
# Utilising Computer Vision and Machine Learning to create a Weightlifting Form Analysis Application

by

Bradley Greaves

Supervisor: Dr Gerald Schaefer

Department of Computer Science
Loughborough University

CS : 23COC251
F015143

May 2024

**Abstract**

This project details the conceptualisation, development and implementation of a weightlifting form analysis system, utilising the popular pose-recognition library, OpenPose, in conjunction with a collection of machine learning models to analyse weightlifting videos, produce a scoring and provide advice to the user on how they could improve their technique to improve muscle engagement and prevent injury.

# Table of Contents

# List of Figures

# 1 Introduction

## 1.1 Problem Domain

In the fitness and strength training industry, poor weightlifting form continues to stand as a continuous and pervasive issue that brings with it potentially hazardous consequences, despite the unprecedented level of access to information we now have. Neglecting proper form not only diminishes the effectiveness of workouts, but also increases the risk of injury, ranging from minor strains to joint injuries and severe musculoskeletal damage. Persistent use of sub optimal form can also slow down progress, limiting muscular strength and size development. Addressing this issue requires an understanding in bio-mechanics and exercise physiology to effectively provide tailored corrective advice.

## 1.2 Project Aims and Objectives

This project aims to develop an application designed to assist users in improving their technique in a range of weightlifting exercises by utilising a combination of computer vision, image analysis and machine learning techniques, to produce suitable personalized feedback on how to increase muscle engagement, promote muscle growth and reduce risk of injury. Users will be able to record or upload video of them performing on of the supported weightlifting exercises in order to receive feedback on how they can improve their lifting technique.

Objectives of this project include:

- Identifying a gap in the market for a form analysis system by researching alternative applications in the problem space and identifying the positive and negative aspects of each.
- Performing a literature review to become informed on the currently accepted principles in lifting bio-mechanics and exercise physiology, and to identify which current technologies can be utilised to implement the system.
- Creating a dataset of weightlifting videos containing examples of what is considered good lifting form, along with examples of common errors in technique made when performing specific exercises.
- Developing a Python application capable of analysing the form of a user in order to provide a scoring based on the quality of the user's lift, alongside feedback specific to the user based on commonly accepted weightlifting principles.
- Evaluate the objectivity of the scoring system and the quality of the advice provided using real-life examples during testing.

## 1.3 Project Motivations

The motivations for creating this project include:

- Assisting novice weightlifters in making significant muscle strength and size developments by providing tailored advice on adjustments that could be made in their lifting technique.
- Reducing the barrier of entry into the gym for users by making tailored advice readily available instead of being locked behind the typical paywall of personal training sessions.
- Reducing the likelihood of injury for weightlifters whilst training and reduce the additional strain on medical services caused by training-related injuries.
- Developing a deeper understanding of lifting bio-mechanics and exercise physiology.
- Developing an understanding of increasingly prevalent machine learning techniques.
- Utilising skills developed during university to contribute to solving a real-life problem of personal interest.

## 1.4 Assumptions and Limitations

For this project to succeed, the assumption has been made that the user has enough prior experience with weightlifting to be able to perform the chosen exercises accurately enough for the system to provide relevant form scoring and advice.

Whilst this project was initially conceived as a mobile application for ease of use and portability, upon research into commonly used pose recognition models, it became apparent that the current range of pose recognition models available are unsuitable for mobile development due to either lack of relevant keypoint capturing or increasingly high system requirements for mobile application development.

This project could require hands-on advice from an experienced weightlifter or professional in Sports Science so as to ensure that the guidance provided by the system is correct and accurate, especially given the nuances in weightlifting such as differing body proportions changing the optimal technique for a variety of exercises.

At the time of writing, there exists no large, readily available video datasets in weightlifting and as a consequence, the success of this project hinges on being able to collate a large enough lifting video dataset for the machine learning

models in the scoring system to accurately produce a score based on user's technique, especially considering a separate model will need to be trained for each of the four exercises selected for the project.

# 2 Literature Review

The UK fitness industry has grown considerably in recent years. As of 2024, there are currently estimated to be around 10.7 million gym members in the UK, consisting 16% of the population, up from 14% the previous year (Pure Gym Limited, 2024). In this same time, the number of employed physiotherapists in the UK has only increased by 3.9% (statista, 2023). As such, injury prevention via technique guidance could reduce the strain on medical services caused by the likely increase in weightlifting related injuries.

This section aims to provide a detailed insight into the currently available options for lifting technique guidance, identifying gaps in their functionality that can be filled with the proposed system. It also aims to summarise the knowledge gained through research in the field of biomechanics and sports science that will be utilized by the system to provide accurate advice, as well as summarise the technologies considered by the project and the reasonings behind the technologies chosen to be utilized in the solution.

## 2.1 Gap Analysis

The landscape of readily available form analysis applications is extremely sparse, however there do exist some systems similar to the one proposed and so in the following section, these systems will be analysed for gaps in their functionality that can be built upon and expanded.

### 2.1.1 Analysis Criteria

In order to evaluate the existing implementations of similar systems in the form analysis landscape, the current state of each application or system will be compared to the desired objectives of the proposed solution.

The points of consideration will include:

- The range of exercises the system is capable of analysing.
- The complexity of the pose recognition models used.
- The quality of the feedback provided to the user.
- The relevancy of the system to the weightlifting community. of personal interest.

### 2.1.2 Computer Vision Weightlifting Coach

'Computer Vision Weightlifting Coach' (SravB, 2019), is a system developed by 'SravB' that analyses deadlifting videos/images and scores the quality of the user's lifting technique on a scale from 0-1. This system utilises a machine learning model trained on a dataset consisting of 11 deadlifting example videos. The system utilised OpenPose for its pose detection model. The location of the joints identified using OpenPose were then used as the features for a machine learning classification model to predict whether the user is using correct deadlifting posture. The confidence in the positive class is then used to formulate a scoring for the quality of the user's lifting form for each given frame.


Figure 1: An example scoring from Computer-Vision-Weightlifting-Coach.

This system utilises OpenPose's MPI model for pose recognition. The MPI model was made obsolete by the BODY_25 model released in 2018, with the MPI model being slower and less accurate than its successor (CMU Perceptual Computing Lab, 2019).

Whilst this system does appear to be a useful aid for beginner lifters trying to improve their form, the system does not provide any advice to the user, only a scoring, instead requiring the user to experiment with their lifting technique in an attempt to achieve a higher score. This guidance-less approach could lead to an increased risk of injury whilst the user is still learning and does not alleviate the need for a personal trainer.

There is room for improvement for the flexibility of this system. Currently restricted to providing a score for a single exercise, the system would benefit from including support for a larger range of compound exercises.

A final gap identified in the system is the size of the dataset used to train the machine learning model utilised in this system. The dataset has a total sample size of 11 videos, and so the dataset used is likely reducing the accuracy of the scoring provided for each lift, as shown in Figure 1, where a deadlift that seems to be initiated with arguably good technique from a biomechanical standpoint is scored just 0.452.

### 2.1.3 Pose Estimation and Correcting Exercise Posture

'Pose Estimation and Correcting Exercise Posture' (Kanasu, 2021), is a paper detailing the development of a system that analyses weightlifting videos in order to guide the user on how to employ correct technique from a specified list of exercises.

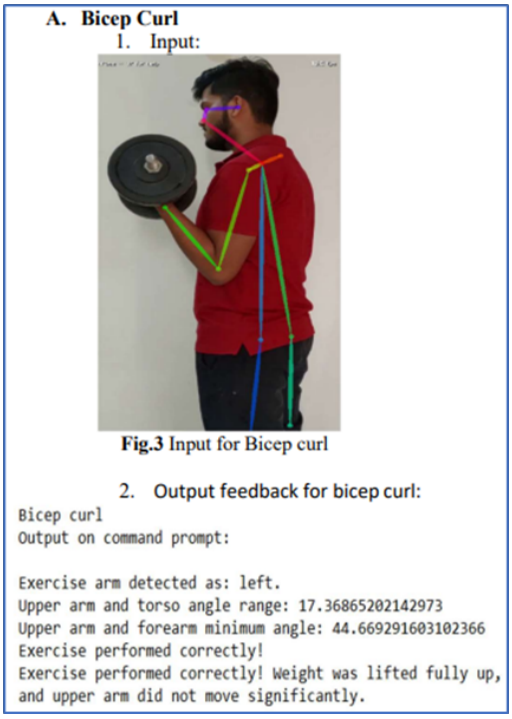


Figure 2: Example from the system developed in Pose Estimation and Correcting Exercise Posture

This system utilised OpenPose for its MPII keypoint detection model. The location of the joints identified using OpenPose were then used to calculate different body vectors. Each exercise then employs geometry evaluation in order to generate advice for the user. An example of the geometry evaluations employed by this system is using the change in angle between the upper arm and torso vectors to determine whether the bicep curl was performed correctly.

A drawback identified in this system is lack of compound exercise selection. Compound exercises are often the most important exercises to perform as they involve multiple muscle groups and are the staple of many workouts. The system specified fails to accommodate any lower body exercises.

Like the previous example, this system also utilises OpenPose's MPII model for its pose recognition functionality, which can lead to inaccuracies as stated within the OpenPose documentation (CMU Perceptual Computing Lab, 2019).

### 2.1.4 WODProof

WODProof (WODProof App inc, 2024), is a mobile application designed for the CrossFit community. The application allows users to record, review, and analyse their CrossFit workouts by capturing video footage of their workouts, and allowing them to annotate key moments and add timers/overlays.

In 2022, WODProof Bionic has implemented an AI rowing coach that provides real time feedback on a user's rowing technique, as shown in Figure 3. As of writing, rowing is currently the only exercise that has AI coach support. Despite this functionality being similar to the proposed solution outlined previously, the system lacks any options for diversity in the exercise selection. The AI rowing coach is also locked behind a hefty subscription fee which reduces the availability of this system.

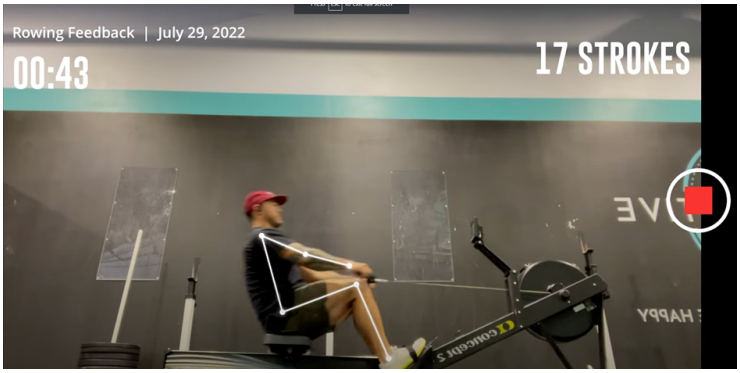


Figure 3: WODProof's AI Rowing Coach

This feature is powered by Sency's AI powered technology however, details on the implementation of the AI rowing coach are unavailable to the public domain due to WODProof being a flagship commercial product in the CrossFit community.

Although through further development this system would likely be of great use to the weightlifting community, as it stands at the time of writing, this system is not especially relevant to the weightlifting community for injury prevention due to the niche exercise selection, and so the gap aiming to be filled by the proposed solution developed in this project remains.

## 2.2 Bio-mechanics Research

It was decided that the proposed application will support a total of four weightlifting exercises, comprised of the Barbell Squat, Barbell Deadlift, Barbell Overhead Press, and Barbell Bicep Curl. In order to provide accurate, scientifically sound advice to the user regarding their technique, a good understanding of the bio-mechanical principles for each lift needs to be acquired. This section will serve as a collation of the understanding gained from bio-mechanics literature regarding what is considered to be the correct technique in each exercise.

### 2.2.1 Barbell Squat

The barbell squat is a lower body compound exercise that primarily targets the quadriceps and glutes, responsible for knee extension and hip extension respectively. The squat also engages the spinal erectors and abdominal muscles isometrically to create core stability and provide support to the spine against shearing forces throughout the movement. Mistakes that are commonly made whilst performing the barbell squat are described in this section.

Knee valgus, (colloquially known as knee cave), is when the knees collapse inwards towards the midline of the body during functional exercises such as squatting and jumping. Commonly occurring during weighted squats, knee valgus can be a predictor for risk of injury to the anterior cruciate ligament (Timothy E Hewett 1, 2005), and as such should be avoided.

The half-squat is a term used for a squat that fails to reach an appropriate depth, appropriate usually meaning that the femur breaks parallel to the floor. Failing to reach depth in the barbell squat reduces the effectiveness of the workout by reducing muscle strength and size developments in the quadriceps that could otherwise be achieved by performing a full-depth squat. (Keitaro Kubo 1, 2019).

Spinal flexion, also known as lower back rounding, is a mistake often occurring when performing squat and deadlift based movements, and is a the primary cause of injury to the lumbar spine, due to the increased shearing forces placed

on the lumbar spine during flexion, (Sjöberg, Aasa, Rosengren, & Berglund, 2020).

Stance width whilst squatting is highly individual, although narrow squat stances (below 1.06x hip width), have been shown to increase quadriceps force generation and enhance peak power production during the squat and wide squat stance widths (above 1.2x hip width), may improve glute hypertrophy, (Jonathan Sinclair, 2022). For those aiming to achieve both considerable quadriceps and glute stimulation by performing squats, a stance width between these two values is advised.

### 2.2.2 Barbell Deadlift

The barbell deadlift is a lower body compound exercise that primarily targets the hamstrings and glutes, to produce hip extension. The deadlift also engages the trapezius, latissimus dorsi, spinal erectors and abdominal muscles isometrically to create core stability and provide support to the spine against shearing forces throughout the movement, similar to the barbell squat. Mistakes that are commonly made whilst performing the barbell deadlift are described in this section.

Utilising an inappropriate hip height relative to the knees and shoulders. Good conventional deadlifting technique typically requires the hips to be placed slightly below the half squat position, hinging at the hips to reach the bar. This position places the initial loading of the exercise onto the quadriceps and reduces stress on the lumbar spine, (Groves, 2000).

Spinal flexion, similar to the squat, (Sjöberg, Aasa, Rosengren, & Berglund, 2020). The bar path during the deadlift should remain as vertical as possible to reduce the work done to move the bar through the movement. Loss of control of the bar, often indicated by being unable to keep the bar directly under the shoulder joint, has shown to be a leading cause of injury to the shoulder, including fractures and dislocations, during the deadlift, (Victor Bengtsson, 2018).

### 2.2.3 Barbell Overhead Press

Also known as the military press, the barbell overhead press is an upper body compound exercise that primarily targets the front deltoids, triceps, and upper chest. The overhead press also engages the spinal erectors and abdominal muscles isometrically to prevent excessive arching whilst performing the exercise. Common mistakes that occur whilst performing the overhead press are described in this section.

Utilising excessive leg drive allows the lifter to generate additional momentum in the beginning portion of the lift, reducing the demand placed to the primary

movers for the exercise. Whilst this allows the lifter to press a larger weight, and still an effective exercise in its own right, as a result of the demand being moved away from the target muscles, this diminishes the effectiveness of the exercise in terms of front deltoid muscle development, (Hadim, 2024).

Lumbar spine hyperextension, also known as lower back arching, often occurs when a lifter is training with heavy loads at which form breakdown can occur. Lower back arching places additional stress on the lumbar spinal disks and can result in back pain or in severe cases, spondylolysis, and spondylolisthesis complications, (Kendrick, n.d.).

### 2.2.4 Barbell Bicep Curl

The barbell bicep curl is an upper body isolation exercise that primarily targets the biceps brachii. The movement is typically performed by gripping the barbell with palms facing up and elbows extended and raising the palms towards the shoulder joint by contracting the biceps. Common mistakes encountered when performing the barbell bicep curl are described in this section.

Overarching the lumbar spine, similar to the barbell overhead press. Whilst the risk of injury is reduced due to the barbell curl typically being performed with a lighter load than the overhead press, there is still a risk. Despite this, a controlled and non-excessive arch has the possibility positively affect the effectiveness of the barbell bicep. Leaning backwards modifies the resistance profile of the exercise by moving the point at which the working weight is travelling with a vector that directly opposes gravity closer to the point of full elbow extension. This means the bicep is placed under the greatest tension when the bicep is in a more stretched position than an standard, upright barbell curl. The increased tension placed on the stretched position could lend itself to improved stretch-mediated hypertrophy, although the current understanding is that this is not yet certain, (Konstantin Warneke, 2023). This also applies the other commonly occurring mistake of not using a full range of motion whilst performing the exercise.

## 2.3 Technology Review

The proposed application will utilise technologies from within the landscapes of both Computer Vision and Machine learning and so the appropriate considerations must be made as to what relevant libraries or frameworks will be utilised. The following section makes comparisons between these technologies to decide which are the most suitable for the project.

### 2.3.1 Pose Recognition Libraries

This section aims to provide an insight into the options of pose recognition libraries available to be utilized within this project. In order to compare the

suitability of the libraries for this project, they will be evaluated by accuracy, speed and ease of integration into a machine learning application.

**OpenPose**

OpenPose is a real-time pose estimation library developed by the Carnegie Mellon University Perceptual Computing Lab, (CMU Perceptual Computing Lab, 2020). The library utilises deep learning in order to detect and track body keypoints. It is capable of multi-person detection which whilst not relevant to this project, could enable additional future developments. At the time of writing, OpenPose currently offers three body pose recognition models that could all potentially be used whilst developing the application; MPI, COCO and BODY_25.

MPI is an 18-keypoint detection model trained on the MPII dataset, (Max Planck Institute for Informatics). The MPI model is the fastest model of the three whilst running on the CPU but is also the least accurate.

COCO is an 18-keypoint detection model trained on Microsoft's COCO (Common Object in Context) dataset, (Lin, 2014). The COCO model is faster than BODY_25 but slower than MPI when running on the CPU, and faster than BODY_25 when run on the GPU. It offers higher accuracy than the MPI model.

BODY_25 is a 25-keypoint detection model trained on a combination of the COCO and MPII datasets. It is the fastest model when run on the GPU and is also the most accurate. It also offers an additional 7 keypoints, some of which have great applicability to the proposed application, such as a keypoints in the neck and mid-hip.

OpenPose also comes bundled with PyOpenPose, a Python API enabling seamless integration of OpenPose into Python-based applications via OpenCV. Considering Python is a top programming language for machine learning based applications, this makes OpenPose a strong contender for the pose recognition library used in this project.

**PoseNet**

PoseNet is an open-source machine learning model created by Google Creative Lab for pose recognition, (BeomJun Jo, 2022). It is a 17-keypoints detection model trained on the COCO dataset. The library is accessible through various libraries and frameworks such as TensorFlow.js, which makes it suitable for integration into web applications and alleviates the need for specialist hardware. Being a 17-keypoint detection model, PoseNet offers limited flexibility in keypoint extraction in comparison to OpenPose, with fewer torso keypoints

than the competitor. PoseNet is also limited to 2D pose estimation which could limit future development of the application, although it could still be suitable for the project given the current project scope at the time of writing.

#### Google ML Kit Pose Detection API

The Google ML Kit Pose Detection API is a machine learning-based solution currently being developed by Google as part of the ML Kit SDK, (Google., 2024). The API includes a 33-keypoint detection model including keypoints for the face and extremities. The ML Kit is designed to integrate seamlessly into mobile applications, abstracting away the complexities of pose detection.

The drawbacks of the ML Kit Pose Detection API are the limited number of keypoints available for the torso in comparison to OpenPose. Whilst the API seems to provide reasonably accurate pose detection, the lightweight nature of the API likely will not provide the level of accuracy obtainable through the use of more specialised pose estimation models like OpenPose or PoseNet. The ML Kit Pose Detection API currently in the beta stage of development and so additional changes to the API are still being made, and it has been stated that changes made may break backwards compatibility, (Google., 2024).

### 2.3.2 Machine Learning

This section will contrast and compare the popular available machine learning libraries available for use within the project. In order to compare the suitability of the machine learning libraries for this project, they will be evaluated by flexibility, ease of use and the level of community support provided by each library by its developers to create as seamless an implementation phase as possible.

#### TensorFlow

TensorFlow is an open-source machine learning framework developed by Google Brain, (Martín Abadi, 2024). TensorFlow offers an ecosystem of tools and libraries for creating flexible, scalable machine learning applications and supports a variety of programming languages such as Python, C++ and JavaScript. TensorFlow is statically graphed and so is well optimized for use with hardware acceleration, lending it to having high performance. Whilst typically statically graphed, TensorFlow has introduced features such as Eager Execution that enable graph behaviour that is dynamic similar to PyTorch.

TensorFlow was one of the first comprehensive libraries developed for deep learning and so the combination of its longevity and association to Google has lead it to garner a sizeable community, abundant in tutorials, along with robust documentation to aid in development.

### PyTorch

PyTorch (Paszke, 2019), is an open-source machine learning library developed by Facebook's AI Research Lab (FAIR). PyTorch is known for its dynamic computational graphing, meaning that the graphs are generated as operations are performed, allowing for more flexibility in model architecture when compared to alternative typically statically graphed frameworks such as TensorFlow. Dynamically computing graphs does make PyTorch marginally slower that TensorFlow, but the performance would still be adequate for the proposed application.

Despite releasing later than TensorFlow, PyTorch has managed to develop a high reputation within the machine learning research community due to the quality of its documentation and flexibility.

### Scikit-Learn

Scikit-learn (Pedregosa, 2011), is a popular machine learning library initially developed by David Cournapeau. The library is known for its simplicity and ease of use, focusing on traditional machine learning algorithms optimized for CPU-based computation, (Cohen, 2023). This makes Scikit-learn less suitable for deep learning applications than alternatives such as PyTorch and TensorFlow, instead excelling at simpler machine learning applications.

Whilst a respectable alternative to TensorFlow and PyTorch, the simplified approach and reduced suitability for deep learning applications makes Scikit-learn a less likely candidate to be chosen for this project, instead lending itself more to data mining related applications.

# 3 Requirements Analysis

This section aims to outline a set of requirements that will be met in order to evaluate the success of the proposed application and provide structure to the design and implementation stages. The requirements analysis consists of two sections; an overview of the project scope including technical requirements of the system, and a user requirements section in which the requirements of each user interface are described, along with details on the screen flow that will be adhered to in the applications.

## 3.1 Project Scope

The aim of this project is to create an application capable of providing advice to the user on their weightlifting form. In order to do so, the application will be comprised of machine learning models trained to recognise features extracted from a recording of the user's exercise using OpenPose. Using this description of the proposed application, alongside aspects discussed in the literature review section, the following general requirements have been created.

**General requirements:**

- The application should include a form scoring system powered by machine learning, supporting a variety of lifting exercises.
- The machine learning models comprising the application should be trained on lifting video datasets that successfully encapsulate the lifting principles discussed whilst researching the bio-mechanics of weightlifting.
- The application should be able to accept video file input from a user's device.
- The application should allow the user to record an exercise using the user device's camera as an alternative method of input.
- The application should be able to process the input from the user to generate the keypoint information necessary for generating a score and providing relevant advice.
- System should use thresholds created based on the discussions had during research to provide accurate and relevant advice.

## 3.2 User Requirements

Expanding upon the general system requirements outlined in the previous section, this section aims to provide a list of more detailed, user-centric requirements. In order to do this, the proposed application has been divided into its relevant features and the following screen flow has been created to better describe what a typical user journey would consist of in this application. User requirements have then been created for each interface specified in Figure 4 in order to provide greater insight into the functionality of the application.

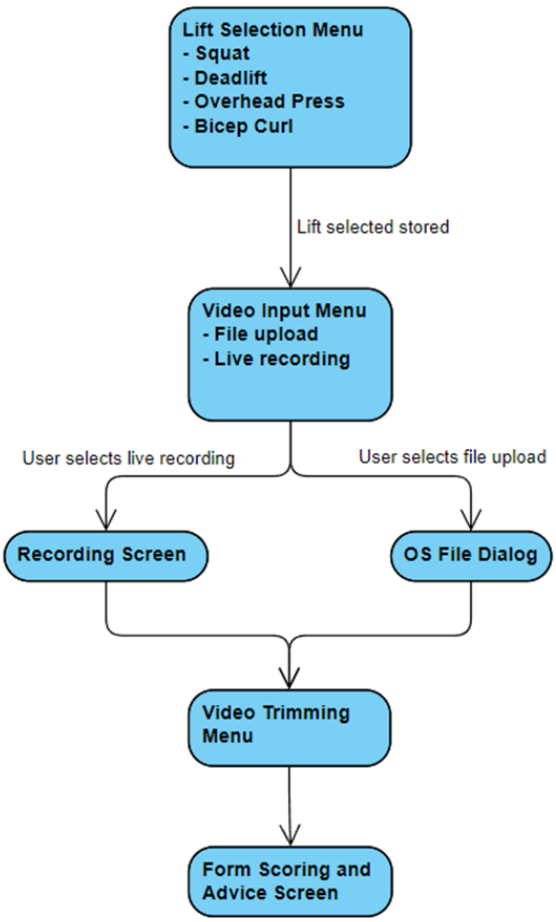


Figure 4: User journey / Screen flow

**Lift Selection Menu**

- Users should be able to browse the list of exercises the application supports.
- Users should be able to select an exercise from the list that they wish to perform.

**Video Input Menu**

- Users should be able to see the list of supported methods for video input.
- Users should be able to select the method of video input that they wish to use.
- Upon selecting the live recording option, the user should be taken to the live recording menu.
- Alternatively, upon selecting the file upload option, the user should be taken shown a file dialog that allows the user to upload a video file through their file explorer.

**Recording Screen**

- Users should be shown a live feed from their device's camera.
- Users should be able to start and stop recording using provided buttons.
- Users should be able to submit their recording by pressing a submit button.
- Upon submitting a recording, users should be taken to the video trimming menu.

**Video Trimming Menu**

- Users should be shown a frame-by-frame slideshow of the video input.
- Users should be able to move forwards and backwards through the slideshow using the associated buttons.
- Users should be able to select the frames corresponding to the start and end of the performed exercise.
- Users should be able to submit the trimmed video for processing using the submit button.
- Upon submitting a trimmed video, the user should be taken to the scoring and advice screen.

**Scoring and Advice Screen**

- Users should be provided a score based on the quality of their lifting form.
- Users should be provided advice on how they can improve their lifting form, along with imagery from the provided video highlighting mistakes, in the form of a scroll-able list.

# 4 Design

## 4.1 Functional Design

The aim of this section is to outline the architecture, important technical components, data structures and technologies used in order to create the proposed application, using the requirements as a guideline. In order to better visualize how the different components of the application interact, see Figure 5.

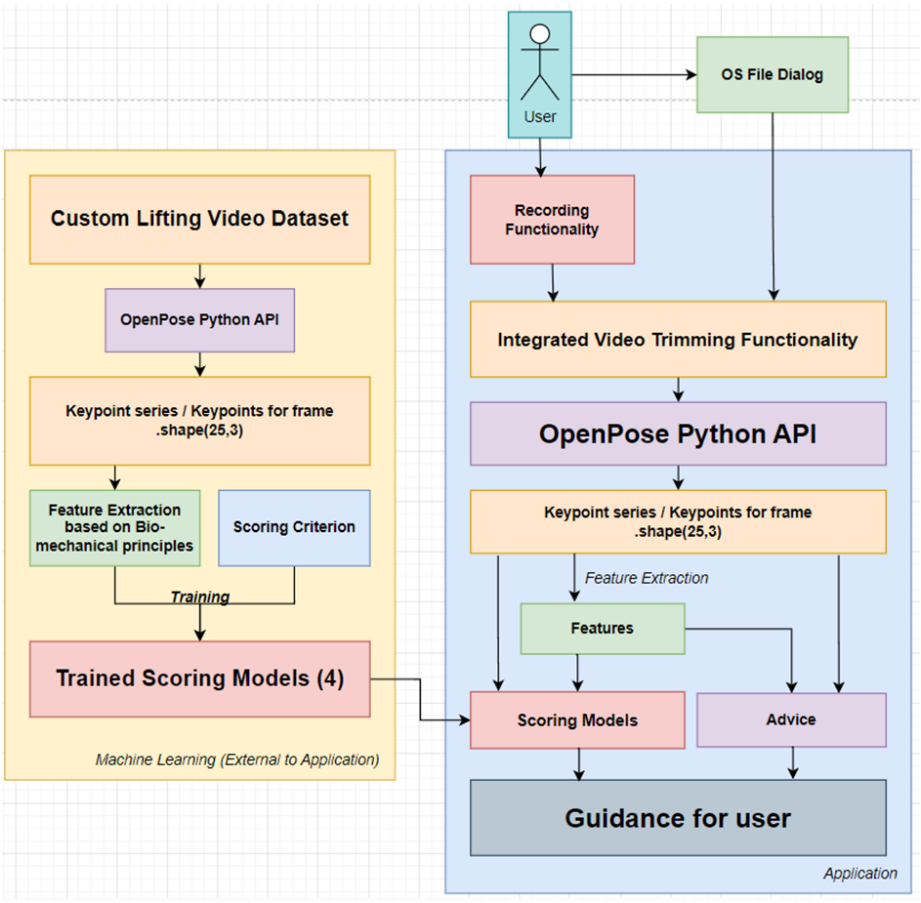


Figure 5: Component Diagram

Having performed research into the popular pose recognition and machine learning libraries available, the following libraries were chosen to be implemented within the proposed application.

OpenPose was chosen as the pose recognition library of choice due to it being the library that offered the highest degree of accuracy when utilizing its flagship BODY_25 model. Its bundled Python API also provides seamless integration with the chosen machine learning library of PyTorch. Although TensorFlow and PyTorch are similar in terms of features, PyTorch's improved ease of use and intuitive Pythonic approach was deemed more suitable for this project.

### 4.1.1 Video Input Processing

OpenPose will analyse each frame and provide data for each keypoint. There are keypoints for a variety of joints in the body, including the shoulders, hips, knees, and ankles. Data from keypoints relevant to the selected exercise will be extracted and normalised in order to be fed into a trained recurrent neural network.

### 4.1.2 Form Scoring Models

A surface level scoring will be given to the user when they perform an exercise. This scoring will be a separate system to the advice functionality and will be used exclusively to provide the user with an instant and easy to understand judgement of their form correctness, using a scoring system. The model will be trained on a dataset of lifting videos, with each video manually scored and fed into OpenPose in order to generate the corresponding keypoint data. Details on the scoring criteria for the labelling of the video examples in the created dataset can be found in the implementation section.

Feature extraction will form a crucial part of the applications ability to provide scoring and advice. Careful feature extraction will allow the neural networks to identify crucial patterns from the keypoint data. The keypoint data provides information regarding the spatial positions of important joints responsible for each exercise and provides rich source of data to perform feature extraction on.

### 4.1.3 Model Architecture

The input layer will receive the normalised keypoint data provided by OpenPose. The number of input nodes will be determined by the number keypoints specified by the OpenPose parameters provided. Normalizing the inputs will help improve training stability and will ensure consistent scaling. As the system will need to analyse a sequence of frames, the OpenPose keypoints will be arranged as a time series input, including recurrent layers in the neural network will allow it to capture time-based dependencies.

There are two types of neural network architectures that will be considered during implementation, a recurrent neural network, or a convolutional neural network. In the case of a recurrent neural network, the following types of RNN will be considered; Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). LSTMs are best used when there is the need to capture long-term dependencies and context over long-length sequences. This is done by using a memory cell which lets the network selectively remember or forget information over these long sequences. This makes them better for tasks such as natural language processing and speech recognition, where understanding context over long sequences is very important. They require more training time and computational power in comparison to GRU's as they have a much more complex layer architecture.

GRUs have a simpler architecture, utilising two gates, a reset gate, and an update gate. The simpler architecture lets GRUs be more efficient and considerably faster to train than an LSTM. They also require fewer parameters than an LSTM, which can further reduce the number of computational resources required, which is especially useful as OpenPose will already require a significant amount of system resources. The task of form analysis will likely only require the neural network to consider short-term dependencies, such as the previous frame.

As the spatial relationship between joints will be important for training the model, convolutional layers will be considered in order to identify patterns between neighbouring joints. Complex machine learning models typically require extensive datasets in order to train and generalise well. The collection and processing of such a large dataset may not be feasible within the timeframe of the project and so a convolutional neural network architecture will also be considered. Such an approach would artificially increase the size of the created dataset as the loss of a need for a coherent time series would enable each video example in the dataset to instead be decomposed into its individual frames.

### 4.1.4 Loss and Activation Functions

As the goal of this model is to provide a numeric rating of the user's weightlifting form, it appears to be a regression task. As such, it would be appropriate to use a loss function such as Mean Squared Error (MSE), Mean Absolute Error (MAE), Huber Loss or Log-Cosh Loss.

MSE measures the average squared differences between the predicted value and their true value. This lets it focus on larger errors, making it good to use when small errors are not as important. MAE measures the average absolute difference between the predicted values and their true value. This loss function is helpful when there are outliers as it is not as sensitive to large errors as MSE and is easier to interpret as the average absolute error is given.

Huber Loss combines MSE and MAE using a delta parameter. This makes it robust against outliers and provides a balance between the advantages of MSE and MAE. This loss function does, however, require the adjustment of the delta parameter. The loss functions will be experimented with during development and the most appropriate will be selected.

For the Activation Function, ReLU, Leaky ReLU, Sigmoid, SoftMax and Tanh will be considered. As the model will not be trained on an extremely large dataset, the model is unlikely to become affected by the Vanishing Gradient Problem, therefore there was not an activation function that immediately appeared to be the best. As a result, these activation functions will be experimented with during development, and again, the most appropriate will be selected.

An alternative approach outlined in the 'Computer Vision Weightlifting Coach', (SravB, 2019) involved treating this problem as a classification task and using the confidence values to calculate a scoring. Such an approach will be experimented with during implementation and would involve using binary cross entropy loss as the loss function, along with the sigmoid activation function on the output.

### 4.1.5 Advice Generation and Feedback

The keypoint data provided for each frame by OpenPose will also be used to calculate whether or not the user is positioning themselves optimally by calculating joint angles using the information extracted from keypoints. A portion of these key angles will already be obtained through the feature extraction process, and additional information can be calculated directly using the keypoint information. These values can be compared against thresholds developed based on criterion created from the knowledge gain in the research chapter on lifting bio-mechanics, in order to provide the correct advice to the user. A full breakdown of the criterion created can be found within the implementation section.

The form score provided by the neural network model, and the advice realised through the leverage calculations will then be shown to the user. Please note, as the project aims to include the functionality for multiple compound movements, a separate neural network will need to be trained for each movement, and likewise the angle calculations will need to be adjusted per lift.

## 4.2 Visual Design

A robust user interface for the application would abstract some of the complexities away from the user and so it was deemed necessary to include one. This section describes the functionality that each page must be comprised of, based on the requirements created prior, along with Figma, (Figma, Inc, 2014), wireframes that attempt to encompass the described functions of each page.

### 4.2.1 Lift Selection Menu

The user will initially be met with a menu where they are able to select the exercise that they wish to perform and receive feedback on. The design of this menu aims to continue the thematic consistency prevalent on websites within the fitness community, such as PureGym, (Pure Gym Limited, 2024).

The layout of the menu is designed to enable intuitive navigation, with exercises ordered logically based on popularity, (© StrengthLog, 2024), and grouped by muscle group, with lower body exercises in the top row, and upper body exercises in the bottom row.

Ample spacing remains between the images to reduce the feeling of a cluttered menu and to create a clean, modern aesthetic. The font used throughout the wireframes is 'Jockey One', providing a crisp, minimalist typeface.

Upon clicking on the image corresponding with the selected lift, the user will be taken to the next page.

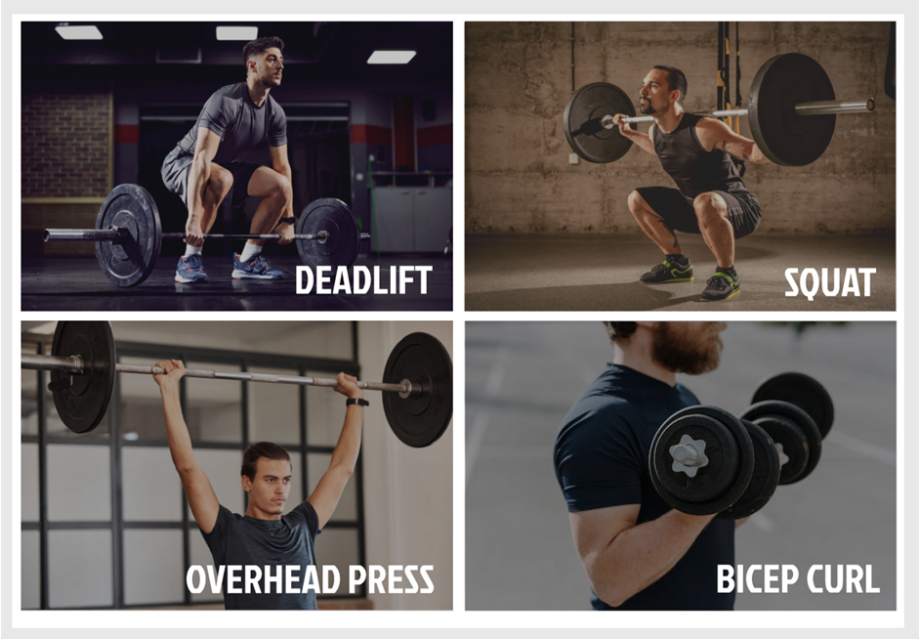


Figure 6: Lift Selection Menu Wireframe

### 4.2.2 Video Input Menu

The user will then need to select whether they would like to upload an existing video or record a new one themselves. This menu will consist of two horizontally offset buttons to create a more stylistic appearance on an otherwise simple menu. Appropriate imagery has been used within each button, alongside the well-contrasted text similar to that in the lift selection menu. During implementation, the buttons will include interactive elements such as hover effects to provide the user with visual feedback and improve visual clarity as to what elements of the page are select-able.

Upon selecting the 'Use Existing File' option, the user will be led to their file explorer where they would be able to select an MP4 file to be processed by the application. Alternatively, should the user want to record themselves performing a lift in real-time, selecting the record lift button will take the user to the live recording menu.

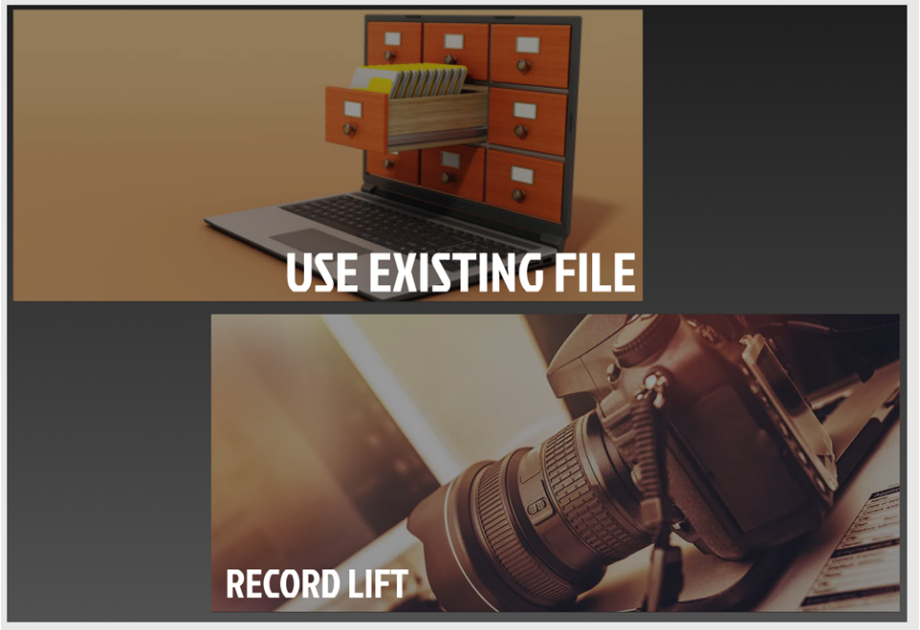


Figure 7: Video Input Menu Wireframe

### 4.2.3 Live Recording Menu

If choosing to live record an exercise, the user is shown a live feed from their device's camera. The feed from the camera takes up the majority of the screen real-estate to provide the user with a clear view of themselves whilst performing the exercise.

Positioned across the bottom of the screen resides the array of buttons available to the user. The start/stop recording button is coloured green/red based on the state of the button, with the button turning red and the text changing to 'STOP RECORDING' whilst the user is currently recording. This makes the buttons easily distinguishable and uses the visual cue of the button colour to improve menu clarity.

Once the user has finished recording, the submit button can be pressed in order to take the user to the video trimming menu. The colour of the submit button will be greyed out until the user has made a successful recording to further increase the visual clarity and quality of the user experience whilst using the application.

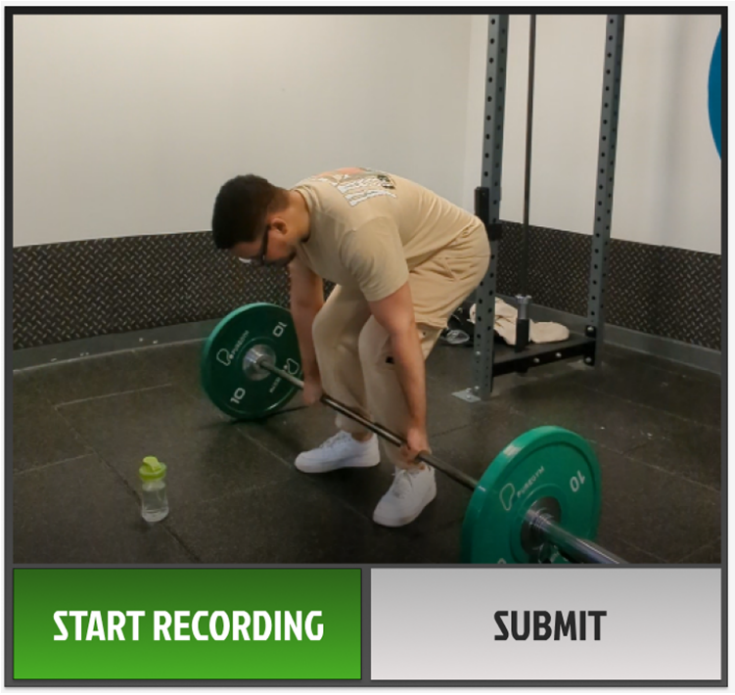


Figure 8: Live Recording Menu Wireframe

### 4.2.4 Video Trimming Menu

Once the user has submitted a video input, the user is shown a frame-by-frame slideshow of the exercise performed. The aim of this window is to allow the user to crop the start and end of the video, so the application is only processing the lift as it is being performed. The slideshow occupies the central portion of the screen, so the user is able to clearly tell what frame the exercise begins and ends on.

An array of buttons similar to the live recording menu is available, with styling consistent with the application interface thus far. Dedicated buttons exist for selecting the start and end frame of the lifting video. When selected, the process lift button will take the user to the advice screen. During implementation, an additional feature will be added to the process lift button will be greyed out until a start and end frame have been selected to avoid user confusion and provide a seamless interaction with the menu.

Existing separately from the select start/end frame and process lift buttons, the frame navigation buttons reside on the left and right of the currently displayed frame.

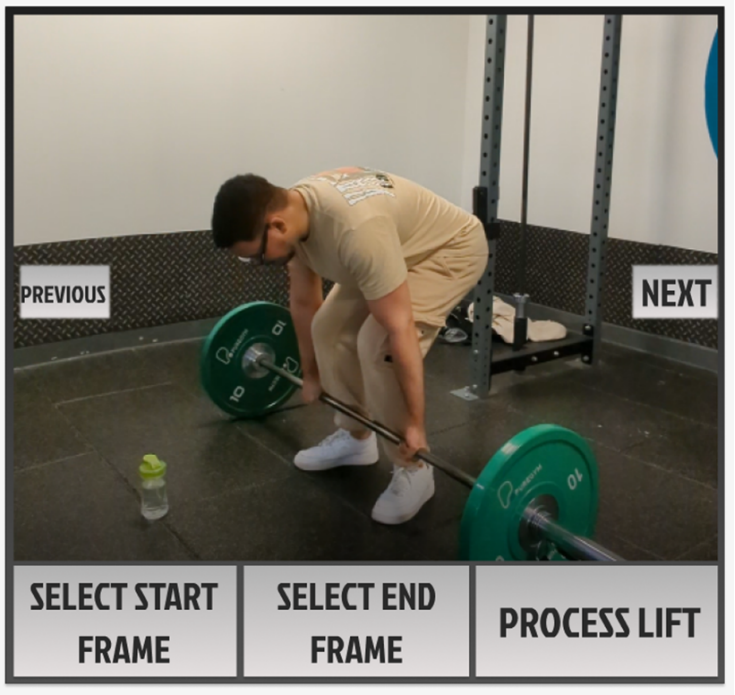


Figure 9: Video Trimming Menu Wireframe

### 4.2.5 Scoring and Advice Screen

The user interface for the scoring and advice menu has the score calculated based on the user's lift displaying prominently in the top centre of the screen to serve as an immediate visual indicator of the quality of the user's lifting technique.

The tailored advice will display alongside relevant imagery taken from the user's video input, with the image coinciding with the point during the lift that the user's mistake was made.

In the case of multiple pieces of advice being shown, they will be displayed in rows consisting of the image extracted and the associated advice. In the instance where the user is provided with too many pieces of guidance that they cannot fit on the screen without the user interface becoming cluttered, scroll functionality will be implemented to improve visual clarity.

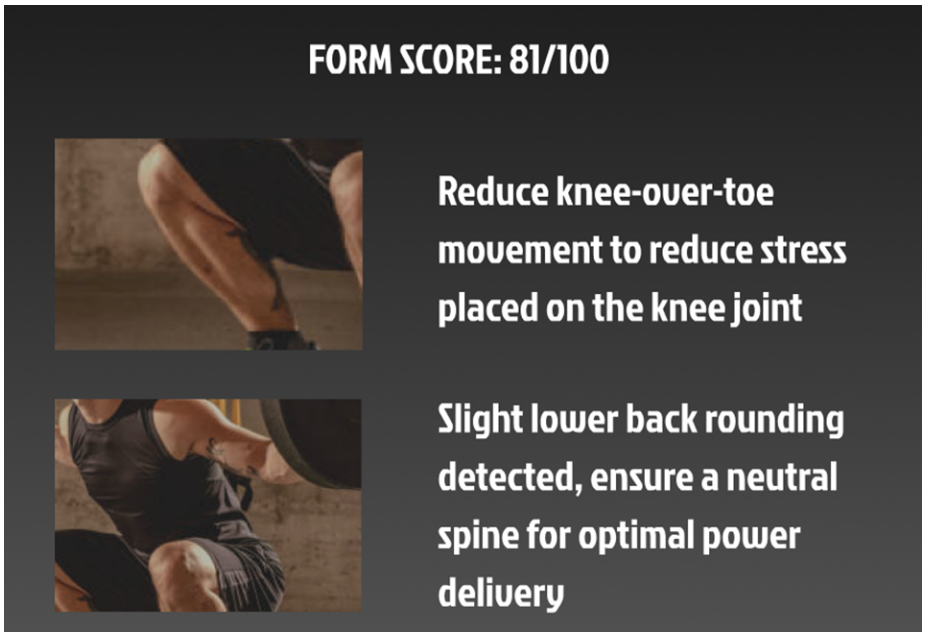


Figure 10: Scoring and Advice Menu Wireframe

Using these wireframes, the visual design of the application is able to maintain consistency with the applications stylistic decisions to provide a cohesive user experience, using colours, typeface, and design elements to provide the application with a clear sense of visual identity.

# 5 Implementation

This section outlines the stages of implementation undergone in order to develop the functionality required for the system, including dataset creation, neural network training, advice generation and development of the user interface.

## 5.1 Dataset Creation

### 5.1.1 Video Collecting

Collecting data for the scoring models involved performing a number of repetitions across the 4 designated lifting exercises. This process aimed to capture a diverse range of variations and form deviations typically found in the specific exercises. Each repetition consisted of the concentric portion of the exercise being performed, and the data extracted provided valuable insight into the nuances of form breakdown across the given lifts. This extensive dataset served as the foundation for training the machine learning models.

During data collection, focus was given to capturing the various factors that could influence the quality of a user's lifting form, such as the target muscle of the exercise being performed, suitable range of motion, as well as a variety of lifting modifiers such as different grips, stances, and repetition speeds.

Making these considerations and including them within the dataset allowed the dataset to stay diverse and ensured that the models would be able to generalize well across as broad a spectrum of users and scenarios as possible, improving the applicability of the models in real-life scenarios. By including a range of lifting techniques into the training process, the models would also become better at identifying subtle differences between correct and incorrect form, enabling them to deliver reliable assessments to users.

Analysing the lifts using OpenPose keypoints allows for the extraction of various features important in assessing a user's lifting form. The following features were selected as they each contribute to what is widely considered to be optimal form in the given exercises.

### 5.1.2 Deadlift Feature Extraction

#### Knee Angles

The angle formed at each of the knee joints can be calculated using the extracted keypoints from the ankle, knee and hip and provides insights into the degree of knee flexion, indicating the level of engagement of both the quadriceps and hamstrings, particularly at the beginning of the lift.

Excessive knee flexion may suggest inefficient force transmission, leading to potential strain on the knee joint. Insufficient knee flexion on the other hand, can

suggest reduced engagement of the quadriceps, and can often lead to a reliance on lumbar spine rounding to initiate the lift, increasing risk of injury from the higher/ shearing forces placed upon the lumbar spine.

**Hip Angle**

The angle formed at the hip joint can be calculated using the extracted keypoints from the knee, hip and neck and provides insights into the degree of hip flexion. This can be an indication of the level of engagement of the posterior chain muscles, namely the glutes, hamstrings, and spinal erectors, all of which are fundamental muscles utilised when using proper deadlift technique.

An optimal hip hinge will ensure that the posterior chain muscles are being recruited effectively, minimizing stress on the lumbar spine. An excessive hip hinge can indicate an increased risk of injury from the undue stress on the lumbar spine, whilst an insufficient hip hinge may indicate reduced glute and hamstring recruitment.

**Lateral Distance between Shoulder and Wrists**

The x-axis distance between the mid-point of the wrists and the mid-point of the shoulders can be used to offer insight into the alignment of the shoulders in relation to the working weight. With optimal form, the shoulders will stay stacked directly above the wrists to minimize stress on the upper arms and shoulders, ensuring that the target muscles are used exclusively as the movers for the lift.

A large lateral shoulder-wrist distance can be an indicator of excessive forward-backward motion or 'swinging', leading to an inefficient bar path which increases the amount of force the prime movers must generate to complete the lift, as well as causing instability about the shoulder joint from the additional stress.

**Knee Separation**

The distance between the knees throughout the lift can be an solid indicator of ankle and hip mobility, as well as proper tracking of the knees over the toes. This distance can also provide an indication of the strength of the hip abductors and adductors, responsible for moving the leg away from, and towards the midline of the body.

Excessive knee separation may suggest inadequate activation of the adductors or poor knee over toe tracking, increasing the risk of injury to the knees and impeding force transfer from the quadriceps and hamstrings.

Insufficient knee separation or 'knee cave' can suggest inadequate activation of the abductors and glutes, leading to similar effects to excessive knee separation, as well as general instability throughout the execution of the lift.

**Elbow Angles**

The angle formed at the elbow joints can be calculated using the extracted keypoints from the shoulders, elbows and wrists and provides insights into the degree of elbow flexion.

Optimal elbow positioning ensures efficient force transfer from the posterior chain into the barbell whilst minimizing strain on the biceps and elbow joint. An optimal elbow angle would be as close to full elbow extension as possible, forming a straight and vertical line from the shoulder joint, through the elbow and into the wrists.

**Vertical Distance between Shoulder and Hip**

The y-axis distance between the shoulders and hips can be used to offer insight into the degree of torso tilt relative to standing, whilst also being a stronger indicator of spinal flexion throughout the lift.

Maintaining a relatively consistent vertical alignment helps to distribute load evenly across the spine. With optimal form, the vertical distance between the shoulder and hips should stay fairly consistent, increasing slightly as the lifter approaches lockout.

An insufficient vertical distance between the shoulder and hip often indicates a lack of knee flexion as the hips remain high relative to the shoulders and increases the risk of excessive stress and injury to the lumbar spine.

### 5.1.3 Squat Feature Extraction

**Knee Angles**

The angle formed at each of the knee provides insights into the degree of knee flexion, indicating the level of engagement of both the quadriceps. Optimal squat form involves maximising knee flexion so as to place the quadriceps under as deep as stretch as possible, increasing the hypertrophic response caused by the exercise as well as strengthening the knees.

Insufficient knee flexion may suggest the user is performing what is colloquially known as a 'half-squat', reducing the beneficial effects of the exercise and ,in the context of competitive weightlifting or powerlifting, is considered a failed lift.

**Hip Angle**

The angle formed at the hip joint provides insights into the degree of hip flexion. This can be an indication of the level of engagement of the posterior chain muscles, particularly the glutes, and spinal erectors, which are the main muscles responsible for keeping the body upright as the user descends into the squat.

An optimal hip angle allows the user to remain as upright as possible during the squat to reduce the shearing forces placed on the lumbar spine.

Depending on an individual's femur-torso proportions, it may be difficult to maintain an upright posture. To facilitate a range of body types, hip angle will only be marked detrimental when it is deemed very small, indicating that the user is folding forwards underneath the working weight, leading to instability. This is often associated with an inefficient bar path, as the forward lean introduces a new axis of movement for the working weight, causing inefficient power transfer into the bar.

**Knee Separation**

Similar to the deadlift, the distance between the knees throughout the squat can be an indicator of hip mobility, as well as proper tracking of the knees over the toes. This distance can also provide an indication of the strength of the hip abductors and adductors, responsible for moving the leg away from, and towards the midline of the body.

Excessive knee separation may suggest inadequate activation of the adductors or poor knee over toe tracking, increasing the risk of injury to the knees and causing impeded force transfer from the quadriceps and glutes.

Insufficient knee separation or 'knee cave' can suggest inadequate activation of the abductors and glutes, leading to similar effects to excessive knee separation, as well as instability throughout the execution of the lift.

**Vertical Distance between Hip and Knees**

The y-axis distance between the hips and knees can be used to offer insight into the depth of the individual's squat.

Optimal squat form typically entails a femur that reaches parallel in relation to the floor, meaning the y-axis distance between the hips and knees can be used to calculate whether or not the user has reached an appropriate depth, and is performing a 'half-squat'.

**Stance Width**

The user's stance width can be calculated using the extracted keypoints from the left and right ankles. This can be compared with the shoulder width of the user, calculated using the information extracted from the left and right shoulder keypoints, to provide insight into the width of a user's stance relative to the size of their torso.

Standard squat form typically involves using placing the feet between 1-1.5x shoulder width apart. Whilst there are examples of longer femur lifters having preference for a wider squat stance, this range should comfortably encompass

the majority of individuals in terms of optimal foot placement.

An excessively narrow stance-width reduces the stress placed on the glutes whilst increasing the stress placed on the quadriceps which, whilst beneficial for those prioritising quadriceps development, places extra stress on the knee joint and can increase risk of injury. On the other extreme, an excessively wide squat stance places the hip adductors under a greater load and makes it more difficult to properly track the knees over the toes.

### 5.1.4 Overhead Press Feature Extraction

#### Knee Angle

The angle formed at each of the knee provides insights into whether the individual is using additional leg drive during the pressing motion.

Optimal overhead pressing form utilizes a slight bend in the knees to create additional stability whilst performing the lift. This bend in the knees should be minimal and will likely be indistinguishable from locked out knees due to the obscuring factor of the user's clothing, but it is worth nothing that locking out of the knees can cause slight instability.

An excessive knee bend during the lift indicates the lifter is using their legs to generate additional momentum to propel the weight upwards. Whilst this technique is often used in competitive weightlifting, for example, in the Clean & Jerk, in this context, the added movement creates an additional risk of injury, as well as reducing the stress placed on the front deltoid and triceps, the prime movers in the exercise.

#### Hip Angle

The angle formed at the hip joint provides insights into the degree of hip extension. This can be an indication of the level of engagement of the abdominal muscles, and the spinal erectors, which keep the torso stable throughout the lift.

A large hip angle indicates that the user is keeping their torso sufficiently upright, ensuring that the shoulders stay stacked above the hips for maximum force transfer into the bar.

An insufficient hip angle could be an indicator that the user is arching or leaning, both of which can place additional stress on the lumbar spine, as well as introducing general instability when performing the pressing motion.

#### Lateral Distance between Shoulder and Wrists

The lateral distance between the shoulders and the wrists can be used to determine whether the user is keeping the working weight reasonably close to the body during the initiation of the lift. It can also be an indicator of proper

lockout positioning.

Optimal lifting form involves keeping the wrists stacked over the shoulder joint throughout the pressing motion. Excessive distance can indicate imbalance, asymmetrical loading, or an inefficient bar path, leading to increase shoulder strain, risk of injury and reduced force transfer.

**Vertical Distance between Shoulders and Wrists**

Analysing the vertical displacement between the wrist and shoulder joints allows the models to identify proper range of motion for a repetition.

Optimal pressing form includes having the wrists as close to the height of the shoulder joint as possible to maximise the stretch placed on the front deltoid. An insufficient vertical distance between the shoulders and wrists can be an indicator of a reduced range of motion, which could also imply the individual is attempting to use a weight that is too heavy for them to handle safely.

**Elbow Angle**

The angle formed at the elbow joints can be calculated using the extracted keypoints from the shoulders, elbows and wrists and provides insights into the degree of elbow extension.

In proper overhead pressing form, the aim of the movement is to maximise elbow flexion at the bottom of the motion and maximise elbow extension at the top of the movement. This allows for a deep stretch on the triceps, one of the prime movers in the shoulder press. An insufficient elbow angle at the top of the movement can be an indicator that the user has failed to approach the lockout position to maximise the stress placed on the triceps. Too great an elbow angle at the bottom of the movement can be another indicator, alongside the feature listed prior, that the user has is performing the exercise with a limited range of motion.

**Shoulder Angle**

The angle formed at the shoulder joints can be calculated using the extracted keypoints from the elbows, shoulders, and hips to provide insights into shoulder alignment and stability throughout the lift. This angle can be used to determine whether the user is performing the exercise with sufficient range of motion, as well as identifying pressing asymmetry.

**Grip Width**

The user's grip width can be calculated using the extracted keypoints from the left and right wrists. This can be compared with the shoulder width of the user, calculated using the information extracted from the left and right shoulder keypoints, to provide insight into the width of a user's grip relative to their shoulders.

With optimal form, the wrists stay slightly wider that shoulder width apart, approximately 1-1.2x shoulder width. An overly narrow or wide grip width can cause additional stress on the wrist joint as it becomes increasingly difficult to keep the elbows stacked directly underneath the wrists. An excessively wide grip width can also cause additional elbow flare, placing the shoulder joint itself under unnecessary stress, increasing the risk of injury.

### 5.1.5 Barbell Curl Feature Extraction

**Elbow angle**

The angle formed at the elbow joints can be calculated using the extracted keypoints from the shoulders, elbows and wrists and provides insights into the degree of elbow extension, and therefore, the completeness of any given repetition based on the starting and ending angle of the elbows.

With proper form, the aim of the barbell bicep curl is to maximise elbow flexion at the top of the motion and maximise elbow extension at the bottom of the movement to create a stretch on the bicep. An insufficient elbow angle at the bottom of the movement can be an indicator that the user has cut short the eccentric portion of the repetition, reducing bicep muscle engagement. Similarly, too great an elbow angle at the top of the movement can be an indicator of a restricted range of motion.

**Grip Width**

Similarly to the overhead press, the user's grip width can be calculated using the extracted keypoints from the left and right wrists. This can be compared with the shoulder width of the user, calculated using the information extracted from the left and right shoulder keypoints, to provide insight into the width of a user's grip relative to their shoulders.

With optimal form, the wrists stay slightly wider that shoulder width apart, approximately at shoulder width, aligning closely with the shoulders provides favourable leverages for the lift, whilst also increasing stability. An overly narrow or wide grip width can cause additional stress on the wrist joint and forces the shoulders to perform compensatory internal/external rotation.

**Shoulder Angle**

The angle formed at the shoulder joints can be calculated using the extracted keypoints from the elbows, shoulders, and hips to provide insights into shoulder alignment and stability throughout the lift.

With ideal lifting form, the angle of the shoulder should stay consistent throughout the lift, ensuring that the target muscle is solely utilized to perform the exercise. Movement of the upper arms and excessive shrugging can be identified through the angle of the shoulder and may indicate muscular imbalance, instability and increase risk of shoulder injury.

**Torso Angle**

The angle of the torso relative to vertical can be calculated using the hip, neck and an imaginary keypoint directly below the neck.

With ideal lifting form, the torso should stay as upright as possible, maintaining a neutral spine throughout. An excessive torso angle relative to vertical can indicate inadequate core bracing and a lack of stability. It may also be an indicator of the user performing compensatory movement to make up for muscular weakness, which detracts from the goal of the exercise, being maximal bicep tension.

### 5.1.6 Form Scoring Criteria and Labelling

Each video undergoes a labelling process in which a label is allocated to the lift, representing a score assessed using a deduction-based scoring system, initially starting with the maximum score of 10 points, and deducting points for errors or mistakes identified in the users lifting form, down to a minimum score of 1.

A form scoring criteria was developed for each exercise, utilising the features extracted from the dataset, in addition to standard lifting principles, in order to create a system that minimizes subjectivity in the labelling process.

**Deadlift Form Scoring Criteria**

| ERROR / MISTAKE | POINTS DEDUCTED |
|---|---|
| Lumbar spine flexion | 3 – 5 (based on severity of curvature) |
| Thoracic spine flexion | 1 |
| Insufficient knee flexion | 2 |
| Excessive knee flexion | 1 |
| Elbow flexion | 1 |
| Hands not under shoulder joint | 1 |
| Knee cave | 2 |

Table 1: Deadlift Scoring Criteria

**Squat Form Scoring Criteria**

| ERROR / MISTAKE | POINTS DEDUCTED |
|---|---|
| Lumbar spine flexion | 2 – 4 (based on severity of curvature) |
| Insufficient range of motion | 3 |
| Knee cave | 2 |
| Excessive forward lean | 2 |
| Excessively narrow stance | 2 |
| Excessively wide stance | 2 |
| Inefficient bar path | 1 |

Table 2: Squat Scoring Criteria

**Overhead Press Form Scoring Criteria**

| ERROR / MISTAKE | POINTS DEDUCTED |
|---|---|
| Insufficient range of motion | 3 |
| Use of leg drive | 2 |
| Excessive backwards lean | 2 |
| Excessively narrow grip width | 2 |
| Excessively wide grip width | 2 |
| Inefficient bar path | 1 |

Table 3: Overhead Press Scoring Criteria

**Barbell Curl Form Scoring Criteria**

| ERROR / MISTAKE | POINTS DEDUCTED |
|---|---|
| Insufficient range of motion | 3 |
| Excessive backwards lean | 2 |
| Excessively narrow grip width | 2 |
| Excessively wide grip width | 2 |
| Compensatory movement (E.g. swinging) | 2 |
| Movement of elbow joint | 1 |

Table 4: Barbell Curl Scoring Criteria

## 5.2 Neural Network Training

### 5.2.1 Model Architecture Approaches

Three different approaches were considered when developing the form scoring system were explored, each utilizing different neural network architectures. Each approach leveraged the same keypoint dataset but used different features to attempt to generate a form scoring.

The first approach was to create a regression model utilizing an RNN (Recurrent Neural Network). RNNs are very well suited for sequential data analysis, providing an option for analysing the video sequences found in the dataset. The aim of this approach was to use the RNNs capacity for recognising patterns with temporal dependencies to help generate a score representative of the lift provided.

The second approach was to instead create a classification model utilizing an RNN. The initial dataset would undergo data augmentation on the labels, with video examples scoring an 8 or above being classified as 'good' and lift examples scoring a 7 or below being classified as 'bad'. The confidence score for the classification could then be used to generate a form scoring for the lift provided.

The final approach considered was a simplified regression model utilizing a CNN (Convolutional Neural Network) instead of a RNN. This approach would instead split each video entry in the dataset into a number of frames split evenly across the video input, with each frame being assigned the scoring of the video entry it was extracted from. When generating a form score, the average score of all frames in the video would be used as the form scoring for the lift. The reasoning behind this approach was that with the size of the dataset being between 200-300 video examples per exercise, having video consist of a number of frames each of between 4 and 10 features may impede the model's ability to learn due to the small dataset. Artificially increasing the size of the dataset would allow the model to stay accurate despite losing the ability to recognise temporal patterns in the data. As an example, this process increases the example/feature ratio of the deadlifting dataset from around 1.5 to around 600.

Each of these approaches offered different advantages and trade-offs in terms of complexity and performance. In the following section, deeper insights into the experimentation and implementation process of each approach will be discussed.

### 5.2.2 Regression Model (RNN)

To follow this approach, the datasets had to be arranged a way such that each input to the model consisted of a series of frames, with each in turn, consisting of a number of features extracted. The model architecture experimented with in order to implement this approach consisted of the following:

- A series of LSTM layers to facilitate sequence modelling. The LSTM layers were bi-directional in order to identify dependencies from previous and future frames.
- A series of fully connected layers of varying sizes in order to allow the model to recognise complex patterns in the data provided. The size of each fully connected layer was experimented with during training.
- Batch normalization layers were used following each fully connected layer. These were implemented to speed up the training process and make the model resilient to the vanishing gradient problem.
- Dropout layers following each batch normalization layer. Dropout layers prevent overfitting by randomly setting a percentage of the inputs to zero during training. Different values for the probability of the dropout were experimented with during training.

Different configurations for neural network were experimented with, modifying the size of hidden layers, order of layers, number of layers, as well as additional hyperparameter settings, however this type of model seemed incapable of recognizing useful patterns in the data provided.

This approach was unable to succeed likely due to size limitations of the dataset relative to the number of features provided. With between 200-300 video examples in each dataset, each consisting of 20 frames of between 4-10 features, the insufficient dataset led to overfitting with all network configurations, having the model predict the same value regardless of the input. Below is one of many examples of the failed training of the models.

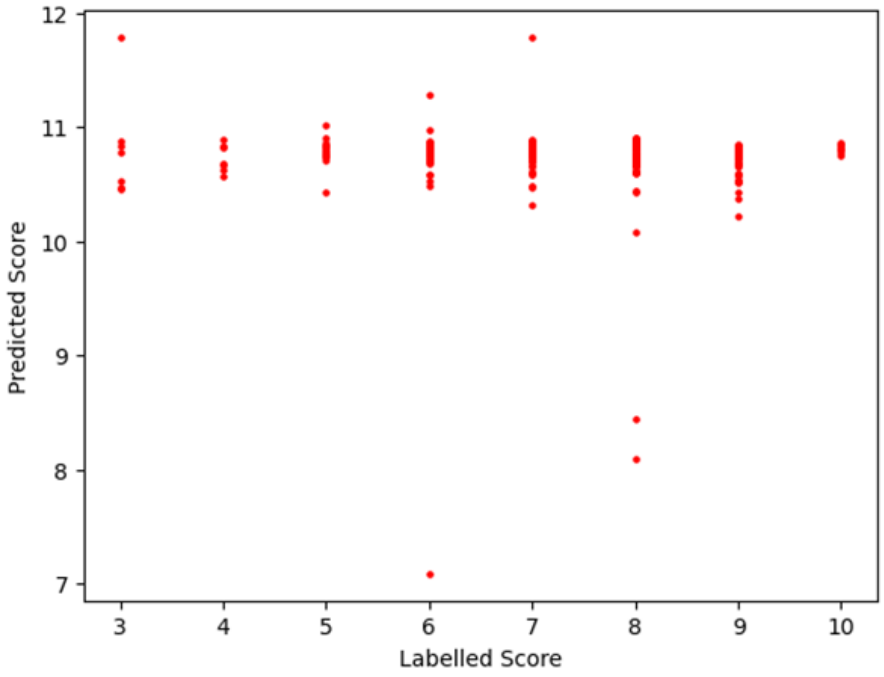


Figure 11: Graph depicting performance of Regression RNN

### 5.2.3 Binary Classification Model (RNN)

To follow this approach, the dataset was arranged in a way similar to the first approach. The labels for each video example were modified in a way such that any scores of 8 and above were classified as 'good' and any scores of 7 or below were classified as 'bad'. This converted the regression problem into a classification problem. This approach would then use a model architecture similar to the previous approach, but instead of a continuous value output from the model, the confidence value in the lift being 'good' would be used to create a score.

Similarly to the first approach however, this neural network configuration was again incapable of learning to recognise patterns in the data, likely due to the size limitations of the dataset.

Instead of simply overfitting, this approach to configuring the network seemed to produce a model that was simply picking a random value, likely due to the fact that it was unable to predict the same score every time due to the nature of the training labels being restricted to binary classes. If the model was learning as intended, one would expect to see a positive correlation between the predicted score and the true score.

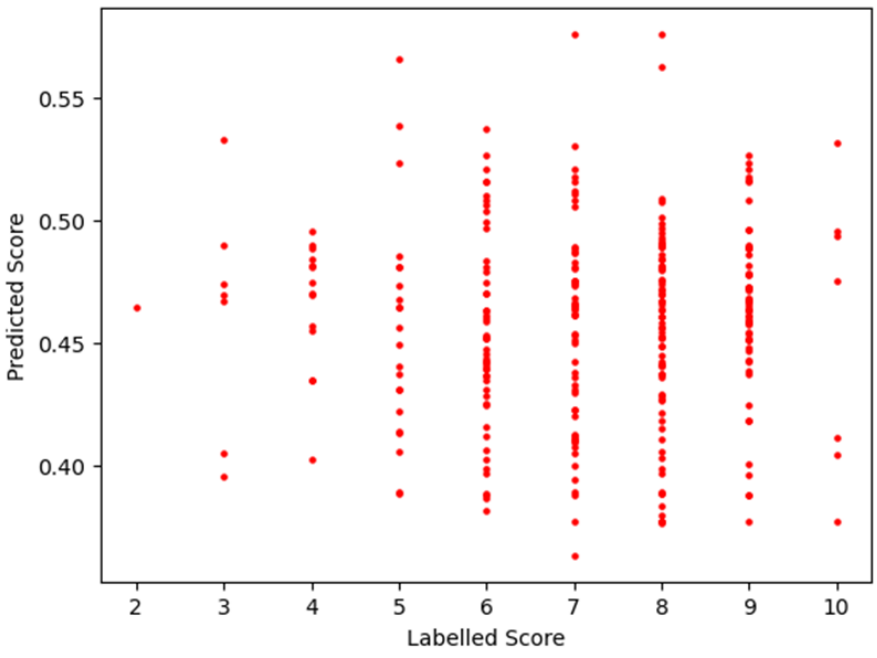


Figure 12: Graph depicting performance of Binary Classification RNN

### 5.2.4 Regression Model (CNN)

For the final approach, the datasets were rearranged in a way such that each video example was broken apart into its individual frames. Each frame taken from a video example was assigned the same scoring as the video example it was extracted from. When passing a video through the model, it would also be broken down into its individual frames, and the mean score from all frames in the series would then serve as the score for the lift.

This approach removes the ability for the network to include LSTM layers as each frame is passed individually and not as a time series. Whilst this means that the model is unable to recognise temporal dependencies, this form of data augmentation allows the size of the training dataset to increase 20x over, whilst reducing the amount of input features by 20x as well. This simplified approach nullifies the issues caused by training with a small dataset, as seen in the previous approaches.

In order to implement this approach, the following model architecture was developed:

- A series of convolutional layers to facilitate special pattern recognition.
- Batch normalization layers following each convolutional layer to accelerate training and remove concerns regarding the vanishing gradient and exploding gradient problem.
- A series of fully connected layers of varying sizes in order to allow the model to recognise complex patterns in the data provided. The size of each fully connected layer was experimented with during training.
- ReLU activation layers after each batch normalization layer / fully connected layer to add additional non-linearity to the model and to further allow it to recognise complex patterns.
- Dropout layers following each ReLU activation layer to prevent overfitting by randomly setting a percentage of the inputs to zero during training.

The baseline model created utilising this approach was capable of recognising patterns in the data and was able to make a correlation, albeit fairly weak, between the predicted score and the labelled score, with a spread that could be improved on through hyperparameter tuning. The model consistently over-predicted poor lifts, and under-predicted quality lifts, likely because the majority of the training examples consisted of lifts that scored in the region of 6-7. This was of little concern as, if the issue was irremediable through hyperparameter tuning, the scores could instead be normalised in the range 5-10 and redistributed across a 1-10 scale. Despite its shortcomings, this approach was the most successful by a significant margin and so it was decided that this would be the model architecture used in the scoring system.

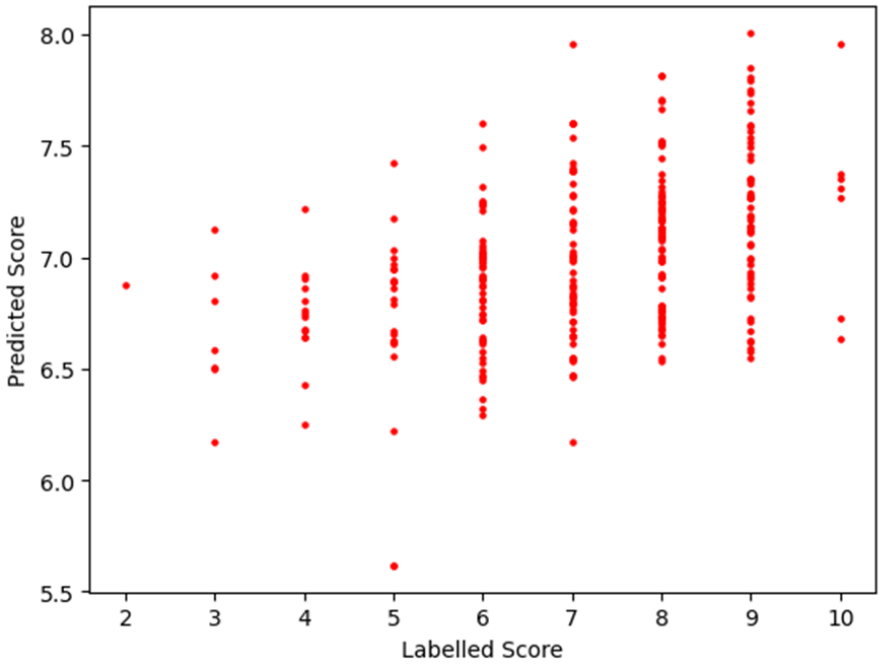


Figure 13: Graph depicting performance of Regression CNN

#### 5.2.5 Hyperparameter Tuning

Hyperparameter tuning was performed using an exhaustive grid search. Whilst typically very computationally expensive, this cost was reduced by selectively choosing which hyperparameters should be optimised in the grid search.

The following hyperparameters and the values included in the grid search are as follows:

| Hyperparameter | Search space | Optimal value |
|---|---|---|
| Hidden Size | [32, 64, 128] | 32 |
| Dropout Chance | [0.25, 0.5] | 0.5 |
| Learning Rate | [0.001, 0.005, 0.01] | 0.001 |
| Number of Epochs | [20,40,60,80,100] | 40 |
| Patience | [5,10,15] | 5 |
| Batch Sizes | [32, 64, 128] | 128 |

Table 5: Table of hyperparameter search space

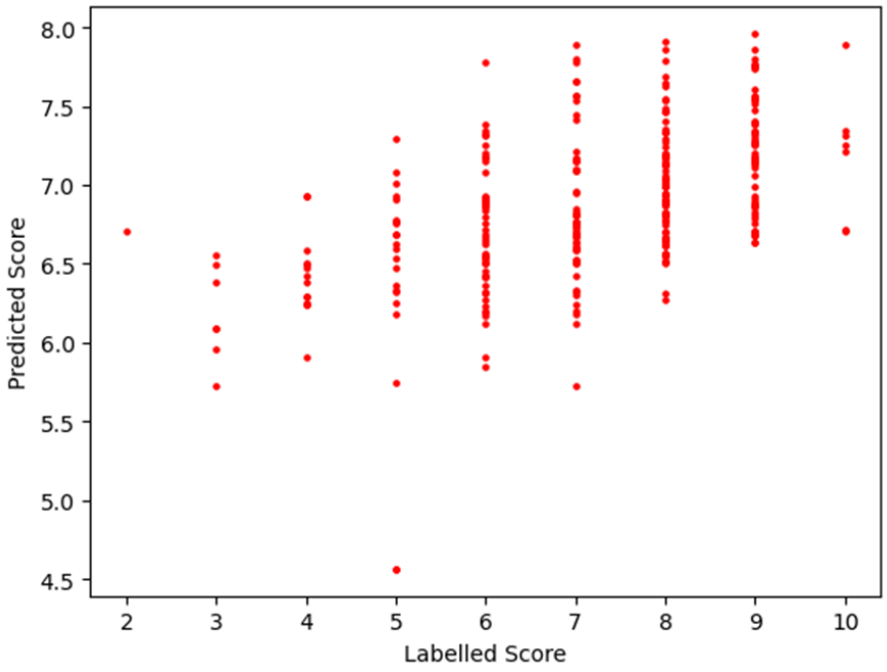


Figure 14: Graph depicting performance of tuned model

Hyperparameter tuning reduced the spread of values and slightly improved the strength of the correlation between the predicted score and the true score, however the model continued to under-perform slightly. The lack of access to a sizeable dataset restricted the model's ability to recognise the complex patterns needed improve the accuracy of the model to the extent initially intended.

In addition to this, despite efforts to reduce subjectivity in the labelling process via a stringent scoring criterion, inconsistencies in the labelling of the video examples remained, causing some large outliers, and overall decreasing the accuracy of models. The models are also heavily dependent on the recording angle used, with best performance at approximately a 15-25° to the user. With anecdotal testing however, the model was capable of differentiating between badly performed exercises, reasonably well performed exercises, and very well performed exercises.

### 5.2.6 Score Calculation

Due to the models providing a score predominantly between the values of 6 and 8, in order to create a reasonably scaled metric, the values will be normalised in the range of 6 to 8 across a scale of 1-10. This provides a more noticeable change in the score predicted and has the scoring system performing more typically to how a user would expect it to behave.

```
def calculateDeadliftScore(features):
    current_dir = os.path.dirname(os.path.abspath(__file__))  # Get the current directory of the script
    model_path = os.path.join(current_dir, '..','deadliftRegressionModelTuned.pth')
    model = torch.load(model_path)

    # Create DataLoader
    dataset = TensorDataset(torch.Tensor(features))
    loader = DataLoader(dataset, batch_size=20, shuffle=False)

    model.eval()
    with torch.no_grad():
        for data in loader:
            inputs = data[0]
            outputs = model(inputs)  # pass input data through model
        score = torch.mean(outputs, dim=0).item()

    # normalize outputs from range 6-8 into a 1-10 scale
    if (score <= 6):
        score = 0
    elif (score >= 8):
        score = 10
    else:
        score = ((score - 6) / 2) * 10
    return score
```

Figure 15: Score calculation code snippet

## 5.3 Advice Generation

One of the core functionalities specified for the system was to provide the user with feedback based on a lift. A criterion was established based on widely accepted lifting principles in bio-mechanics and exercise science, and this criterion was used to determine breakpoints for what constitutes good and bad lifting form. These breakpoints would then serve as the guidelines for whether an aspect of the user's lifting form is acceptable, and if not, would trigger a message informing the user of their mistakes. The thresholds for acceptable values were obtained by comparing examples of what would be considered good and bad lifting form (based on the details specified in Chapter 5.1), and measuring the angles made at each joint / the spatial relation of specific joints throughout the lift. In order to compensate for possible inaccuracies in the keypoint information received from the pose recognition libraries, the range of acceptable values for each feature were expanded to provide a reasonable degree of leniency.

The data needed to compare the user's lifting form to the criterion was created by using the features extracted previously when creating the model architecture, as well as using additional spatial information taken from the keypoints identified by OpenPose.

### 5.3.1 Deadlift Criterion

| Aspect of Lift | Acceptable Values | Associated Message(s) |
|---|---|---|
| Wide Grip-Shoulder width ratio. | $0.8 < x < 1.6$ | 'Grip width is too wide.' |
| Narrow Grip-Shoulder width ratio. | $0.8 < x < 1.6$ | 'Grip width is too narrow.' |
| Insufficient knee flexion. | $120° < x < 160°$ | 'Try to incorporate more leg drive.' |
| Excessive knee flexion. | $120° < x < 160°$ | 'Hinge further about the hips to allow yourself to reach the bar without excessive knee bend.' |
| Insufficient elbow extension. | $155° < x$ | 'Keep both arms straight.' |
| Low ratio of knee separation throughout lift compared to lockout. | $0.6 < x < 1.4$ | 'Significant knee cave detected.' |
| High ratio of knee separation throughout lift compared to lockout. | $0.6 < x < 1.4$ | 'Sumo deadlift detected. Please note, this system is designed for conventional.' |
| Shoulder-wrist lateral distance. | $x < 70$ | 'Try to keep the bar underneath the shoulders. Avoid swinging the bar laterally.' |
| Shoulder-hip vertical distance at start. | $x < 20$ | Starting hip position is too high. Hips should be higher than knees and lower than shoulders. This may also indicate back rounding. Maintain a neutral spine. |

Figure 16: Table of deadlift criterion

### 5.3.2 Squat Criterion

| Aspect of Lift | Acceptable Values | Associated Message(s) |
|---|---|---|
| Barbell path lateral movement. | x < 30 | 'Try to keep the bar path vertical.' |
| Insufficient lockout knee extension. | x > 150 | 'Try to get closer to lockout.' |
| Excessive hip flexion. | x > 70° | 'Try to stay more upright and avoid folding forwards.' |
| Hip-knee vertical distance. | x < 30 | 'Try to break parallel and squat deeper for a better stretch on the quads.' |
| Low ratio of knee separation throughout lift compared to lockout. | x < 0.8 | 'Significant knee cave detected.' |
| High ratio of foot separation to shoulder width. | 0.8 < x < 1.3 | 'Squat stance is too wide.' |
| Low ratio of foot separation to shoulder width. | 0.8 < x < 1.3 | 'Squat stance is too narrow.' |

Figure 17: Table of squat criterion

### 5.3.3 Overhead Press Criterion

| Aspect of Lift | Acceptable Values | Associated Message(s) |
|---|---|---|
| Wide Grip-Shoulder width ratio. | $1.4 < x < 2.8$ | 'Grip width is too wide.' |
| Narrow Grip-Shoulder width ratio. | $1.4 < x < 2.8$ | 'Grip width is too narrow.' |
| Excessive knee flexion. | $x > 160°$ | 'Try to keep your legs straight to maximise shoulder recruitment.' |
| Elbow flexion at lockout. | $x > 125°$ | 'Try to get closer to lockout.' |
| Hip angle throughout lift. | $x > 160°$ | 'Try to keep hips stacked underneath shoulders and avoid leaning.' |
| Shoulder-wrist lateral distance. | $x < 120$ | 'Try to keep the bar path vertical. Avoid pushing the bar laterally.' |
| Shoulder-wrist vertical distance. | $x < 20$ | 'Start the rep closer to the height of your shoulders for a better stretch.' |

Figure 18: Table of overhead press criterion

### 5.3.4 Barbell Curl Criterion

| Aspect of Lift | Acceptable Values | Associated Message(s) |
|---|---|---|
| Wide Grip-Shoulder width ratio. | 0.7 < x < 1.13 | 'Grip width is too wide.' |
| Narrow Grip-Shoulder width ratio. | 0.7 < x < 1.3 | 'Grip width is too narrow.' |
| Torso angle relative to vertical. | x < 5° | 'Keep your core stable and upright, swinging and/or leaning detected.' |
| Starting elbow angle at start of lift. | x > 145° | 'Make sure to extend arms close to being locked out in the starting position.' |
| Elbow angle at end of lift. | x < 45° | 'Partial range of motion detected. Make sure to squeeze at the top.' |

Figure 19: Table of barbell curl criterion

### 5.3.5 Comparing with Criterion

The data extracted from the provided video consists of the following; A features array, containing the values of the features extracted for each of the 20 frames taken from the video, of shape (20,num_features). A keypoints array, containing the x, y values for the spatial location of the joint keypoints from the video provided.
Whilst the features are typically used for the neural network pass, some of their values are also used whilst checking against the criterion for the lift. An example criterion check would look like the following:

```
# check for knee cave / inconsistent knee space throughout lift
lockout_knee_width = features[len(features) - 1][4]
for i in range(len(features)):
    knee_width = features[i][4]
    # check knee cave
    if (knee_width / lockout_knee_width) < 0.6:
        advice.append("Significant knee cave detected.")
        issue_frames.append(frames[i])
        break
    # check sumo style
    if (knee_width / lockout_knee_width) > 1.4:
        advice.append("Sumo deadlift detected. Please note, this system is designed for conventional.")
        issue_frames.append(frames[i])
        break
```

Figure 20: Example threshold comparison for advice generation

Due to the lifts using separate criterion, each exercise has its own extract advice function. Each piece of advice is saved into an array that is then returned at the end of the function call. This array can then be utilised to retrieve data and display the advice to the user via the user interface.

An issue to note is that both the scoring system and advice system are entirely dependent on the keypoint information extracted from the video input being substantial enough to be able to perform the various distance and angle calculations necessary to pass through the model and also to compare against the exercise criterion. As such the system requires that the user ensure that no keypoints are hidden whilst performing the exercise, otherwise causing an error with the two systems in place. A catch case was therefore added when processing the video input to inform the user of such an issue.

## 5.4 User Interface

The user interface for the application consists of 4 sections, a main menu which acts as the landing interface when the application is run, a submenu to provide the user with an option as to how they wish to input their video data into the system, an interface in which a user can choose to live record and trim the recording, and a final splash screen that displays the user's score and advice, along with close-ups of the problem area(s) in their form.

### 5.4.1 Main Menu

The design for the main menu was for it to consist of a quadrant of stylised images. As the high-fidelity prototype was created using Figma, it made it possible to import each quadrant component, including its styling and associated text, as an image. The user would then be able to click on the image representative of their chosen life, in order to be taken to the next screen.

In order to display the images as quadrants, a window size for the application had to be declared, and then using height and width of the window, resize the images to have dimensions half that of the window. Each resized image could then be allocated a location within a 2x2 grid, comprising the full window.

```
# Create the application window
app = Tk()
app.title("Form Analysis System")
app.configure(bg='#444745')
app.option_add("*Font", ('Jockey One', 12))
# Set window dimensions
window_width = 1600
window_height = 900
app.geometry(f"{window_width}x{window_height}")

# Resizing all images
squatImage = resize_image("assets/squat.jpg", window_width // 2, window_height // 2)
deadliftImage = resize_image("assets/deadlift.jpg", window_width // 2, window_height // 2)
ohpImage = resize_image("assets/ohp.jpg", window_width // 2, window_height // 2)
curlImage = resize_image("assets/curl.jpg", window_width // 2, window_height // 2)
uploadImage = resize_image("assets/upload.jpg", 1000, (window_height // 2) - 1)
recordImage = resize_image("assets/record.jpg", 1000, (window_height // 2) - 1)
instructionsImage = resize_image("assets/instructions.png", 500, 260)
instructionsDemoImage = resize_image("assets/instructionDemo.png", 500, 600)

# Defining all labels
upload = Label(app, image=uploadImage, cursor="hand2")
record = Label(app, image=recordImage, cursor="hand2")
squat = Label(app, image=squatImage, cursor="hand2")
deadlift = Label(app, image=deadliftImage, cursor="hand2")
ohp = Label(app, image=ohpImage, cursor="hand2")
curl = Label(app, image=curlImage, cursor="hand2")
instructions = Label(app, image=instructionsImage, bg='#444745')
instructionsDemo = Label(app, image=instructionsDemoImage)

displayMainMenu()

app.mainloop()
```

Figure 21: Code snippet for main menu

Once the images had been displayed, the functionality to be able to select a lift still needed to be added. This was achieved by creating a function that would save the user's choice of exercise, before leading the user to the video input method selection screen. This function could then be bound to the labels on the main menu, passing in a parameter based on the selected lift. The labels responsible for displaying the imagery in each quadrant could then be hidden in preparation for displaying the following video input method selection menu.

```
# Bind click events to the labels
squat.bind("<Button-1>", lambda event: selectLift('squat'))
deadlift.bind("<Button-1>", lambda event: selectLift('deadlift'))
ohp.bind("<Button-1>", lambda event: selectLift('ohp'))
curl.bind("<Button-1>", lambda event: selectLift('curl'))
```

Figure 22: Code snippet for binding click events

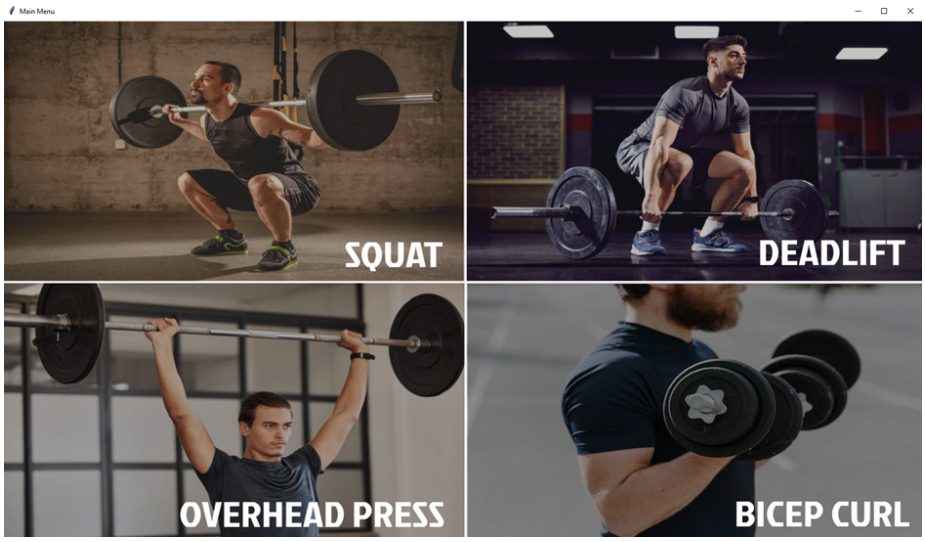


Figure 23: Main menu user interface

### 5.4.2 Video Input Selection Menu

The design for the video input selection menu consists of two selectable images, similar to the main menu, with each being offset from the centre. The two options provided would be to submit an existing video file, or to live record a new one.

If the option submit existing video file is selected, the system will initiate a file dialog, prompting the user to select a video file via the file explorer. Upon selecting the file from the file explorer, the file at the video path is passed onto the trimming screen. If the option to create a live recording is selected, the user is instead taken to the video recording section of the application.

In the high-fidelity prototype, the buttons for each option were offset to provide a modern feel. Upon creating the models however, it was deemed necessary to include a section explaining the optimal recording angles so that the system can provide meaningful scoring and advice to the user. As such, the layout of the menu was adjusted to accommodate this.

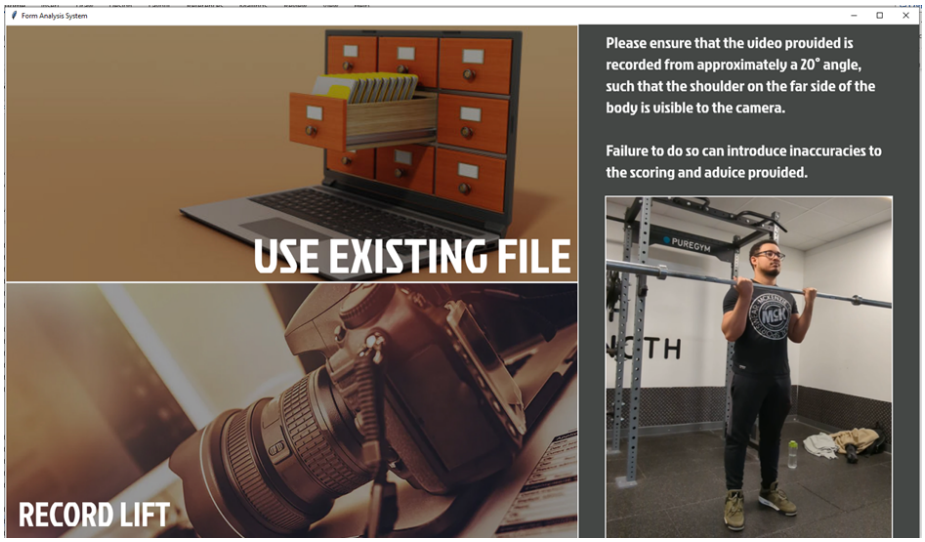


Figure 24: Video input user interface

Whilst the system was still in the design stage, no considerations were made regarding the option to trim the input video in order to remove unnecessary sections. Whilst this functionality is already implemented in prevalent operating systems such as Windows, an approach that foregoes having this functionality would severely impact the accuracy of the system on live recorded video inputs.

### 5.4.3 Live Recording Functionality

In order to implement live recording, CV2 was utilized to retrieve data from the device's main camera. The frames received from the camera object are converted to RGB as the default image format in CV2 is BGR. Each frame received from the camera is then displayed to the user via a Tkinter label in order to display a live feed from the camera. The live feed from the camera is not stored until the user toggles the recording button. Once toggled all frames coming from the camera are appended to an array. This functionality was implemented in such a way that the user has unlimited attempts to record.

```python
# update camera feed
def update_frame():
    ret, frame = camera.read()
    if ret:
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        # resize recording window to match camera resolution
        height, width, _ = frame.shape
        resize_window(width, height)

        img = Image.fromarray(frame)
        img = ImageTk.PhotoImage(image=img)

        # display frame as photo image on panel
        panel.config(image=img)
        panel.image = img

        # save frames if the user is recording
        if is_recording:
            recorded_frames.append(frame)

    # continuously update video feed
    panel.after(1, update_frame)
```

Figure 25: Code snippet for updating the frame


Figure 26: Recording menu user interface

The ability to submit a recording is initially disabled, only becoming available once the user has successfully recorded. Submitting a recording passes the array of frames to the video trimming menu.

### 5.4.4 Video Input Trimming and Processing

In order to maximise the accuracy of the feedback provided by the system, the user is requested to trim the video input in a way such that the first frame of the cropped video consists of the start of the concentric portion of the exercise being performed, and the final frame of the cropped video is selected where the concentric portion of the exercise has been performed, and the working weight is stationary.

To accommodate this request made of the user, this section of the application consists of a frame-by-frame feed of the video provided by via either the file dialog or through live recording and allows the user select manually the frame at which to start and stop the input. This was accomplished by tracking the index in the array of frames currently being displayed, and once the user has selected the frames representing the start and end of the concentric phase of the exercise, removing the sections of the array not between the start and the end.

To implement this, the menu consists of a series of buttons responsible for, incrementing the frame being viewed, decrementing the frame being viewed, marking a frame as the beginning of the concentric, marking a frame as the end of the concentric, and submitting the video to be processed using OpenPose and the appropriate machine learning model.

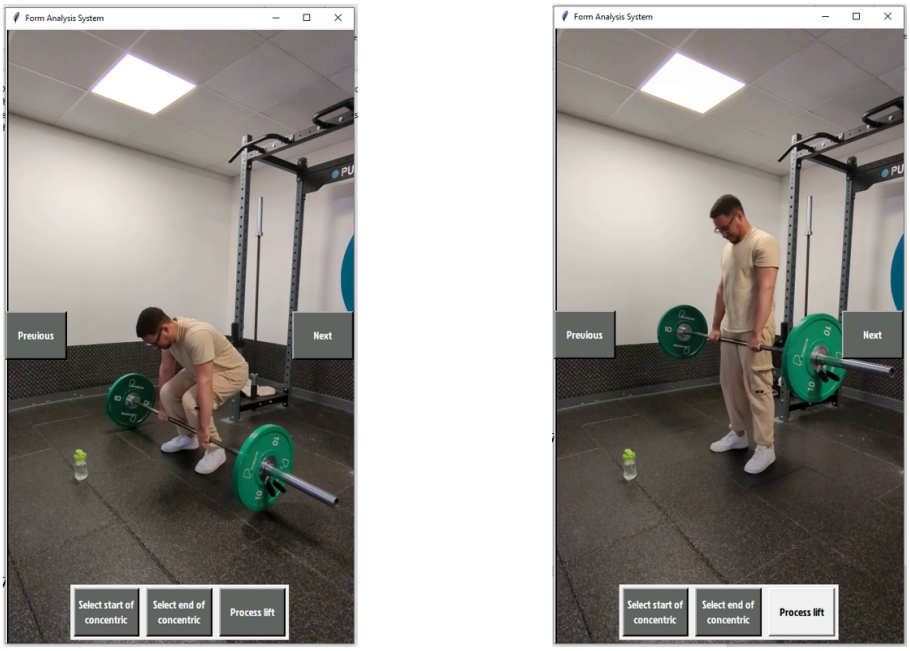


Figure 27: Video Trimming Menu user interface

During keypoint extraction, 20 frames are processed, evenly distributed across the length of the input video. As such, this requires the trimmed video created by the user to have a length of at least 20 frames. Should the user attempt to select a video portion of length less than this lower limit, the system notifies the user.

Once the user has submitted a valid video input, the trimmed video is saved as an mp4 and the video path to the newly saved file is passed into the function responsible for providing scoring and advice for the chosen exercise.

The function responsible for processing the user's video is wrapped in a try-except to protect against the case where the user's input is not suitable, most often caused by the video submitted not containing imagery capable of being recognised with OpenPose. The score, advice and issue frames generated from the user input are then passed onto the scoring screen.

```python
def processDeadlift(video_path):
    # extract keypoints with OpenPose
    keypoints, frames = extractKeypoints(video_path)

    # feature extraction for neural network and advice generation
    features = extractDeadliftFeatures(keypoints)

    # advice extraction using features and thresholds
    advice, issue_frames = extractDeadliftAdvice(features, keypoints, frames)

    # score calculation from trained models
    score = calculateDeadliftScore(features)

    # return score, advice and frames containing mistakes found during advice extraction
    return score, advice, issue_frames
```

Figure 28: Code snippet example of lift processing steps

### 5.4.5 Scoring and Advice Menu

The design for the scoring and advice menu was further developed upon throughout implementation. Initially, advice and the associated imagery was to be displayed to the user in the form of a scrollable list. This approach was revised in order to provide consistency with previous menus and to conserve screen real-estate and was replaced with a pair of buttons responsible for incrementing through each set of advice and its associated imagery.

Implementing this functionality involved iterating through a zipped array of advice and issue frames, creating the appropriate label for each of them, storing the labels in an array and then systematically removing and replacing the image label and text label as the user increments a counter representing the index of said array using the provided buttons.

```
# array of advice + imagery
advice_screens = []

# iterate through zipped arrays
for advice, frame in zip(advice, issue_frame):

    # retrieve image data as RGB
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    frame_image = Image.fromarray(frame)

    # resize image to fit label whilst keeping aspect ratio
    widget_width = 700
    widget_height = 700
    ratio = min(widget_width / frame_image.width, widget_height / frame_image.height)
    new_width = int(frame_image.width * ratio)
    new_height = int(frame_image.height * ratio)
    resized_image = frame_image.resize((new_width, new_height), Image.ANTIALIAS)
    frame_image = ImageTk.PhotoImage(resized_image)

    # create image label
    image_label = tk.Label(app, image=frame_image, bg="#444745")
    image_label.image = frame_image

    # create text label
    text_label = tk.Label(app, text=advice, fg='white', bg="#444745", font=("Jockey One", 30), wraplength=500)

    # append to array
    advice_screens.append([image_label,text_label])

# place initial advice / imagery
advice_index = 0
advice_screens[advice_index][0].place(x=150, y=180)
advice_screens[advice_index][1].place(x=800, y=300)
```

Figure 29: Code snippet of zipped arrays

```
def next_advice():
    # hide currently displayed advice / imagery
    global advice_index
    advice_screens[advice_index][0].place_forget()
    advice_screens[advice_index][1].place_forget()

    # increment counter and wrap with modulo
    advice_index += 1
    advice_index = advice_index % len(advice_screens)

    # display new advice / imagery
    advice_screens[advice_index][0].place(x=150, y=180)
    advice_screens[advice_index][1].place(x=800, y=300)

def prev_advice():
    global advice_index
    advice_screens[advice_index][0].place_forget()
    advice_screens[advice_index][1].place_forget()

    advice_index -= 1
    advice_index = advice_index % len(advice_screens)

    advice_screens[advice_index][0].place(x=150, y=180)
    advice_screens[advice_index][1].place(x=800, y=300)

prev_advice_button = tk.Button(app, text="Previous", command=prev_advice)
prev_advice_button.pack(side=tk.LEFT)
style_buttons(prev_advice_button)

next_advice_button = tk.Button(app, text="Next", command=next_advice)
next_advice_button.pack(side=tk.RIGHT)
style_buttons(next_advice_button)
```

Figure 30: Code snippet for incrementing buttons

An oversight made during the design stage was that the designs lacked any option for the user to be able to return to the main menu after having been provided with a scoring and advice. To implement this, a home label was added, consisting of a clickable Unicode character. This clickable label triggered a function that would remove any existing labels from the application window and re-display the main menu, such that another pass through the application could be made.

```
# create label for home button
home = tk.Label(app, text="Return to Main Menu \U0001F3E0", font=("Jockey One", 24),
                bg='#444745', fg='white', cursor="hand2")

def toMainMenu(event):
    global bad_lift
    home.destroy()
    score_label.destroy()

    if advice_label:
        advice_label.destroy()

    if bad_lift:
        advice_screens[advice_index][0].destroy()
        advice_screens[advice_index][1].destroy()
        next_advice_button.destroy()
        prev_advice_button.destroy()

    bad_lift = False
    displayMainMenu()

home.bind("<Button-1>", toMainMenu)
home.place(x=0, y=0)
```

Figure 31: Home button code snippet

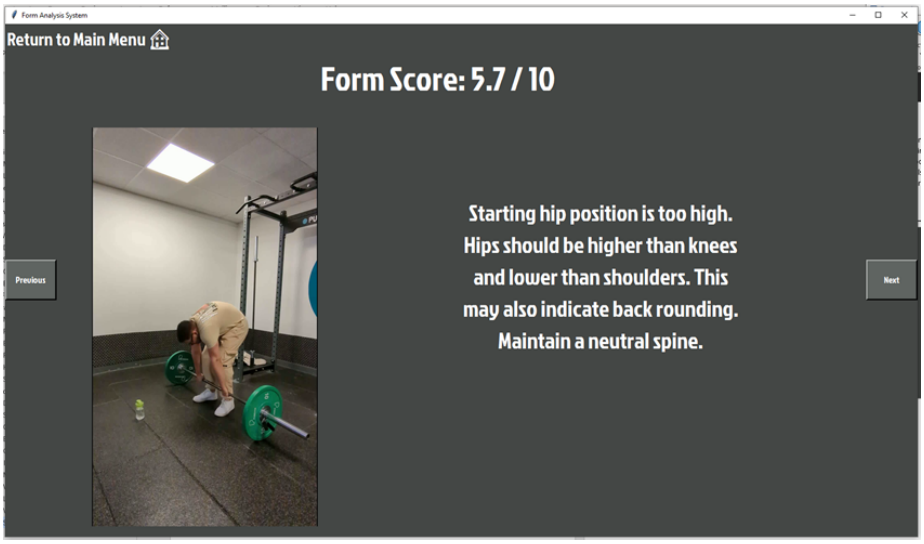


Figure 32: Scoring and Advice Menu user interface

# 6 Testing

This section aims to provide an insight into the testing process undergone for the application. The application's functionalities will be compared with the requirements specified to ensure that the application behaves as expected and meets the requirements outlined. A structured approach has been taken for functionality testing, with testing segmented by the menu structure. Any deviation of the application's functionality with regards to the requirements is noted and in the case of failed tests, notes on remediation required and action taken is included. Additional testing on the scoring system was undertaken during model evaluation and can be seen in the following evaluation chapter.

## 6.1 Main Menu

| Feature | Expected Behaviour | Test Passed / Failed | Action Taken / Notes |
|---|---|---|---|
| Squat Button | User is taken to input selection menu and lift variable is set to 'Squat'. | Passed | N/A |
| Deadlift Button | User is taken to input selection menu and lift variable is set to 'Deadlift'. | Passed | N/A |
| Overhead Press Button | User is taken to input selection menu and lift variable is set to 'OHP'. | Passed | N/A |
| Curl Button | User is taken to input selection menu and lift variable is set to 'Curl'. | Passed | N/A |

Figure 33: Main menu testing table

## 6.2 Video Input Selection Menu

| Feature | Expected Behaviour | Test Passed / Failed | Action Taken / Notes |
|---|---|---|---|
| File Upload Button | File dialog is opened and once a file is selected, the user is taken to the video trimming menu, unless the user fails to select a video in which case a warning is thrown. | Passed | N/A |
| Record Menu Button | User is taken to the live recording menu unless a device is not found, in which case a warning is displayed. | Passed | N/A |

Figure 34: Video input selection menu testing table

## 6.3 Live Recording Menu

| Feature | Expected Behaviour | Test Passed / Failed | Action Taken / Notes |
|---|---|---|---|
| Live Feed | User is displayed a live feed from their device's webcam / camera. | Passed | N/A |
| Start Recording Button | Frames from the live feed begun to be saved to an array. The button changes to the recording state. | Passed | N/A |
| Stop Recording Button | Frames from the live feed are no longer saved. Button changes back to the not recording state. | Passed | N/A |
| Submit button | Button is not active until a recording has been made. User is taken to the video trimming menu. | Passed | N/A |

Figure 35: Live recording menu testing table

## 6.4 Video Trimming Menu

| Feature | Expected Behaviour | Test Passed / Failed | Action Taken / Notes |
|---|---|---|---|
| Mark start button | Index of the currently displayed frame is marked as the start. | Passed | N/A |
| Mark end button | Index of the currently displayed frame is marked as the end. | Passed | N/A |
| Submit button | Video is processed and user is taken to the advice menu, so long as the length of the trimmed video is greater than 20 frames, if not a warning is thrown. | Passed | N/A |
| Next frame button | Next frame in the video is displayed. | Passed | N/A |
| Previous frame button | Previous frame in the video is displayed. | Passed | N/A |

Figure 36: Video trimming menu testing table

## 6.5 Scoring and Advice Menu

| Feature | Expected Behaviour | Test Passed / Failed | Action Taken / Notes |
|---|---|---|---|
| Form Scoring | Appropriate form scoring is displayed to the user | Failed | In situations where the joints required for feature extraction are obscured, a score was not able to be calculated and the application crashes. Try/Except has been added to prevent application crash but score will still not be displayed, instead throwing warning to console. |
| Form breakdown imagery | Imagery related to the mistake made by the user is displayed. | Passed | N/A |
| Form guidance | Where necessary, textual advice is displayed to the user. | Failed | Similarly to the form scoring, in situations where the joints required for guidance calculations were obscured, guidance would not be given. The chance of this occurring has been attempted to be reduced by specifying the angle the user should record from, but sometimes still remains. |
| Next/Prev Guidance buttons | The image and associated textual advice are iterated through if there are multiple. | Passed | N/A |
| Return Home Button | User is returned to the main menu on click. | Passed | N/A |

Figure 37: Scoring and advice menu testing table

# 7 Evaluation and Conclusion

This section aims to compare the developed application with the requirements specified during requirements analysis. The impact of the application on its problem domain will be examined to determine whether the application can be considered a success.

## 7.1 System Evaluation

The objectives originally set out for the project were as followed:

- To identify a gap in the market for a form analysis system by researching alternative applications in the problem space and identifying the positive and negative aspects of each.
- To performing a literature review to become informed on the currently accepted principles in lifting bio-mechanics and exercise physiology, and to identify which current technologies can be utilised to implement the system.
- To create a dataset of weightlifting videos containing examples of what is considered good lifting form, along with examples of common errors in technique made when performing specific exercises.
- To develop a Python application capable of analysing the form of a user in order to provide a scoring based on the quality of the user's lift, alongside feedback specific to the user based on commonly accepted weightlifting principles.
- To evaluate the objectivity of the scoring system and the quality of the advice provided using real-life examples during testing.

A thorough gap analysis was completed, comparing the functionalities of existing systems within the problem domain, and using the information gathered to identify the need for such a form analysis application.

A broad literature review was carried out that aimed to collate the information utilised in the project, by encompassing the bio-mechanical and technical aspects of a potential solution.

A dataset consisting of over 600 lifting examples was successfully compiled, containing a variety of exercises performed with a varying degree of technique quality.

A python application was developed that is fully capable of detecting mistakes in a user's lifting technique across four exercises, based on pre-determined criteria, and meets all the requirements outlined within during requirements analysis.

## 7.2 Evaluating Scoring Model Performance

In order to evaluate the objectivity of the scoring system, a selection of video samples collected during dataset collection but excluded from training, supplemented with a collection of publicly accessible exercise videos, were evaluated and scored based on the scoring criterion described in the implementation section. This testing set consisted of 40 examples across the four exercises supported by the application. An effort was made to include examples recorded from varying angles to gain an insight into how the recording angle affects model volatility.

### 7.2.1 Deadlift Scoring Model

| Criterion-based score | Model score | Notes |
|---|---|---|
| 3 | 0 | Used sumo deadlift technique |
| 4 | 5.9 | N/A |
| 5 | 6.2 | N/A |
| 6 | 7.2 | N/A |
| 7 | 3.2 | Recorded from straight on angle |
| 7 | 7.1 | N/A |
| 7 | 6.7 | N/A |
| 7 | 6.8 | N/A |
| 8 | 0 | Recording angle shifting throughout lift |
| 9 | 8.2 | N/A |
| 9 | 9.2 | N/A |

Table 6: Deadlift model performance on testing set

The average absolute difference for the deadlift video testing set was 2.1 when including examples purposefully recorded from alternative angles or when the lifter is utilising an alternative lifting style. The average absolute difference for the video testing set removing such outliers was 0.84. This does require lifts to be performed from specific angles in order for the model to be the most accurate, perhaps raising questions regarding the models ability to generalise as training dataset consisted of video examples where the majority were recorded from the same angle, at around a 20° offset to the direction the user is facing. The model appears to consistently overestimate the scoring for a lift, likely due to the majority of the training dataset consisting of examples in the 6-8 range, because users are unlikely to make multiple large mistakes when performing lifts. Overall the accuracy of the model is satisfactory under the recording conditions requested within the application, see figure 24.

### 7.2.2 Squat Scoring Model

| Criterion-based score | Model score | Notes |
|---|---|---|
| 5 | 3.7 | N/A |
| 6 | 4.2 | N/A |
| 6 | 4.8 | N/A |
| 7 | 4.9 | N/A |
| 7 | 5.6 | N/A |
| 8 | 5.3 | N/A |
| 9 | 6.2 | N/A |
| 9 | 6.7 | N/A |
| 9 | 7.8 | N/A |
| 10 | 8.1 | N/A |

Table 7: Squat model performance on testing set

The average absolute difference for the squat video testing set was 1.97. This value is higher than what would have been expected, and the model appears to consistently under-predict. Despite this, the model is able to very consistently score lifts accurately relative to each other, with better lifts scoring consistently higher than poorly executed lifts. Due to this, the model inaccuracy is not of a concern as, with fine tuning of the scoring algorithm by normalizing the scores outputted by the squat scoring model, the scoring could be made to more closely resemble the criterion.

### 7.2.3 Overhead Press Scoring Model

| Criterion-based score | Model score | Notes |
|---|---|---|
| 3 | 4.4 | N/A |
| 6 | 6.5 | N/A |
| 7 | 7.3 | N/A |
| 7 | 5.7 | N/A |
| 7 | NaN | Keypoints obscured during lift |
| 8 | 7.3 | N/A |
| 8 | NaN | Keypoints obscured during lift |
| 8 | 7.6 | N/A |
| 9 | 8.2 | N/A |
| 9 | 8.0 | N/A |

Table 8: Overhead press model performance on testing set

The average absolute difference for the overhead press video testing set was 0.8, excluding outliers where generating a score was no possible. Whilst the

accuracy of the model is satisfactory, there remains issues caused when the exercise is not recorded from the angle specified in figure 24. The model appears to slightly over-predict poorly executed lifts, and under-predict well executed exercises. The model has likely over-fitted on the training data where the average lift was scored between 6-7 according to the criterion. Despite this, similar to the other models, it is able to consistently determine whether a lift was executed well or not, albeit differently scaled to the criterion.

### 7.2.4 Barbell Curl Scoring Model

| Criterion-based score | Model score | Notes |
|---|---|---|
| 3 | 2.8 | N/A |
| 6 | 6.5 | N/A |
| 7 | 7.5 | N/A |
| 7 | 0.2 | Recording angle obscuring far-side arm |
| 7 | 8.1 | N/A |
| 8 | 0 | Recording angle obscuring far-side arm |
| 8 | 8.7 | N/A |
| 9 | 8.6 | N/A |
| 9 | 8.0 | N/A |

Table 9: Barbell Curl model performance on testing set

The average absolute difference for the barbell curl video testing set was 2.13. The model did not perform well when variation was introduced to the recording angle, likely because the barbell curl scoring model predominantly uses features that are calculated using the positions of both arms relative to each other, and so when one of the arms is obscured, the amount of features input to be used to calculate a scoring is reduced significantly. Combined with the model using at most 6 features, this leads to the model performing poorly with variable recording angles. In an ideal recording environment however, the model is able to perform very well, with an absolute difference being reduced to 0.65 when removing outliers.

### 7.2.5 Scoring Model Conclusion

Overall, the models perform well with an optimal recording environment, able to determine the quality of a lifters form, but struggle to provide reasonably accurate scoring when these recording conditions are not met. This issue could likely be remedied with more refined feature extraction and a dataset that better represented the variety of recording angles that a user would typically use. The system would also consistently over/under predicted a scoring depending on the exercise model used, likely due to the scoring criterion being deduction-based, leading to most examples scoring similarly as they were often representative of a single type of mistake.

## 7.3 Discussions on Machine Learning Viability

Throughout this project, there was an overarching question as to the viability of applying machine learning techniques to a form scoring system compared to using static calculations and as such a combination of both were used. The resultant machine learning models, whilst capable of detecting whether an exercise is performed using good or bad technique, were not as performant as what was initially hoped. This raised concerns as to the viability of using machine learning for a system that was very possible to be created using only static comparisons. Whilst the choice was made to use machine learning in conjunction with static comparisons, during implementation, two ideas were conceived that could have been viable alternative approaches to the form scoring and advice creation systems.

The first would involve simply removing the machine learning models and instead using the criterion used for advice generation to help calculate a scoring for a user's lift. This may have allowed the system to calculate a more accurate scoring. Using a static algorithm would, however, make it more challenging to account for the nuances and variations in lifting technique, especially across users with varying limb proportions.

Another approach that could have been considered would be to instead use a multi-label classification model to identify the specific mistakes made by a user during a lift and calculating a score based on this. This would also mean that the classifications received from the model could also be used during advice generation to completely remove the need for any static algorithms in the application. The main limiting factor behind such an approach would be the size of the dataset used to train the models, which is already a concern for the current model architectures, let alone increasingly complex models.

To conclude, whilst both an entirely machine-learning based approach or exclusive use of static calculation could have been used in this project, both have their own unique benefits and drawbacks and so the decision to encompass both to take a balanced approach to the problem, experimenting with aspects of both.

## 7.4 Future Development

Whilst not currently within the scope of the project, there remains room for possible future development of the application. This section outlines some of the improvements that could be considered if the project was to be developed further, given more development time.

The exercise selection of the application is currently limited to four exercises and could easily be expanded upon, though would require additional models trained on new datasets including the new exercises. User interface accommodations would have to be made to include this, but with an extensive selection of exercises, a user would be able to instead receive advice on entire workouts, also introducing the possibility of meta-reviews of users training over extended periods of time and providing insights into technique breakdown over multiple workouts based on exercise choice and ordering. In addition to this, another feature that could be implemented in future developments would be support for including the eccentric phase of the exercises into the users video input, which could provide additional insights into a user's lifting form and create new avenues for advice generation.

As existing pose recognition libraries are improved, or new libraries are developed, these could be integrated into the application in order to improve the accuracy of the systems included in the application, as well as provide the possibility of porting the application to mobile devices if the libraries become lightweight enough to be run on lower-spec mobile devices.

If the application were to be developed over a long enough time-frame, changes would likely need to be made in order to accommodate the growing body of sports science literature, to ensure that the advice provided to the user remains relevant.

A final development that could be made would be to increase the size of the dataset created to enable this project. Due to time restraints and the physically taxing data collection process, of which all exercises were self-performed, the results of the trained machine learning models suggest that the size of the dataset was insufficient and so model accuracy may have suffered as a result. Given a longer project timeline, a more comprehensive dataset could be created, using videos from individuals with a variety of body proportions, to further improve model accuracy and enable the user of more complex model architectures that are currently infeasible.

## 7.5 Personal Learning and Development

In order for this project to be a success, I had to draw upon knowledge and experience acquired through the course thus far in addition to knowledge acquired externally.

My placement as a Software Engineer and the university course provided me with solid fundamentals on Software Engineering principles, such as project lifecycles that I have attempted to incorporate into the creation of this project. This then better enabled me to create an well-structured report based on the stages typically taken during a development lifecycle.

The course also provided me with a basic understanding of machine learning, however the majority of the machine learning content on the course was provided after having already started the implementation stage of the project, and so a considerable amount of my knowledge in the subject had to be self-taught. In addition to this, for the Computer Vision aspect of the system, I had no prior knowledge of field and of pose recognition libraries having not taken the Computer Vision module and so had to do considerable self-learning in that topic throughout the project.

At the time of writing, having spent the last 18 months consistently weightlifting and undergoing the process of improving my own weightlifting technique, I felt like I had a reasonable grasp on what was considered good lifting technique. This project was a great opportunity to apply the knowledge I had gained over that time, whilst also gaining new knowledge through additional reading into the field of bio-mechanics as necessary to develop the application.

I feel like my understanding of Python has greatly improved throughout the time spent on this project, requiring me to utilise libraries that I otherwise would not have ever touched, namely OpenCV, and will be able to apply this new knowledge well in future projects, or perhaps even future developments to the application developed in this project.

# 8 References


StrengthLog. (2024). Most Popular Exercises. Retrieved from StrengthLog: https://www.strengthlog.com/most-popular-exercises/

BeomJun Jo, S. K. (2022). Comparative Analysis of OpenPose, PoseNet, and MoveNet Models for Pose Estimation in Mobile Devices. Traitement du Signal.

CMU Perceptual Computing Lab. (2019). Installation: Additional Settings. OpenPose Documentation. Retrieved from cmu-perceptual-computing-lab: https://cmu-perceptual-computing-lab.github.io/openpose/web/html/doc/md_doc_installation_2_additional_settings.html

CMU Perceptual Computing Lab. (2020). openpose. Retrieved from GitHub: https://github.com/CMU-Perceptual-Computing-Lab/openpose

Cohen, A. (2023, August 19). A Comparative Analysis of TensorFlow, PyTorch, MXNet, and scikit-learn. Retrieved from Iamitcohen: https://iamitcohen.medium.com/a-comparative-analysis-of-tensorflow-pytorch-mxnet-and-scikit-learn-2072fe566df7

Figma, Inc. (2014). Retrieved from Figma: https://www.figma.com

Google. (2024). Pose detection. Retrieved from Google Developers, ML Kit: https://developers.google.com/ml-kit/vision/pose-detection

Groves, B. (2000). Powerlifting. Human Kinetics.

Hadim, M. (2024). How to Overhead Press with Proper Form: The Definitive Guide. Retrieved from Stronglifts: https://stronglifts.com/overhead-press/#common-mistakes

Jonathan Sinclair, 1. P. (2022). A Multi-Experiment Investigation of the Effects Stance Width on the Biomechanics of the Barbell Squat. Sports (Basel).

Kanasu, R. K. (2021). Pose Estimation and Correcting Exercise Posture. Mumbai: ITM Web Conf.

Keitaro Kubo 1, T. I. (2019). Effects of squat training with different depths on lower limb muscle volumes. Eur J Appl Physiol.

Kendrick, A. (n.d.). LUMBAR HYPEREXTENSION INJURY IN YOUNG ATHLETES. Retrieved from OrthoNeuro: https://orthoneuro.com/lumbar-hyperextension-injury-in-young-athletes/#:~:text=Lumbar%20hyperextension%20injuries%20occur%20when,the%20normal%20anatomical%20position%20repeatedly.&text=Such%20additional%2C%20repetitive%20stress%20may,and%20nerves%20in%20t

Konstantin Warneke, L. H. (2023). Physiology of Stretch-Mediated Hypertrophy and Strength Increases: A Narrative Review. Sports Med 53.

Lin, T.-Y. (2014). Microsoft COCO: Common Objects in Context. CoRR.

Mart n Abadi, A. A. (2024). TensorFlow. Retrieved from TensorFlow: Large-scale machine learning on heterogeneous systems,: https://www.tensorflow.org/

Mykhaylo Andriluka and Leonid Pishchulin and Peter Gehler and Schiele, B. (2014). 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Paszke, A. a. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. Advances in Neural Information Processing Systems 32.

Pedregosa, F. a. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research.

Pure Gym Limited. (2024). Retrieved from PureGym: https://www.puregym.com/

Pure Gym Limited. (2024). THE UK FITNESS REPORT 2023/24 GYM STATISTICS. Retrieved from Puregym: https://www.puregym.com/blog/uk-fitness-report-gym-statistics/

Sj berg, H., Aasa, U., Rosengren, M., & Berglund, L. (2020). Content Validity Index and Reliability of a New Protocol for Evaluation of Lifting Technique in the Powerlifting Squat and Deadlift. Journal of Strength and Conditioning Research .

SravB. (2019). Computer-Vision-Weightlifting-Coach. Retrieved from GitHub: https://github.com/SravB/Computer-Vision-Weightlifting-Coach

statista. (2023). Annual number of physiotherapists in the United Kingdom (UK) from 2010 to 2023. Retrieved from statista: https://www.statista.com/statistics/318906/numbers-of-physiotherapists-in-the-uk/

Timothy E Hewett 1, G. D. (2005). Biomechanical measures of neuromuscular control and valgus loading of the knee predict anterior cruciate ligament injury risk in female athletes: a prospective study. Am J Sports Med.

Victor Bengtsson, 1. L. (2018). Narrative review of injuries in powerlifting with special reference to their association to the squat, bench press and deadlift. BMJ Open Sport Exerc Med.

WODProof App inc. (2024). WODProof Bionic. Retrieved from WODProofApp: https://wodproofapp.com/