Real-Time Analysis of Squat Form Using Computer Vision and Pose Estimation Techniques

William Sakyi

Summary

The squat is a fundamental exercise performed daily that requires precise technique to avoid injuries. Traditional methods for learning correct squat form are often costly or lack real-time feedback, limiting their effectiveness and accessibility. This study addresses these limitations by developing an innovative real-time analysis system using pose estimation and machine learning technologies. The system employs the MediaPipe BlazePose model to capture and analyse squat motions frame-by-frame, extracting landmark data that is then evaluated by a Squat Classification Model (SCM). Based on a 1D convolutional neural network, the SCM assesses squat quality using three critical metrics: the optimal knee angle between 55° and 65°, correct gaze direction (forwards or upwards), and spine neutrality. Trained on the synthetic InfiniteForm dataset, the SCM achieved an overall accuracy of approximately 90% after 50 epochs, with the depth measurement showing particularly strong performance (average metric of 89.9%). However, the model was less effective at recognising incorrect gaze and spine alignments, with average metrics of 63.1% and 55.9%, respectively. Real-world testing confirmed these findings, highlighting the model's robustness in depth classification but indicating areas for improvement in gaze and spine analysis. Future research will refine the SCM by utilising a more suitable dataset and extending the model's application to other exercises and sports, potentially enhancing training techniques and injury prevention strategies across various physical activities.

Supervised by Dr M Valero

Department of Mechanical Engineering

University of Bristol

2024

Declaration

Contents

1 Introduction

1.1 Background Information

The squat is one of the most fundamental and beneficial exercises. It is also a common movement carried out daily (e.g. picking a box off the floor). It engages all the muscles in the lower extremities and the posterior chain [1]. However, it is also a very technical exercise, requiring correct form to minimise the risk of injury and increase the amount of muscle activation. A popular way to learn the proper form is to hire a personal trainer (PT) who would be experienced and qualified to teach correct form. With the presence of a PT, a trainee can be assured that they will learn the correct form and technique while receiving immediate feedback. However, one key drawback is the price of a PT. The average cost of a PT for a 60-minute session in the South West of England is £45-£90 [2]. The high price of hiring a personal trainer has given rise to self-taught methods of learning exercise techniques. Especially during the COVID-19 pandemic, people have turned to online means such as YouTube to learn essential exercise techniques [3]. YouTube channels such as Squat University which is run by Dr Aaron Horschig, who has a doctorate in physical therapy [4] can provide safe and reliable information. However, using such methods again has its drawbacks. The lack of real-time feedback drastically increases learning time and the risk of injury. The most effective way to learn correct technique is to first be shown the exercise and then attempt it while receiving real-time feedback for safety and efficiency. There are various versions of the squat (barbell back and front squats, bodyweight squats); however, in this paper, bodyweight squats will be the focus.

1.2 Project Aim and Objectives

This project aims to use sensor technology to develop a cheap, easy and accessible way for a person to speed up the squat form learning process. To achieve this aim, a set of objectives have been devised:

1. **Define metrics to quantitively assess a person's squat form.** This will allow any analysis method to have accurate metrics to judge against.
2. **Conduct a review of current sensor technology.** This will determine the current state of the art and allow for the selection of suitable sensor technology.
3. **Develop a system for real-time analysis of a user's squat that is available to anyone.** The squat is a universal movement, and anyone should be able to learn how to perform it. The system should be suitable for all body types and be simple to use with a short learning curve. The squat analysis should be real-time to allow users to gain instant feedback to decrease learning time.

2 Literature Review

2.1 Squat Mechanics

To develop a solution to assess squat form, the aspects of a good squat needed to be first established. There are multiple components of a squat that contribute to its overall quality. Suchomel, Comfort and McMahon, in their paper [1] have stated key findings on what parts of a squat form affect muscle activation and safety. Firstly, a squat depth where the knee angle is between 55° and 65° (measured as the angle between the thigh and shin) results in the maximum amount of muscle activation in the quads and glutes. This was the case as long as a neutral spine was maintained throughout the movement. A neutral spine is defined as the zone where the passive spinal column offers minimal resistance [5]. A neutral spine is not necessarily a completely straight spine but tends to have the natural curves of the vertebral column. A neutral spine position ensures safety during movement as extra pressure is not applied to the back or neck. It was found that foot orientation and stance width (if greater than 75% shoulder width) had no bearing on the squat quality apart from the fact that they should be in a comfortable position for the trainee. A larger stance width, for example, may allow a person with longer femurs to squat deeper. Knee movement should be unrestricted (i.e. heels should be kept on the ground). Also, the trainee's gaze should be forward or upward but never downward. There is a strong correlation between hip flexion and gaze [6], a forward or upward gaze minimises undesirable flexion. Finally, the squatter's knees should track over their toes without caving inwards. To help those learning a squat, PTs often prompt exercisers with physical cues to help an individual complete it optimally. For example, one popular cue for a squat is to brace your core.

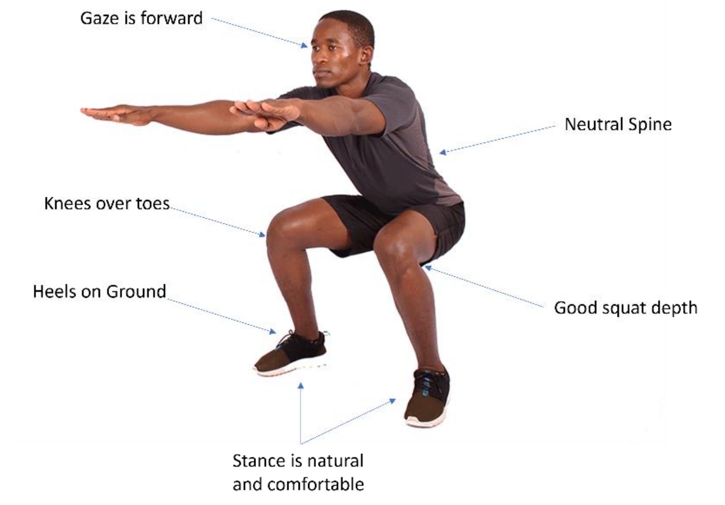


Figure 1 - Features of an optimal squat [7].

For the remainder of this report, the five aspects that define a good squat are optimal depth, neutral spine, forward or upward gaze, heels on the ground and knees over toes. These features are shown in Figure 1.

2.2 Sensor Technology

There are various sensors that are used to monitor physical training. This includes some general sensors like smartwatches and fitness trackers that are already widely available. These do not provide accurate data for specific exercises like the squat but are very useful for general health tracking as they can monitor heart rate, calories burnt and steps taken [8]. There are also research and professional-grade sensors that provide much more accurate and detailed information and, in tandem with an algorithm, could provide information on squat technique. These sensors include Inertial Measurement Units (IMUs), Surface electromyography (sEMG) sensors, insole pressure sensors, Electrocardiogram (ECG) chest bands [9] and computer vision. sEMGs focus on measuring muscle activity, while ECG chest bands record cardiac output. Without a large amount of interpolation, these sensors cannot provide beneficial information for analysing squat form and, therefore, will not be considered. IMUs and pose estimation can provide positional data about a user's entire body, which can be analysed to determine squat quality.

2.2.1 Inertial Measurement Unit (IMU)

An Inertial Measurement Unit (IMU) is a device that measures a body's acceleration, angular velocity, and rotation angle. It can measure up to 9 degrees of freedom using a combination of a gyroscope, accelerometer and magnetometer.

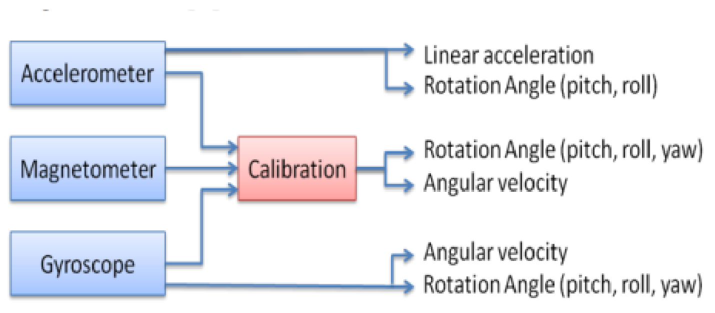


Figure 2 - An IMU based on three types of sensors [10].

Figure 2 shows an IMU based on three types of sensors and how the data received from each sensor combines to provide accurate output data. This output data is then fused together using various filter algorithms, such as the Kalman Filter [11]. This allows for orientation, position and velocity data to be extracted and used.

In the context of squat form classification, IMUs have been used before to classify squats. The approach was done by Lee et al. in a 2020 paper [12] where they placed 5 IMUs on a participant (one on each thigh and calf and one on the lumbar region) and recorded sensor data while the participants performed a squat. After, the data was used as an input for a Deep Learning (DL) model and a Random Forest Machine Learning (ML) model to evaluate the squat. It was found that the DL and ML models had an accuracy of 91.7% and 75.4%, respectively.

IMUs are well-established technology dating back to the 1930s [10] and therefore there is a lot of research behind it. However, uptake by health professionals is low and this is largely because IMUs have been developed with mainly researchers in mind without involving the opinion of clinical end-users [13]. For example, the usability of IMUs have been neglected. There is not much customisation and interacting with the data it provides is complicated with further processing being required. A sole exerciser trying to improve their squat form on their own would not be able to use a set of IMUs without difficulties. Also, IMUs are high-priced; the one used in the 2020 paper cost about £280 per IMU [14].

2.2.2 Computer Vision and Pose Estimation

Computer vision is a field of computer science and AI that attempts to enable computers to identify and understand objects and people in images and videos [15]. Human Pose Estimation (HPE), a sub-field of computer vision, is the task of predicting the joints of a human (called keypoints or landmarks) from an input image or video. To maintain consistency, the most common keypoint output is the 17-point skeleton developed for the COCO dataset. Common Objects in Context (COCO) [16] is a large-scale object detection, segmentation and captioning dataset developed by Microsoft. COCO also includes HPE data, so using a consistent skeleton allows for synergy in the pose estimation research field. The output from pose estimation models can be further processed for various applications, including video surveillance, medical assistance and sports motion analysis [17]. In the context of squat form analysis, a pose estimation model can be used to extract keypoints data from a video or image, which can then be fed into a classification model to determine the quality of the squat.

Pose detection has been used in health and fitness before. Both in yoga to detect yoga poses [18,19] and in sport and physical exercise [20–22]. It has also been used for measuring gait kinematics [23]. In a 2022 paper by Youssef et al. [24] machine learning was used to analyse the squat exercise. This was done by using a pose estimation model (namely BlazePose [25]) to extract keypoints data for an entire squat repetition. For each frame of the rep, they computed a 33×33 difference matrix that contained all the distances between keypoints. The difference matrix for each frame of the squat rep was then combined. The set of 33×33 matrices were then fed into a convolutional neural network. The model achieved a 96% accuracy when classifying a squat. The report mentions the possibility for use in real-time, however, since the full squat repetition needed to be completed before the squat could be analysed, this meant that although, feedback was fast, it cannot be live. A user, therefore, cannot adjust their squat live but must wait until they complete the full repetition before they can adjust. This limits a person's ability to determine which part of their squat is faulty.

Pose estimation is the method that will be used for squat form analysis in this report. Using pose estimation aligns highly with the project's aim to develop a cheap, easy-to-use method for anyone to assess their squat. All that is required from the end-user is a mobile phone camera, which 87% of UK adults have [26]. The analysis method that will be used will be a frame-by-frame method, where each individual frame will be considered a data point, and the model or algorithm will take the landmark data from a single frame as its input. This will allow for near-instant feedback to the user. It will also simplify any models developed, as the temporal domain will not have to be considered.

2.2.2.1 Comparing Different Pose Estimation Models

Various pose estimation models have been developed. Selecting the correct one for this use case was important. To select a model, various metrics were considered and a Pugh matrix was used to decide. The metrics used are shown in Table 1. A model with 3D output dimensions could be useful as any further model or algorithm may be able to pick up patterns that are only noticeable in 3 dimensions. A fast frame rate would improve the performance of the model. One of the project objectives is to have real-time analysis; a fast model will allow for this. The average percentage of detected joints (PDJ) information was obtained from a 2022 paper by Chung, Ong and Loew, who completed a comparative study of skeleton-based human pose estimation [17]. In this study, they compared OpenPose [27], PoseNet [28], MediaPipe BlazePose [25] and MoveNet [29] models. The release year of the model was also considered, as machine learning models tend to improve with time.

Training a complex model can take a substantial amount of time; therefore, those released more recently will have had more time to train and be of better quality.

Table 1 - Key performance metrics of various pose estimation models.

Metrics	**Metric Description**	**OpenPose**	**PoseNet**	**MediaPipe BlazePose**	**MoveNet Lightning**
Output Dimensions	The output type of the keypoints data	2D [27]	2D [28]	3D [25]	2D [29]
Performance/Speed (fps)	The framerate at which the model runs on similarly powered machines	34 [30]	51 [31]	92 [32]	104 [33]
Average Percentage of Detected Joints (PDJ) (%) [17]	In the referenced paper, the percentage of joints that were detected in a series of tests	37.18	65.95	68.47	69.85
Maximum Number of Keypoints	How many keypoints the model can detect	135 [27]	17 [28]	33 [25]	17 [29]
Release Year	The year the model was released	2017 [27]	2017 [28]	2020 [25]	2021 [29]

Table 2 - A Pugh matrix comparing the HPE models based on key metrics.

Criteria	**PoseNet (Datum)**	**OpenPose**	**BlazePose**	**MoveNet Lightning**
Output Dimensions	0	0	1	0
Performance	0	-1	1	1
Average PDJ	0	-1	1	1
Number of Keypoints	0	1	1	0
Release Year	0	0	1	1
Total	**0**	**-1**	**5**	**3**

Table 2 shows the selection matrix that was used to decide the best HPE model. The comparison metrics defined in Table 1 were compared to quantitatively determine the best model. MediaPipe BlazePose was the best and was used due to its 3D output, relatively high performance, and greater number of keypoints. It was also one of the more recent models.

2.2.2.2 MediaPipe BlazePose

MediaPipe BlazePose is a lightweight convolutional neural network (CNN) architecture for human pose estimation that was developed by Google Research [25]. BlazePose runs a top-down pipeline. This means the model identifies each person instance within a frame and bounds them in a box before identifying the keypoints for each person instance. This is opposed to a bottom-up approach where all keypoints in an image are first found and then grouped into the different person instances of that image. BlazePose works by taking an image or video (array of images) as an input and returning a set of keypoints in the shape (33,4) for each image.

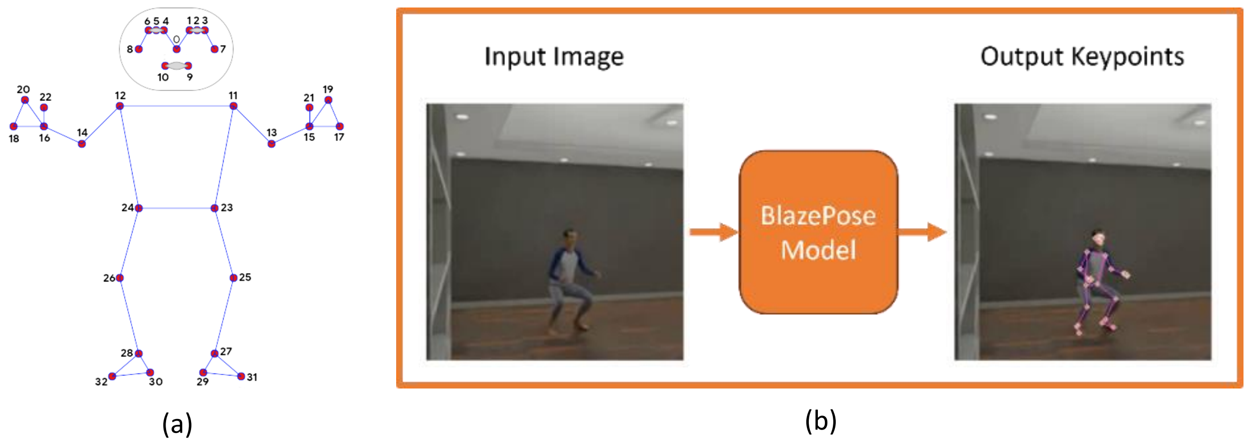


Figure 3 - BlazePose skeleton with all the identifying landmarks (a) and an example output of the BlazePose model (b).

Figure 3a shows the 33 landmarks that the model outputs. Each landmark has 4 features. These are the normalised x, y, z coordinates, which have an origin at the centre of the hips, and a visibility (v) confidence score. An example of the BlazePose model's output is shown in Figure 3b. The 33 landmarks the BlazePose model uses are a superset [25] of BlazeFace [34], BlazePalm [35] and the 17-point COCO skeleton.

3 Method

A method to classify squats was developed using a ML pipeline that classified squats based on the flaws they contained. The model is described as follows: the BlazePose model will take an input frame, it will then output normalised landmark data which will then be fed into a Squat Classification Model (SCM) that will classify the squats.

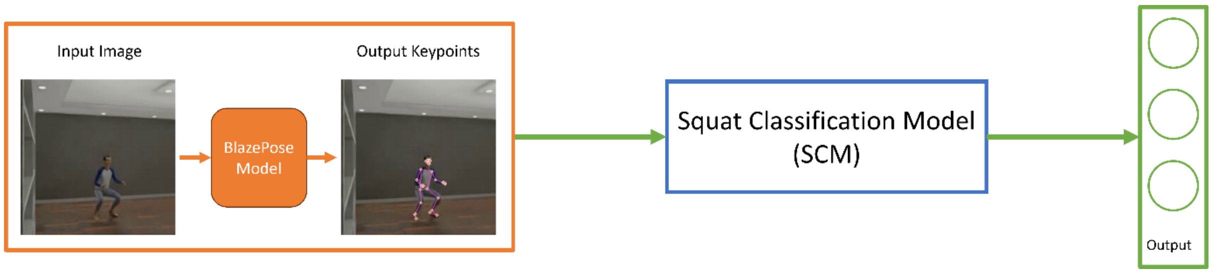


Figure 4 - Overview of the machine learning pipeline.

An overview of the proposed pipeline is presented in Figure 4. The model returns a 1 or 0 for each label (squat metric category) depending on whether it has detected an errancy with that part of the squat.

3.1 Machine Learning Platform

The machine learning platform that was used was TensorFlow (TF). TensorFlow is an end-to-end open-source machine learning platform developed by Google. TF was chosen above Pytorch and Scikit Learn due to its simplicity and the option to use the high-level Keras API, making it easy for a beginner to create ML models. MediaPipe BlazePose was also developed by Google, which meant that using TF would provide a seamless development experience.

3.2 Dataset

To train the model, a dataset of squats was needed so that the model could learn how to classify squatting errors. The dataset selected was InfiniteForm [36] created by InfinityAI. InfiniteForm is a synthetic, minimal-bias dataset for fitness applications. It is a dataset created using Blender and a physics-based rendering engine with raytracing to emulate photo-realism. This dataset was used as it was large and well annotated, making it suitable for training. It also introduced a large amount of body shape variance that could be difficult to recreate through a self-created dataset. An example of the dataset is shown in Figure 5. The entire dataset contained other exercise movements however, only the bodyweight squat dataset was used for the SCM.

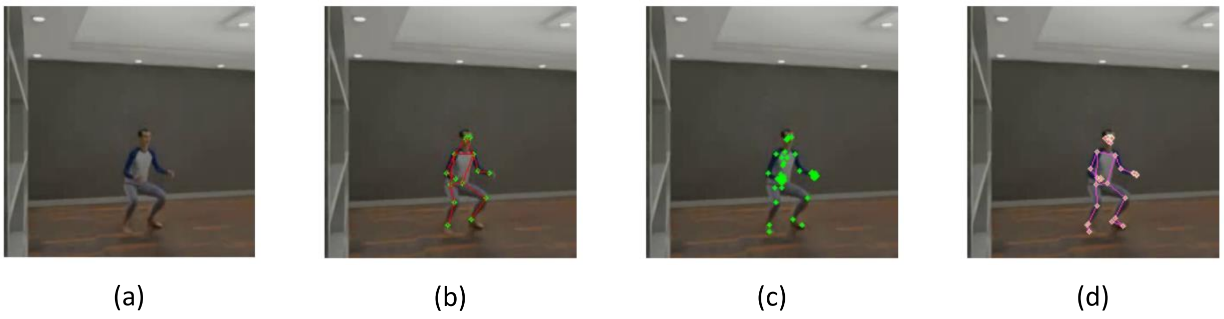


Figure 5 – An example squat dataset without any annotations (a) with the COCO keypoints annotations (b), armature keypoints annotations (c) and BlazePose landmarks (d).

The Fitness Basic Squat Dataset [37] contained 100 separate videos of various person models completing a squat. Factors such as location, lighting conditions and camera angle varied between videos. Within each video, there were 5-10 reps each. The dataset contained variance between each rep to the point where no two squats were the same, emulating the same variance an actual person would have when performing an exercise. Alongside each video, the dataset included a .JSON file that contained a vast amount of annotation data. This annotation data included 2D 17-point COCO skeleton data (Figure 5b) and 3D 55-point armature keypoints data (Figure 5c). To maintain consistency between training and deployment of the SCM, the BlazePose model was run on the entire squat dataset, and that data was used instead of the already provided annotations for training, validation and deployment.

3.2.1 Labelling the Dataset

The SCM worked by classifying each frame according to errors within the squat. The InfiniteForm dataset contained a total of 38,692 different frames. Labelling the data manually frame by frame would have been too time-consuming; therefore, several scripts in tandem with the annotations provided by the InfiniteForm dataset were used to label the data. The classification categories by which the squats would be classified were selected. These were selected from the 5 aspects of a good squat defined in section 2.1, and they were selected based on whether it was possible to label them using scripts and the annotation data. The depth of the squat, the gaze direction and the neutrality of the spine were the chosen metrics. As the squat analysis was defined as a classification problem, binary labels were used to label the dataset.

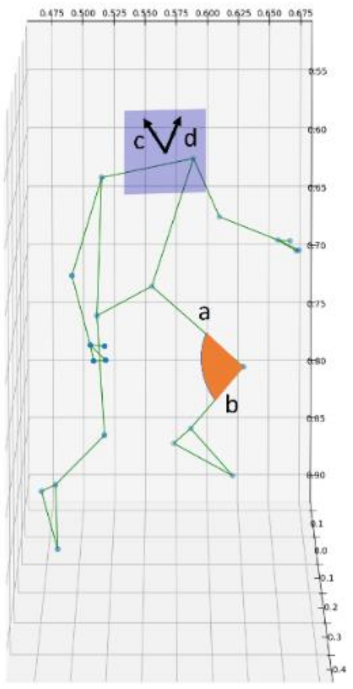


Figure 6 - BlazePose skeleton in 3D space with knee angle and face plane visible.

3.2.1.1 Squat Depth

The optimal squat depth requires the angle between the shin and thigh to be between 55° and 65°. Using the 3D BlazePose keypoints data and the dot product rule, knee angle can be calculated for each frame. The dot product is shown in Equation 1:

$$a \cdot b = |a||b|cos(\theta) \tag{1}$$

where a and b are direction vectors and θ is the angle between the direction vectors. In the context of knee flexion angle, a and b represent the thigh and shin bone, respectively, as shown in Figure 6.

The average angle of both knees was found, and the frame was labelled with a 1 if the squat was at the correct depth or a 0 if the squat was not deep enough.

3.2.1.2 Gaze Direction

The gaze of a squatter throughout the movement should be forward or upwards. To determine this, the angle between a face plane and the vertical was calculated. A face plane was created with 3 points on the face (the 2 eyes and nose) as shown in Figure 6. The angle is then calculated as:

$$sin(\theta) = \frac{n \cdot v}{|n||v|} \tag{2}$$

where $v = (0, -1,0)$ is the vector direction of vertical and n is the normal to the face plane. n is found as the cross product of the direction vectors:

$$n = c \times \mathrm{d} \tag{3}$$

where c and d are direction vectors to each respective eye from the nose as shown in Figure 6.

The gaze angle for each frame was then found. If the gaze was down and the angle was negative the frame was labelled with a 1. Otherwise, the gaze was deemed ok and labelled with a 0.

3.2.1.3 Spine Neutrality

A squatter's spine should stay neutral throughout the entire movement to maintain a rigid trunk, preventing injury and increasing a person's load capacity. The armature keypoints (Figure 5c) contain position data for 3 points on the spine (top, middle, bottom). Linear regression was used to determine the "straightness" of the back during squatting. This was done by evaluating the R^2 factor from linear regression of the 3 spine points. After evaluating the data and results of this linear regression, it was deemed that an R^2 factor of 0.4 was a sensible limit to deem a user's spine not neutral. This value was selected after analysing a small sample of squats. Examining by eye showed that the squats that had an R^2 value greater than 0.4 were distinctly not neutral.

3.2.1.4 Labelling Results

After running the BlazePose model, it was found that 3,377 frames were not useable as the model could not generate keypoints due to low visibility. This meant that the total number of frames used for training was 35,315. 8.75% of the frames had a depth in the correct range, 2.68% had a bad gaze direction and 0.23% had a spine that wasn't neutral. This is visualised in Figure 7 and it shows that the data is heavily skewed.

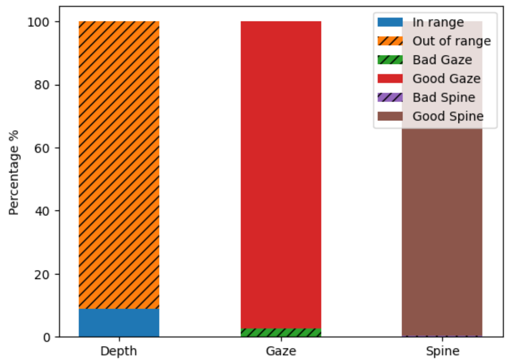


Figure 7 - Visual representation of the labelling results.

3.2.2 Preprocessing

The dataset currently consists of 2 large arrays of shapes (35315,33,4) and (35315,4) containing the landmark data and label data, respectively. Before the dataset could be used for training and validation, pre-processing was required. To be able to use the data in TF, a TensorFlow Dataset object was created. This combined the features (landmark data) and labels into a single TF Dataset. The dataset was split into training data and validation data via an 80%/20% split. Then, the dataset was placed into batches of 32. A batch size of 32 was selected as it is the most commonly used batch size in ML applications [38]. Batching helps manage memory usage and provides computational efficiency.

Model weights were applied to each category to compensate for the class imbalance of the data. To apply the weights, the ratio between the binary labels for each category was found. The model was then penalised by a factor that was inversely proportional to that ratio. For example, for the depth category, if the model incorrectly predicts a squat that is in range, it will be penalised by a factor 10.4 times greater than if it incorrectly predicts a squat that is out of range. This forces the model to pay more attention to patterns amongst minority classes to avoid being heavily penalised, resulting in a better-trained model.

3.3 Squat Classification Model (SCM)

The aim of the Squat Classification Model (SCM) was to determine the relationship between input landmark data and their classification. Due to the nature and complexity of the problem, a neural network was used. Neural networks are a series of algorithms designed to recognise relationships between input data and output classification [39]. Neural networks can vary vastly and can solve almost any data-driven problem. The layers suitable for the SCM specifically, needed to be determined. It was found that the model would consist of 2 convolutional 1D layers, a flatten layer to flatten all the parameters into a single dimension, and a dense layer which was connected to the output layer. This was a multi-label classification problem, which meant that the SCMs 3 output labels (depth, spine, gaze) are not mutually exclusive, and a frame can contain any combination of them.

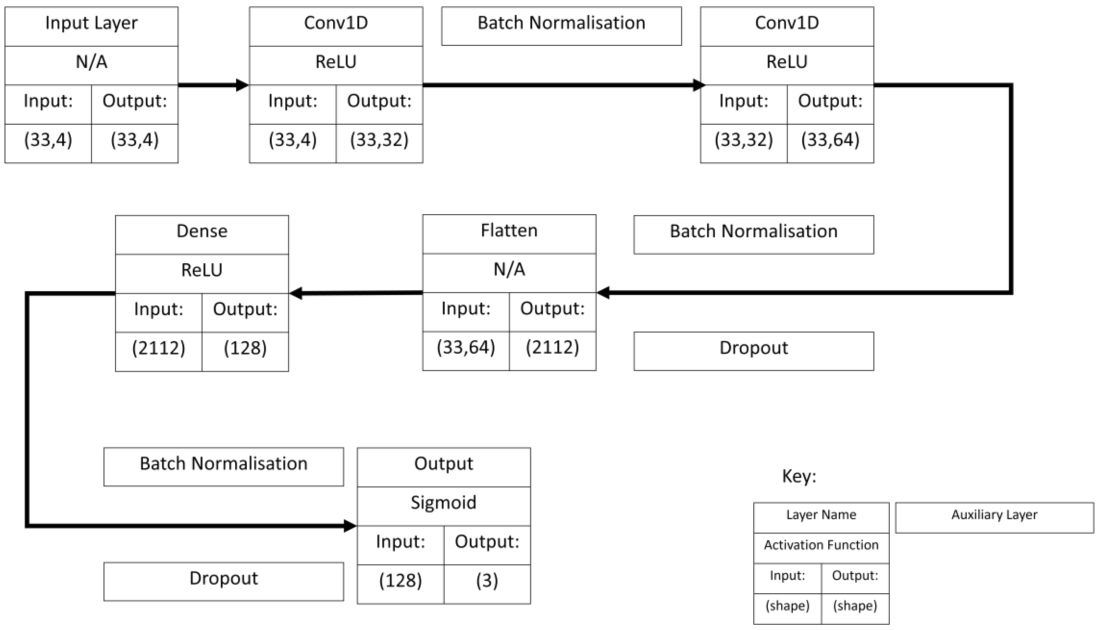


Figure 8 – A flow chart summary of the Squat Classification Model.

Figure 8 shows all the layers in the SCM and how they flow together to create the model. Each block contains a description of the layer's name, its activation function and the input and output shapes of the data it receives. The flow chart also shows the auxiliary layers of the model.

3.3.1 Layers

3.3.1.1 Input Layer

This layer takes the input data and feeds it into the model. The input of this layer has the shape (32,33,4), representing the batch size, the 33 landmarks, and the 4 features (x, y, z, v) for each of those landmarks.

3.3.1.2 Convolutional 1D (Conv1D) layers

The SCM has 2 1D Convolutional Neural Network (CNN) layers. 1D CNNs are a type of neural network that is used for processing sequential data and can detect patterns within them. A CNN has a kernel which "slides" across the data performing the convolution operation. The convolution of two discrete functions is defined as:

$$(f * g)[n] = \sum_{k=-\infty}^{\infty} f[k]\, g[n-k] \tag{4}$$

where n is the sequence index and k is the summation index which shifts g across f. In the context of the SCM, f is the landmark data while g is the kernel matrix.

The kernel is a separate matrix whose size is defined during model creation. The convolution of the data and the kernel allows the layer to detect patterns. For each Conv1D layer, the number of filters is also defined. This specifies the different number of kernels that should be created. Various kernels are created to pick up the different patterns within the data.

Compared to 2D CNNs, which are used to process data with higher dimensions (images or videos), 1D CNNs are much less computationally intensive. For an image with $N \times N$ dimensions and a $K \times K$ kernel size, it would have a computational complexity $\sim O(N^2K^2)$ whereas a 1D equivalent would have a complexity of $\sim O(NK)$ [40]. Therefore, in situations that do not require a 2D CNN, a 1D should be used for better performance.

In the context of the SCM, a 1D CNN was used due to the format of the data. The set of keypoints were sequential in nature. Figure 9 visualises the first Conv1D layer of the SCM. It shows how a kernel would convolve over it and pick up patterns. A kernel size of 3 was selected and Figure 9 shows what that means for the convolution operation in the first layer. A pattern that a specific kernel may recognise, for example, is the relationship between the hip, knee and ankle landmarks and the squat depth label.

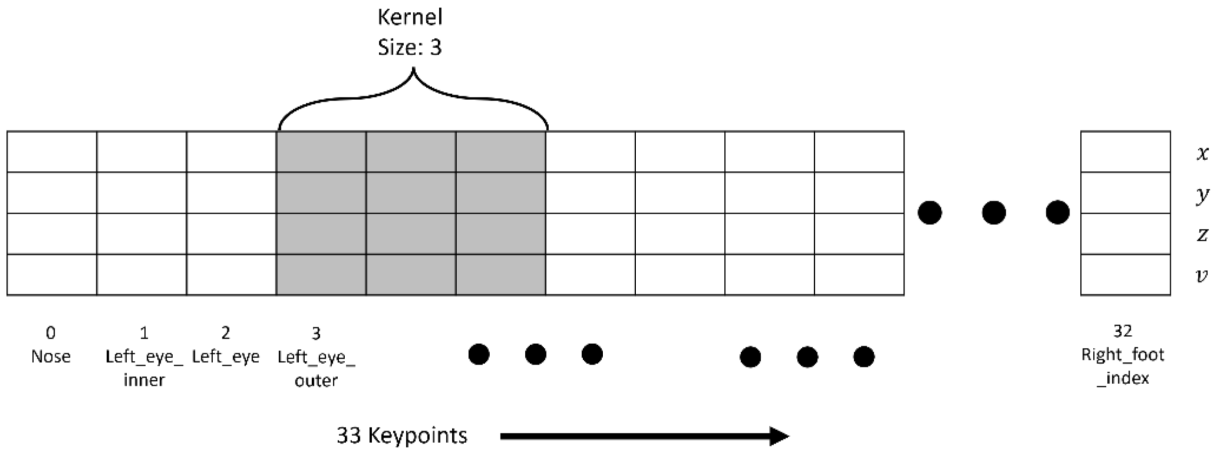


Figure 9 – A visualisation of the first Conv1D layer of the SCM.

3.3.1.3 Dense Layer

This is the most basic neural network layer, where every input neuron is connected to each output neuron. Each dense layer neuron contains a vector of weights corresponding to each input neuron that is connected to it. Weights determine the connection strength of each input. Each neuron has a bias value that shifts the neuron's output, enabling finer adjustments to the overall function of the network. A dense layer calculates the output of each output neuron via Equation 5:

$$output = activation\big((w \cdot x) + b\big) \tag{5}$$

where x is the input vector to the neuron, w represents the weights vector, b is the bias. $activation$ is a function that introduces nonlinearity to the model (see section 3.3.2).

A dense layer was selected for the SCM as it works as a bridge between the convolutional layers and the output layers. The dense layer finds relationships between the patterns found by the convolutional layers and the outputs. Figure 10 shows the dense layer configuration of the SCM. It shows the nodes after the data has been flattened and how it is connected to the 128 nodes of the dense layer which are then connected to the output layer.

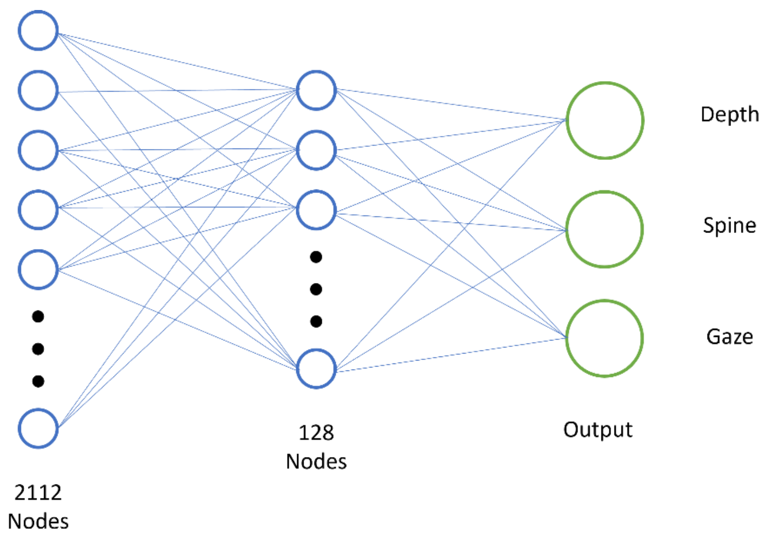


Figure 10 – The flatten, dense and output layers of the SCM.

3.3.1.4 Output Layer

For the SCM, the output layer contains 3 neurons, one for each squat classification category: depth, spine, and gaze. Each of them are activated independently by a sigmoid function (see section 3.3.2.1) and can each return a value between 0 and 1. Figure 10 shows how the output layer is connected to the dense layer.

3.3.1.5 Auxiliary Layers: Batch Normalisation, Dropout and Flatten

Auxiliary layers do not perform any primary computations essential to the model's function but rather support the core layers to improve the model's performance.

Batch normalisation layers are placed after each convolutional layer and before the dense layer to normalise the outputs from the previous layer, aiming to maintain a mean output close to 0. Since each of these layers include activation functions that introduce non-linearity to the model, internal covariate shift [41] can occur. This shift refers to the changes in the distribution of layer inputs as the network's weights and biases are updated during training. As the model learns, these updates can cause unpredictable shifts in the data each layer processes. By reducing this shift, batch normalisation helps stabilise and accelerate training, making the model more efficient.

Dropout layers help prevent overfitting. Overfitting is when a model adapts so much to the training data that it can get an effectively perfect accuracy during training, but when it is presented with new data it hasn't seen before, it cannot adapt and has a much poorer accuracy. Dropout works by randomly choosing neurons to omit during the training process, meaning that these neurons do not contribute to the model's output [42]. During testing, however, the entire network is used, and neurons' outputs are scaled down based on whether they were dropped out during training. In this model, a dropout probability of 50% was used. 50% was selected as this would result in the highest amount of regularisation [43].

Flatten layers "flatten" multidimensional outputs from the previous layer into a single 1-dimensional vector. In the SCM, the dense layer requires a 1-dimensional input so the flatten layer reshapes the data from (33,64) to (2112). This allows for a proper connection between the convolutional and dense layers.

3.3.2 Activation Functions

Activation functions are mathematical equations that determine the output of a neural network neuron. This is done by applying the desired function to the weights and bias calculations of each neuron (see Equation 5). The primary purpose of activation functions is to introduce non-linearity, which allows the model to learn non-linear relationships. Activation functions can also normalise outputs into a desirable range of values. There are multiple types of activation functions. The two activation functions used in the SCM are the sigmoid function and the Rectified Linear Unit (ReLU) function. These functions are shown in Figure 11.

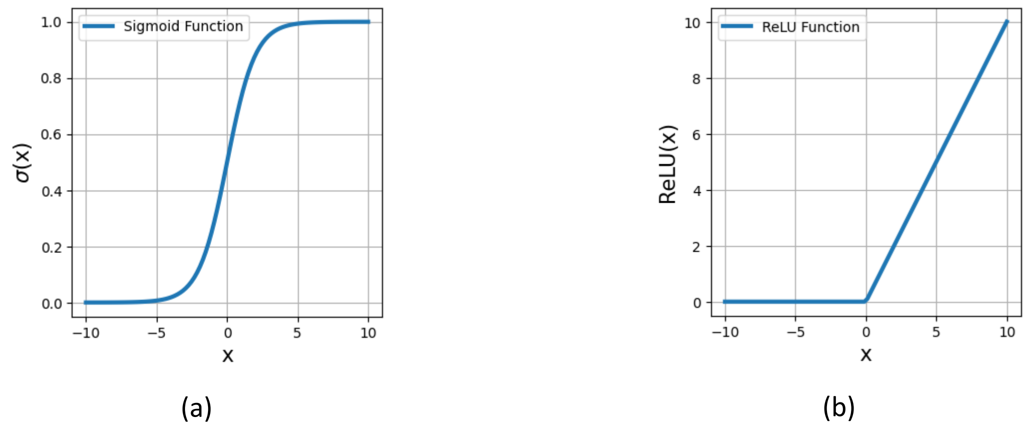


Figure 11 - Graphical representation of the Sigmoid function (a) and ReLU function (b).

.

3.3.2.1 Sigmoid Function

The sigmoid function maps any input value to a value between 0 and 1. This means that sigmoid functions are used for binary problems. The sigmoid function is defined as:

$$f(x) = \frac{1}{(1 + e^{-x})} \tag{6}$$

This output is shown in Figure 11a. There are, however, some limitations to sigmoid functions, for very large or very small input values the output becomes very close to 1 or 0 and therefore loses its sensitivity to small changes in input. Also, the sigmoid function suffers from the vanishing gradient problem [44] at those values. The vanishing gradient problem arises in activation functions with a gradient that tends to 0 at very large or very small inputs. Due to the way a model back-propagates, this problem causes a model to learn slowly. In the context of the SCM, a sigmoid function will be used for each output neuron to classify the squat in a binary fashion.

3.3.2.2 Rectified Linear Unit (ReLU) Function

In the SCM, the ReLU activation function is used for both Conv1D layers and the dense layer, as it is a relatively simple but effective function. It overcomes the vanishing gradient problem mentioned previously. When the function is activated ($x > 0$) the gradient is always equal to 1 and never vanishes, allowing for faster convergence. The ReLU function introduces non-linearity into the network, allowing the SCM to learn non-linear relationships between the landmark data and the outputs. The ReLU function is shown in Figure 11b and is defined as:

$$f(x) = max(0, x) \tag{7}$$

3.3.3 Machine Learning, Loss and Back Propagation

There are various steps a model takes to learn effectively. These include a forward pass, loss function calculation, backpropagation, and then optimisation. This is iterated over many epochs (iterations), decreasing the loss value with each epoch until convergence.

3.3.3.1 Forward Pass

During the forward pass stage, the input data is fed into the model and the model computes an output based on the current weights and biases. This output is the prediction for that epoch.

3.3.3.2 Loss Function Calculation

The loss function calculates the error of a model prediction. The error is how far off a prediction is from the ground truth, this is shown mathematically as:

$$Loss = abs\left(Y_{pred} - Y_{actual}\right) \tag{8}$$

As the model trains it aims to minimise the value of the loss function. There are different loss functions that can be used depending on the problem. In the case of the SCM, there are 3 labels (depth, spine, gaze) that have each been labelled in a binary fashion. Cross-entropy loss functions, in general, are used for binary classification problems. There are 2 types of cross-entropy loss functions: Binary Cross-Entropy (BCE) and Categorical Cross-Entropy (CCE) functions. BCE was selected as opposed to CCE as it is more suitable for multi-label classification problems in which the output can be any combination of the categories. On the contrary, CCE is used for multi-class classification problems in which the output can only be a single class. The BCE loss function is defined as:

$$loss_{BCE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M}\left(y_{ij}\log\left(p_{ij}\right) + \left(1 - y_{ij}\right)\log\left(1 - p_{ij}\right)\right) \tag{9}$$

where N is the number of samples, $M = 3$ is the number of classes. y_{ij} represents the ground truth label for that sample and class and p_{ij} represents the respective model prediction. This function allows the loss to be calculated at every epoch.

3.3.3.3 Back Propagation and Optimisation

Once the loss has been calculated, back propagation is used to propagate the loss back through the network to update the weights and biases. This is done by calculating the gradient of the loss function with respect to each weight and bias in the network. This gradient specifies in which direction the weights and bias should be adjusted to minimise loss. The calculation is done in reverse from the output to the input layer applying the chain rule to determine the relationship between the loss function and a specific neuron’s weights and bias.

To optimise the model, an optimisation algorithm is used. Gradient descent [45] is the simplest and works by using the gradient value of the loss function calculated during backpropagation to determine the direction of greatest descent. This determines how the weights and biases should be updated.

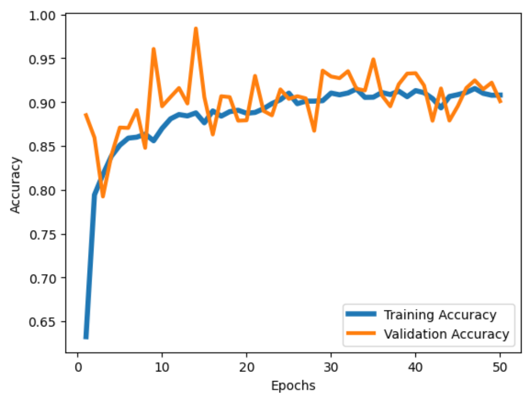
The SCM was trained on a large dataset, so using gradient descent would have resulted in slow convergence. Adam (Adaptive Moment Estimation) is an extension of gradient descent. Adam has separate learning rates (step sizes) for each parameter, allowing for more efficient convergence. Therefore, it was chosen as the optimiser for the SCM.

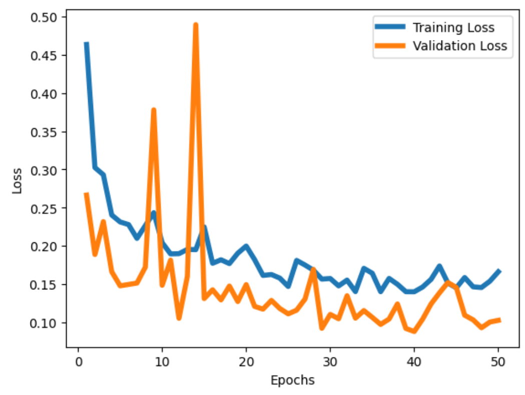
3.4 Method Summary

To summarise, a synthetic dataset from InfiniteForm was used to train the data. The dataset was pre-processed and labelled to allow it to be used for training and validation. Squats would then be classified frame-by-frame using a ML pipeline. The pipeline will begin by detecting BlazePose landmarks, it will then be fed into a squat classification model. This model contained two Conv1D layers, a dense layer and an output layer. The output layer has 3 categories: depth, spine and gaze, and each frame will be classified as any combination of them. The model will learn and optimise the weights after every epoch to improve its performance.

4 Results

The model was run for a total of 50 epochs, totalling a training time of 11 minutes and 43 seconds. The loss value and accuracy were calculated for both the training data and the validation data at every epoch. These results are shown in Figure 12. The accuracy increased throughout training, settling at around 90% after 50 epochs. The loss function value decreased over time, settling around 0.15.



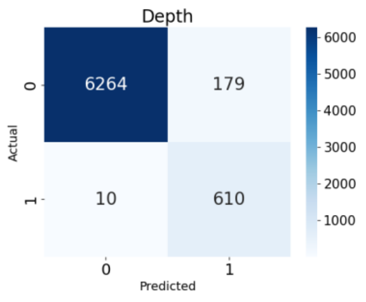


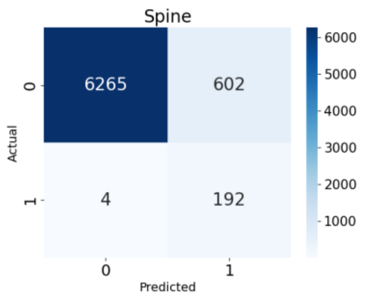
(a) (b)

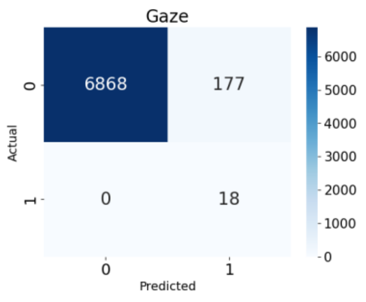
Figure 12 - Accuracy (a) and Loss (b) plots of the SCM.

Once the model had completed its training, it was run on a validation dataset. This was done to extract key metrics to determine the quality and effectiveness of the SCM in detecting depth, gaze and spine faults.

A confusion matrix was used to show the effectiveness of the SCM. Confusion matrices allow for determining the type I and type II errors that are present. Figure 13 presents the confusion matrices for each output label. For the depth category, an output of 1 denotes that a squat is in the correct range of 55° to 65°. For the spine, an output of 1 identifies that a person's spine is not neutral, and for the gaze, an output of 1 denotes that a person's gaze is downwards.







(a) (b) (c)

Figure 13 - Confusion Matrix for the depth (a), spine (b) and gaze (c) labels.

The confusion matrices show that most squat frames were predicted correctly. However, the model was more confused when trying to classify spine neutrality with 602 false positives. The results also reiterate the dataset's imbalance. To further validate the model, evaluation metrics for each label were found. These were accuracy, precision, recall and F1 score. Each evaluation metric has its purpose in evaluating the SCM. Accuracy is the simplest and measures the proportion of true results among the entire validation set. Accuracy is a less useful metric in a dataset with large imbalances, such as the dataset used for the SCM. Precision determines the percentage of true positive events that were found by the model among all predicted positive cases. Recall is the ratio of true positively predicted events against all true positive events. In the context of the spine label, for example, recall shows the ratio of how many non-neutral spines the SCM detects against how many it fails to pick up. The F1 score is a weighted average of both precision and recall, allowing a balance to be struck between them.

Table 3 - Key Evaluation metrics.

Metrics	**Depth**	**Spine**	**Gaze**	**Average**
Accuracy	97.3%	91.4%	97.4%	**95.4%**
Precision	77.3%	24.2%	9.2%	**36.9%**
Recall	98.4%	98.0%	100.0%	**98.8%**
F1 Score	86.6%	38.8%	16.9%	**47.4%**
Average	**89.9%**	**63.1%**	**55.9%**	**69.6%**

Table 3 shows all the evaluation metrics for each label. The SCM performed best for the depth label and performed worse for the gaze label. Overall, precision was the weakest metric. In comparison to a similar approach with IMUs [12], which had an accuracy of 91.7%, the average accuracy was slightly higher at 95.4%. The low precision metric shows that there were a high number of false positives. The recall score for the gaze label is 100%, but it is worth mentioning that there were only 18 true positives in the validation dataset, and all of them were predicted correctly by the model.

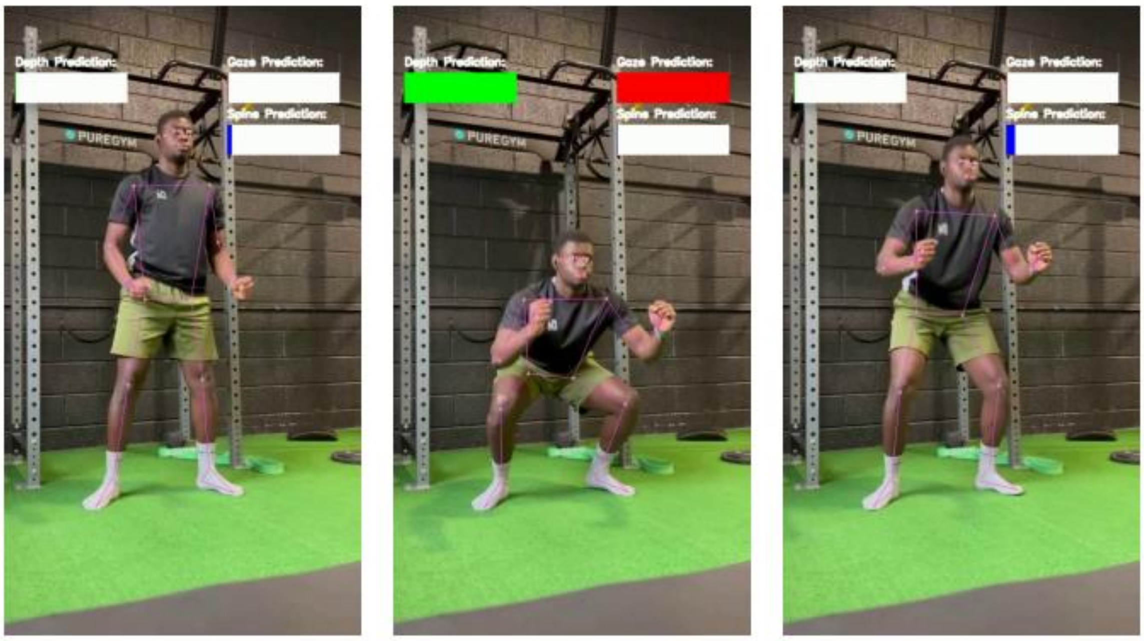


Figure 14 - An example of a real-world test of the SCM. The left bar predicts the squat's depth, and the bar on the right predicts the gaze.

Real-world tests were conducted. In this test, poor squat technique was purposely present for certain reps to test the model's effectiveness. Figure 14 shows a squat that begins with a downward gaze, then reaches the correct depth and returns to a quarter squat position. The aim of these tests was to confirm that the model worked as specified in real-life situations. The average speed of the pipeline during these tests was 382 ms. During these tests, it was found that the depth classifier could effectively predict the user's depth. The test shown in Figure 14 had an accuracy of 83.8% and an F1 score of 86.1% for the depth category. With respect to the gaze category, in the first frame of Figure 14, a downward gaze is present, but the SCMs gaze predictor does not activate. In the second frame, when the correct depth is present, but the gaze is forward, the gaze predictor is activated. Obtaining accuracy, precision, recall and F1 scores for both the gaze and spine categories was not possible due to the inability to label the real-life data frame by frame. To label spine neutrality, the 3 spine annotations (see section 3.2.1) provided by the InfiniteForm dataset were required and therefore was not available for real-world tests.

5 Discussion

5.1 Review of Project Objectives

5.1.1 Objective 1: Define metrics to quantitatively assess a person's squat form.

This objective was met. During the literature review (see section 2.1), it was found that there were 5 features of a squat that determined its quality. These were the depth of the squat, having a neutral spine, ensuring that the gaze is forward or upwards, ensuring heels are on the ground, and the knees should track over the toes. It was also found that foot stance and width had no bearing on the squat form and quality except for the comfortability of the squatter. Out of these, 3 metrics were chosen: squat depth, spine neutrality and gaze direction. These metrics were the easiest to evaluate based on the information that was available (i.e. dataset annotations). It is also important to note that it is possible for a person to tick the boxes of the 5 features and still result in a subpar squat. When reviewing a squat, a holistic view that considers body dimensions like femur length may result in a more accurate analysis of a person's squat. The quality of a squat is not entirely quantitative, so including an "overall" label that is based on a professional's judgment may improve the model's ability to assess squat form.

5.1.2 Objective 2: Conduct a review of current sensor technology.

This objective was also met. It was found that there were various types of sensor technology available. Some were more general, like smartwatches, and some were more sophisticated and specific, such as IMUs and computer vision. It was found that IMUs had been used before to evaluate squat quality, however, IMUs are expensive and are not geared towards general public use but rather to researchers. It was also found that computer vision and pose estimation is a method that has been used in the past for squat evaluation. However, the squat was not evaluated on a frame-by-frame basis, which does not allow for near-instant feedback.

5.1.3 Objective 3: Develop a system for real-time analysis of a user's squat that is available to all.

This objective was partially met. Pose estimation was the system developed to analyse squat form in real time. This was due to the possible development of a free, easy-to-use platform that anyone can download to assess their own squat. A pipeline was devised, which began with the BlazePose estimation model, which would extract landmark data from a frame. Then, this data would be passed into the squat classification model and an output classification would be presented. This pipeline allows for the quickest feedback loop as the model is constantly being run on every frame that is passed in. The model took an average of 382 ms per frame. For a live feedback loop, this is relatively slow as a frame passes every 33 ms, meaning by the time the model has assessed a squat, 10 frames would have already passed. For each category the model performed differently. It performed the best for the depth metric with high accuracy, precision and recall. However, for the spine and gaze, it did not work as well. Small-scale real-life tests were also conducted. These tests confirmed that the depth category is precise and accurate and can predict when the user's depth is correct. It is thought that the depth category can be correctly classified due to the significant distinction between a squat with a good depth and a bad one. The model was able to recognise depth patterns as they were consistent for every squat. However, for the gaze category, it is clear that the SCM had picked up on patterns related to depth as it activates when a user is at the correct depth rather than when they have a downward gaze. This is assumed to be because of dataset imbalance. When completing a squat, the bottom of the squat is the point where the most effort is exerted. At that point, there is a greater chance for form to suffer. Therefore, there is a higher chance that a user's gaze is lowered. As there were a small number of frames with a downward gaze in the dataset, it was assumed that the SCMs gaze classifier picked up on the depth pattern rather than the downward gaze patterns as that was the more apparent pattern. For spine neutrality, the model never predicts a non-neutral, even when multiple reps in the real-world test had a non-neutral spine. This was once again because of the lack of diversity in the dataset. Also, the definition for labelling the spine as non-neutral was based on how straight it was; however, a straight spine does not necessarily constitute a neutral spine and defining a better way to determine if a spine was neutral may improve results.

5.2 Limitations

The greatest limitation was that of the dataset. The synthetic squat dataset by InfiniteForm was not designed to classify incorrect form. None of the data points had a squat that was purposely incorrect. This made it harder for the SCM to detect gaze and spine faults. If a dataset where participants purposely created errors in their squats was used, it would have made it easier for the model to learn patterns. Due to time constraints and the difficulty of labelling such a dataset, this was not carried out.

Another way the model could have been improved was by isolating landmarks specifically for each category. For example, when calculating if the squatter's gaze is correct, the only landmarks that are of interest are the face landmarks (landmarks 0-10 in Figure 3a). This would reduce the computational power needed as only the landmarks that are required are used as input data. Real-world tests showed that the gaze classifier picked up on the wrong patterns, if the landmarks that were important for gaze classification were isolated it would have eliminated that possibility. This could also speed up the model as it can focus on patterns of the keypoints that are important allowing for better performance and more responsive feedback to the user.

The use of a linear regression model as opposed to binary classification for squat depth may have helped to provide more accurate depth information to the user. Currently, the model only determines if the squat is in range or out of range with no in-between. Using a linear regression model, the SCM can more accurately determine the depth of the squat and give the user specific information on how deep they are in their squat by percentage. This will help prevent users from squatting too deep, which isn't accounted for in the current model.

Finally, using a larger variety of people for both the dataset and the real-world tests would have helped improve the model's effectiveness. There is a high correlation between squat mechanics and a person's lower extremity biomechanics [46]. A person with a longer femur will squat differently than someone with a shorter femur; however, both squats may be considered good. Having a balanced data set that accounts for these differences is important.

5.3 Future Work

Future work may include recreating the SCM using a dataset that is more suitable for a squat quality classification task. Using a faster pose estimation model such as MoveNet Lightning [29], will speed up runtime, allowing for better performance. Isolating landmarks is also integral to ensure each category can learn the correct patterns and will also speed up train time and runtime. Also, implementing a linear regression model for depth classification will present more intricate detail about the user's squat. Similar models could also be developed for use in other exercises like deadlifts and bench press. It could also be adapted for use in sports. Being able to receive real-time feedback about form in sports like football or basketball during solo training sessions will be integral to making rapid progress. However, as with most problems solved with machine learning, obtaining a diverse and well-annotated dataset to train any model is the greatest problem.

6 Conclusion

A SCM was developed that could accurately classify whether a person's squat was in the correct depth with an accuracy of 97.3%. However, when classifying the neutrality of the spine or the gaze direction, the model picked up incorrect patterns, resulting in poor classification. Future work could include optimising the current model using a more suitable dataset or creating a similar model for other exercises or sports.

7 References

1. Comfort P, McMahon JJ, Suchomel TJ. Optimizing Squat Technique—Revisited. Strength & Conditioning Journal. 2018 Dec;40(6):68–74.

2. How Much is a Personal Trainer? Latest 2023 Guidance [Internet]. 2024 [cited 2024 May 2]. Available from: https://www.thefitnessgrp.co.uk/how-much-is-a-personal-trainer/

3. Sui W, Rush J, Rhodes RE. Engagement With Web-Based Fitness Videos on YouTube and Instagram During the COVID-19 Pandemic: Longitudinal Study. JMIR Form Res. 2022 Mar 8;6(3):e25055.

4. Dr. Aaron Horschig [Internet]. Squat University. 2015 [cited 2024 May 2]. Available from: https://squatuniversity.com/about/

5. Panjabi MM. The stabilizing system of the spine. Part II. Neutral zone and instability hypothesis. J Spinal Disord. 1992 Dec;5(4):390–6; discussion 397.

6. THE EFFECT OF THE DIRECTION OF GAZE ON THE KINEMATICS OF THE SQUAT EXERCISE [Internet]. [cited 2024 May 2]. Available from: https://oce-ovid-com.bris.idm.oclc.org/article/00124278-200602000-00023/HTML

7. Anonymous. English: A man performing a bodyweight squat. He begins by standing with his arms by his sides. He then squats down and moves his arms up in front of himself. He then stands again and moves his arms back to his sides. [Internet]. 2021 [cited 2024 May 7]. Available from: https://commons.wikimedia.org/wiki/File:Man-Doing-Air-Squats-A-Bodyweight-Exercise-for-Legs.png

8. Bender CG, Hoffstot JC, Combs BT, Hooshangi S, Cappos J. Measuring the fitness of fitness trackers. In: 2017 IEEE Sensors Applications Symposium (SAS) [Internet]. 2017 [cited 2024 May 2]. p. 1–6. Available from: https://ieeexplore.ieee.org/abstract/document/7894077

9. De Beukelaar TT, Mantini D. Monitoring Resistance Training in Real Time with Wearable Technology: Current Applications and Future Directions. Bioengineering. 2023 Sep 14;10(9):1085.

10. Ahmad N, Ghazilla RAR, Khairi NM, Kasi V. Reviews on Various Inertial Measurement Unit (IMU) Sensor Applications. IJSPS. 2013;256–62.

11. Zhao H, Wang Z. Motion Measurement Using Inertial Sensors, Ultrasonic Sensors, and Magnetometers With Extended Kalman Filter for Data Fusion. IEEE Sensors Journal. 2012 May;12(5):943–53.

12. Lee J, Joo H, Lee J, Chee Y. Automatic Classification of Squat Posture Using Inertial Sensors: Deep Learning Approach. Sensors. 2020 Jan 8;20(2):361.

13. Routhier F, Duclos NC, Lacroix É, Lettre J, Turcotte E, Hamel N, et al. Clinicians' perspectives on inertial measurement units in clinical practice. Papa F, editor. PLoS ONE. 2020 Nov 13;15(11):e0241922.

14. Buy MTw Awinda Wireless Motion Tracker [Internet]. Movella. [cited 2024 May 2]. Available from: https://shop.movella.com/product-lines/wearables/products/mtw-awinda-wireless-motion-tracker

15. What Is Computer Vision? | Microsoft Azure [Internet]. [cited 2024 May 2]. Available from: https://azure.microsoft.com/en-gb/resources/cloud-computing-dictionary/what-is-computer-vision

16. Lin TY, Maire M, Belongie S, Bourdev L, Girshick R, Hays J, et al. Microsoft COCO: Common Objects in Context [Internet]. arXiv; 2015 [cited 2024 May 2]. Available from: http://arxiv.org/abs/1405.0312

17. Chung JL, Ong LY, Leow MC. Comparative Analysis of Skeleton-Based Human Pose Estimation. Future Internet. 2022 Dec 15;14(12):380.

18. Girase S, Dutta O, Mahadar A, Ghodmare A, Bedekar M. Identifying Incorrect Postures While Performing Sun Salutation Using MoveNet. In: Sharma H, Shrivastava V, Bharti KK, Wang L, editors. Communication and Intelligent Systems. Singapore: Springer Nature; 2023. p. 575–87.

19. Parashar D, Mishra O, Sharma K, Kukker A. Improved Yoga Pose Detection Using MediaPipe and MoveNet in a Deep Learning Model. RIA. 2023 Oct 31;37(5):1197–202.

20. Badiola-Bengoa A, Mendez-Zorrilla A. A Systematic Review of the Application of Camera-Based Human Pose Estimation in the Field of Sport and Physical Exercise. Sensors. 2021 Jan;21(18):5996.

21. Patil A, Rao D, Utturwar K, Shelke T, Sarda E. Body Posture Detection and Motion Tracking using AI for Medical Exercises and Recommendation System. Patil MD, Vyawahare VA, editors. ITM Web Conf. 2022;44:03043.

22. Joseph R, Ayyappan M, Shetty T, Gaonkar G, Nagpal A. BeFit—A Real-Time Workout Analyzer. In: Shakya S, Balas VE, Kamolphiwong S, Du KL, editors. Sentimental Analysis and Deep Learning. Singapore: Springer; 2022. p. 303–18.

23. Washabaugh EP, Shanmugam TA, Ranganathan R, Krishnan C. Comparing the accuracy of open-source pose estimation methods for measuring gait kinematics. Gait & Posture. 2022 Sep;97:188–95.

24. Youssef F, Zaky A, Gomaa W. Analysis of the Squat Exercise from Visual Data: In: Proceedings of the 19th International Conference on Informatics in Control, Automation and Robotics [Internet]. Lisbon, Portugal: SCITEPRESS - Science and Technology Publications; 2022 [cited 2024 May 3]. p. 79–88. Available from: https://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0011347900003271

25. Bazarevsky V, Grishchenko I, Raveendran K, Zhu T, Zhang F, Grundmann M. BlazePose: On-device Real-time Body Pose tracking [Internet]. arXiv; 2020 [cited 2024 May 3]. Available from: http://arxiv.org/abs/2006.10204

26. Boyle M. Mobile phone and internet usage statistics in the UK [Internet]. Finder UK. 2019 [cited 2024 May 3]. Available from: https://www.finder.com/uk/banking/mobile-internet-statistics

27. Cao Z, Hidalgo G, Simon T, Wei SE, Sheikh Y. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields [Internet]. arXiv; 2019 [cited 2024 May 3]. Available from: http://arxiv.org/abs/1812.08008

28. Papandreou G, Zhu T, Chen LC, Gidaris S, Tompson J, Murphy K. PersonLab: Person Pose Estimation and Instance Segmentation with a Bottom-Up, Part-Based, Geometric Embedding Model [Internet]. arXiv; 2018 [cited 2024 May 3]. Available from: http://arxiv.org/abs/1803.08225

29. Chen YH, Votel R. MoveNet.SinglePose Model Card.

30. Pose Detection comparison : wrnchAI vs OpenPose | LearnOpenCV [Internet]. 2019 [cited 2024 May 3]. Available from: https://learnopencv.com/pose-detection-comparison-wrnchai-vs-openpose/

31. Oviyum. Pose Estimation Benchmarks on intelligent edge [Internet]. Medium. 2019 [cited 2024 May 3]. Available from: https://medium.com/@oviyum/pose-estimation-benchmarks-on-intelligent-edge-afadcdd6c039

32. High Fidelity Pose Tracking with MediaPipe BlazePose and TensorFlow.js — The TensorFlow Blog [Internet]. [cited 2024 May 3]. Available from: https://blog.tensorflow.org/2021/05/high-fidelity-pose-tracking-with-mediapipe-blazepose-and-tfjs.html

33. Next-Generation Pose Detection with MoveNet and TensorFlow.js — The TensorFlow Blog [Internet]. [cited 2024 May 3]. Available from: https://blog.tensorflow.org/2021/05/next-generation-pose-detection-with-movenet-and-tensorflowjs.html

34. Bazarevsky V, Kartynnik Y, Vakunov A, Raveendran K, Grundmann M. BlazeFace: Sub-millisecond Neural Face Detection on Mobile GPUs [Internet]. arXiv; 2019 [cited 2024 May 3]. Available from: http://arxiv.org/abs/1907.05047

35. On-Device, Real-Time Hand Tracking with MediaPipe [Internet]. [cited 2024 May 3]. Available from: https://research.google/blog/on-device-real-time-hand-tracking-with-mediapipe/

36. Weitz A, Colucci L, Primas S, Bent B. InfiniteForm: A synthetic, minimal bias dataset for fitness applications [Internet]. arXiv; 2021 [cited 2024 May 3]. Available from: http://arxiv.org/abs/2110.01330

37. Fitness Basic Dataset [Internet]. Infinity AI. [cited 2024 May 3]. Available from: https://marketplace.infinity.ai/products/fitness-basic-dataset

38. Bengio Y. Practical recommendations for gradient-based training of deep architectures [Internet]. arXiv; 2012 [cited 2024 May 3]. Available from: http://arxiv.org/abs/1206.5533

39. Ian Goodfellow, Yoshua Bengio, Aaron Courville. Deep Learning [Internet]. MIT Press; 2016 [cited 2024 May 8]. Available from: https://www.deeplearningbook.org/

40. Kiranyaz S, Avci O, Abdeljaber O, Ince T, Gabbouj M, Inman DJ. 1D convolutional neural networks and applications: A survey. Mechanical Systems and Signal Processing. 2021 Apr 1;151:107398.

41. Zhan H, Chen G, Lu Y. Applying Batch Normalization to Hybrid NN-HMM Model For Speech Recognition. In: Tan T, Li X, Chen X, Zhou J, Yang J, Cheng H, editors. Pattern Recognition. Singapore: Springer; 2016. p. 427–35.

42. Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR. Improving neural networks by preventing co-adaptation of feature detectors [Internet]. arXiv; 2012 [cited 2024 May 3]. Available from: http://arxiv.org/abs/1207.0580

43. Baldi P, Sadowski PJ. Understanding Dropout. In: Advances in Neural Information Processing Systems [Internet]. Curran Associates, Inc.; 2013 [cited 2024 May 8]. Available from: https://proceedings.neurips.cc/paper_files/paper/2013/hash/71f6278d140af599e06ad9bf1ba03cb0-Abstract.html

44. Hochreiter S. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. 1998 Apr 1;6:107–16.

45. Du S, Lee J, Li H, Wang L, Zhai X. Gradient Descent Finds Global Minima of Deep Neural Networks. In: Proceedings of the 36th International Conference on Machine Learning [Internet]. PMLR; 2019 [cited 2024 May 3]. p. 1675–85. Available from: https://proceedings.mlr.press/v97/du19c.html

46. Kim S, Miller M, Tallarico A, Helder S, Liu Y, Lee S. Relationships between physical characteristics and biomechanics of lower extremity during the squat. Journal of Exercise Science & Fitness. 2021 Oct 1;19(4):269–77.