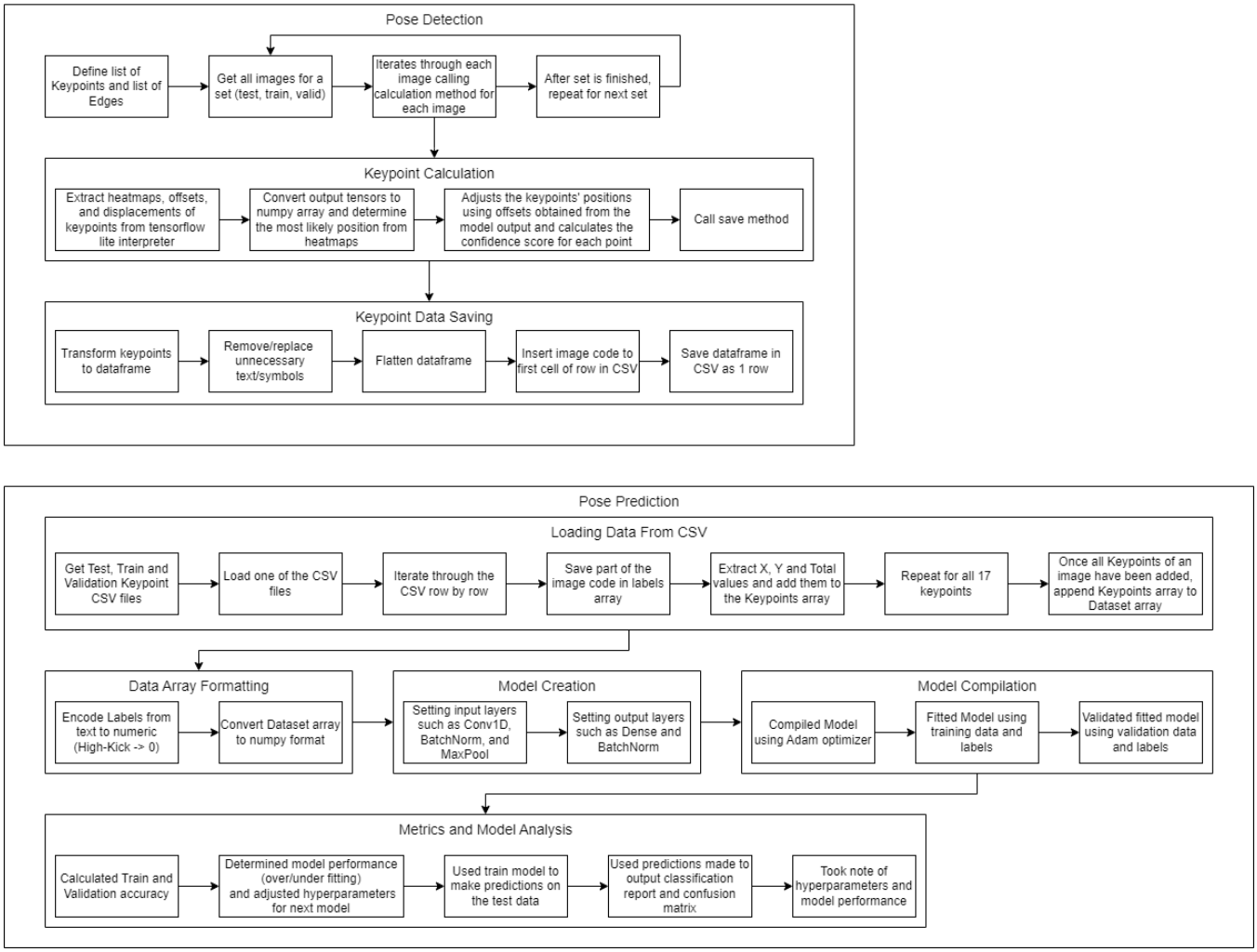

Figure 3.6: *Detailed diagram of artefact process flow.*

A.1 of the appendices. The second part of the artefact carried out pose prediction using a Convolutional Neural Network (CNN). CNN was also seen to be the ideal choice of model for pose detection by [3], [5], [10]. The data relating to keypoints extracted from images in the dataset was read from a CSV. This data was separated into 6 arrays: validation_dataset, validation_labels, train_dataset, train_label, test_dataset, test_labels and encoded into a numeric format. The data arrays were also reshaped in NumPy format and converted to 2-dimensional arrays where the number of rows was the same as the length of the dataset; this was done so that the model could read the required data from the arrays where each row contained the flattened version of one of the elements in the original dataset. To construct the CNN, a sequential Keras model was implemented using Tensorflow libraries; each layer had exactly one input and one output tensor. The first layer determined the shape of the data utilized by the model – in this case, the shape was (17 * 3) as each pose had 17 keypoints and each keypoint had 3 measures (X and Y coordinates and a score). The proposed model is an amalgamation of the models by [16] and [4]. The model was structured to have a Conv1D layer followed by BatchNormalisation and MaxPooling, this was repeated 3 times. The Conv 1D layers specified a kernel size of 3 and relu activation. [16]'s model was composed of three convolutional layers: a recurrent, dense, and softmax layer. Each of [16]'s convolutional layer had 3x3xN kernels with stride 1, where N doubled each layer from 8 to 32 and used the rectified linear units (ReLUs) as the activation functions and Max pooling was implemented at the end of every convolutional layer. The configuration of the proposed model's filers for the Conv1D layers were 64 for the first layer, 128 for the second and

256 for the third. After each Conv1D layer, batch normalisation was applied to stabilise training by normalizing layer activations. Finally, MaxPooling1D was added after each batch normalisation layer with a pool size of 2 and 2 strides. After 3 sets, a flatten layer was added to convert the multi-dimensional data into a single dimension. The final layers of the model were a dense layer followed by a batch normalisation layer for 3 times. The first dense layer used 128 neurons along with relu activation. The second dense layer changed the number of neurons down to 64 while still using relu activation. The final dense layer used 3 neurons signifying the 3 classes while using softmax activation. [4]'s model consisted of three 1D convolutional layers, each with 128, 256 and 128 feature maps, respectively. Each convolutional layer was followed by batch normalization and a rectified linear unit (ReLU). Global average pooling was used after the last convolutional layer. The proposed model was compiled using the Adam optimizer specifying the learning rate to 0.01 and sparse categorical cross entropy as the loss function. When fitting the model 50 epochs were used in conjunction with early stopping, restoring the best weights. The loss and accuracy for training testing and validation were calculated to give an indication if the model is over or underfitting. Based on these values the model's hyperparameters were adjusted accordingly to fine-tune to a satisfactory point.

3.5 Model Evaluation

To test the model's performance, the research questions were used as guidelines to create 3 separate experiments which were carried out sequentially.

3.5.1 Experiment 1

The first experiment aimed to answer the first research question i.e. "What is the optimal model for pose detection in kickboxing, considering factors such as accuracy and performance?". To answer this question, existing literature, as discussed in Chapter 2, was used to determine which model architecture and hyperparameters provided the best results. The research conducted by [12] collected recent data to determine the most prominent technology used for pose detection. Given that a trend in certain technologies was noticed further investigation led to research by [3]. [3] tested MoveNet lightning and thunder alongside PoseNet and OpenPose in an attempt to extract features such as the efficiency of detecting 1000 images given the keypoints the technology has available. For each technology represented in [12] a section in the literature review was created. Papers using OpenPose, PoseNet, CNNs and other miscellaneous technologies were found and using results of performance, accuracy and what the technology is best suited to, the ideal model for pose detection in kickboxing was determined and implemented based on the needs of this study. The model indicated to be the optimal choice, based on existing literature, was determined to be a CNN and was later trained and tested using the kickboxing dataset created for the purposes of this research. Once the model was trained, classification reports, confusion matrices, training and validation accuracy were outputted. Based on the results obtained the ability of what literature considers to be the most optimal model for pose detection in kickboxing was determined.

3.5.2 Experiment 2

The second experiment was tied to the second research question: "How effective is the optimal model in classifying poses for other sports when comparing its accuracy to pose detection in kickboxing?". A second dataset comprised of yoga pose images was compiled and the model was trained and tested, using the yoga data et, to determine its applicability to pose detection in a different domain. The aim was to determine if the model was suited specifically to kickboxing pose detection or if it could be used for general pose detection in sports, provided it was trained with a suitable dataset/ To ensure the validity of the experiment, the yoga dataset adhered to 3 classes and was as close to the original number of images in the kickboxing dataset as possible. Figure 3.7 illustrates the classes and number of images in each class for the yoga dataset. Compiling the yoga with more classes and a similar image count was perceived as being detrimental to model's performance due to having less data to train and test on for each pose. By compiling the yoga dataset using the same number of categories but more images than the Kickboxing dataset, performance between the 2 models might also have been affected and would have prevented a fair like with like comparison. This was just my thought process to ensure a fair test. The train: validation: test split for the Yoga dataset did not result in the exact same image count as the original, however, the variance was small enough to be considered as having a negligible effect on the results obtained. The performance of the model, when using the yoga dataset, was evaluated using the same metrics as outputted for the model trained with the Kickboxing dataset; results achieved by both models were then compared to determine the model's effectiveness in

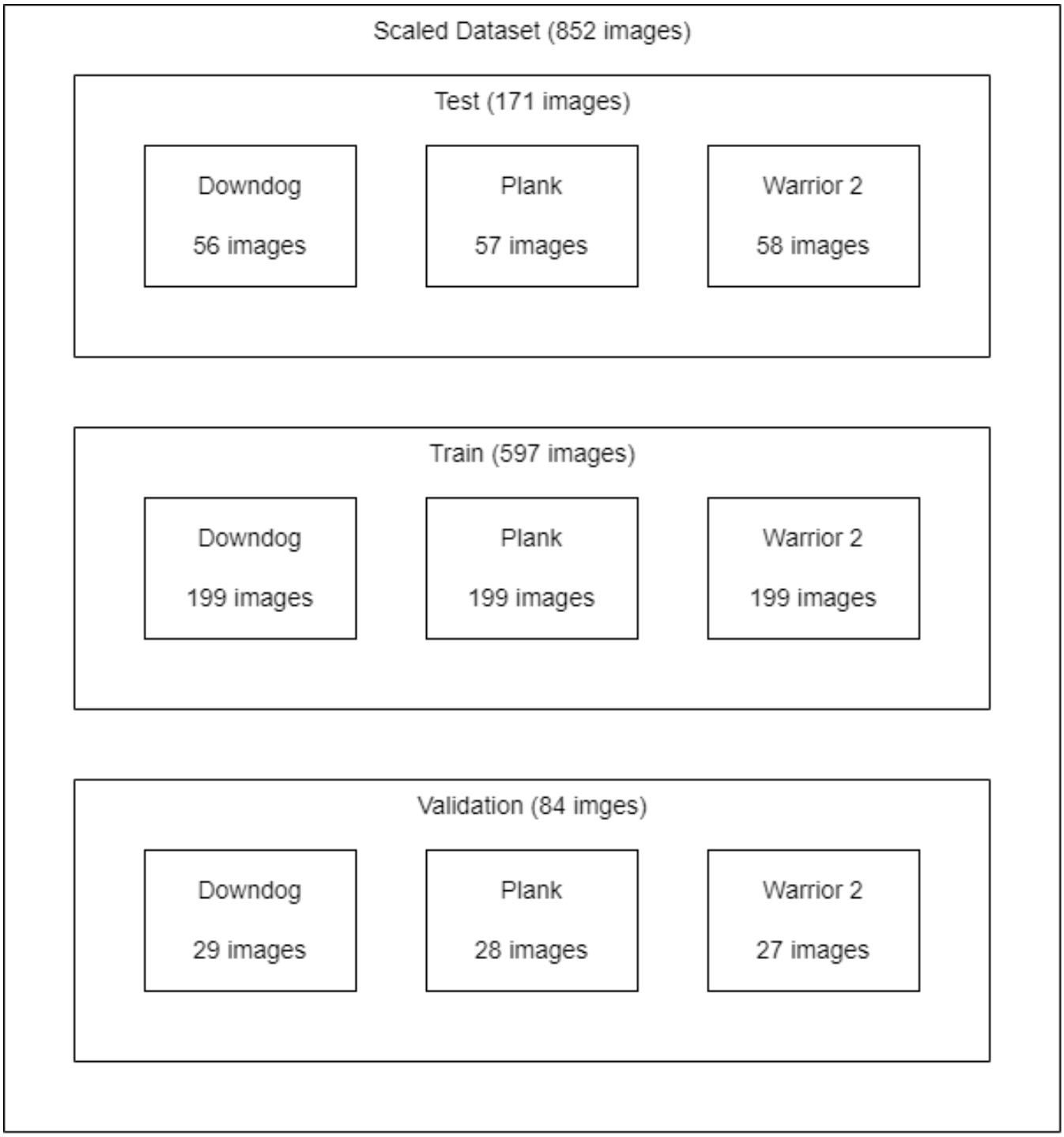

Figure 3.7: *Diagram of same scale yoga dataset*

classifying poses for other sports.

3.5.3 Experiment 3

The third and final experiment focused on Research Question 3: “How does an increase in the number of poses being detected affect the optimal model in terms of accuracy and performance?” To address this question. the yoga dataset was expanded to contain 2 additional classes called ”tree” and ”goddess” was compiled. Figure 3.8 illustrates the classes and number of images in each class for the expanded yoga dataset. The yoga dataset was chosen to be expanded upon to further test the model’s performance and adaptability when dealing with more classes and using static poses. Similarly to the other experiments, the same metrics were outputted and used to assess the model’s performance as the dataset was scaled up.

Based on the literature reviewed in Chapter 2, the methodologies discussed were constructed with the aim of addressing the research questions in light of testing the hypothesis. This was done by constructing a dataset comprised of kickboxing images, with factors such as background and equipment worn taken into consideration. Two additional yoga datasets were compiled, one using the same image count and ratio as the kickboxing dataset and another implementing 2 additional classes; however, the second yoga dataset did not follow the image ratios. Utilizing PoseNet for keypoint extraction and a Convolutional Neural Network (CNN) for pose prediction, the artifact demonstrated a meticulous approach to data preprocessing and model construction. The CNN architecture, inspired by prior research, integrated Conv1D layers, batch normalization, and max pooling, with hyperparameters fine-tuned through iterative training and evaluation. The Ex-

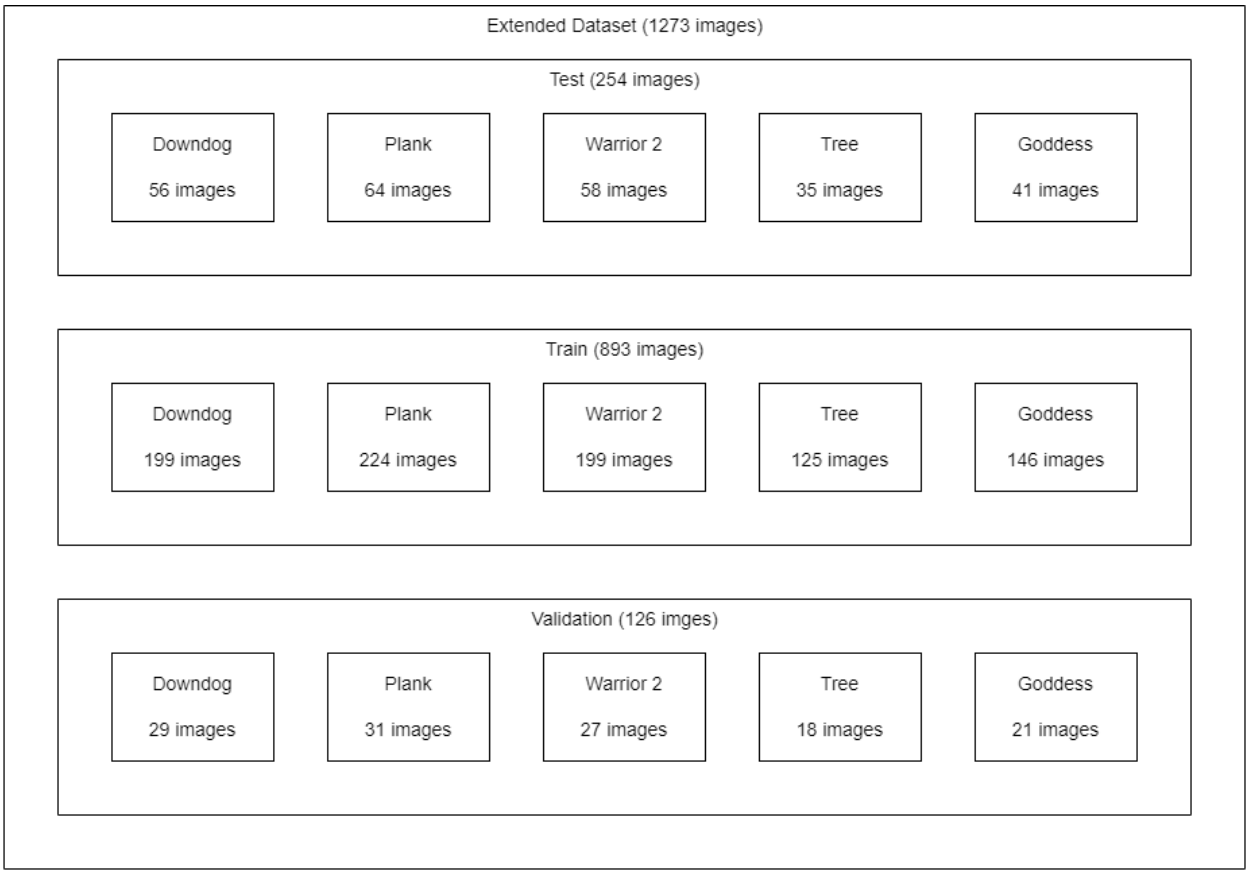

Figure 3.8: *Diagram of yoga dataset with 5 classes*

periments detailed were used to evaluate the proposed model in a way to answer the research question tied to the respective Experiment.

Chapter 4: Analysis of Results and Discussion

This chapter aims to address and analyse the results of the experiments conducted in the previous chapter. Each experiment's results were compared with experiments conducted and with the results of some papers in the literature review. This chapter also aims to answer the research questions posed and the hypothesis.

4.1 Assessing Experiment 1

Experiment 1 was used to create a baseline of the model's performance. The proposed model was tested using the created kickboxing dataset. The model refinement was done on this dataset as it was the scope of this research. The model architecture chosen was a CNN based on works by [14], [18], [2]. These papers all provided insights into their proposed models and a detailed analysis of the parameters used to achieve a high performance. Overall, the model's performance with the kickboxing dataset proved to yield promising results. The overall F1 score for pose detection given a dataset of 852 images was 82.67 with the overall accuracy being 83%. The F1 score achieved for each class in the data set is illustrated in the classification report 4.2 and a visualisation of the image's classification can be found in the confusion matrix 4.3 which was generated using Python's sklearn.metrics library. As shown in table 4.1 the history was saved to use some of the important variables and output the training and validation accuracies and losses. These were important due to them giving an indication of the

model's performance. The first usage is noting if both accuracies are of an acceptable level which in this case, they are acceptable. The second usage is comparing the difference between the accuracies, if both are similar but low then this indicates the model is underfitting. If the training accuracy is high but the validation is low then the model is overfitting, in this case, the model is very slightly overfitting. Although the above results indicated that the model could be considered useful for pose detection in kickboxing, the training accuracy and validation accuracy varied by 0.221; this could be an indication of the model suffering from a slight degree of overfitting. In order to reduce overfitting as much as possible, the model implemented batch normalisation which acts as a regularizer, reducing sensitivity to the initial weights. Dropout layers and early stopping were also implemented to combat overfitting as dropout prevents the model from becoming too dependent on particular neurons and early stopping monitors the validation loss and stops the training process once the validation loss stops improving for a specified number of epochs. [18] utilised dropout layers as the model parameters were gaining bias towards the training data. Overfitting may be attributed to the small size of the dataset. Furthermore, due to the small size of the dataset, a few misclassifications heavily impacted the results. [14] achieved a 99% accuracy using their proposed CNN model however, the model was designed around a very large dataset. The dataset by [14] was roughly 13k images with a 60:20:20 train:test:val ratio. This showed that CNN models can achieve very high results in pose detection applications, and the proposed model in this research demonstrated its ability to do so for dynamic movement. Further fine-tuning of the model and an expanded dataset may contribute towards resolving the issue with underfitting,

Train loss	0.441
Train accuracy	0.830
Val loss	1.172
Val accuracy	0.609

Table 4.1: *Loss and Accuracy of the Kickboxing model in training and validation phase*

F1 0	0.82
F1 1	0.86
F1 2	0.80
avg F1	0.826
accuracy	0.83

Table 4.2: *Final results captured from the kickboxing model*

however, such fine-tuning could not be carried out due to the increased complexities required to the architecture of the model and the time constraints governing this research. Experiment 1 aimed to answer the research question "What is the optimal model for pose detection in kickboxing, considering factors such as accuracy and performance?" The results from experiment 1 showed that the proposed model has reached a level of performance which can answer research question 1. This is also supported by the research conducted, in papers like X and Y where CNNs were employed for pose detection and estimation and provided promising results.

4.2 Assessing Experiment 2

Experiment 2 aimed to test the adaptability of the proposed model. This was done by creating a secondary dataset composed of yoga images while still keep-

	High-Kick	**Knee-Strike**	**Low-Kick**
High-Kick	45	4	5
Knee-Strike	5	48	3
Low-Kick	7	5	48

Table 4.3: *Confusion matrix of the kickboxing model*

ing the same image count and ratios as the kickboxing dataset.

The classification report illustrated in 4.5 shows the results obtained by the same model used in Experiment 1 when trained on a yoga dataset with the same amount of images and classes. In experiment 2, the model performed similarly to when this was trained with the kickboxing dataset, only a slight increase of 1% in both the average F1 score and overall accuracy were observed. The training accuracy achieved by the model, trained with the yoga dataset, was 6.4% higher than that for the model trained with the kickboxing dataset, whilst validation accuracy, for the model trained with the yoga dataset, was 13.6% higher. There was also only 1 less misclassification by the model trained with the yoga dataset when compared to the model trained for experiment 1. Similar results were perceived to be due to the configuration of the model being identical between Experiment 1 and Experiment 2, with the only variable being different datasets aimed at pose detection in different fields. In view of this, it was perceived that the model could adapt well to the detection of poses for other sports, thereby indicating that some degree of generalization was achieved. Experiment 2 aimed to answer the research question "How effective is the optimal model in classifying poses for other sports when comparing its accuracy to pose detection in kickboxing?" The results from experiment 2 showed that the proposed model was able to adapt and detect poses for a different sport, and when compared to the results of experiment 1 the proposed model was able to do so at a similar proficiency.

Train loss	0.294
Train accuracy	0.894
Val loss	0.780
Val accuracy	0.745

Table 4.4: *Loss and Accuracy of the same scaled Yoga model in training and validation phase*

F1 0	0.79
F1 1	0.83
F1 2	0.89
avg F1	0.836
accuracy	0.84

Table 4.5: *Final results captured from the same scaled Yoga model*

4.3 Assessing Experiment 3

Experiment 3 aimed to determine the model's ability to handle more poses than the initial 3. To do so 2 poses "Tree" and "Goddess" were added to the yoga dataset used in experiment 2. However, these classes did not keep the same image ratio as the previous experiments but the train:test:val split was kept the same. As illustrated in 4.7, the expanded yoga dataset yielded lower results, given the same model, when compared to experiment 1 and experiment 2. The Average F1 score dropped to 76.0%, while the overall accuracy of the model to 76.0%. The drop in performance could be attributed to the 2 new classes, added to the yoga dataset to enable it to detect 5 poses instead of 3, not adhering to the ratio of images between classes, thus leading to less data to train, validate and test the new classes. Most misclassifications were noted to have happened in the 'Downdog' and 'Plank classes. W when the image ratio was taken into ac-

	Downdog	**Plank**	**Warrior2**
Downdog	44	8	4
Plank	8	49	0
Warrior2	4	4	50

Table 4.6: *Confusion matrix of the same scaled Yoga model*

count both Downdog and Plank had the highest misclassification percentage with Downdog misclassifying 32.1% of test images and Plank misclassifying 31.7% of test images. This could be attributed to multiple factors related to the pose detection algorithm i.e. PoseNet, due to multiple factors, such as background, and resolution of the new classes. Experiment 3 aimed to answer the research question "How does an increase in the number of poses being detected affect the optimal model in terms of accuracy and performance?" The proposed model did not perform on a similar level to that in Experiments 1 and 2. The decrease in performance may stem from the model itself being slightly overfitting, the PoseNet detection algorithm or, the quality of the images themselves. Given that the model had to classify from a larger pool of classes, it is speculated that the confidence of classifying the same image correctly is less than in previous experiments, which contained fewer classes. Each experiment was tailored to answer a specific research question which aimed to test the hypothesis while also identifying the proposed model's durability and shortfalls. The hypothesis: "Machine Learning Models can be used for accurate pose detection in Kickboxing." Has been proven to an extent. The proposed artefact can accurately detect and predict poses in kickboxing which is a sport with dynamic movement. The artefact also proved its strength in detecting and predicting yoga poses. However, when the model encountered a dataset with a larger class pool and an unbalanced image count the performance suffered. Whether or not the increase in classes or the image count was the root cause of the decrease has to be determined in future experiments.

F1 0	0.72
F1 1	0.72
F1 2	0.82
F1 3	0.81
F1 4	0.73
avg F1	0.76
accuracy	0.76

Table 4.7: *Final results captured from the larger dataset Yoga model*

Train loss	0.549
Train accuracy	0.802
Val loss	0.855
Val accuracy	0.677

Table 4.8: *Loss and Accuracy of the larger dataset Yoga model in training and validation phase*

	Downdog	**Plank**	**Warrior2**	**Tree**	**Goddess**
Downdog	38	2	12	0	4
Plank	0	28	2	5	6
Warrior2	7	1	56	0	0
Tree	1	0	1	32	1
Goddess	3	6	1	7	40

Table 4.9: *Confusion matrix of the larger dataset Yoga model*

Chapter 5: Conclusions and Recommendations

This research aimed to test the hypothesis "Machine Learning Models can be used for accurate pose detection in Kickboxing" by devising Experiments to answer the research questions posed. Experiment 1 was the creation of the proposed model and was tested using the created kickboxing dataset and directly addressed the research question "What is the optimal model for pose detection in kickboxing, considering factors such as accuracy and performance?". The results of this Experiment were used as a baseline to determine the performance of the subsequent Experiments. The model was based on the works of [2], [18] and, [14] using layers such as Conv1D, MaxPooling, BatchNormailization and Dropout. Methods such as Adam optimisation and relu activation to adjust the learning rate and introduce non-linearity. The results of Experiment 1 displayed that the model constructed has promising results with the overall F1 score being 82.67 and an accuracy of 83%. When taking a closer look at the F1 score of each individual class the metrics were also promising, being 82, 86 and 80. A look at the training and validation accuracies showed that the model has a minor overfitting issue. Due to the simplified architecture of the model which was designed around the relatively small size of the dataset, optimising the model to the point where there is virtually no overfitting nor underfitting would have required major architectural changes which was not possible due to time constraints. Despite this, the proposed model has shown it was able to successfully identify and classify kickboxing poses. The research question has been answered given the

model's performance despite the model encountering minor issues such as overfitting. Experiment 2 was based on the research question "How effective is the optimal model in classifying poses for other sports when comparing its accuracy to pose detection in kickboxing?" and aimed at testing the adaptability of the proposed model by training and testing on another sport. The chosen sport was yoga as it is a static sport which has already been heavily researched in regard to pose detection. The yoga dataset constructed for Experiment 2 consisted of 852 images and used the same 70:10:20 ratio. This was done to ensure a fair comparison between both Experiments where the only variable was the dataset type. Experiment 2 displayed the model's performance was consistent with that in Experiment 1. The overall accuracy of the model was 84% and the overall F1 score was 83.6. On a class level, the model's range was slightly larger than in Experiment 1. The class metrics were 79, 83 and 89. The training and validation accuracies saw an improvement with them being 89.4% and 74.5% respectively indicating little overfitting compared to the 83.0% and 60.9% from Experiment 1. This discrepancy in training and validation accuracy could be attributed to the pose detection algorithm since the dataset was the only change to the prediction model. Due to yoga being a static sport, its poses show the keypoints more clearly than kickboxing making them easier to detect which could attribute to the difference. As depicted in Figure 5.1 and Figure 5.2. Despite the minor issues the model still performed well showing its adaptability and answering the research question.

Experiment 3 aimed to test the scalability of the model, answering the research question "How does an increase in the number of poses being detected

Figure 5.1: *Image WAR_165 with keypoints and edges drawn*

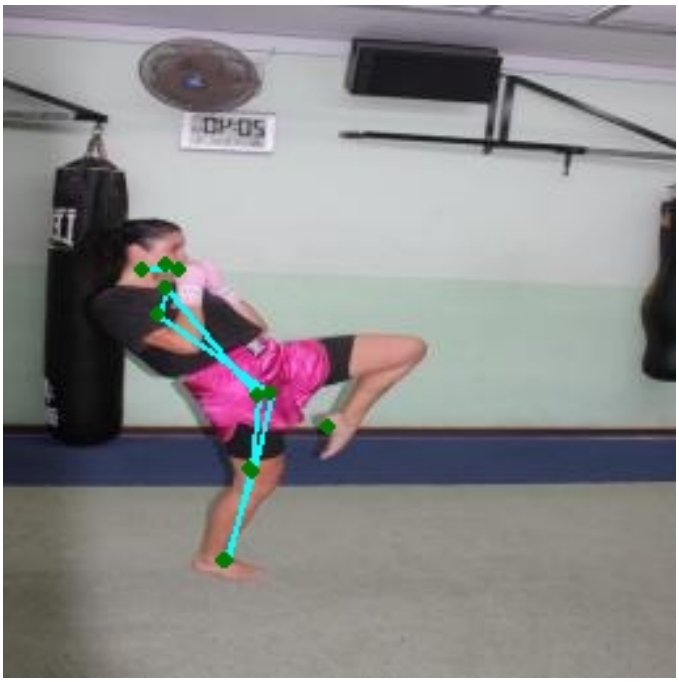

Figure 5.2: *Image KNE_007_1_R with keypoints and edges drawn*

affect the optimal model in terms of accuracy and performance?" This was done by taking the yoga dataset used in Experiment 2 and having 2 additional classes added bringing the total number of classes to 5. These added classes were the Downdog and Goddess poses. The same split ratio of 70:20:10 was used however the image ratio was not kept and therefore the dataset was imbalanced. This happened due to time constraints resulting in having to conduct the Experiment with the dataset as it was.

The results of Experiment 3 depicted the model's performance being lower than that in Experiments 1 and 2. The overall accuracy of the model was 76% and the overall F1 score was 76.0. The individual F1 scores peaked at 82 and reached 72 at its lowest. Similar to Experiment 1, the training and validation accuracies were 80.2 and 67.7 respectively. This difference indicates that the model is encountering overfitting which may have also attributed to the decrease in performance. However, this does not rule out the fact that an imbalance exists in the dataset and that the dataset is larger than the previous datasets. The research question has been partially answered. It is confirmed that the performance was worse than that in Experiments 1 and 2 however it is not known what portion of the decrease is attributed to the overfitting, what portion is attributed to the imbalance and what portion is attributed to the scale increase. For a visual representation comparing the main metrics of all 3 models/experiments refer to Fig: 5.3

Time was the main limiting factor for this research and as a result, it left room for improvement. For future work, the model should be refined to remove the overfitting encountered especially for Experiments 1 and 3. The larger dataset

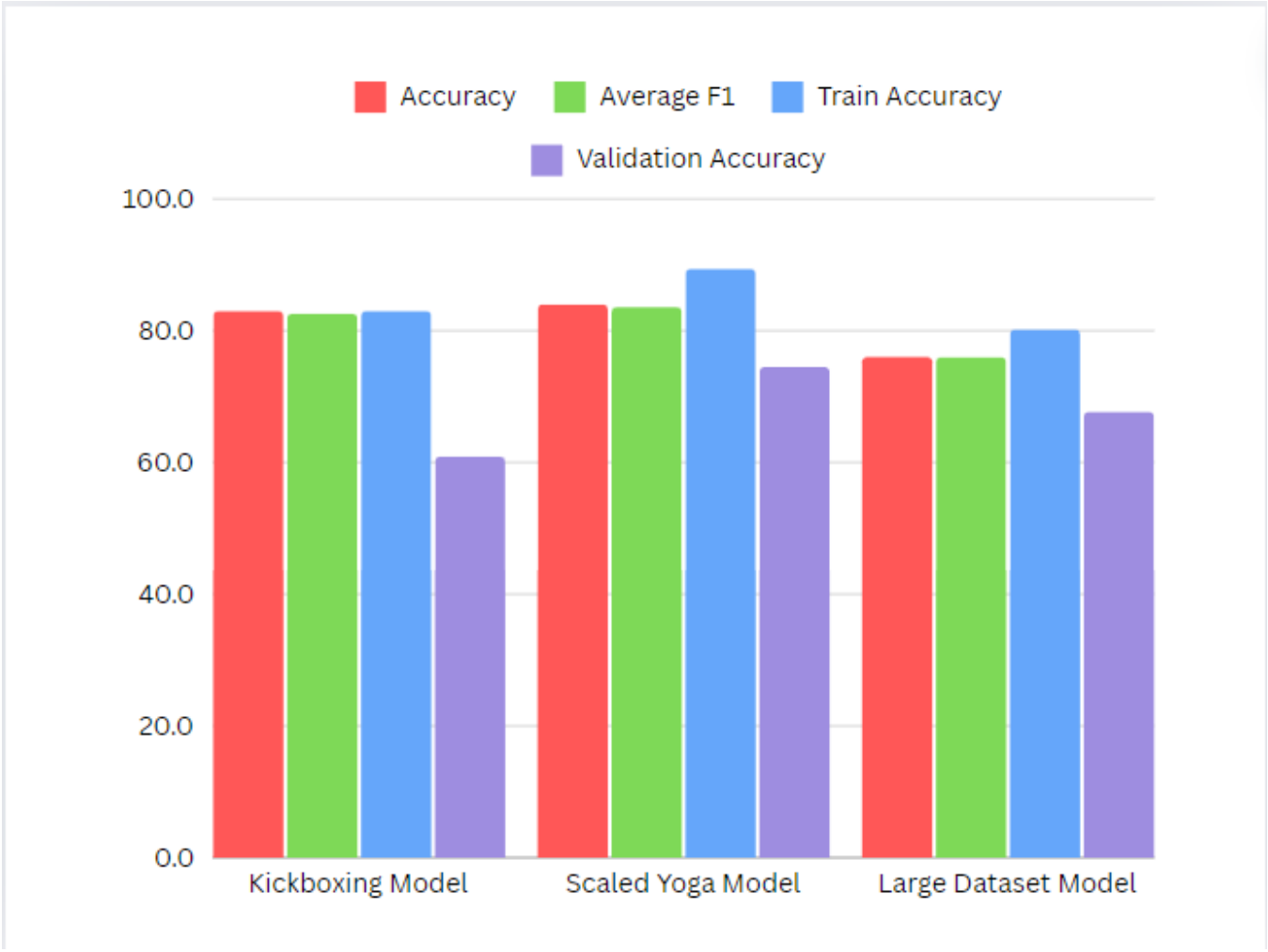

Figure 5.3: *Barchart of performance across all experiments*

used for Experiment 3 needs to be expanded upon to adhere to the image ratio, hence eliminating the imbalance. Code optimisation is necessary to decrease the run time of the code, for small-scale datasets it does not take too long however for large-scale datasets the increased runtime may pose issues. Overall, the hypothesis "Machine Learning Models can be used for accurate pose detection in Kickboxing." has been answered, further research will aid in closing the research gap, given the time constraints and the learning curve involved this research has been a step in the right direction.

List of References

[1] Rutuja Gajbhiye, Snehal Jarag, Pooja Gaikwad, and Shweta Koparde. Ai human pose estimation: Yoga pose detection and correction. *international journal of innovative science and research technology*, 7:1649–1658, 2022.

[2] Fazil Rishan, Binali De Silva, Sasmini Alawathugoda, Shakeel Nijabdeen, Lakmal Rupasinghe, and Chethana Liyanapathirana. Infinity yoga tutor: Yoga posture detection and correction system. In *2020 5th International conference on information technology research (ICITR)*, pages 1–6. IEEE, 2020.

[3] BeomJun Jo and SeongKi Kim. Comparative analysis of openpose, posenet, and movenet models for pose estimation in mobile devices. *Traitement du Signal*, 39(1):119, 2022.

[4] Colin Arrowsmith, David Burns, Thomas Mak, Michael Hardisty, and Cari Whyne. Physiotherapy exercise classification with single-camera pose detection and machine learning. *Sensors*, 23(1):363, 2022.

[5] Kosei Yamao and Ryosuke Kubota. Development of human pose recognition system by using raspberry pi and posenet model. In *2021 20th International Symposium on Communications and Information Technologies (ISCIT)*, pages 41–44. IEEE, 2021.

[6] Leah Nagay. Yoga pose classification with tensorflow's movenet model, Aug 2022.

[7] Utkarsh Bahukhandi and Shikha Gupta. Yoga pose detection and classification using machine learning techniques. *Int Res J Mod Eng Technol Sci*, 3(12):13–15, 2021.

[8] Eriny Wessa, Abdelaziz Ashraf, and Ayman Atia. Can pose classification be used to teach kickboxing? In *2021 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, pages 1–6. IEEE, 2021.

[9] Maybel Chan Thar, Khine Zar Ne Winn, and Nobuo Funabiki. A proposal of yoga pose assessment method using pose detection for self-learning. In *2019 International conference on advanced information technologies (ICAIT)*, pages 137–142. IEEE, 2019.

[10] Varsha Bhosale, Pranjal Nandeshwar, Abhishek Bale, and Janmesh Sankhe. Yoga pose detection and correction using posenet and knn. 05 2022.

[11] Amit Raja Naik. Inside movenet, google's latest pose detection model, May 2021.

[12] Atima Tharatipyakul and Suporn Pongnumkul. Deep learning-based pose estimation in providing feedback for physical movement: a review. 2023.

[13] Prisma 2020 checklist, 2020.

[14] Deepak Kumar and Anurag Sinha. *Yoga pose detection and classification using deep learning*. LAP LAMBERT Academic Publishing London, 2020.

[15] TensorFlow. Real-time human pose estimation in the browser with tensorflow.js, Sep 2018.

[16] Jaehyun Lee, Hyosung Joo, Junglyeon Lee, and Youngjoon Chee. Automatic classification of squat posture using inertial sensors: Deep learning approach. *Sensors*, 20(2):361, 2020.

[17] Martin Friedrich, Thomas Cermak, and Petra Maderbacher. The effect of brochure use versus therapist teaching on patients performing therapeutic exercise and on changes in impairment status. *Physical Therapy*, 76(10):1082–1088, 1996.

[18] Gnana Priya. Action classification for karate dataset using deep learning, Oct 2018.

[19] Gregory Rogez, Philippe Weinzaepfel, and Cordelia Schmid. Lcr-net++: Multi-person 2d and 3d pose detection in natural images. *IEEE transactions on pattern analysis and machine intelligence*, 42(5):1146–1161, 2019.

[20] Sumaira Ghazal and Umar S Khan. Human posture classification using skeleton information. In *2018 international conference on computing, mathematics and engineering technologies (iCoMET)*, pages 1–4. IEEE, 2018.

[21] Sunil Kale, Nipun Kulkarni, Sumit Kumbhkarn, Atharva Khuspe, and Shreyash Kharde. Posture detection and comparison of different physical exercises based on deep learning using media pipe, opencv. *International Journal of Scientific Research in Engineering and Management (IJSREM)*, 2023.

Chapter A:

HIG_002_8_L	nose:{'x': 220; 'y': 121; 'score': 0.45829568429627726}	left eye:{'x': 130; 'y': 122; 'score': 0.23640388677536736}	right eye:{'x': 219; 'y': 119; 'score': 0.3447963791605671 6}	left ear:{'x': 131; 'y': 122; 'score': 0.5803951430923541}
HIG_009_15_R	nose:{'x': 148; 'y': 120; 'score': 0.2875145537695163}	left eye:{'x': 132; 'y': 111; 'score': 0.2875581364895796}	right eye:{'x': 150; 'y': 117; 'score': 0.3264874789762051}	left ear:{'x': 134; 'y': 111; 'score': 0.284320289554549}
HIG_004_12_L	nose:{'x': 182; 'y': 124; 'score': 0.40444884628094585}	left eye:{'x': 184; 'y': 122; 'score': 0.3620094400454749}	right eye:{'x': 182; 'y': 123; 'score': 0.32266542635907}	left ear:{'x': 129; 'y': 160; 'score': 0.20768805039967148}
HIG_007_13_L	nose:{'x': 233; 'y': 149; 'score': 0.046545338843262 82}	left eye:{'x': 45; 'y': 69; 'score': 0.0669096509521953 9}	right eye:{'x': 40; 'y': 72; 'score': 0.03338741160698570 4}	left ear:{'x': 135; 'y': 156; 'score': 0.06269441858026238}

Figure A.1: *Sample of the Keypoint data as CSV*

Headguard worn	Gloves worn	Shin guards worn	Background type
No	No	No	Normal
No	No	No	Cluttered
No	No	Yes	Normal
No	No	Yes	Cluttered
No	Yes	No	Normal
No	Yes	No	Cluttered
No	Yes	Yes	Normal
No	Yes	Yes	Cluttered
Yes	No	No	Normal
Yes	No	No	Cluttered
Yes	No	Yes	Normal
Yes	No	Yes	Cluttered
Yes	Yes	No	Normal
Yes	Yes	No	Cluttered
Yes	Yes	Yes	Normal
Yes	Yes	Yes	Cluttered

Table A.1: *Table of all combinations of variables*

Nose	
Left eye	Right eye
Left ear	Right ear
Left shoulder	Right shoulder
Left elbow	Right elbow
Left wrist	Right wrist
Left hip	Right hip
Left knee	Right knee
Left ankle	Right ankle

Table A.2: *Table of Keypoints declared in the pose detection algorithm*

Keypoint A	Keypoint B
nose	left eye
nose	right eye
nose	left ear
nose	right ear
left ear	left eye
right ear	right eye
left eye	right eye
left shoulder	right shoulder
left shoulder	left elbow
left shoulder	left hip
right shoulder	right elbow
right shoulder	right hip
left elbow	left wrist
right elbow	right wrist
left hip	right hip
left hip	left knee
right hip	right knee
left knee	left ankle
right knee	right ankle

Table A.3: *Table of Keypoint relationships to create edges*