HAW HAMBURG

BACHELORTHESIS
Leonard Bardtke

Machine-Learning-basierte Qualitätserkennung von Fitnessübungen auf Grundlage von Pose Estimation

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Computer Science and Engineering
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Leonard Bardtke

Machine-Learning-basierte Qualitätserkennung von Fitnessübungen auf Grundlage von Pose Estimation

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Informatik Technischer Systeme*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Stephan Pareigis
Zweitgutachter: Prof. Dr. Kai von Luck

Eingereicht am: 25.04.2022

Leonard Bardtke

Thema der Arbeit

Machine-Learning-basierte Qualitätserkennung von Fitnessübungen auf Grundlage von Pose Estimation

Stichworte

Pose Estimation, Human Activity Recognition, Quantified Self, Zeitreihen, LSTM

Kurzzusammenfassung

Für einen gesunden Lebensstil sind körperliche Aktivitäten ein wichtiger Bestandteil. Mit Hilfe von Human Activity Recognition ist es möglich, die eigene Gesundheit im Auge zu behalten und Verletzungen zu vermeiden. Hierbei wird zwischen den beiden Haupttechniken der Datenerfassung im Gebiet der Human Activity Recognition differenziert. Zum einen handelt es sich dabei um den Bereich der Computer Vision und zum anderen um den Ansatz, welcher auf dem Verwenden von tragbaren Sensoren basiert. In dieser Arbeit erfolgt die Qualitätserkennung von Fitnessübungen anhand von Liegestützen, welche als Videodaten vorliegen. Um die zu analysierenden Parameter auf die Informationen zu reduzieren, die für die Qualitätserkennung relevant sind, erfolgte eine Reduzierung der Merkmale mit Hilfe von Pose Estimation. Diese liefert die Koordinaten bestimmter Körperschlüsselpunkte (Keypoints), mit denen das Erstellen eines Skeletts der trainierenden Person möglich ist. Bei den Keypoints handelt es sich um Gelenke und Körperteile, wie den Ellenbogen oder der Hüfte, wobei die Anzahl der zu erkennen Keypoints je nach Modell variiert. Um die Qualität der Liegestütze zu bestimmen, wurden Machine Learning in Form von Long Short-Term Memory (LSTM) verwendet. Neben LSTMs, welche die Koordinaten der Keypoints für die Bestimmung der Ausführungsqualität verwenden, wurden zudem Modelle trainiert, die die Klassifikation anhand der Winkel zwischen ausgewählten Körperteilen vornehmen. In dieser Arbeit konnte gezeigt werden, dass sich die Winkeldaten für eine Klassifikation deutlich besser eignen. Die binäre Klassifizierung, welche lediglich zwischen einer korrekten und einer nicht korrekten Ausführung unterscheidet, wies eine Testgenauigkeit von 91,5% auf. Um die Art der Fehlausführung spezifizieren zu können wurde zudem eine Mehrklassen-Klassifzierung untersucht. Hierbei gelang es die Qualität der Ausführung lediglich mit einer Genauigkeit von 60,71% zu bestimmen. Dies ist zum einen auf die Ungleichverteilung des Datensatzes zurückzuführen und zum andern auf die Schwierigkeit, zwischen den definierten Klassen zu unterscheiden.

Leonard Bardtke

Title of Thesis

Machine-learning based quality recognition of fitness exercises based on pose estimation

Keywords

Pose Estimation, Human activity recognition, quantified self, time series, LSTM

Abstract

Physical activity is an important part of a healthy lifestyle. With the help of human activity recognition, it is possible to keep track of one's health and avoid injuries. There are two main techniques of data acquisition in the field of human activity recognition. On the one hand there is the field of computer vision and on the other hand the approach, which is based on the use of wearable sensors. In this work, the quality of fitness exercises is identified using push-ups, which are available as video data. Pose estimation is used to reduce the parameters to the information relevant for quality recognition. This provides the coordinates of certain keypoints of the body with which it is possible to create a skeleton of the person being trained. The keypoints are joints and parts of the body, such as elbows or the hip, whereby the number of keypoints that can be recognized varies depending on the model.

Machine learning in the form of long short-term memory (LSTM) was used to determine the quality of the push-ups. In addition to LSTM-model, which uses the coordinates of the keypoints to determine the quality of the execution, models were also trained to perform the classification based on the angles between selected body parts. This work has shown that the angle data are much better suited for classification. The binary classification, which only distinguishes between correct and incorrect execution, had a test accuracy of 91.5%. To be able to specify the type of incorrect execution, a multi-class classification was also examined. It was only possible to determine the quality of the execution with an accuracy of 60.71%. On the one hand, this is due to the unequal distribution of the data set and, on the other hand, to the difficulty of distinguishing between the defined classes

Danksagung

Ich bedanke mich recht herzlich bei Prof. Dr. Kai von Luck und Prof. Dr. Stephan Pareigis, die mich bei dieser Bachelorarbeit unterstützt und motiviert haben. Außerdem gebührt mein Dank Jan Schwarzer und André Jeworutzki aus dem MoGaSens-Team, für die konstruktive Kritik und Unterstützung während der Anfertigung dieser Arbeit.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Motivation

„Quantified Self“ beschreibt das Konzept der Selbstverfolgung von bestimmten Aspekten des Lebens. Hierbei werden personenbezogene Informationen aufgezeichnet, um diese zu analysieren und anschließend eine Optimierung durchzuführen [30]. Aufgrund des technologischen Fortschritts und der immer kleiner und günstiger werdenden Sensoren, erfolgt die Selbstoptimierung heutzutage in verschiedenen Bereichen des Lebens. Die Selbstvermessung findet hierbei Anwendung in Bereichen, wie dem Aufzeichnen der täglichen Bewegungen, der eigenen Gefühlszustände, des Blutdrucks, der Ernährung oder der Überwachung des eigenen Schlafverhaltens[25].

Bei Quantified Self handelt es sich um ein Teilgebiet der Human Activity Recognition. Die Human Activity Recognition zielt darauf ab, Aktionen oder auch Intentionen von Personen zu erkennen. Diese kann zudem dabei helfen, den Gesundheitszustand einer Person zu bestimmen und dem Auftreten von Verletzungen vorzubeugen. Da es bei fehlerhaften Ausführungen von Fitnessübungen zu Verletzungen kommen kann, beschreibt dies ein Anwendungsbeispiel der Human Activity Recognition. Die zwei Haupttechniken für die Datenerhebung, um eine automatische Erkennung der Aktivitäten möglich zu machen, sind zum einen das Verwenden von tragbaren Sensoren und zum anderen durch das Verwenden von Kameras. Die kamerabasierte Methode weist jedoch einige Nachteile auf, wie beispielsweise, dass die Erkennung durch Änderung des Hintergrundes oder unterschiedliche Lichtverhältnisse beeinträchtigt werden kann. Infolgedessen haben die tragbaren Sensoren eine höhere Aufmerksamkeit in der Forschung erhalten [33]. Der Anstieg des Deep Learnings hat zu großen Fortschritten im Bereich der Computer Vision beigetragen [34]. Eine Aufgabe der Computer Vision ist die Human Pose Estimation. Hierbei handelt es sich um ein Verfahren, um die Position von einer bestimmten Anzahl an Körperschlüsselpunkten von Personen in einem Bild zu lokalisieren. Mit Hilfe dieser Punkte lässt sich anschließend auf die Pose der Person schließen.

1.2 Zielsetzung

Das Ziel dieser Arbeit ist es, die Qualität von Liegestützen zu erkennen, um die Form der Ausführung zu verbessern und Verletzungen vorbeugen zu können. Hierbei wird bei der Qualitätserkennung mit Videodaten gearbeitet.
Mit Hilfe von Human Pose Estimation soll die Pose der trainierenden Person erkannt werden. Für die Klassifizierung wird maschinelles Lernen in Form von Long Short-Term Memory Netzwerken (LSTM) verwendnet. Hierbei ist auch zu untersuchen, in welcher Form die Ergebnisse der Pose Estimation dem LSTM vorzuliegen haben, um eine präzise Klassifikation zu ermöglichen.

Da das Prinzip des Quantified Self das Ziel hat Aktionen zu optimieren, wird neben einer binären Klassifizierung auch eine Mehrklassen-Klassifizierung angestrebt. Somit kann der trainierenden Person eine detailliertere Rückmeldung zur ausgeführten Liegestütze geliefert werden.

1.3 Gliederung

Diese Arbeit ist in mehrere Kapitel unterteilt. Im Analysekapitel wird auf den Hintergrund der Arbeit eingegangen und es werden vergleichbare Arbeiten vorgestellt. Es erfolgt die Darstellung des Datensatzes, der in dieser Arbeit verwendet wird, sowie die Kriterien, um diesen Datensatz zu klassifizieren. Im dritten Kapitel wird auf die Implementation eingegangen. Es werden die Prozessschritte der Vorverarbeitung und Auswertung der Daten vorgestellt, die für die Umsetzung der Experimente notwendig sind. Im Evaluationskapitel wird der Versuchsaufbau beschrieben und es folgt die Auswertung der Ergebnisse. Abschließend wird in Kapitel 5 ein Fazit aus dieser Arbeit gezogen und es erfolgt ein Ausblick über mögliche Weiterentwicklungen des Projektes, sowie die Darstellung weiterer Einsatzmöglichkeiten der in dieser Arbeit verwendeten Techniken.

2 Analyse und vergleichbare Arbeiten

Im folgenden Kapitel wird zunächst der Hintergrund dieser Arbeit erläutert. Außerdem wird der Stand der Technik vorgestellt, zusammen mit Arbeiten, die dem Ansatz dieser Arbeit ähnlich sind. Es folgt die Darstellung des Datensatzes und der Kriterien, auf der die Klassifikation dieser Arbeit beruht. Im Anschluss werden die verwendeten Techniken vorgestellt.

2.1 Hintergrund des Projektes

Im Zuge dieser Arbeit wurde mit dem MoGaSens-Team zusammengearbeitet. Dieses beschäftigt sich mit dem Messen von medizinisch- und trainingsrelevanter Daten, um darauf basierend Aussagen über den Fitnesszustand der trainierenden Person treffen zu können. Die Aufzeichnung der Daten erfolgte mit Hilfe von Sensoren, welche an dem Körper der trainierenden Person angebracht waren [11]. Um eine Bewertung der Sensordaten vornehmen zu können, erfolgt zusätzlich eine Videoaufzeichnung der Übungen. Diese Videoaufzeichnungen dienen dieser Arbeit als Datensatz. Das MoGaSens-Team arbeitet unter anderem mit der Universität Hamburg und einem Team der Medizintechnik der HAW Hamburg zusammen. Diese haben zum Erstellen der Labels der Fitnessübungen beigetragen. Da sich diese Arbeit auf Liegestütze konzentriert, werden ausschließlich die Liegestütze des MoGaSens-Datensatzes verwendet.

2.2 Stand der Technik

Es existieren verschiedene Ansätze, um die Bewegungen oder das Verhalten von Personen zu analysieren. Unterscheiden lässt sich hierbei zwischen dem Ansatz, welcher tragbare Sensoren für die Datenaufzeichnung verwendet und dem Computer Vision basierten Ansatz.
Bei dem sensorbasierten Ansatz werden inertiale Messeinheiten (IMU), bestehend aus Gyroskopen, Beschleunigungssensoren und Magnetometern, an den Körperteilen der Personen angebracht, dessen Verhalten anschließend analysiert werden soll [22]. Bei der visuellen Analyse von Bewegungen oder dem Verhalten von Personen liegen die Informationen als Videodaten vor. Diese enthalten, neben der zu analysierenden Person, eine Menge an Hintergrundinformation, die für die Verhaltensanalyse irrelevant sind.
Video Instance Segmentation bietet die Möglichkeit Objekte in einem Video von ihrem Hintergrund zu segmentieren, diese über das Video hinweg zu verfolgen und sie, basierend auf einer Menge an vordefinierten Klassen, zu klassifizieren.[23] Dies wird ermöglicht, indem jeder Pixel eines Bildes einer bestimmten Klasse zugeordnet wird. [35] Mit Hilfe von Instance Segmentation wäre es also möglich, die trainierende Person vom Hintergrund des Videos zu separieren und anschließend eine Klassifikation durchzuführen. Eine weitere Methode liefert die Human Pose Estimation. Bei diesem Ansatz werden eine bestimmte Anzahl an Keypoints von einer Person erkannt und im Bild lokalisiert. Anhand dieser Keypoints lässt sich anschließend auf die Pose einer Person schließen.

Einen zu dieser Arbeit ähnlichen Ansatz bietet der Pose-Trainer [16]. Hier wurde Pose Estimation verwendet, um die Qualität von ausgewählten Fitnessübungen zu bestimmen. Es wurde das zweidimensionale Pose Estimation Model von „OpenPose“[13] verwendet. Die generierten Keypoints wurden anschließend normalisiert, um für eine Unabhängigkeit in Bezug auf die Statur der trainierenden Person und der Entfernung zur Kamera zu sorgen. Bei der Qualitätserkennung wurde hier zwischen einem heuristisch-basierten und einem Machine Learning-basierten Ansatz unterschieden. Im Zuge des Machine Learning-basierten Ansatzes wurde Dynamic Time Warping verwendet. Dynamic Time Warping ist ein Verfahren um Sequenzen miteinander zu vergleichen, wobei die Länge der Sequenzen, sowie deren zeitliche Ausrichtung vernachlässigt wird [31].
Einen ähnlichen Ansatz bietet zudem Ofli u. a. [29]. Hier wurde mit Hilfe der Microsoft Kinect ein Coaching-System entwickelt, um das Sturzrisiko der älteren Bevölkerung bei Fitnessübungen zu reduzieren und gleichzeitig die Leistung der täglichen Aktivitäten zu verbessern.

2.3 Analyse des Datensatzes

Der Trainingsdatensatz dieser Arbeit stützt sich auf die von dem MoGaSens-Team aufgenommenen Videodaten. Dieser beinhaltet verschiedene Übungen, wobei sich im Zuge dieser Arbeit ausschließlich auf die Liegestütze konzentriert wird. Die Liegestütze wurden von verschiedenen Personen, unterschiedlichen Geschlechts mit unterschiedlichen Körperproportionen durchgeführt, wodurch eine Generalität in Bezug auf die trainierende Person gegeben ist. Die Aufnahmen der Übungen erfolgten aus zwei unterschiedlichen Perspektiven, um die Qualität präziser bestimmen zu können. Hierbei wurden die Übungen zum einen von der Seite aufgenommen, um Qualitätsmerkmale, wie die Haltung der Körperspannung oder des Kopfes überprüfen zu können. Um in der Lage zu sein die Stellung der Arme während der Ausführung einer Liegestütze zu analysieren, erfolgte zudem die Aufnahme aus der frontalen Perspektive.
Wie in Abbildung 2.1 zu sehen, wurden die Aufnahmen zusammen in einem Video abgespeichert. Die Aufnahmen erfolgten mit zwei „Logitech C922“ Webcams, welche eine Auflösung von 1280×720 Pixeln lieferten. Die Videodaten lagen mit einer Bildfrequenz von 30 Bildern pro Sekunde vor und verfügten durchschnittlich über eine Länge von 54 Sekunden.
Der Datensatz beinhaltet 18 Sätze an Liegestützen. In dieser Arbeit konnten jedoch nur 17 Sätze des Datensätzes verwendet werden, da in einem Satz die Aufnahme aus der frontalen Perspektive teilweise von der Aufnahme aus der seitlichen Perspektive verdeckt wurde. In den Aufnahmen der Liegestütze haben die Probanden [1] so viele Wiederholungen durchgeführt, wie sie nacheinander ausführen konnten. Zudem wurden teilweise absichtlich falsche Ausführungen durchgeführt, um auch diese bei der Qualitätserkennung erfassen zu können. Die Sätze der Liegestützen dauern durchschnittlich 54 Sekunden und enthalten im Durchschnitt 19 Wiederholungen. Im Gesamten liegen 356 Wiederholungen vor.
Die Qualität der einzelnen Wiederholungen wurden von Sportwissenschaftler*Innen der Universität Hamburg bestimmt. Auf den daraus resultierenden Labels ist die Qualitätserkennung dieser Arbeit aufgebaut. Bei der Bestimmung der Ausführungsqualität wurden 209 Wiederholungen als korrekt und 147 als fehlerhaft gewertet.

[1] In dieser Arbeit wird aus Gründen der besseren Lesbarkeit das generische Maskulinum verwendet. Weibliche und anderweitige Geschlechteridentitäten werden dabei ausdrücklich mitgemeint.

Abbildung 2.1: Zeigt die Originalaufnahme aus dem Datensatz

Qualitätsdefinition

Die Definition der Qualitätsmerkmale der Liegestütze erfolgten durch zwei Sportwissenschaftler*Innen der Universität Hamburg. Die Qualität einer Liegestütz-Wiederholung wird als fehlerhaft definiert, wenn die Plank-Position nicht gehalten werden kann und somit keine Spannung im Oberkörper vorhanden ist. Ein Indiz dafür ist, dass die Punkte des Kopfes, der Brust und des Beckens keine gerade Linie ergeben. Ein weiteres Merkmal für eine fehlerhafte Ausführung ist, dass die Oberarme nicht nahe genug am Oberkörper anliegen und der Winkel zwischen Ellenbogen und Rumpf 60 Grad übersteigt. Des Weiteren ist die Range of Motion einzuhalten. Diese wird so definiert, dass die Arme beim Start der Liegestütze durchgestreckt sein sollen und der Winkel zwischen Ober-und Unterarm dementsprechend bei ca. 170 bis 180 Grad liegen sollte. In der Endposition der Liegestütze sollte die Brust den Boden beinahe berühren (ca. 20 cm über dem Boden) und der Winkel zwischen Ober- und Unterarm sollte 90 Grad entsprechen. Während einer Liegestütze ist der Kopf als Verlängerung der Wirbelsäule zu halten.
Bei der Ausführung einer Liegestütze sollten sowohl die Hände als auch die Füße circa schulterbreit aufgestellt ein. In dieser Arbeit liegen Ausführungen mit abweichender Hand- oder Fußstellung jedoch in dem Toleranzbereich von korrekt ausgeführten Liegestützen.

Anhand der genannten Kriterien lassen sich die Klassen bilden, die in Tabelle 2.1 definierten sind.

Klasse	Label	Definition
Klasse 0	Korrekt	Korrekte Ausführung
Klasse 1	Nicht korrekt	Es konnte keine komplette Liegestütze ausgeführt werden. Die Startposition der Liegestütze konnte nicht wieder erreicht werden.
Klasse 2	Nicht korrekt	Der Ellenbogen-/ Rumpfwinkel übersteigt 60 Grad
Klasse 3	Nicht korrekt	Die Plank Position kann nicht gehalten werden. Das Gesäß muss höher gebracht werden
Klasse 4	Nicht korrekt	Die Plank Position kann nicht gehalten werden. Das Gesäß muss niedriger gebracht werden
Klasse 5	Nicht korrekt	Die Bewegungsamplitude muss maximiert werden. Ellenbogen-/ Rumpfwinkel übersteigt 60 Grad. Der Kopf dient nicht als Verlängerung der Wirbelsäule
Klasse 6	Nicht korrekt	Gegen Ende der Übung fällt der Kopf nach unten ab. Der Kopf dient nicht als Verlängerung der Wirbelsäule.

Tabelle 2.1: Darstellung der Klassen, aus denen der Datensatz besteht

2.4 Feature Extraction

Für das Bestimmen der Qualität von Liegestützen auf Basis von Videodaten sind nicht alle Informatienen der Videodaten von Nutzen. Die trainierende Person macht dabei lediglich einen kleinen Bestandteil des Bildes aus. Im Folgenden werden zwei Möglichkeiten vorgestellt, welche es erlauben die zu analysierenden Daten zu verringern.

Human Instance Segmentation

Human Instance Segmentation ermöglicht die Segmentierung von Personen aus einem Bild. Hierbei wird jeder Pixel des Bildes einer bestimmten Klasse zugeordnet.[35] Somit wäre es möglich, die trainierende Person im Bild zu lokalisieren und dadurch die Menge der zu analysierenden Daten deutlich zu verringern.
Dies ist in Abbildung 2.2. dargestellt. Die Genauigkeit dieser Methode weist jedoch eine Anfälligkeit in Bezug auf die zu analysierende Person und dessen Körperproportionen auf.

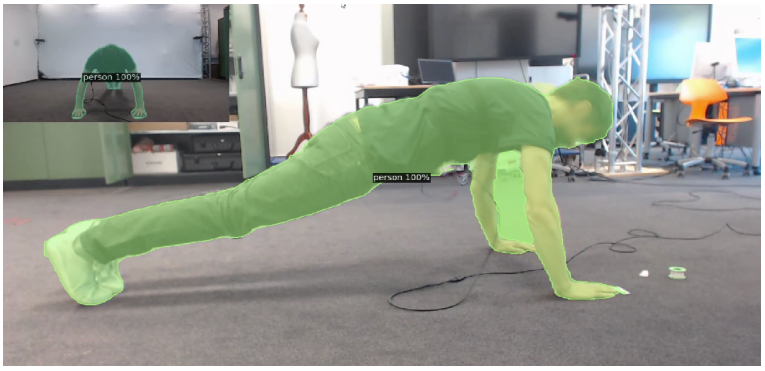


Abbildung 2.2: Darstellung der Human Instance Segmentation auf dem Datensatz dieser Arbeit

Human Pose Estimation

Human Pose Estimation beschreibt einen Prozess aus der Computer Vision. Wie in Abbildung 2.3 dargestellt, wird hierbei ermöglicht die Position von Körperschlüsselpunkten (Keypoints) einer oder mehrerer Personen in einem Bild zu bestimmen.[26] Die Anzahl der Keypoints hängt von dem verwendeten Pose Estimation Modell ab.

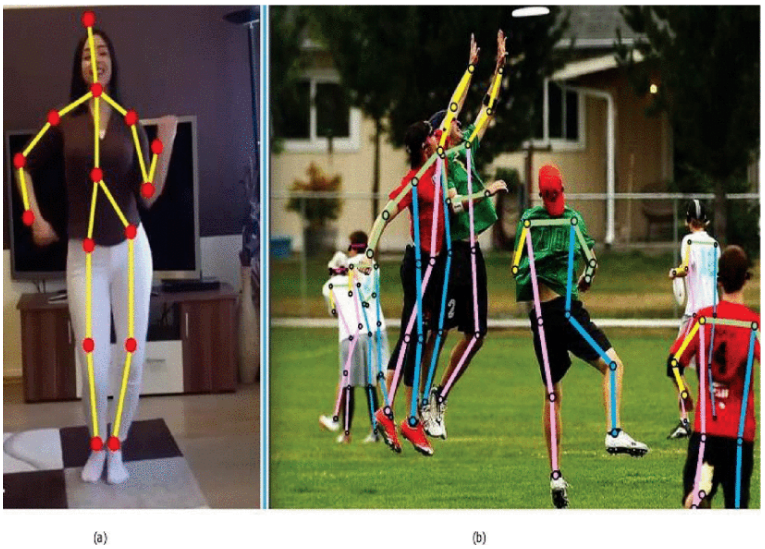


Abbildung 2.3: Gegenüberstellung der Human Pose Estimation einer oder mehrerer Personen[27]

Im ersten Schritt der Pose Estimation gilt es die Keypoint zu lokalisieren. Anschließend kann die Pose der Person, mit Hilfe dieser Keypoint dargestellt werden [26]. Zur Erkennung und Lokalisierung der Keypoints gibt es zwei verschiedene Verfahren.
Bei dem Top-Down Verfahren dient Object Detection zunächst dazu die Personen im Bild mittels Bounding Boxes zu lokalisieren. Daraufhin folgt die Lokalisierung der Keypoint in jeder erzeugten Bounding Box. Die Rechenzeit dieses Verfahren ist jedoch direkt proportional zu der Anzahl an Personen, die sich im Bild befinden, da die Erkennung der Keypoints der Personen einzeln erfolgt.
Bei dem zweiten Verfahren handelt es sich um das Bottom-Up Verfahren. Hierbei werden anfangs alle zu erkennenden Keypoints im Bild lokalisiert und anschließend den jeweiligen Personen zugeordnet. Dieses Verfahren weist andererseits Probleme auf, wenn sich die Keypoints unterschiedlicher Personen überschneiden und es kann zu fehlerhaften Erkennungen führen.

Um ein Pose Estimation Modell zu finden, welches für diesen Anwendungsfall geeignet ist, wurden mehrere Modelle miteinander verglichen.

Pose Estimation Modelle

- Bei „OpenPose“ [15] handelt es sich um ein open-source Pose Estimation Modell, welches die zwei- und dreidimensionale Echtzeiterkennung des Gesichtes, der Hände und des ganzen Körpers bereitstellt. Außerdem bietet OpenPose die Erkennung von mehreren Personen in einem Bild an.
- „MediaPipe“ ist ein von Google entworfenes Framework, welches eine Vielzahl von Anwendungsgebieten bereitstellt.[9] Bei „MediaPipe Pose“ handelt es sich um ein Modell zur Human Pose Estimation im zweidimensionalen Raum von einer Person. Hierbei werden bei der Erkennung 33 Keypoints lokalisiert. Des Weiteren ist die Verwendung von „MediaPipe Pose“ auch auf Android und IOS möglich.
- „EfficientPose“ [18] konzentriert sich auf die Erkennung einzelner Person.

2.5 Multivariate Zeitreihendaten

Für diese Arbeit wurde sich dazu entschieden Human Pose Estimation zu verwenden, da dies die Möglichkeit bietet die Informationen aus den Videodaten auf die Merkmale zu reduzieren, die für die Erkennung der Qualität relevant sind.
Hierbei wurde sich für das Modell von MediaPipe entschieden. Die Gründe für diese Entscheidung waren, neben der zuverlässigen Erkennung der Keypoints und der Performance, auch die Möglichkeit diese Arbeit auf eine mobile Anwendung zu erweitern. Wie die Abbildung 2.4 zeigt, ist es mit Pose Estimation möglich aus einer Videosequenz eine Sequenz von ausgewählten Keypoints zu generieren.

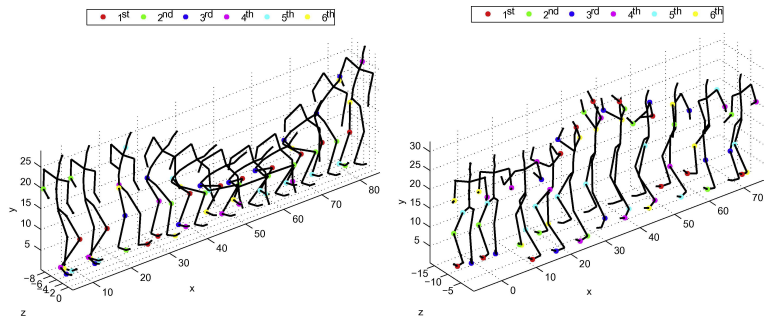


Abbildung 2.4: Eine beispielhafte Darstellung der Sequenz ausgewählter Keypoint zweier Aktionen [28]

Durch diese Keypoint-Sequenzen liegen die Daten dieser Arbeit nach der Verwendung der Human Pose Estimation als multivariate Zeitreihendaten vor.

Dabei handelt es sich um Daten, die in einer bestimmten Sequenz angeordnet sind. Somit ist neben den Daten selber, auch die Reihenfolge der Sequenz von Bedeutung. [17] Zeitreihendaten entsprechen der Datengrundlage der realen Welt und sind beispielsweise das Ergebnis von Sensor- und Kameraaufnahmen. Um sie korrekt interpretieren zu können, müssen auch die vorherigen Zustände mit in die Evaluierung einbezogen werden.
Somit ist es bei der Evaluierung der Qualität von Liegestützen nicht ausreichend Bilder zu betrachten, sondern es muss der komplette Vorgang einer Ausführung bewertet werden.

Long Short-Term Memory (LSTM)

Bei Long Short-Term Memory Networks handelt es sich um eine spezielle Art von rekurrenten neuronalen Netzen (RNNs)[19]. RNNs führen die Zeitkomponente in das Modell ein, indem sie „recurrent connections“ verwenden. Hierbei handelt es sich um Selbstverbindungen und Verbindungen zwischen Nodes, welche benachbarte Zeitschritte darstellen. Somit wird ein Gedächtnis erzeugt, mit dem ein zeitlicher Kontext betrachtet werden kann.
Bei LSTM handelt es sich um eine erweiterte Form von RNNs. Die Erweiterung liegt in der Einführung von Gedächtniszellen. Der Zustand dieser Zellen wird mit Hilfe von drei Gates gesteuert, welche sich aus dem Input-Gate, dem Forget-Gate und dem Output-Gate zusammensetzen. Diese Gates entscheiden, wie viele Informationen aufgenommen, vergessen oder an die nächste Zelle weitergegeben werden. Dadurch sind LSTMs in der Lage sich die wichtigen Informationen der bisherigen Sequenz zu merken und die irrelevanten Informationen zu vergessen. [20]

3 Umsetzung der Experimente

In diesem Kapitel wird die Implementation der Prozessschritte erläutert, die für die Umsetzung der Experimente nötig waren. Hierbei handelt es sich unter anderem um die Feature Extraction, sowie um die Vorverarbeitung der erzeugten Keypoints.

3.1 Rahmenbedingungen

Die Experimente, sowie die Prozessschritte der Datenvorverarbeitung, wurde in der Programmiersprache Python umgesetzt. Um die Keypoints der Videodaten zu erhalten, wurde das Pose Estimation Modell von Mediapipe[10] verwendet. Die Schritte der Vorverarbeitung dieser Keypoints wurden mit Hilfe der Programmbibliotheken „NumPy“[3] und „SciPy“[7] realisiert. Des Weiteren wurde die Bibliothek „OpenCV“ [4] für die Bilderverarbeitung verwendet. Für die Machine-Learning-basierte Qualitätserkennung mit Hilfe von LSTMs, dienten die Bibliotheken „Tensorflow“[8], „Keras“[2] und „Sklearn“ [6].

Der experimentelle Ablauf dieser Arbeit ist an den Prozessablauf der Activity Recognition Chain angelehnt. Die Activity Recognition Chain beschreibt den Prozessablauf und hat das Ziel die Aktivität von Personen bestimmen zu können. Hierfür sind fünf Prozessschritte definiert, bestehend aus: Raw Data, Preprocessing, Segmentation, Feature Extraction und der Classification.[14]

3.2 Implementation

Im Folgenden wird der Ablauf der Implementation beschrieben. Im Ursprung dieser Arbeit war es vorgesehen, dass zu Beginn eine Vorverarbeitung der Videodaten erfolgt, welche anschließend in die einzelnen Liegestütz-Wiederholungen unterteilt werden. Daraufhin sollten mit Hilfe von Mediapipe-Pose die ausgewählten Keypoints bestimmt werden. Jedoch stellte sich dabei heraus, dass Mediapipe-Pose bei einigen Probanden erst nach einiger Zeit in der Lage ist die Keypoints zuverlässig zu liefern. Aus diesem Grund erfolgte die Segmentierung nicht auf den Videodaten, sondern auf den Keypoint-Sequenzen.

Der Ablauf der Prozessschritte, die in dieser Arbeit verwendet wurden, ist in Abbildung 3.1 dargestellt. Zunächst erfolgte die Vorverarbeitung der Videodaten, da die beiden Perspektiven zusammen in einem Video abgespeichert wurden. Anschließend wurde mit der Erkennung der Keypoints im Zuge der Feature Extraction fortgefahren. Im Anschluss erfolgte der erste Schritt der Vorverarbeitung der Keypoints, bestehend aus dem Glätten der X- und Y-Werte der erkannten Keypoints. Im nächsten Schritt wurden die Videodaten in die einzelnen Liegestütz-Wiederholungen unterteilt und konnten den jeweiligen Klassen zugeordnet werden. Da die Sequenzlängen der Keypoints unterschiedlich lang waren, folgte eine Interpolation. In dieser Arbeit werden zwei unterschiedliche Varianten untersucht, die Pose der trainierenden Person darzustellen. Zum einen durch die Verwendung der Winkel zwischen ausgewählten Körperteilen und zum anderen durch die Koordinaten der Keypoints. Die Variante, welche die Koordinaten der Keypoints verwendet, benötigt zusätzlich eine Normalisierung der Position im Bild und der Körperproportionen. Diese Schritte sind bei der Berechnung der Winkel irrelevant, da es sich bei den Winkeldaten um relative Werte handel, welche unabhängig von der Position der Pose sind. Bevor die Qualitätserkennung durchgeführt werden kann, ist zunächst eine Transformation nötig.

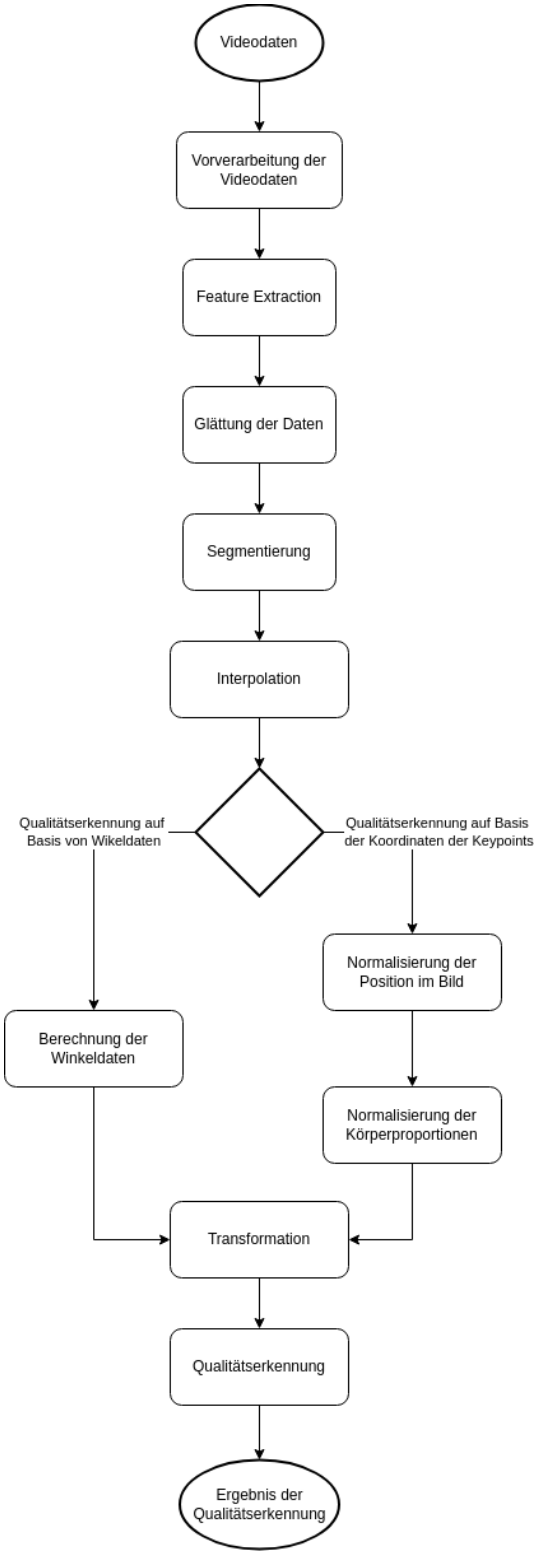


Abbildung 3.1: Ablaufdiagramm, welches die Prozessschritte darstellt, aus denen die Implementation dieser Arbeit besteht.

3.2.1 Vorverarbeitung der Videodaten

In den vorliegenden Videodaten wurden die beiden Perspektiven zusammen in einem Video abgespeichert. Im ersten Schritt der Vorverarbeitung wurden diese mit Hilfe von OpenCV voneinander isoliert, um sie separat in das Pose Estimation Modell geben zu können. Die Aufnahmen aus den beiden Perspektiven verfügten anschließend über unterschiedliche Auflösungen. Dies war jedoch nicht problematisch, da das Mediapipe-Pose nicht die Bildkoordinaten der Keypoints liefert, sondern die Bildkoordinaten mittels der Bildbreite bzw. Bildhöhe zuvor auf [0.0, 1.0] abbildet. Wie in Abbildung 3.2 dargestellt, wird somit beispielsweise ein Keypoint, der sich mit den Koordinaten (512, 126) in einem Bild mit einer Bildbreite von 1280 Pixeln und einer Bildhöhe von 720 Pixeln auf die Koordinaten (0.4, 0.175) abgebildet.

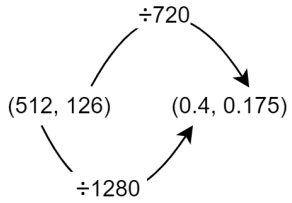
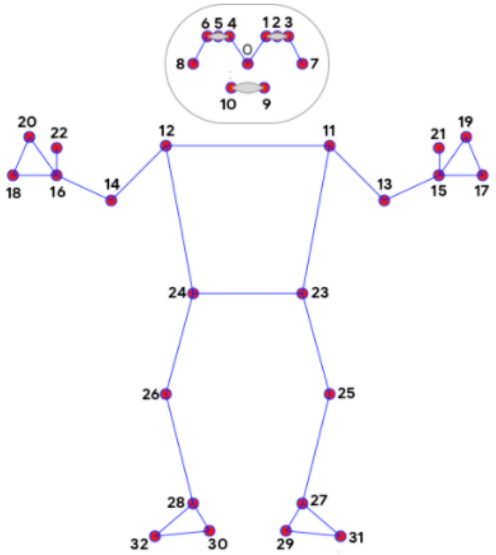


Abbildung 3.2: Beispielhaft dargestellt ist die Umrechnung der Koordinaten der Keypoints von Mediapipe Pose. Mit Hilfe der Division des X-Wertes der Koordinate durch die Bildbreite und der Division des Y-Wertes der Koordinate durch die Bildhöhe, werden die Koordinaten auf [0.0, 1.0] abgebildet. Die Koordinate (512, 126) wird somit bei einem Bild mit den Abmessungen 1280x720 auf (0.4, 0.175) abgebildet.

3.2.2 Feature Extraction

Die Feature Extraction der Videodaten erfolgt mit Hilfe des Pose Estimation Modells von Mediapipe [9]. Dieses Modell, namens Mediapipe-Pose, liefert die in Abbildung 3.3 dargestellten 32 Keypoints, wobei die X- und Y-Werte der Keypoints als Prozentwert relativ zur Videoauflösung dargestellt sind.
Für den Anwendungsfall dieser Arbeit ist es nicht notwendig alle 32 Keypoints zu verwenden. Es wurden nur die Keypoints ausgewählt, die auch konstant von der Kamera erfasst werden konnten und der Qualitätserkennung dienlich sind. Beispielsweise ist Mediapipe-Pose in der Lage 11 Keypoints (landmarks 0-10) zu erkennen, die das Gesicht einer Person betreffen. Da dies für diese Arbeit nicht notwendig ist, wird hierbei nur die Position der Nase weiterverwendet, um eine Aussage über die Haltung des Kopfes treffen zu können.



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right_knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

Abbildung 3.3: Darstellung der 33 Keypoints, die Mediapipe-Pose zu erkennen in der Lage ist

Für die seitliche und die frontale Ansicht auf die trainierende Person, wurden jeweils folgende Keypoints verwendet:

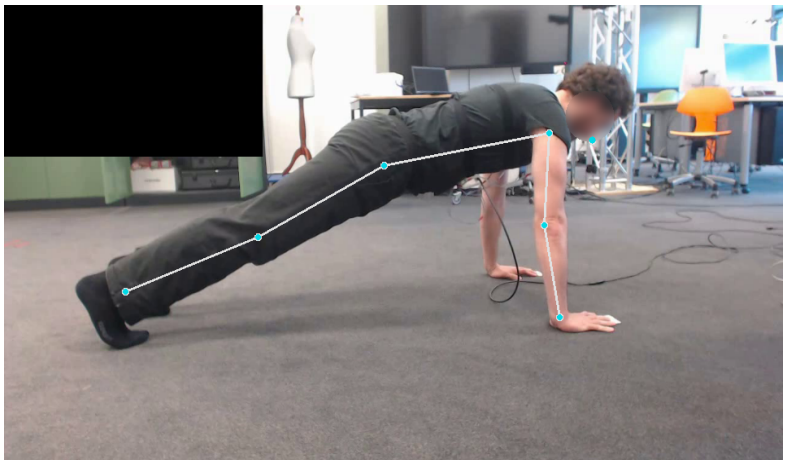
Perspektive	Keypoints
	• Nase • Rechtes Handgelenk • Rechter Ellenbogen • Rechte Schulter • Rechte Hüfte • Rechtes Knie • Rechter Knöchel

Tabelle 3.1: Darstellung der verwendeten Keypoints bei der Seitenansicht

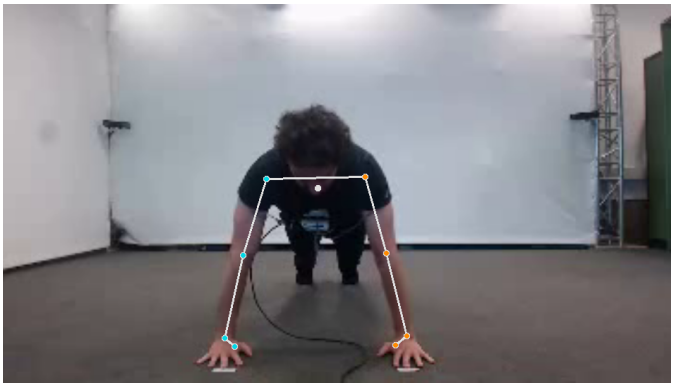
Perspektive	Keypoints
	• Nase • Rechte Schulter • Linke Schulter • Rechter Ellenbogen • Linker Ellenbogen • Rechtes Handgelenk • Linkes Handgelenk • Rechter Daumen • Linker Daumen

Tabelle 3.2: Darstellung der verwendeten Keypoints bei der frontalen Ansicht

3.2.3 Glättung der Daten

Bei den erkannten Keypoints sind teilweise Sprünge und zudem eine Art „Zittern“ festzustellen. Um dies zu unterdrücken, wurde der „Savitzky-Golay-Filter“ [32] der Bibliothek „SciPy“ [5]verwendet. Die Auswirkung, welche der Filter auf die Koordinaten der Keypoints hat, sind in Abbildung 3.4 dargestellt. Die Glättung wird hier anhand des Y-Wertes des Keypoints des rechten Ellenbogens aus der seitlichen Perspektive gezeigt. Die Ausführung erstreckt sich lediglich über 37 Zeitschritte, da noch keine Interpolation stattgefunden hat. Der Verlauf der Daten wurde durch das Anwenden des Savitzky-Golay-Filters erhalten, wobei die Sprünge und das Zittern herausgefiltert wurde.
Bei dem Savitzky-Golay-Filter handelt es sich um einen Polynom-Glättungsfilter, welcher anhand von zwei Parametern angepasst werden kann. Bei den Parametern handelt es sich zum einen um die Fenstergröße des Filters und zum anderen um die Ordnung des Polynoms. In dieser Arbeit wurde die Fenstergröße 25 und ein Polynom der dritten Ordnung gewählt.

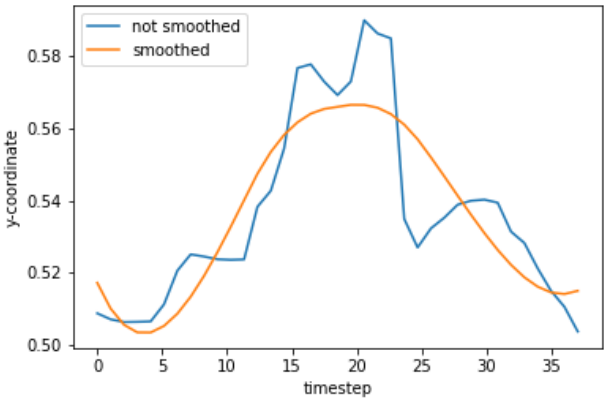


Abbildung 3.4: Dargestellt ist der Vergleich zwischen den nicht verarbeiteten Rohdaten und den Daten nach der Glättung durch den Savitzky-Golay-Filter. Hierbei handelt es sich um den Y-Wert der Koordinate des rechten Ellenbogens aus der seitlichen Perspektive über eine Ausführung hinweg.

3.2.4 Segmentierung

Die Keypoint-Sequenzen wurden mit Hilfe der Videodaten in die einzelnen Liegestütz-Wiederholungen unterteilt. Dies war möglich, da die Videodaten bei der Feature Extraction Bild für Bild in das Pose Estimation Modell gegeben wurden, wodurch die Länge der Keypoint-Sequenz der Länge der jeweiligen Videosequenz entsprach. Dies erfolgte für die beiden Perspektiven der jeweiligen Ausführung.

3.2.5 Interpolation

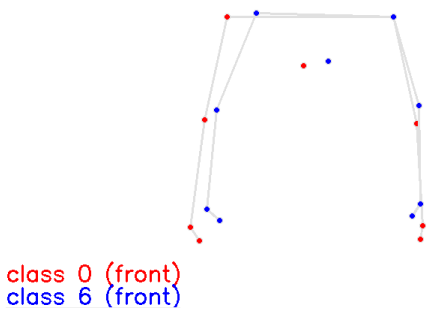
Die Sequenzen der Liegestütz-Wiederholungen besitzen nicht dieselbe Länge, da die Ausführung gegen Ende eines Satzes, aufgrund der Anstrengung, zunehmend langsamer ausgeführt werden. Des Weiteren ist die variierende Sequenzlänge darauf zurückzuführen, dass die Liegestütze von mehreren Probanden mit unterschiedlichen physischen Voraussetzungen durchgeführt wurden. Aus diesem Grund ist eine Interpolation der generierten Keypoints erforderlich.
Zunächst wurde festgelegt, dass eine Wiederholung aus 60 Zeitschritten zu bestehen hat. Anschließend konnte mit Hilfe der Funktion „interpolate.interp1d“ der Bibliothek „SciPy“ eine Interpolationsfunktion generiert werden, mit der die Werte jedes Keypoints interpoliert werden konnten. Somit konnten fehlende Werte bei Ausführungen, die nicht der definierten Sequenzlänge entsprachen, angenähert werden.

3.2.6 Normalisierung der Position im Bild

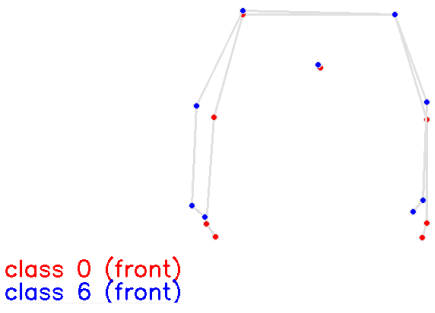
Die in dieser Arbeit verwendeten Videos wurden stets aus denselben Perspektiven aufgenommen. Aus diesem Grund ist es nicht notwendig eine perspektivische Tranformation vorzunehmen.
Die Positionierung der trainierenden Personen im Bild ist jedoch nicht immer dieselbe, weshalb die Keypoints in ein einheitliches Koordinatensystem gebracht werden müssen. Hierfür wurde festgelegt, dass der Keypoint der rechten Schulter, sowohl bei der Frontal- als auch bei der Seitenansicht, als statischer Punkt mit vorgegebenen Koordinaten definiert ist. Die Differenz zwischen der vorgegebene Koordinate und der rechten Schulter der trainierenden Person wird anschließend auf alle Keypoints der Person angewandt. Dadurch ist gegeben, dass sich alle Personen in derselben Position im Bild befinden.

3.2.7 Normalisierung der Körperproportionen

Da die Liegestützen des Datensatzes von unterschiedlichen Personen durchgeführt wurden, ist eine Generalität der Qualitäserkennung, in Bezug auf die trainierende Person gegeben. Jedoch ist eine Vorverarbeitung der Keypoints, aufgrund der unterschiedlichen Körperproportionen dieser Personen notwendig.
In der Frontalansicht wird die Normalisierung basierend auf der Schulterbreite durchgeführt. Eine beispielhafte Darstellung dieser Normalisierung ist in Abbildung 3.5 zu sehen. Verwendet werden hier die Keypoints einer Ausführung der Klasse 0 und der Klasse 6. Zunächst wird eine Schulterbreite definiert, welche als Norm gewertet wird. Im nächsten Schritt wird die Distanz zwischen dem Keypoint der rechten und dem der linken Schulter berechnet und die Differenz zur vorgegebenen Norm gebildet. Anschließend folgt die Normalisierung, indem die Keypoints der rechten Körperhälfte, bestehend aus der rechten Schulter, dem rechten Ellenbogen, dem rechten Handgelenk und dem rechten Daumen, um die berechnete Differenz in X-Richtung verschoben werden.



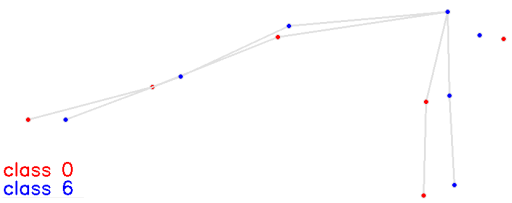
(a) Die Normalisierung der Körperproportionen hat noch nicht stattgefunden. Die Differenzen der Schulterbreiten zur ausgewählten Norm werden berechnet.



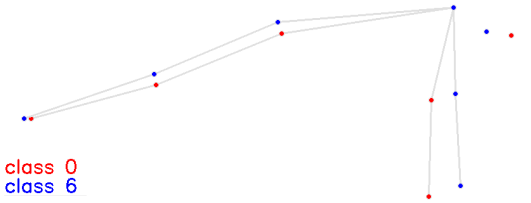
(b) Nach der Normalisierung der Körperproportionen. Die Keypoints der rechten Körperhälfte, welche aus der rechten Schulter, dem rechten Ellenbogen, dem rechten Handgelenk und dem rechten Daumen bestehen, werden in X-Richtung um die Differenz der Schulterbreite nur Norm verschoben.

Abbildung 3.5: Dargestellt sind die Keypoints einer Wiederholung aus der frontalen Ansicht der Klasse 0 und der Klasse 6. Eine Normalisierung der Position im Bild hat bei diesen Keypoints bereits stattgefunden.

Bei der seitlichen Ansicht erfolgt die Normalisierung auf Basis der Länge des Torsos. Dies ist in Abbildung 3.6 dargestellt. Zunächst wird eine Torsolänge definiert, die es einzuhalten gilt. Daraufhin wird die Distanz zwischen dem Keypoint der rechten Schulter und dem Keypoint der rechten Hüfte berechnet. Basierend auf der Differenz zwischen dieser Distanz und der vordefinierten Torsolänge, wird anschließend die Länge des Torsos und die Länge des rechten Beines angepasst. Hierfür werden die Keypoints der Hüfte, des Knies und des Fußgelenks in X-Richtung verschoben.



(a) Die Normalisierung der Körperproportionen hat noch nicht stattgefunden. Die Differenzen der Torsolänge zur ausgewählten Norm werden berechnet.



(b) Nach der Normalisierung der Körperproportionen. Die Keypoints der Hüfte, des Knies und des Fußgelenks werden in X-Richtung um die Differenz der Torsolänge nur Norm verschoben

Abbildung 3.6: Dargestellt sind die Keypoints einer Wiederholung aus der seitlichen Ansicht der Klasse 0 und der Klasse 6. Eine Normalisierung der Position im Bild hat bei diesen Keypoints bereits stattgefunden.

3.2.8 Berechnung der Winkeldaten

Zusätzlich zu Modellen, welche die Klassifikation anhand der Koordinaten der Keypoints vornehmen, sind Experimente mit Modellen durchzuführen, welche die Winkel zwischen ausgewählten Körperteilen verwenden. Für die Berechnung dieser Winkel wurde der Kosinussatz verwendet.

Bei der Videosequenz der frontalen Perspektive wurden zum einen die Winkel zwischen dem Ober- und Unterarm verwendet (Abbildung 3.7). Außerdem wurde der Winkel zwischen Unterarm und Daumen betrachtet, um die Stellung der Hand während der Ausführung überprüfen zu können. Diese Winkel wurden jeweils für beide Arme berechnet.

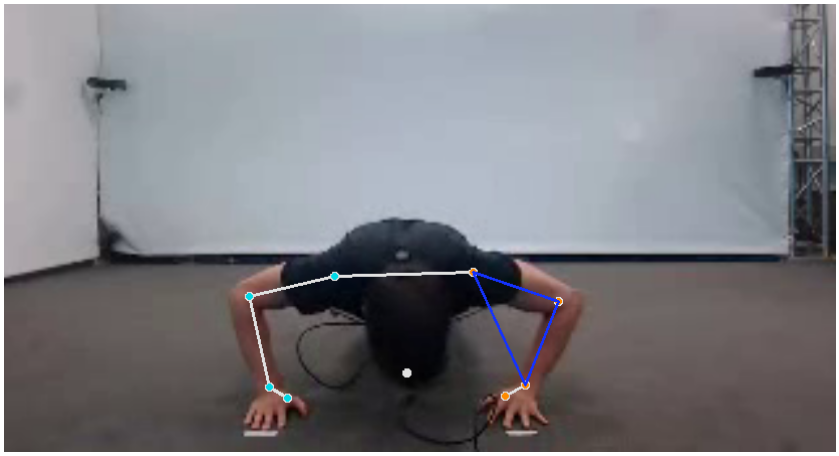


Abbildung 3.7: Aufnahme aus der frontalen Ansicht mit eingezeichnetem Dreieck (dunkelblau) am linken Arm, um den Winkel berechnen zu können.

In den Videosequenzen aus der seitlichen Ansicht wurden hauptsächlich Winkel gewählt, welche die Körperspannung der trainierenden Person repräsentieren. Hierfür dienten die Winkel zwischen Unter- und Oberschenkel und zwischen Oberschenkel und Oberkörper. Um die Haltung des Kopfes erfassen zu können, wurde der Winkel zwischen dem Oberkörper und dem Nacken verwendet. Dies ist in Abbildung 3.8 dargestellt. Außerdem wurde der Winkel zwischen dem Oberkörper und dem Oberarm verwendet, um eine präzisere Bestimmung der Armstellung möglich zu machen.

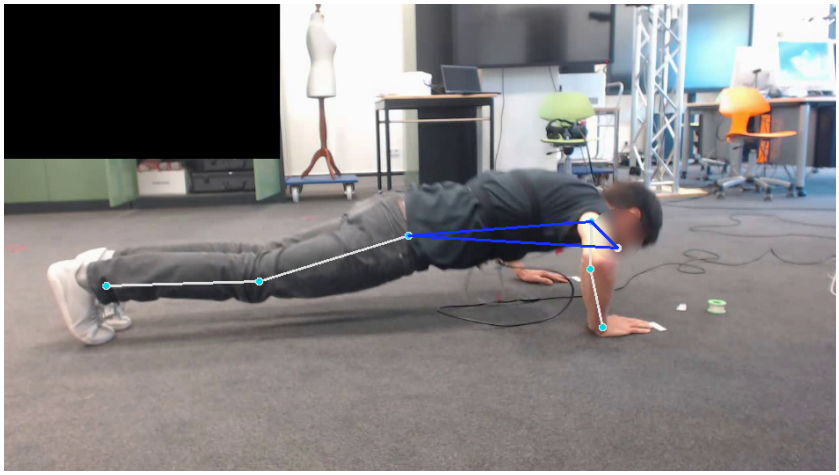


Abbildung 3.8: Aufnahme aus der seitlichen Ansicht mit eingezeichnetem Dreieck (dunkelblau), um die Haltung des Kopfes bestimmen zu können.

3.2.9 Transformation

Bevor die Daten mittels Machine Learning analysiert werden können, muss zunächst eine Transformation stattfinden. Die Trainingsdaten für das LSTM benötigen folgendes Format: [Samples, Timesteps, Features]
„Samples“ beschreibt die Anzahl an vorliegenden Trainingsdaten. „Timesteps“ drückt dabei aus, über wie viele Zeitschritte ein Sample verfügt, wobei „Features“ angibt, wie viele Daten die pro Timesteps vorliegen.
Um dieses Datenformat zu gewährleisten wurde die Funktion „reshape“ der Bibliothek NumPy [1] verwendet. Im Anschluss wurden die Keypoint-Sequenzen der beiden Perspektiven einer Wiederholung mit Hilfe der Funktion „hstack“ der Bibliothek NumPy [1] horizontal zusammengeführt.
Die Labels der Trainingsdaten wurden als ein eindimensionales Array generiert, wobei der Inhalt des Array der jeweiligen Klasse der zugehörigen Liegestütz-Wiederholung entspricht. Für das Trainieren des LSTMs Netzes muss auch bei den Labels eine Transformation vorgenommen werden. Verwendet wurde hierfür das Hot-One-Encoding, bei dem jedes Label in eine binäre Vektorform gebracht wird. Die Länge des Vektors entspricht hierbei der Anzahl an Klassen, zwischen denen bei der Klassifikation unterschieden werden kann. Im Vektor steht an der Stelle eine 1, dessen Klasse das Label angehört. An den restliche Stellen des Vektors liegt eine 0 vor.

4 Evaluation

Im Evaluationskapitel wird zunächst die Ausführung der Experimente beschrieben. Im Anschluss erfolgt die Auswertung der Ergebnisse, wobei die beiden unterschiedlichen Darstellungsmöglichkeiten der Pose der trainierenden Personen miteinander verglichen werden.

4.1 Ablauf der Experimente

Die Experimente wurden in einem Python Projekt durchgeführt. Das Ziel der Experimente ist es zu prüfen, ob eine Qualitätserkennung von Fitnessübungen möglich ist. In Teilabschnitt 3.2 sind die Prozessschritte beschrieben, in denen die Keypoints der Pose Estimation verarbeitet wurden. Hierbei wurde zwischen zwei unterschiedlichen Varianten unterschieden. Durch die Experimente soll außerdem herausgefunden werden, mit welcher dieser Varianten eine genauere Klassifizierung möglich ist. Des Weiteren wird neben einer binären Klassifizierung, zusätzlich eine Mehrklassen-Klassifzierung untersucht.

4.2 Versuchsaufbau

Die Trainingsdaten liegen in zwei unterschiedlichen Varianten vor. Zum einen als Koordinaten der Keypoints und zum anderen als Winkeldaten von ausgewählten Körperteilen. Die Daten wurden nach der Transformation, beschrieben in Abschnitt 3.2.9, als CSV-Dateien abgespeichert.
Bevor mit dem Trainieren der LSTM-Netze begonnen werden kann, müssen die Trainingsdaten und die dazugehörigen Labels eingelesen werden. Die Trainingsdaten müssen den LSTMs in einem bestimmten Format vorliegen. Dieses Format ist als dreidimensionales Array definiert, welches sich in Reihenfolge aus der Anzahl der Samples, der Anzahl an

Timesteps pro Sample und der Anzahl an Features in jedem Timestep zusammensetzt. Anschließend folgte das Trennen des Datensatzes in Trainingsdaten, Validierungsdaten und Testdaten. Die Trainingsdaten dienen dazu, ein Modell zu trainieren. Mit den Validierungsdaten wird ein Auswendiglernen der Trainingsdaten während des Trainings verhindert. Der Testdatensatz dient im Anschluss zum Evaluieren des Modells.
Es wurden zusätzliche Liegestütze der unterschiedlichen Klassen aufgenommen, um das Modell darüber hinaus mit Ausführung von einer Person zu testen, welche dem Modell unbekannt ist.

Im ersten Durchlauf gelang es dem Modell eine Validierungsgenauigkeit von 98% zu erreichen. Jedoch gelang es nicht die Genauigkeit auch im Testen des Modells zu wiederholen. Der Grund dafür ist die unausgewogene Verteilung des Datensatzes. Bei der Untersuchung einer Mehrklassen-Klassifizierung wird zwischen sieben Klassen unterschieden, welche in Teilabschnitt 2.3 beschreiben wurden. Die Verteilung der Daten wird in Abbildung 4.1 dargestellt.

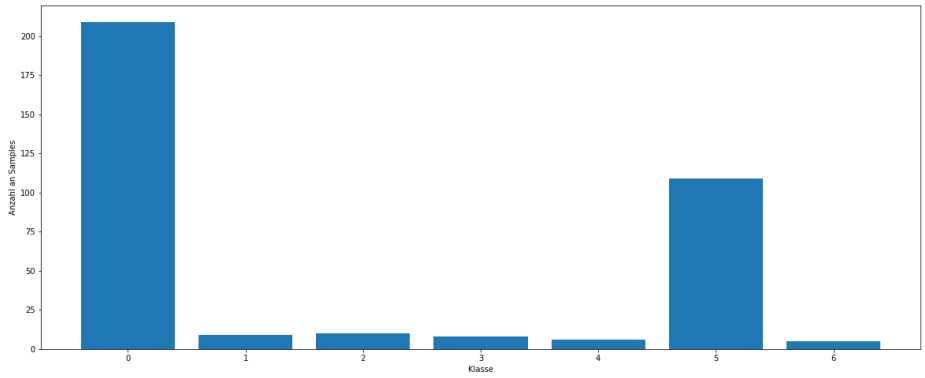


Abbildung 4.1: Abgebildet ist die Verteilung des Datensatzes auf die jeweiligen Klassen. Bei der Klasse 0 handelt es sich um die Liegestütze in korrekter Ausführung. Die Klassen 1 bis 6 beschreiben jeweils nicht korrekt ausgeführte Liegestütze. Die Fehlerklassifizierung ist in Teilabschnitt 2.3 definiert.

Bei der Aufteilung der Daten in Trainings- und Validierungsdaten, besteht somit die Möglichkeit, dass das Auswendiglernen für Klassen, die weniger vertreten sind, nicht gewährleistet wird. Um dies zu verhindern, wurde dafür gesorgt, dass auch Daten dieser Klassen im Validierungsdatensatz vorhanden sind.

Architektur

Bei den LSTMs, die in dieser Arbeit verwendet wurden, handelt es sich um mehrschichtige LSTM-Netzwerke. Die Architektur der LSTMs ist in Abbildung 4.2 dargestellt. Zum Erstellen der Modelle wurden die Bibliotheken „Tensorflow“ und „Keras“ verwendet.
Ein Bestandteil der LSTM-Netzwerke, welche in dieser Arbeit verwendet wurden, sind LSTM-Layer. Die LSTM-Layer werden mit einer Menge an Neuronen definiert. Das erste LSTM-Layer erhält zudem die Struktur der transformierten Features als Eingabeformat. Um Overfitting, also das spezialisieren des Modells auf die Trainingsdaten, zu verhindern, befindet sich nach jeder LSTM Layer eine Dropout-Layer. Als Dropout-Rate wird hier 0,5 gewählt. Insgesamt verfügt die in dieser Arbeit verwendeten LSTMs über zwei LSTM-Hidden-Layer. In Anschluss wurden zwei Dense-Layer verwendet, um die Ergebnisse mit Hilfe der Softmax-Aktivierungsfunktion auf die jeweiligen Klassen abzubilden.

Das Trainieren der LSTM-Netze erfolgte jeweils in 100 Epochen. Als Fehlerfunktion wurde Categorical Cross-Entropy [24] verwendet und als Optimizer diente der Adam Optimizer [21].

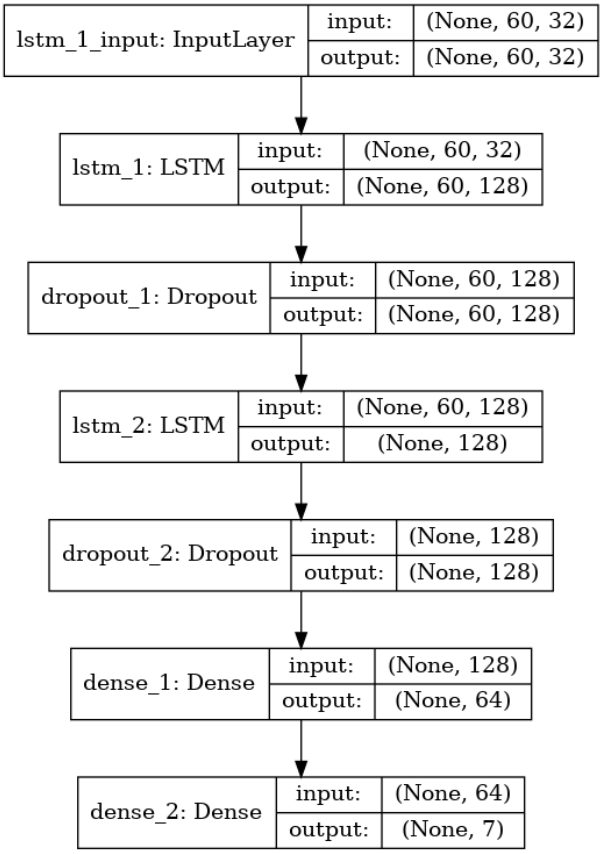
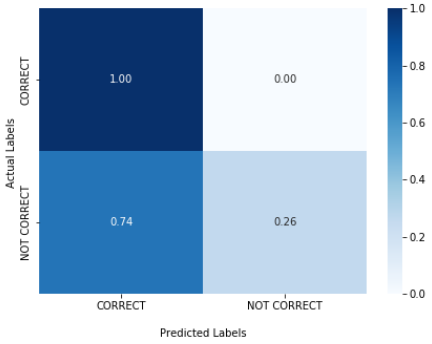


Abbildung 4.2: Die Architektur der LSTMs, die in dieser Arbeit verwendet wurde. Bestehend aus dem Input-Layer, gefolgt von zwei LSTM-Layern. Nach jeder LSTM-Layer befindet sich eine Dropout-Layer, um das Auftreten von Overfitting zu verhindern. Am Ende des LSTMs befinden sich zwei Dense-Layer, um das Ergebnis auf die jeweiligen Klassen abzubilden.

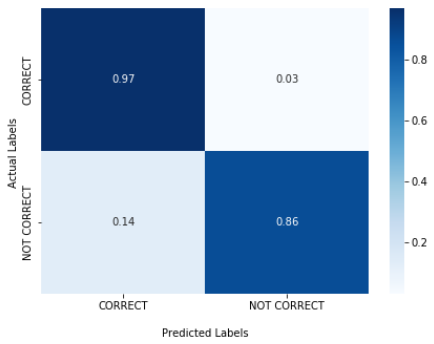
4.3 Auswertung der Ergebnisse

Binäre Klassifikation

Zu Beginn wurde die binäre Qualitätserkennung untersucht. Bei der binären Klassifizierung wird ausschließlich zwischen einer korrekten und einer nicht korrekten Ausführung unterschieden. Verglichen wird hierbei zwischen dem Modell, welches die Qualität anhand der Koordinaten der Keypoints beurteilt und dem Modell, welches die Winkeldaten verwendet. Mit Hilfe einer Konfusionsmatrix lässt sich die Genauigkeit des Klassifikators beurteilen.
Wie in Abbildung 4.3 zu erkennen, sind beide Modelle in der Lage die korrekten Ausführungen auch als solche zu erkennen. Bei dem Modell, welches mit den Koordinaten der Keypoints als Parameter arbeitet (Abbildung 4.3a), liegt die Genauigkeit bei 100%. Dieser Wert ist jedoch nicht realistisch und ist unter anderem auf die geringe Anzahl an Testdaten zurückzuführen. Außerdem werden auch 74% der fehlerhaften Ausführungen als korrekte Ausführungen erkannt. Dies resultiert in einer Testgenauigkeit des Modells von 63%. Die Genauigkeit der Erkennung der Ausführungsqualität liegt bei dem Modell, welches mit den Winkeldaten arbeitet (Abbildung 4.3b), mit 91,5% deutlich höher.



(a) Konfusionsmatrix des Modells, welches die Koordinaten der Keypoints verwendet. Verfügt über eine Testgenauigkeit von 63%

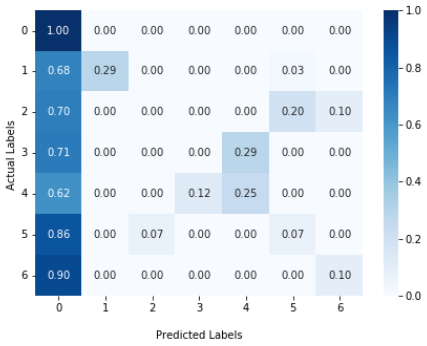


(b) Konfusionsmatrix des Modells, welches die Winkeldaten verwendet. Verfügt über eine Testgenauigkeit von 91,5%

Abbildung 4.3: Konfusionsmatrizen mit einer binären Klassifizierung des gegebenen Datensatzes

Mehrklassen-Klassifikation

Als nächstes wurde die Genauigkeit einer mehrklassigen Klassifizierung untersucht. Die Konfusionsmatrizen dieser Untersuchung sind in Abbildung 4.4 dargestellt. Auch hier ist zu erkennen, dass das Modell, welches mit den Winkeldaten arbeitet, über eine zuverlässigere Klassifizierung der Ausführungsqualität verfügt. Dennoch hat auch dieses Modell bei bestimmten Klassen Probleme die Ausführung richtig zu klassifizieren.



(a) Konfusionsmatrix des Modells, welches die Koordinaten der Keypoints verwendet. Verfügt über eine Testgenauigkeit von 24,46%

(b) Konfusionsmatrix des Modells, welches die Winkeldaten verwendet. Verfügt über eine Testgenauigkeit von 57,29%

Abbildung 4.4: Konfusionsmatrizen mit einer spezifischen Klassifizierung des gegebenen Datensatzes

Ein Grund dafür ist die bereits beschriebene ungleiche Verteilung der Klassen. Um dem entgegenzuwirken wurden Klassen-Gewichte verwendet. „Keras“ [2] bietet an, dem Modell für das Trainieren diese Klassen-Gewichte mit zu übergeben. Diese können verwendet werden, um bestimmten Klassen während des Trainings eine gewisse Wichtigkeit beizumessen oder ein Datensatz, mit ungleich verteilten Daten auf die entsprechenden Klassen, auszubalancieren. Als Klassen-Gewichte wurden jeweils die Differenzen der Anzahl an Daten der Klassen 1 bis 6 zur Anzahl vorliegender korrekter Ausführungen berechnet. Diese wurden dem Modell zum Training übergeben.

Anhand der Abbildung 4.5 lässt sich erkennen, dass das Verwenden von Klassen-Gewichten bei beiden Modellen für eine Verbesserung der Klassifizierung sorgt. Das Modell, welches die Klassifikation auf Basis der Winkeldaten vollzieht, weist eine Testgenauigkeit von

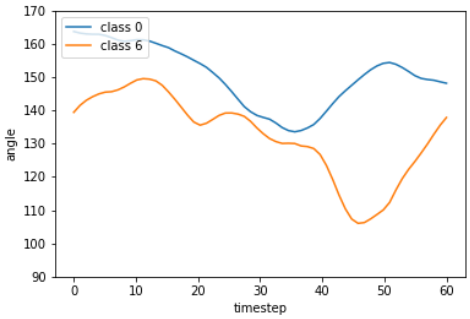
60,71% auf. Das Modell, welches die Koordinaten der Keypoints verwendet, besitzt lediglich eine Testgenauigkeit von 46,42%.

(a) Konfusionsmatrix des Modells, welches die Koordinaten der Keypoints verwendet. Verfügt über eine Testgenauigkeit von 46,42%

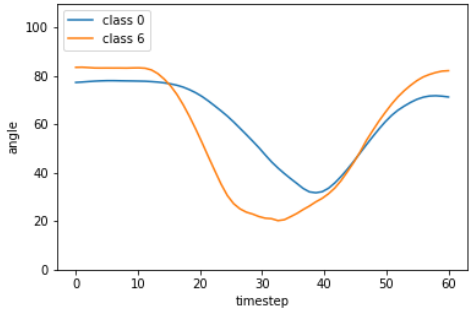
(b) Konfusionsmatrix des Modells, welches die Winkeldaten verwendet. Verfügt über eine Testgenauigkeit von 60,71%

Abbildung 4.5: Konfusionsmatrizen mit einer spezifischen Klassifizierung des gegebenen Datensatzes, unter Hinzunahme der Klassen-Gewichte

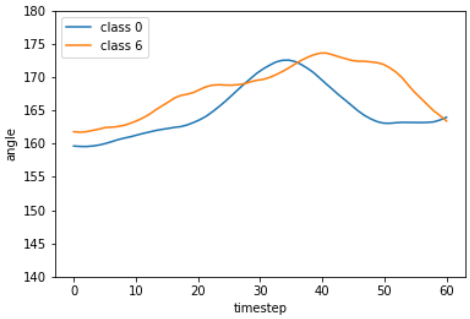
Ein weiterer Grund für die Schwierigkeiten zwischen den verschiedenen Klassen zu unterscheiden sind die definierten Kriterien. Beispielsweise ist die Kopfhaltung in der Aufwärtsbewegung einer Liegestütz-Ausführung das ausschlaggebende Kriterium für eine fehlerhafte Ausführung der siebten Klasse. Hierbei handelt es sich jedoch nur um einen Teil des multivariaten Samples. Dies ist in Abbildung 4.6 zu erkennen, da die Abweichung zwischen der korrekten und der inkorrekten Ausführung lediglich in den Winkeldaten des Kopfes zu erkennen ist (Abbildung 4.6a). Währenddessen sind keine bedeutenden Differenzen zwischen den übrigen Winkeldaten der beiden Personen vorhanden. Beispielsweise ist dies an dem rechten Arm, der Hüfte oder dem rechten Bein zu erkennen (Abbildungen 4.6b - 4.6d).



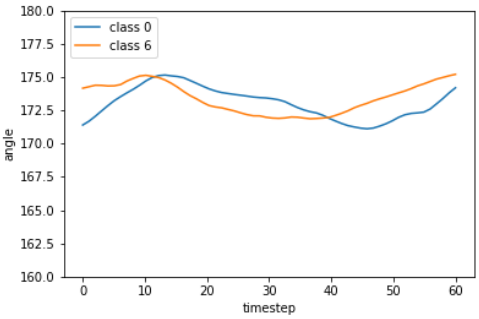
(a) Zeigt den Winkel des Kopfes während einer Ausführung



(b) Zeigt den Winkel des rechten Armes während einer Ausführung



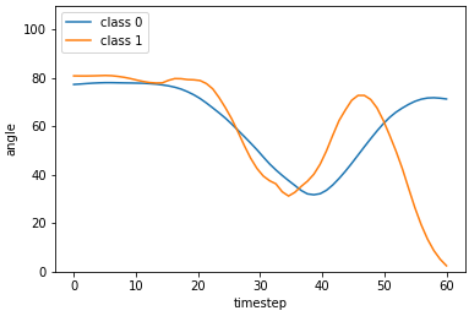
(c) Zeigt den Winkel der rechten Seite der Hüfte während einer Ausführung



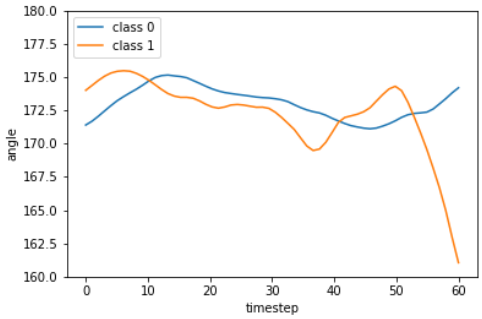
(d) Zeigt den Winkel des rechten Knies während einer Ausführung

Abbildung 4.6: Vergleich von vier Winkeldaten aus der seitlichen Perspektive der Klasse 6 mit der Klasse 0. Bei der Klasse 6 handelt es sich um die Klasse, in der der Kopf der trainierenden Person gegen Ende der Ausführung nach unten abfällt. Bei der Klasse 0 handelt es sich um eine korrekte Ausführung. Das Abfallen des Kopfes ist in Abbildung (a) zu erkennen.

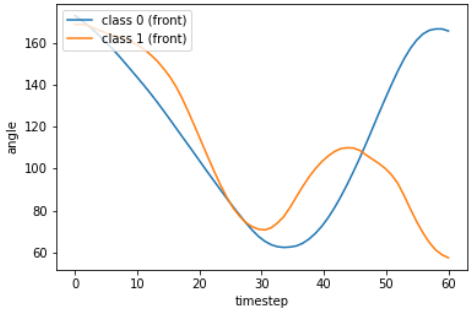
Im Gegensatz dazu sind Ausführungen, bei denen die Liegestütze nicht vollendet werden konnte (Klasse 1), deutlich von den restlichen Klassen differenzierbar. Dies ist in den Abbildungen 4.7a - 4.7c insofern zu erkennen, dass an allen Körperteilen gegen Ende der Ausführung ein Abfall des Winkels auszumachen ist. Hier wird eine Ausführung der Klasse 1 mit einer korrekten Ausführung der Klasse 0 verglichen.



(a) Zeigt den Winkel des rechten Armes während einer Ausführung aus der seitlichen Ansicht



(b) Zeigt den Winkel des rechten Beins während einer Ausführung aus der seitlichen Ansicht

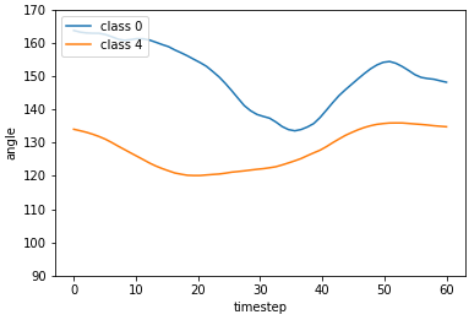


(c) Zeigt den Winkel des linken Armes während einer Ausführung aus der frontalen Ansicht

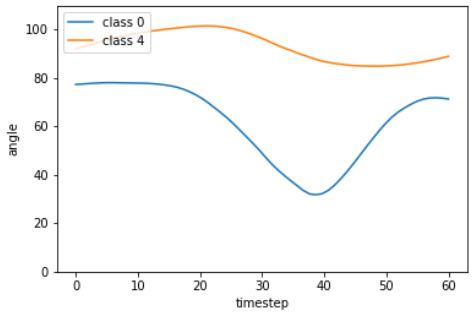
Abbildung 4.7: Vergleich von zwei Winkeldaten aus der seitlichen Perspektive und einer aus der frontalen Perspektive der Klasse 1 mit der Klasse 0. Bei der Klasse 1 handelt es sich um die Klasse, in der keine komplette Ausführung vollzogen werden konnte. Bei der Klasse 0 handelt es sich um eine korrekte Ausführung.

Des Weiteren bestehen einige Klassen aus mehreren Kriterien, wobei einzelne dieser Kriterien wiederum eigene Klassen bilden. Diese Klassen voneinander zu unterscheiden stellt eine Schwierigkeit dar, wie an den Klassen 2 und 5 in der Abbildung 4.5b zu erkennen.

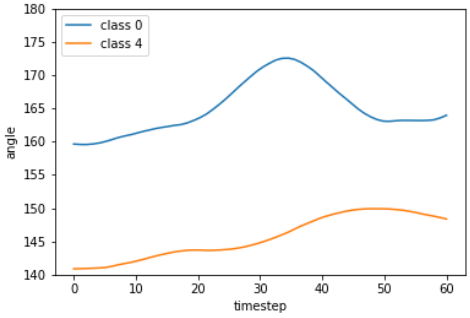
In Abbildung 4.8 ist der Vergleich der Klasse 0 mit der Klasse 4 dargestellt. Zwischen diesen Klassen ist ein deutlicher Unterschied über den gesamten Verlauf der Ausführung erkennbar.



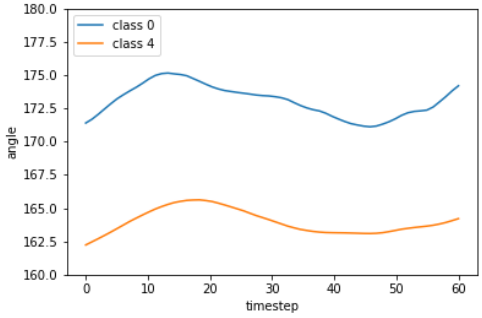
(a) Zeigt den Winkel des Kopfes während einer Ausführung



(b) Zeigt den Winkel des rechten Armes während einer Ausführung



(c) Zeigt den Winkel der rechten Seite der Hüfte während einer Ausführung



(d) Zeigt den Winkel des rechten Knies während einer Ausführung

Abbildung 4.8: Vergleich von vier Winkeldaten aus der seitlichen Perspektive der Klasse 4 mit der Klasse 0. Bei der Klasse 4 handelt es sich um die Klasse, in der die trainierenden Person die Plank Position nicht halten kann. Bei der Klasse 0 handelt es sich um eine korrekte Ausführung.

4.4 Zusammenfassung und Einschätzung

Zusammenfassend ist zu sagen, dass eine visuell basierte Qualitätserkennung möglich ist. Dies zeigt das Modell, welches die Winkeldaten für die Klassifizierung verwendet (Abbildung 4.5b). Hierfür eignet sich Pose Estimation hervorragend, um ausschließlich die relevanten Informationen aus den Videodaten zu erhalten. Bei der Verwendung der Koordinaten der Keypoints sind eine Reihe von Vorverarbeitungen notwendig, die bei der Verwendung der Winkeldaten nicht erforderlich sind.

Bei der binären Klassifizierung anhand der Winkeldaten konnte eine Testgenauigkeit von 91,5% festgestellt werden, während das Modell, welches die Koordinaten der Keypoints verwendete, lediglich eine Testgenauigkeit von 63% aufweisen konnte.Auch bei der Mehrklassen-Klassifizierung zeigte sich dieser Umstand. Aufgrund des ungleich verteilten Datensatzes wurden Klassen-Gewichte verwendet. Die Auswirkung der Klassen-Gewichte machten sich primär bei dem Modell bemerkbar, welches die Koordinaten der Keypoints verwendete. Hier stieg die Testgenauigkeit von 24,46% auf 46,42% an. Allerdings lag die Testgenauigkeit des Modells, welches mit den Winkeldaten arbeitete, mit 60,71% immer noch höher.

5 Fazit

Die Experimente dieser Arbeit haben gezeigt, dass eine Erkennung der Ausführungsqualität von Liegestützen auf Basis von Videodaten möglich ist. Zudem haben sich Human Pose Estimation und LSTMs als geeignete Techniken erwiesen, um dies zu realisieren. Bei der binären Klassifikation ist deutlich geworden, dass sich Winkeldaten für das Klassifizieren der Qualität besser eignen, als die Verwendung der Koordinaten der Keypoints. Zudem ist es bei den Winkeldaten nicht nötig eine Normalisierung der Position im Bild, sowie der Körperproportionen durchzuführen. Die Mehrklassen-Klassifizierung hat das Problem der Ungleichverteilung des Datensatzes deutlich gemacht. Dem konnte mit der Verwendung von Klassen-Gewichten entgegengewirkt werden.
Des Weiteren hat sich gezeigt, dass das Modell Schwierigkeiten hat zwischen bestimmten Klassen zu unterscheiden. Dies ist zum einen auf die teilweise vorhandene Überschneidung der Kriterien der Klassen zurückzuführen. Zum anderen liegt dies daran, dass sich einige Klassen nur leicht von einer korrekt ausgeführten Liegestütze unterscheiden.
Der Vorteil des kamerabasierten Ansatzes gegenüber den sogenannten Wearables oder auch Smart-Textiles liegt zum einen darin, dass es möglich ist mehrere Faktoren mit Hilfe von weniger Komponenten zu erfassen. In dieser Arbeit wurden zwei Kameras verwendet, wobei ein Großteil der Kriterien auch einzig und allein von der seitlichen Kameraperspektive erfasst werden könnten. Die Anzahl der Kameras hängt hierbei von dem zu untersuchenden Verhalten ab. Auch Aspekte wie die Bewegungsfreiheit und die Stromversorgung sind im sensorgestützten Ansatz von großer Bedeutung. Im Gegensatz dazu ist die Sichtbarkeit der zu analysierenden Person bei der Verwendung von Kameras zu gewährleisten. Somit spielen hier Faktoren wie die Beleuchtung und auch die Kameraperspektive eine große Rolle. Außerdem besteht hier die Gefahr, dass Körperteile verdeckt werden, die für die Analyse essenziell sind.

6 Ausblick

Die Human Pose Estimation hat sich in dieser Arbeit als eine Möglichkeit herausgestellt, die Pose einer trainierenden Person auf eine minimale Anzahl an Daten zu reduzieren. Die Human Pose Estimation lässt sich darüber hinaus in einer Vielzahl von Anwendungsgebieten einsetzen. Von der Sturzerkennung von Personen über den Bereich der Mensch-Computer-Interaktion bis hin zur Verhaltensanalyse von Personen. Hierbei ließe sich beispielsweise die Arbeitshaltung von Personen analysieren, womit der sinnvollste Zeitpunkt für eine Pause abgepasst oder möglicherweise auch Haltungsschäden vermieden werden könnten. Aufgrund des technologischen Fortschritts, lassen sich Pose Estimation Modelle, wie beispielsweise Mediapipe-Pose, bereits in Echtzeit auf mobilen Endgeräten verwenden. Dies würde mobile kamerabasierte Activity Recognition-Anwendungen möglich machen.
Das Thema Datenschutz stellt besonders in Systemen, die Kameras verwenden, einen wichtigen Aspekt dar. Im Bereich der Sicherheit wird Videoüberwachung häufig als Aufklärungs- und Präventionsmittel eingesetzt. In diesem Fall ist das Kamerabild essenziell. Systeme, die sich hingegen mit der Verhaltensanalyse beschäftigen, sind nicht auf das Kamerabild angewiesen, sondern auf die Informationen, die das Kamerabild über Personen beinhaltet. Eine Beispielanwendung stellt die Verhaltensanalyse von Passanten im Bereich des autonomen Fahrens dar. Mit Hilfe von Human Pose Estimation könnte das Verhalten der Passanten klassifiziert und vorhergesagt werden, um Unfälle zu verhindern. Um den Datenschutz sicherzustellen und die Privatsphäre der Passanten nicht zu verletzen, wäre es nötig, dass die Kameraaufzeichnung nicht zwischengespeichert werden und die Pose Estimation innerhalb der Kamera stattfindet. Jedoch ist die kamerabasierte Activity Recognition gleichermaßen in bedenklichen Anwendungsgebieten einsetzbar, wie in der Kontrolle von Arbeitsgängen.

Außerdem werden in automatisierten Verfahren ausschließlich die Faktoren analysiert, die auch gemessen werden können. Sobald ein Verhalten einmal in Zahlen dargestellt wurde, geht es ausschließlich um die Optimierung dieser Zahlen, wobei es sein könnte, dass das Wohlbefinden einer Person dabei vernachlässigt wird [12].

Literaturverzeichnis

[1] *API reference: hstack (NumPy).* https://numpy.org/doc/stable/reference/generated/numpy.hstack.html. – Accessed: 2022-01-14

[2] *API reference: Keras.* https://keras.io/api/applications/. – Accessed: 2022-01-14

[3] *API reference: NumPy.* https://numpy.org/doc/stable/reference/. – Accessed: 2021-11-09

[4] *API reference: OpenCV.* https://docs.opencv.org/4.x/index.html. – Accessed: 2021-10-14

[5] *API reference: Savitzky-Golay filter (SciPy).* https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.savgol_filter.html. – Accessed: 2021-12-17

[6] *API reference: Scikit-learn .* https://scikit-learn.org/stable/modules/classes.html. – Accessed: 2021-12-18

[7] *API reference: SciPy .* https://docs.scipy.org/doc/scipy/reference/. – Accessed: 2021-12-23

[8] *API reference: TensorFlow .* https://www.tensorflow.org/api_docs. – Accessed: 2022-01-9

[9] *MediaPipe.* https://google.github.io/mediapipe/. – Accessed: 2021-12-14

[10] *MediaPipe Pose.* https://google.github.io/mediapipe/solutions/pose. – Accessed: 2021-12-14

[11] *MoGaSens.* https://csti.haw-hamburg.de/project/mogasens/. – Accessed: 2021-06-15

[12] *Nora Berg: Ethical Reflections on Quantified Self Devices and their Effects on Humans.* https://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/master/berg.pdf. – Accessed: 2022-02-16

[13] *OpenPose-Git.* https://github.com/CMU-Perceptual-Computing-Lab/openpose#citation. – Accessed: 2021-11-10

[14] Bulling, Andreas ; Blanke, Ulf ; Schiele, Bernt: A Tutorial on Human Activity Recognition Using Body-Worn Inertial Sensors. 46 (2014), jan, Nr. 3. – URL https://doi.org/10.1145/2499621. – ISSN 0360-0300

[15] Cao, Zhe ; Hidalgo, Gines ; Simon, Tomas ; Wei, Shih-En ; Sheikh, Yaser: *OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields.* 2018. – URL https://arxiv.org/abs/1812.08008

[16] Chen, Steven ; Yang, Richard R.: Pose Trainer: Correcting Exercise Posture using Pose Estimation. In: *CoRR* abs/2006.11718 (2020). – URL https://arxiv.org/abs/2006.11718

[17] Esling, Philippe ; Agon, Carlos: Time-Series Data Mining. In: *ACM Comput. Surv.* 45 (2012), dec, Nr. 1. – URL https://doi.org/10.1145/2379776.2379788. – ISSN 0360-0300

[18] Groos, Daniel ; Ramampiaro, Heri ; Ihlen, Espen A.: EfficientPose: Scalable single-person pose estimation. In: *Applied Intelligence* 51 (2021), Nr. 4, S. 2518–2533

[19] Hochreiter, Sepp ; Schmidhuber, Jürgen: Long Short-Term Memory. In: *Neural Computation* 9 (1997), Nr. 8, S. 1735–1780

[20] Karim, Fazle ; Majumdar, Somshubra ; Darabi, Houshang ; Chen, Shun: LSTM Fully Convolutional Networks for Time Series Classification. In: *IEEE Access* 6 (2018), S. 1662–1669

[21] Kingma, Diederik P. ; Ba, Jimmy: *Adam: A Method for Stochastic Optimization.* 2014. – URL https://arxiv.org/abs/1412.6980

[22] Lee, Jaehyun ; Joo, Hyosung ; Lee, Junglyeon ; Chee, Youngjoon: Automatic Classification of Squat Posture Using Inertial Sensors: Deep Learning Approach. In: *Sensors* 20 (2020), Nr. 2. – URL https://www.mdpi.com/1424-8220/20/2/361. – ISSN 1424-8220

[23] Luiten, Jonathon ; Torr, Philip ; Leibe, Bastian: Video Instance Segmentation 2019: A Winning Approach for Combined Detection, Segmentation, Classification and Tracking. In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 2019, S. 709–712

[24] Mannor, Shie ; Peleg, Dori ; Rubinstein, Reuven: The Cross Entropy Method for Classification. In: *Proceedings of the 22nd International Conference on Machine Learning*. New York, NY, USA : Association for Computing Machinery, 2005 (ICML '05), S. 561–568. – URL https://doi.org/10.1145/1102351.1102422. – ISBN 1595931805

[25] Meissner, Stefan: *Selbstoptimierung durch Quantified Self?* S. 217–236. In: Selke, Stefan (Hrsg.): *Lifelogging: Digitale Selbstvermessung und Lebensprotokollierung zwischen disruptiver Technologie und kulturellem Wandel*. Wiesbaden : Springer Fachmedien Wiesbaden, 2016. – URL https://doi.org/10.1007/978-3-658-10416-0_10. – ISBN 978-3-658-10416-0

[26] Munea, Tewodros L. ; Jembre, Yalew Z. ; Weldegebriel, Halefom T. ; Chen, Longbiao ; Huang, Chenxi ; Yang, Chenhui: The Progress of Human Pose Estimation: A Survey and Taxonomy of Models Applied in 2D Human Pose Estimation. In: *IEEE Access* 8 (2020), S. 133330–133348

[27] Munea, Tewodros L. ; Jembre, Yalew Z. ; Weldegebriel, Halefom T. ; Chen, Longbiao ; Huang, Chenxi ; Yang, Chenhui: The Progress of Human Pose Estimation: A Survey and Taxonomy of Models Applied in 2D Human Pose Estimation. In: *IEEE Access* 8 (2020), S. 133330–133348

[28] Ofli, Ferda ; Chaudhry, Rizwan ; Kurillo, Gregorij ; Vidal, René ; Bajcsy, Ruzena: Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition. In: *Journal of Visual Communication and Image Representation* 25 (2014), Nr. 1, S. 24–38. – URL https://www.sciencedirect.com/science/article/pii/S1047320313000680. – Visual Understanding and Applications with RGB-D Cameras. – ISSN 1047-3203

[29] Ofli, Ferda ; Kurillo, Gregorij ; Obdržálek, Štěpán ; Bajcsy, Ruzena ; Jimison, Holly B. ; Pavel, Misha: Design and Evaluation of an Interactive Exercise Coaching System for Older Adults: Lessons Learned. In: *IEEE Journal of Biomedical and Health Informatics* 20 (2016), Nr. 1, S. 201–212

[30] PAILLARD-BORG, STEPHANIE ; WANG, HUI-XIN ; WINBLAD, BENGT ; FRATIGLIONI, LAURA: Pattern of participation in leisure activities among older people in relation to their health conditions and contextual factors: a survey in a Swedish urban area. In: *Ageing and Society* 29 (2009), Nr. 5, S. 803–821

[31] SAKOE, H. ; CHIBA, S.: Dynamic programming algorithm optimization for spoken word recognition. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26 (1978), Nr. 1, S. 43–49

[32] SCHAFER, Ronald W.: What Is a Savitzky-Golay Filter? [Lecture Notes]. In: *IEEE Signal Processing Magazine* 28 (2011), Nr. 4, S. 111–117

[33] SUTO, Jozsef ; ONIGA, Stefan ; LUNG, Claudiu ; ORHA, Ioan: Comparison of offline and real-time human activity recognition results using machine learning techniques. In: *Neural Computing and Applications* 32 (2020), Oct, Nr. 20, S. 15673–15686. – URL https://doi.org/10.1007/s00521-018-3437-x. – ISSN 1433-3058

[34] VOULODIMOS, Athanasios ; DOULAMIS, Nikolaos ; DOULAMIS, Anastasios ; PROTOPAPADAKIS, Eftychios ; ANDINA, Diego: Deep Learning for Computer Vision: A Brief Review. In: *Intell. Neuroscience* 2018 (2018), jan. – URL https://doi.org/10.1155/2018/7068349. – ISSN 1687-5265

[35] XIAO, Bo ; GUO, Lijun ; ZHANG, Yuanyuan ; ZHANG, Rong: Human instance segmentation from video using locally competing 1SVMs with shape prior. In: *2012 8th International Conference on Natural Computation*, 2012, S. 242–245

Erklärung zur selbstständigen Bearbeitung einer Abschlussarbeit

Gemäß der Allgemeinen Prüfungs- und Studienordnung ist zusammen mit der Abschlussarbeit eine schriftliche Erklärung abzugeben, in der der Studierende bestätigt, dass die Abschlussarbeit „— bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit [(§ 18 Abs. 1 APSO-TI-BM bzw. § 21 Abs. 1 APSO-INGI)] — ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt wurden. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich zu machen."

Quelle: § 16 Abs. 5 APSO-TI-BM bzw. § 15 Abs. 6 APSO-INGI

Erklärung zur selbstständigen Bearbeitung der Arbeit

Hiermit versichere ich,

Name: _____

Vorname: _____

dass ich die vorliegende Bachelorarbeit – bzw. bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit – mit dem Thema:

Machine-Learning-basierte Qualitätserkennung von Fitnessübungen auf Grundlage von Pose Estimation

ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

_____	_____	_____
Ort	Datum	Unterschrift im Original