Posenerkennung von Personen innerhalb von Menschenmengen

Bachelorarbeit

KIT – Karlsruher Institut für Technologie
Fraunhofer IOSB – Fraunhofer-Institut für Optronik, Systemtechnik und Bildauswertung

Thomas Dissert

19. September 2018

Verantwortlicher Betreuer: Prof. Dr.-Ing. habil. Jürgen Beyerer
Betreuender Mitarbeiter: Thomas Golda, M. Sc.

Erklärung der Selbstständigkeit

Hiermit versichere ich, dass ich die Arbeit selbständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht habe und die Satzung des Karlsruher Instituts für Technologie zur Sicherung guter wissenschaftlicher Praxis in der gültigen Fassung beachtet habe.

Karlsruhe, den 19. September 2018

(Thomas Dissert)

Kurzfassung

Das Bestimmen von Körperposen in Bildern durch Deep-Learning-Verfahren ist bereits ein weit verbreitetes und stark untersuchtes Themengebiet aus der Computergrafik. Das Ziel dieser Arbeit ist die Evaluation aktueller State-of-the-Art-Verfahren zur Bestimmung von Skelettposen auf Bilddaten hinsichtlich der Echtzeitfähigkeit und Qualität ihrer Ergebnisse. Dazu wird in erster Linie untersucht, inwieweit diese Verfahren auf Datensätzen mit großen Menschenmengen verwendet werden können. Um dieser Frage nachzugehen, wurde ein neuer Datensatz geschaffen, welcher für die quantitative Auswertung der Verfahren annotiert wurde. Zugrunde liegen dem Datensatz Bilder, welche vom Fraunhofer IOSB zur Verfügung gestellt wurden. Die Ergebnisse von zwei ausgewählten Verfahren auf dem Datensatz zeigen, dass mit zunehmender Personenzahl die Nutzbarkeit der Verfahren abnimmt, da die Rate der erfolgreichen Detektionen von Körperposen sinkt. Dennoch lassen unterschiedliche Ergebnisse der quantitativen und qualitativen Analyse darauf schließen, dass noch weitere Untersuchungen sowohl hinsichtlich der Evaluation, als auch der Verfahren erforderlich sind.

Abstract

Body pose estimation with deep learning programs on image-data is a common and heavily studied subtask in computer vision. The objective of this work is to evaluate the current state-of-the-art person pose estimators regarding their real-time capabilities and the quality of their results. Primarily the usability of these algorithms will be examined with respect to datasets containing images with large numbers of people. Therefore a novel dataset was introduced, which was annotated for the subsequent quantitative evaluations. The dataset consists of images made available by the Fraunhofer IOSB. The outcome of the evaluation of two selected person pose estimators concludes that the usability decreases with an increasing number of persons contained in an image, yet the different results of the quantitative and qualitative analysis show that more research has to be done on this topic.

Inhaltsverzeichnis

Einleitung

Posenerkennung von Personen auf Bild- und Videodaten ist schon lange ein wichtiges Forschungsgebiet, welches vor allem für die Bereiche *Robotik*, *Erweiterte Realität*, *Aktivitätserkennung* und *Spielekonsolen* aus der Informatik hochinteressant ist. Breite Aufmerksamkeit bekam die Thematik durch Systeme wie *Microsoft Kinect* oder Spiele wie *Just Dance*.

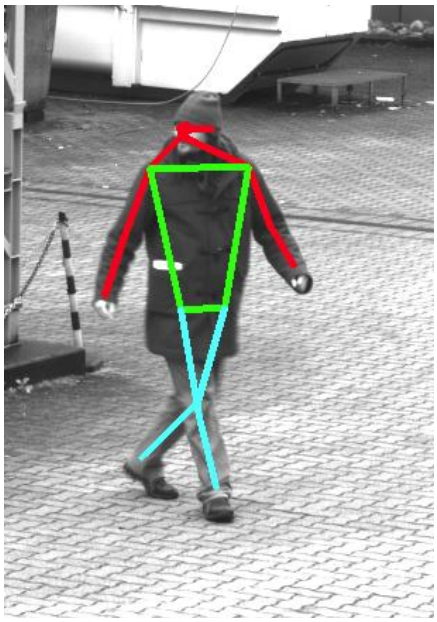


Abbildung 0.1: Eine Darstellung einer Körperpose aus dem Datensatz CrowdPose. [15]

Zur automatisierten Erkennung von Skeletten werden relevante Keypoints aus Rohbilddaten extrahiert, um in weiteren Schritten daraus logische Körperposen abzuleiten. Zur Schätzung dieser Keypoints und Körperposen werden in der Regel Deep-Learning Verfahren verwendet, welche durch ihre Abstraktionsfähigkeit hervorragende Ergebnisse erzielen können. Für gewöhnlich sind die aktuellen Verfahren dabei nur für geringe Personenmengen, oft sogar nur für einzelne Personen pro Bild ausgelegt. Hervorzuheben sind die beiden Frameworks *Alphapose* [10] und *Openpose* [3], welche für die Untersuchungen in dieser Arbeit verwendet werden. Diese Verfahren zeigen bereits sehr gute Resultate auf den öffentliche zugänglichen

Benchmark-Datensätzen COCO [21] und MPII [1]. COCO ist ein sehr großer Datensatz zur Objekterkennung und -segmentation, sowie der Keypointschätzung, während sich MPII auf den zuletzt genannten Bereich fokussiert. Die Datensätze werden in den Abschnitten 2.10 und 2.11 noch weiter beschrieben.

Neben den genannten Themengebieten lassen sich noch viele weitere Anwendungen ausmalen, in denen Posenerkennungen sinnvoll eingesetzt werden könnten, die aber davon abhängig sind, dass diese Verfahren auch bei Szenen mit großen Menschenmengen zuverlässig funktionieren. Ein möglicher Anwendungsfall wäre die Anomalieerkennung auf Bildern, welche zur Verbesserung der Sicherheit an öffentlichen Plätzen genutzt werden könnte. Nach erfolgreicher Posenschätzung könnten in weiteren Schritten Aussagen über das Verhalten von Personen getroffen werden. Dabei könnten Auffälligkeiten in Menschenmengen erkannt und frühzeitig verhindert werden.

Das Ziel dieser Thesis ist es, eine Beurteilung der aktuellen State-of-the-Art-Verfahren zu erarbeiten, welche Auskunft über deren Nutzbarkeit bei Bildern mit großen Menschenmengen gibt. Dafür wird eine Auswahl an State-of-the-Art-Verfahren bezüglich ihrer Eignung zur Schätzung von Körperposen auf Bilddaten mit erheblich größeren Personenmengen pro Bild evaluiert. Dazu wird ein Testdatensatz zur Evaluation existierender Posenschätzungsverfahren erstellt, welcher die gegebenenfalls existierenden Schwachpunkte und Probleme dieser Verfahren aufzeigt. Ein besonderes Augenmerk wird dabei auf die Robustheit und Echtzeitfähigkeit gelegt.

Aufbau der Arbeit

Die Arbeit ist in vier Kapitel unterteilt. Das Erste beschäftigt sich mit den Grundlagen, welche zum Verständnis der Thematik notwendig sind. Dabei handelt es sich neben Erläuterungen zu Teilgebieten der neuronalen Netze auch um eine Beschreibung des Begriffs der Körperpose.

Kapitel 2 geht auf einige Verfahren ein, welche das Problem der Posenerkennung auf Bilddaten lösen. Diese Verfahren werden in ihrer Funktionsweise erklärt und nach den Modellen Top-Down und Bottom-Up kategorisiert.

Nachfolgend befindet sich in Kapitel 3 der Kern der Arbeit, die Evaluation, in der zwei ausgewählte Verfahren miteinander bezüglich ihrer Robustheit verglichen werden. Sowohl quantitative, als auch qualitative Ergebnisse werden hierbei diskutiert.

Kapitel 4 schließt die Arbeit mit einem Fazit ab und stellt weitere mögliche Untersuchungsansätze vor.

1 Grundlagen

Das folgende Kapitel beschäftigt sich mit den erforderlichen Grundlagen, um die im Rahmen dieser Arbeit untersuchten Verfahren zu verstehen. Unter anderem bietet es eine Einführung in das Thema *Deep Learning* und *neuronale Netze*. Weiterhin wird in diesem Kapitel der Begriff *Körperpose* definiert, welcher in dieser Arbeit von zentraler Bedeutung ist.

1.1 Künstliche Neuronale Netze

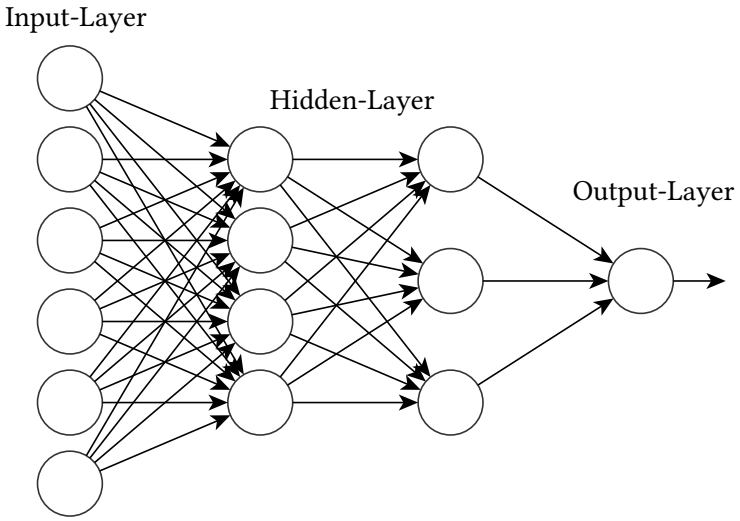


Abbildung 1.1: Eine Skizze der Struktur eines vierschichtigen neuronalen Netzes. Die linke Spalte stellt den Input-Layer dar, die mittleren beiden Spalten sind Hidden-Layer und der rechte Knoten gehört zum Output-Layer. Angelehnt an [24].

Neuronale Netze beschreiben die Struktur des Nervensystems im Gehirns und bestehen aus Neuronen. In der Informatik wird diese Struktur mithilfe *künstlicher neuronaler Netze* nachgebildet, um sowohl Rückschlüsse auf die Funktionsweise des Gehirns zu ziehen, als auch konkrete Anwendungsfälle effizient zu lösen. Im Allgemeinen werden künstliche neuronale Netze ebenfalls als neuronale Netze bezeichnet. Die darin enthaltenen künstlichen Neuro-

nen werden dementsprechend Neuronen oder auch Units, Einheiten oder Knoten genannt. Neuronale Netze werden sowohl für Regressions-, als auch für Klassifikationsprobleme verwendet. Während Klassifikationsnetze das Ergebnis auf einen begrenzten Wertebereich, meist $e \in [0,1] \subset \mathbb{R}$ beschränken, können Regressionsnetze Werte aus kontinuierlichen, unbeschränkten Wertemengen ausgeben. Diese Arbeit beschäftigt sich ausschließlich mit dem Problem der Klassifikation. Wie in Abbildung 1.1 abgebildet, handelt es sich bei einem neuronalen Netz aus mathematischer Sicht um einen Graphen $G = (V, E)$, dessen Knoten $v_i \in V$ die Neuronen darstellen. Sie sind dafür zuständig die Informationsverarbeitung und -weitergabe zu bewältigen, indem sie aufgrund ihrer Eingabeparameter einen lokalen Output berechnen, welcher wiederum als Eingabeparameter für die nächste Schicht zur Verfügung gestellt wird. Die linke Schicht der Neuronen stellt den *Input-Layer* und die rechte Schicht den *Output-Layer* dar. Im Gegensatz zum dargestellten Netz können im Output-Layer mehrere Knoten enthalten sein. Die mittleren Schichten werden auch *Hidden-Layer* genannt, da diese nach dem Black-Box-Prinzip von außen nicht einsehbar sind. Die Kanten $e_j \in E$ des neuronalen Netzes bilden die Verknüpfungen zwischen den einzelnen Neuronen ab. Die Stärke der Verbindung zwischen zwei Knoten v_i, v_j wird durch das Kantengewicht w_{ij} erfasst und stellt den Einfluss des Neurons v_i auf das Neuron v_j dar. Um zu verstehen, wie neuronale Netze arbeiten, ist es sinnvoll, die Funktionsweise eines solchen Neurons genauer zu betrachten.

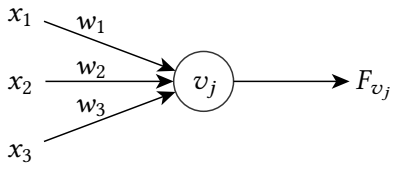


Abbildung 1.2: Schematische Darstellung eines einzelnen Neurons eines neuronalen Netzes. $x_1, x_2, x_3 \in \mathbb{R}$ stellen die Eingabeparameter dar, während F_{v_j} den Aktivierungswert angibt. Angelehnt an [24].

Ein Neuron v_j erhält, wie bereits erwähnt, Eingabeparameter $x_1,\ldots,x_n \in \mathbb{R}$. Diese stammen entweder aus der Umwelt, in diesem Fall handelt es sich um den Input-Layer, oder von anderen Neuronen. Mithilfe der Eingabeparameter, den eingehenden Kantengewichten $\{w_{ij}\}$ und einem Bias $b \in \mathbb{R}$ wird eine Berechnungsvorschrift für den Ausgabewert

$$F_{v_j} = \frac{1}{1 + \exp(-\sum_i w_{ij} x_i - b)} \tag{1.1}$$

des Neurons definiert, welche Aktivitäts- oder Aktivierungsfunktion genannt wird. Diese Funktion ist häufig eine Sigmoidfunktion, wodurch der Aktivierungswert oder Output präzise durch Justierung der Gewichte und des Bias verändert werden kann. Die Besonderheit bei neuronalen Netzen besteht darin, dass diese Gewichte und das Bias und damit die Abstraktionswege zwischen den Schichten nicht durch einen Menschen spezifiziert, sondern durch eine standardisierte Lern-Prozedur optimiert werden. Dazu werden die Gewichte und das Bias initial meistens zufällig gewählt.

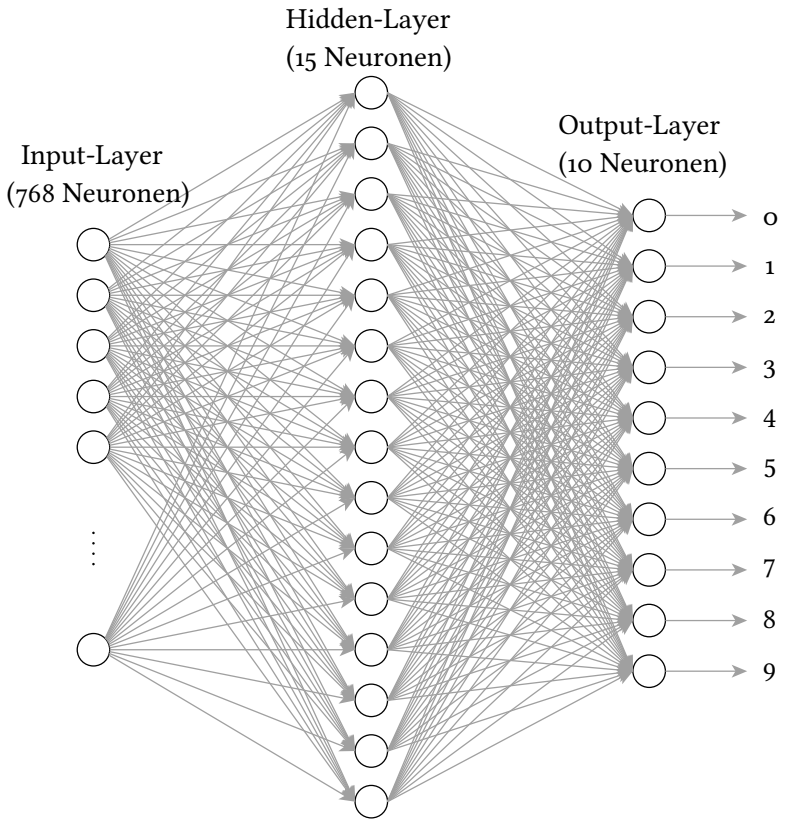


Abbildung 1.3: Eine Darstellung eines neuronalen Netzwerks zur Erkennung von handgeschriebenen Ziffern zwischen 0 und 9 in Graustufenbildern (einkanalig) mit 28×28 Pixeln. Angelehnt an [24].

In dem in Abbildung 1.3 illustrierten neuronalen Netz zur Erkennung von handgeschriebenen Ziffern werden die Bildpixel als Informationen an den Input-Layer übergeben. Dazu besitzt das Netzwerk 784 Neuronen in der ersten Schicht, welche als Output lediglich den Grauwert des zugehörigen Pixels liefern. Jedes dieser Neuronen gibt also den Wert 0.0 für weiß und 1.0

für schwarz weiter, oder einen passenden Wert für einen beliebigen Grauwert dazwischen. Die zweite Schicht, ein Hidden-Layer, beinhaltet lediglich 15 Neuronen. Der Output-Layer besteht aus 10 Neuronen, welche von 0 bis 9 durchnummeriert wurden. Der Output jedes dieser Neuronen kann als Wahrscheinlichkeit $p \in [0,1] \subset \mathbb{R}$, dass es sich bei der zu erkennenden Ziffer um die dem Neuron zugeordnete Zahl handelt, interpretiert werden. [19][24][29]

1.1.1 Überwachtes Lernen

Überwachtes Lernen ist die am weitesten verbreitete Form des maschinellen Lernens. Dazu ist zuerst eine Datenmenge notwendig, die Trainingsdaten, welche annotiert sein müssen. Im Beispiel der handgeschriebenen Ziffern aus Abbildung 1.3 wären dies Bilder von handschriftlichen Ziffern mit den zugehörigen Ziffern. Während des Trainings wird dem neuronalen Netz ein Bild übergeben, woraufhin es einen Ausgabevektor errechnet, dessen Komponenten für die möglichen Ergebniswerte stehen. Für jede Kategorie gibt das Netz also eine Wahrscheinlichkeit zurück, dass das Bild eben jener Kategorie angehört. Das Ziel des Trainings ist es, dass die der korrekten Kategorie zugehörige Komponente des Ausgabevektors die höchste Aktivierung im korrespondierenden Ausgabeneuron besitzt, das Bild also der richtigen Klasse zugeordnet wurde. Die Differenz zwischen der richtigen Klassifizierung und dem Ausgabevektor wird dazu als Fehler aufgefasst, damit das Netz die veränderbaren Parameter (Gewichte und Bias) so anpassen kann, dass dieser Fehler minimiert wird. Dazu wird ein Optimierungsalgorithmus verwendet, worauf in Abschnitt 1.1.2 weiter eingegangen wird. Durch das Wiederholen dieser Schritte für eine große Bildmenge verbessert sich das neuronale Netz fortlaufend, wodurch es im Laufe des Trainings immer bessere Ergebnisse erzielt. Um die Leistung des Systems zu messen, gibt es nach der Trainingsphase eine Testphase, in der das System auf bisher unbekannte Daten angewendet wird. Dadurch kann es auf die Fähigkeit zur Generalisierung getestet werden. [19] [24]

1.1.2 Backpropagation

Der Backpropagation-Algorithmus ist das am häufigsten verwendete Verfahren zur Optimierung der Gewichtsfunktion eines neuronalen Netzes. Die Besonderheit des Verfahrens ist es, jeweils den Parameter mit dem größten Einfluss auf den Output zu verändern, indem der Gradient der Gewichtsfunktion berechnet wird.

Gradientenabstiegsverfahren Zur Backpropagation gehört auch das Gradientenabstiegsverfahren. Dieser Algorithmus verwendet die durch die Backpropagation erhaltenen Gradienten, um neuronale Netze zu trainieren, also die zugrundeliegende Fehlerfunktion zu minimieren.

Der Algorithmus beginnt mit einer zufälligen Gewichtskombination. Abhängig von dem errechneten Gradienten werden die Gewichte derart angepasst, sodass die Ergebnisse des neuronalen Netzes verbessert werden.

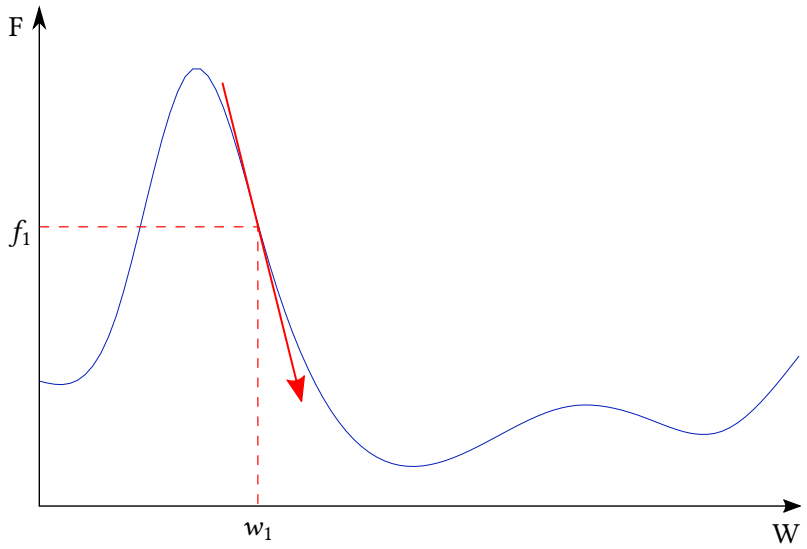


Abbildung 1.4: Darstellung des Gradienten einer Kurve am Punkt w_1 im zweidimensionalen Raum. Mithilfe des Gradientenabstiegsverfahrens wird w_1 in einem nächsten Schritt erhöht, um den Fehler f_1 zu minimieren. Im n-dimensionalen Raum, also einem Netz mit $n-1$ Gewichten, würde es sich um eine Hyperebene handeln, in welcher das tiefste Tal gesucht werden würde. Angelehnt an [29].

Das Gradientenabstiegsverfahren wird so lange ausgeführt, bis es an einem lokalen Minimum angekommen ist. Um über das lokale Minimum heraus ein globales Minimum zu erreichen, existieren diverse Verfahren, welche auf diesem Verfahren aufsetzen. Als State-of-the-Art sind hier *Stochastic Gradient Descent*, *Adam* [17] und *Adagrad* [8] zu nennen, welche große Fortschritte erzielen konnten. [24][29]

1.1.3 Überanpassung - Regularisation

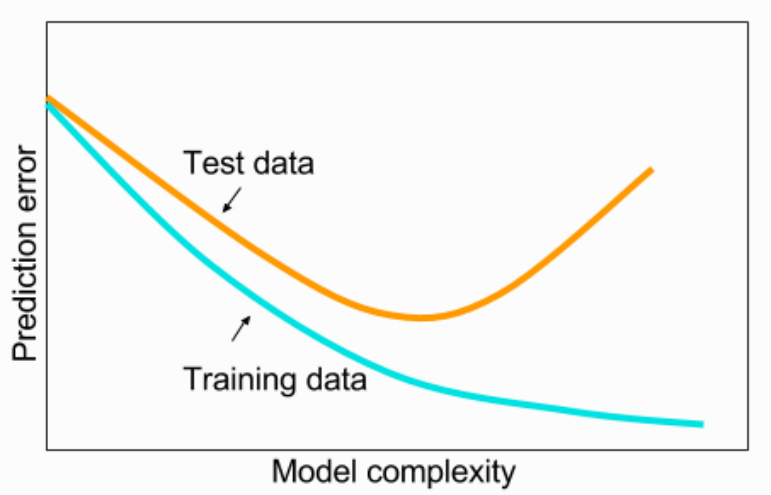


Abbildung 1.5: Die Grafik stellt das Problem der Überanpassung dar. Während die Ergebnisse für die Trainingsdaten immer genauer werden, nimmt die Genauigkeit der Aussage auf den Testdaten zuerst zu, fällt nach einem unbestimmten Zeitpunkt jedoch wieder ab. Die Überanpassung hat eine schlechte Fähigkeit zur Generalisierung zur Folge, weshalb sie im Allgemeinen vermieden werden soll. [22]

Überanpassung (engl. overfitting) beschreibt das Problem, dass ein neuronales Netz zu sehr auf die Trainingsdaten angepasst ist, wodurch die Fähigkeit des Netzwerks zur Generalisierung leidet. Beim Training mit den Trainingsdaten reduzieren sich die Kosten für eine erfolgreiche Berechnung immer weiter, was irrtümlicherweise auf eine Verbesserung des Netzes schließen lassen könnte. Schaut man sich jedoch die Genauigkeit der Ergebnisse an, sieht man, dass neuronale Netze ab einem bestimmten Punkt die Trainingsdaten auswendig lernt. Die Fähigkeit zur Generalisierung wird folglich nicht mehr verbessert. Unter *Regularisation* versteht man die Methoden zur Behebung dieses Problems. Die häufigsten Regularisations-Mechanismen sind *L2-* und *L1-Regularisation*, sowie *Dropout*. L2- und L1-Regularisation fügen der Kostenfunktion dafür einen neuen Term hinzu, welcher abhängig von allen Gewichten w ist (siehe L2-Regularisation in Gleichung 1.2, wobei C_0 die ursprüngliche Kostenfunktion darstellt und r ein Regularisationsparameter ist).

$$C = C_0 + r * \sum_w w^2 \tag{1.2}$$

Dadurch wird erreicht, dass während des Trainings möglichst niedrige Gewichte gewählt werden. Dropout hingegen hilft gegen Overfitting, indem vor dem Training das Netz selbst derart modifiziert wird, dass zufällig eine Hälfte der Hidden-Neuronen entfernt wird. Auf dem modifizierten Netzwerk wird dann trainiert, bevor die Neuronen wieder hergestellt werden und ein neues zufälliges Subnetz erstellt wird. Dies wird fortlaufend wiederholt, woraus ein guter Mittelwert für die Gewichte gefunden wird. [24][11]

1.1.4 Besondere Formen Neuronaler Netze

Die bisher beschriebenen Netze heißen *Feedforward Neural Networks*, denn die Aktivierungswerte einer Schicht l werden lediglich als Eingabe für die darauffolgende Schicht $l+1$ verwendet. Insbesondere bedeutet das, dass es in diesen Netzen keine Schleifen gibt.

Recurrent Neural Networks

Eine weitere Art der neuronalen Netze sind die *Recurrent Neural Networks*, auch *Feedback-Netze* genannt, in welchen Rückkopplungen erlaubt sind. Hervorzuheben sind dabei vier Arten von Verbindungen:

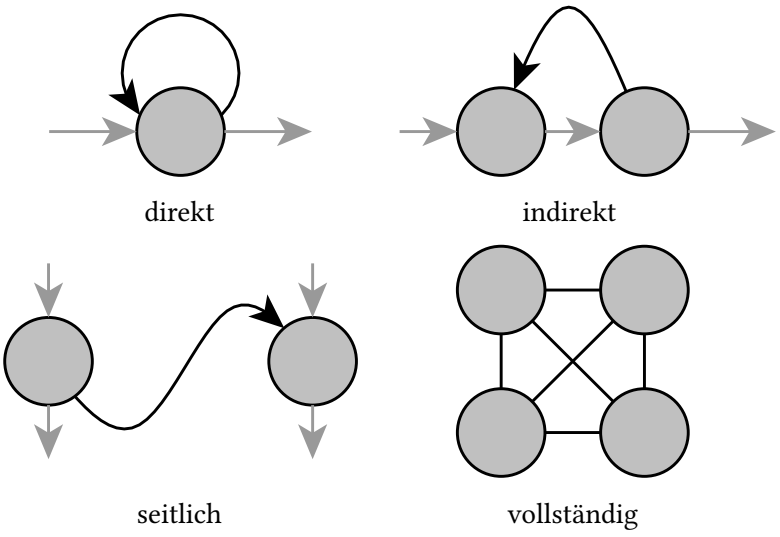


Abbildung 1.6: Darstellung der vier möglichen Arten von Verbindungen in rekurrenten Netzen. Es gibt *direkte*, *indirekte*, *seitliche* und *vollsändige* Verbindungen. Angelehnt an [29].

- Bei der *direkten* Rückkopplung wird der Output eines Neurons als Input des selben Neurons wiederverwendet.
- Besonders bei der *indirekten* Rückkopplung ist, dass der Aktivierungswert eines Neurons an die vorangegangene Schicht zurückgegeben wird.
- *Seitliche* Rückkopplungen sind solche, bei denen der Output eines Neurons an Neuronen derselben Schicht übergeben wird.
- Zuletzt gibt es noch Netze mit *vollständigen* Verbindungen, sie besitzen Verbindungen zwischen allen Neuronen.

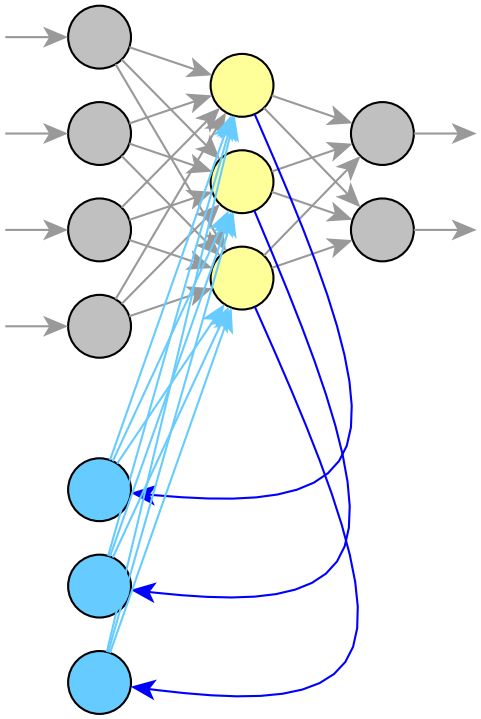


Abbildung 1.7: Ein Simple Recurrent Network (SRN) mit drei Hidden-Layer-Neuronen (gelb) und drei Kontext-Neuronen (blau). Angelehnt an [29].

Sogenannte *Simple Recurrent Networks (SRNs)*, auch *Elman-Netze* genannt, sind Netze mit indirekten Rückkopplungen, wodurch Informationen aus dem vorangegangenen zeitlichen Verlauf berücksichtigt werden können. Dazu werden so viele Kontext-Neuronen eingeführt, wie es Neuronen in Hidden-Layers gibt. Jedes dieser Kontext-Neuronen bekommt von genau

einem Hidden-Unit seinen Input und gibt seinen Output im nächsten Schritt an alle Hidden-Units zurück. Die Gewichte sind dabei, wie alle anderen Gewichte des Netzwerks, durch ein Lernverfahren anpassbar. [24][29]

Convolutional Neural Networks

Eine besondere Art von neuronalen Netzen sind die *Convolutional Neural Networks*, welche vor allem in der Bild- und Audioverarbeitung verwendet werden. Sie zeichnen sich dadurch aus, dass sie effizient auf Daten arbeiten, welche in einer Rasterstruktur vorliegen. Convolutional Neural Networks existieren sowohl als Feedforward Neural Networks, wie auch in Form von Recurrent Neural Networks. Convolutional Neural Networks machen sich wie alle tiefen neuronalen Netze die Hierarchie natürlich vorkommender Strukturen zunutze. In Bildern zum Beispiel können Kombinationen von Kanten und Kurven zu höherwertigen Mustern abstrahiert werden. Im Gegensatz zu klassischen Feedforward Netzen, welche den gesamten Input an die nächste Schicht weitergeben, simulieren CNNs die *rezeptive Felder* des biologischen Vorbilds, welche eine lokale Nachbarschaftsbeziehung darstellen. Hierdurch wird sichergestellt, dass nur lokal miteinander verknüpfte Werte eine Auswirkung aufeinander haben.

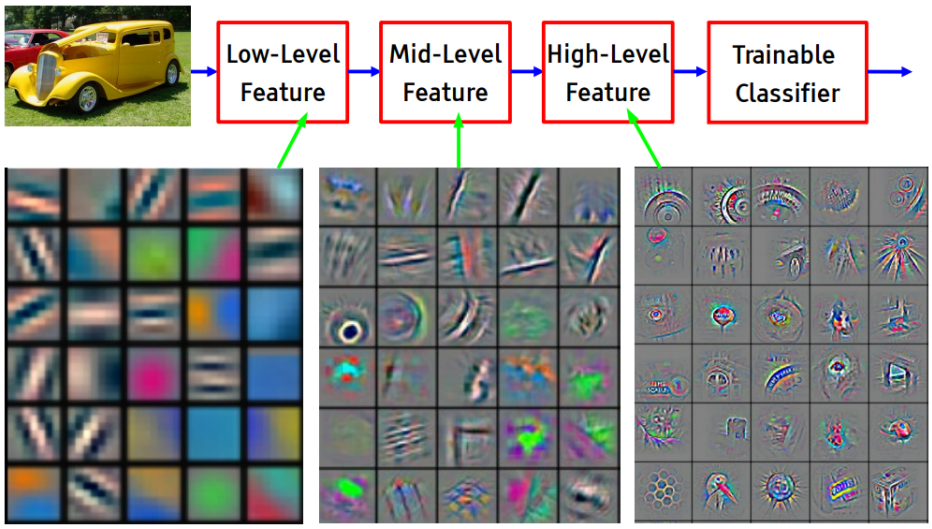


Abbildung 1.8: Visualisierung einiger Filter eines Convolutional Neural Networks. In den ersten Schichten befinden sich einfache Filter zur Erkennung von z.B. Ecken und Kanten, welche im Bild links zu sehen sind. In nachfolgenden Schichten nimmt die Komplexität der erlernten Filter zu. Derartige Filter sind im mittleren und rechten Bild zu sehen. [18]

Der Name der Netze gibt einen Hinweis auf die zugrunde liegende *(diskrete) Faltung* (engl. convolution), welche hier im Gegensatz zur Matrizenmultiplikation der herkömmlichen neuronalen Netze zum Einsatz kommt. Die Faltung ist eine mathematische Operation, bei der ein für gewöhnlich $w_k \times h_k$ großer *Kernel* k, auch *Filter* genannt, über ein Eingabearray I - z.B. ein Bild - wandert. Die mathematische Formel der Faltung lässt sich wie folgt ausdrücken:

$$I^*(x,y) = \sum_{i=1}^{w_k} \sum_{j=1}^{h_k} I(x-i+a,\, y-j+a) \cdot k(i,j) \tag{1.3}$$

Hierbei ist $I^*(x,y)$ ein Ergebnispixel des Bildes. a ist die Koordinate des Mittelpunktes und $k(i,j)$ ein Element des Filters. Durch die Multiplikation des Filters auf die Pixel in der darunterliegenden Region ergibt sich ein Wert, welcher bei einer Übereinstimmung der Pixel hoch ausfällt, andernfalls niedrig. Dieser Wert zeigt an, ob das Feature, welches der Filter beschreibt, erkannt wurde und geht in die resultierende *Feature-Map* ein, siehe Abbildung 1.9.

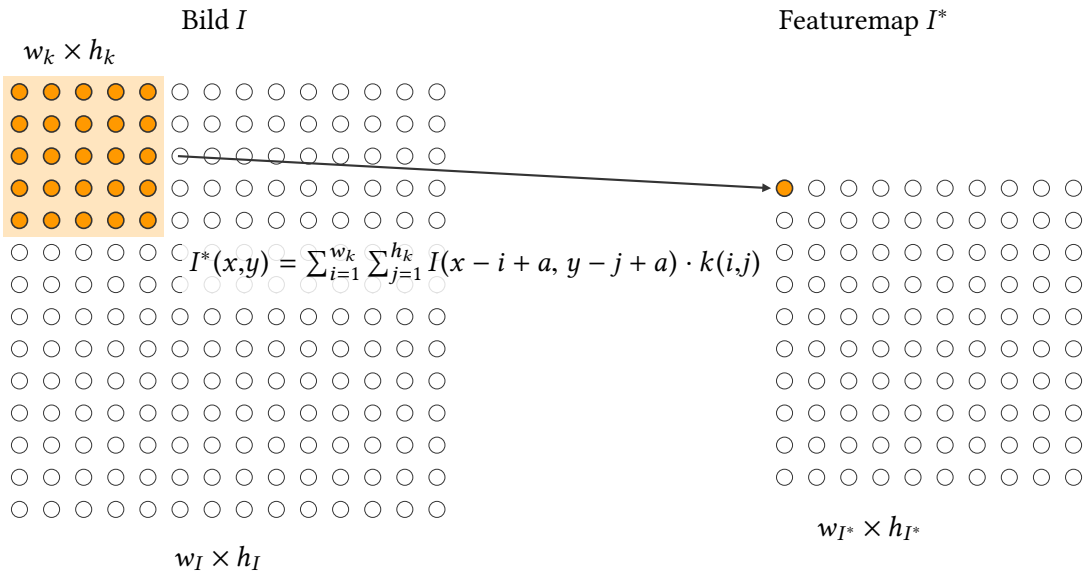


Abbildung 1.9: Schematische Darstellung der diskreten Faltung. Ein quadratischer Filterkern k der Größe $w_k \times h_k$ (hier mit $w_k = h_k = 5$) wird mit dem Eingangsbild I der Größe $w_I \times h_I$ gefaltet und ergibt die Featuremap I^*. Für die Größe von I^* resultiert $w_{I^*} = w_I - w_k + 1$ und $h_{I^*} = h_I - h_k + 1$. In Anlehnung an [24].

Aus dem Einsatz des Filters ergibt sich also eine erste Feature-Map, welche nur noch 28×28 Neuronen beinhaltet. Die Anwendung mehrerer Filter auf den selben Input ergibt einen *Convolutional-Layer*. Somit kann ein Netz mit l Filtern in einem Convolutional-Layer auch l verschiedene Arten von Features auf dieser Stufe erkennen.
Auf einen Convolutional-Layer folgt im nächsten Schritt der sogenannte *Pooling-Layer*, durch den die wichtigen Informationen der Feature-Maps extrahiert werden, zugleich aber eine Reduktion der Dimensionen durchgeführt wird. Eine sehr häufig gewählte Form des Pooling ist das in Abbildung 1.10 einsehbare *Max-Pooling*, bei dem der höchste Wert von 2×2 Neuronen in den Pooling-Layer übertragen wird. Dadurch werden die Dimensionen auf ein Viertel der Ursprungsgröße reduziert, wobei jedoch die markanten Merkmale erhalten bleiben.

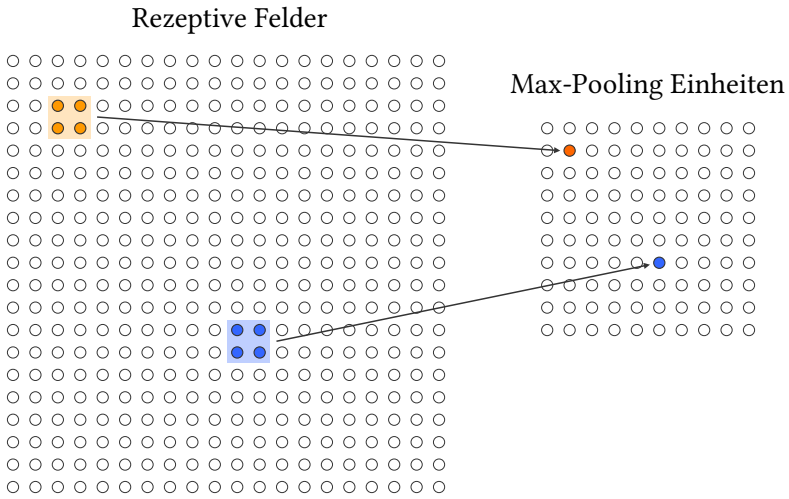


Abbildung 1.10: Das linke Raster stellt die rezeptiven Felder dar, woraus mittels Pooling das rechte Raster erzeugt wird. Rezeptive Felder (links hervorgehoben) werden auf Pooling-Einheiten (rechts hervorgehoben) abgebildet, indem in diesem Fall das jeweilige Maximum der 2×2-Matrix übernommen wird. Angelehnt an [24].

Das überwachte Lernen mit einem Optimierungsverfahren wird genau wie bei herkömmlichen neuronalen Netzen gehandhabt, indem unter anderem die verwendeten Filter variable Parameter sind. [24] [19] [2]

1.2 Deep Learning Frameworks

Machine Learning Frameworks sind eine Abstraktion der theoretischen Modelle neuronaler Netze in Form von Programmbibliotheken oder eigenständigen Anwendungen. Sie erleichtern das Arbeiten mit Machine Learning Algorithmen. Tabelle 1.1 beschreibt einige dieser Frameworks, welche den in dieser Arbeit erwähnten Posenerkennungen zugrunde liegen.

Framework	Programmiersprachen	Entwickler
Caffe [16]	Python, MATLAB, C++	Berkeley Vision and Learning Center
TensorFlow [23]	Python, Java, C, Go	Google Brain
Torch [6]	Lua, C	Ronan Collobert, Koray Kavukcuoglu, Clement Farabet
PyTorch [26]	Python	Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan

Tabelle 1.1: Einige Machine Learning Frameworks und die möglichen Programmiersprachen als Schnittstelle.

1.2.1 Keras

Keras ist ein in Python entwickeltes Framework, welches das Arbeiten mit einigen Deep Learning Frameworks (TensorFlow [23], CNTK [30] und Theano [31]) vereinfachen soll. Dazu setzt es an einer höheren Abstraktionsebene an, sodass Entwickler sich besser auf das Konzipieren eines Modells konzentrieren können und weniger Zeit in die Framework-spezifischen Details investieren müssen. Keras hat aber auch schon große Einflüsse auf TensorFlow gehabt, weshalb Keras seit TensorFlow 1.4 ein Teil der TensorFlow Core API ist. Keras hat eine sehr aktive Community, welche durch den modularen Aufbau des Frameworks viele typische Verfahren und bereits trainierte Netze implementiert hat. Weiter besitzt Keras die Möglichkeit, auf das jeweilige Backend (TensorFlow, CNTK oder Theano) zuzugreifen, um nicht in Keras vorhandene Verfahren zu verwenden. [5] [23]

1.3 Menschliche Körperpose

Für diese Arbeit ist es zunächst wichtig zu definieren, was Posen überhaupt sind. Naheliegend ist, sich darunter die Körperhaltung eines Menschen in einer Momentaufnahme vorzustellen. Aus mathematischer Sicht handelt es sich bei einer Körperpose um einen Graphen $G = (V, E)$, dessen Knoten $v \in V$, auch Keypoints genannt, die markanten Stellen eines Körpers darstellen. Die Kantenmenge $E \subset E_{max}$ ist eine echte Teilmenge des zugehörigen vollständigen Graphen $G_{max} = (V, E_{max})$. Dabei sind V und E abhängig von den verwendeten Verfahren und Datensätzen, weshalb darauf in den Abschnitten noch genauer eingegangen wird. Ein Knoten $v = (x, y, v, t)) \in V$ besitzt Koordinaten x und y sowie eine Sichtbarkeit v und einen Keypoint-Typ t, durch den er eindeutig festgelegt ist. Eine Kante $e = (v, w) \in E$ mit $v, w \in V$ existiert genau dann, wenn v und w zusammenpassende Keypoint-Typen besitzen. Ein Beispiel wären typischerweise die Keypoint-Typen *LINKER KNÖCHEL* und *RECHTES KNIE*, nicht jedoch *LINKER KNÖCHEL* und *RECHTER KNÖCHEL*.

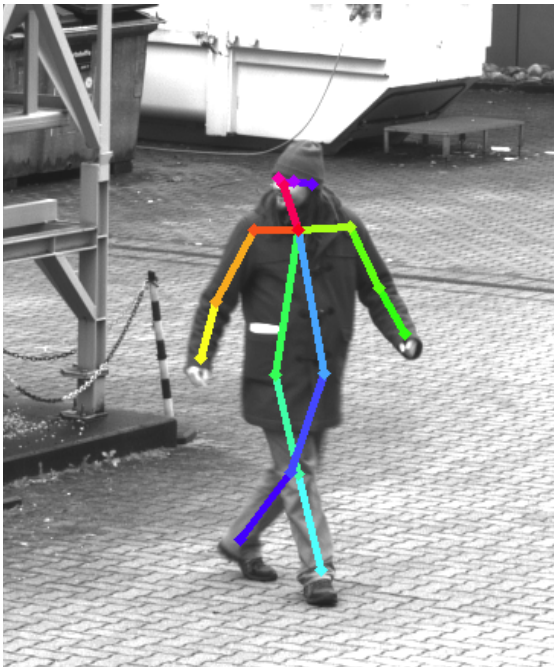


Abbildung 1.11: Darstellung einer menschlichen Körperpose, welche mithilfe des OpenPose-Frameworks aus Abschnitt 2.2 erstellt wurde. [3] [15]

2 Verwandte Arbeiten

Für das Verständnis der weiterführenden Arbeit werden hier einige State-of-the-Art-Verfahren vorgestellt und erläutert. Dabei wird vorwiegend auf die zugrundeliegende Funktionalität eingegangen und weniger auf die notwendige Mathematik. Existierende Ansätze zur Erkennung von Posen multipler Personen teilen sich in zwei Rubriken auf: *Top-Down*, bei der zuerst Bounding-Boxen geschätzt werden und daraufhin Single-Person Pose Estimation auf deren Inhalt angewandt wird und *Bottom-Up*, bei der zunächst alle Keypoints geschätzt und danach zu Personenclustern zusammengefügt werden. Über die Verfahren hinaus beinhaltet dieses Kapitel eine Einführung in die Datensätze COCO und MPII, welche neben eigenen Ansätzen zum Vergleich der Verfahren verwendet werden.

2.1 DeepCut/DeeperCut

DeepCut ist ein State-of-the-Art-Ansatz für Multi-Personen-Posenerkennung. Dazu wird zuerst eine Menge von Keypoint-Kandidaten D identifiziert, wobei jeder erkannte Kandidat $d \in D$ einen Score $p_{dc} \in [0,1]$ besitzt, der angibt, wie wahrscheinlich es sich um einen bestimmten Keypoint-Typ $c \in C$ handelt. Dieser Score geht über die Kostenfunktion $\alpha_{dc} := \log \frac{1-p_{dc}}{p_{dc}}$ in das neuronale Netz ein. Zusätzlich wird für jedes Keypoint-Paar $d,d^{'} \in D$ und jedes Paar von Keypoint-Typen $c,c^{'} \in C$ ein Score $p_{dd^{'}cc^{'}} \in (0,1)$ bestimmt, woraus die Kostenfunktion $\beta_{dd^{'}cc^{'}} := \log \frac{1-p_{dd^{'}cc^{'}}}{p_{dd^{'}cc^{'}}}$ resultiert, die eine Aussage darüber macht, ob die Keypoints zu einer gemeinsamen Person gehören. Diese beiden Kostenfunktion werden mittels ganzzahliger linearer Optimierung minimiert, woraus ein Kandidat für eine Pose resultiert. Das DeeperCut-Verfahren baut auf dem oben beschriebenen DeepCut-Verfahren auf, und bringt einige Verbesserungen mit sich. Die Autoren greifen hierfür drei Stellen auf und optimieren diese: Die Detektoren für Körperteile werden verbessert, indem auf ein ResNet-CNN [12] gesetzt wird. Außerdem wird eine Abstandsfunktion definiert, welche jedem gefundenen Keypoint den Abstand der umliegenden Keypoints zuordnet. Diese Funktion wird als neue Kostenfunktion in das neurale Netzwerk eingeführt. Zuletzt wird die lineare Optimierung verbessert, indem von den am besten erkannten Keypoints ausgehend optimiert wird. [27] [14]

2.2 Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields

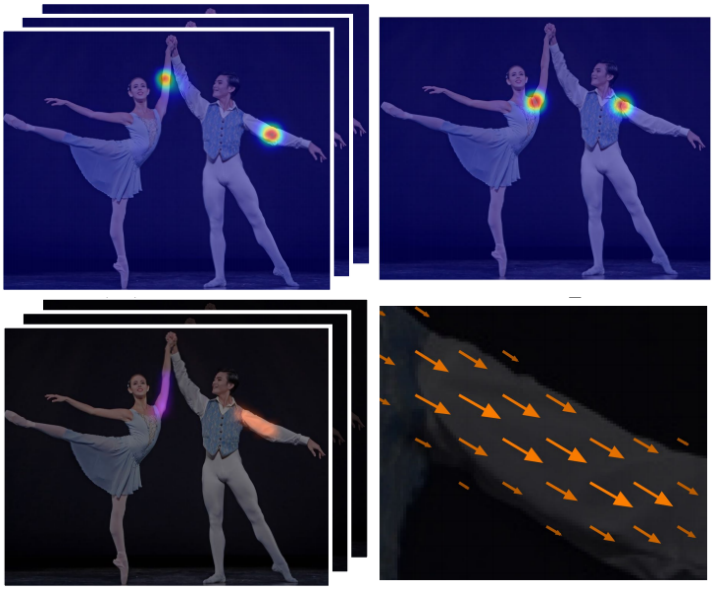


Abbildung 2.1: In der oberen Reihe sind beispielhafte Confidence Maps S_i für die linken Schultern und die linken Ellenbogen als Heatmap dargestellt, bei der die Wahrscheinlichkeit einer korrekten Detektion eines Keypoints farblich kodiert ist. In der unteren Reihe werden Part Affinity Fields zu den linken Oberarmen gezeigt, wobei im linken Bild die Richtung durch die farbliche Markierung dargestellt wird, während dafür im rechten Bild Vektoren genutzt werden. [3]

Die Besonderheit an diesem Verfahren, welches unter anderem als Framework mit dem Namen *OpenPose* umgesetzt wurde, sind die sogenannten Part Affinity Fields, welche Informationen darüber beinhalten, ob sich an einer Position ein Körperglied befindet und in welche Richtung dieses ausgeprägt ist. Zuerst wird eine Menge von Confidence Maps $S = (S_1, S_2, ..., S_J)$, welche die Positionen der J Körperteile darstellen und eine Menge mit Vektorfeldern $L = (L_1, L_2, ..., L_C)$, Part Affinity Fields genannt, welche die Beziehungen zwischen den Körperteilen beinhalten, erzeugt. Einige Confidence Maps und Part Affinity Fields sind beispielhaft in Abbildung 2.1 zu sehen. Zur simultanen Berechnung der beiden Mengen wird ein *two-branch multi-stage convolutional neuronal Network* verwendet, welches in Abbildung 2.2 dargestellt ist. Jede Phase des Netzes erzeugt eine Menge Confidence Maps S^t und eine Menge Part Affinity Fields L^t, sodass nach erfolgreicher Berechnung des CNN die Ergebnisse $S = S^T$ und $L = L^T$ sind und $\forall t : t < T$.

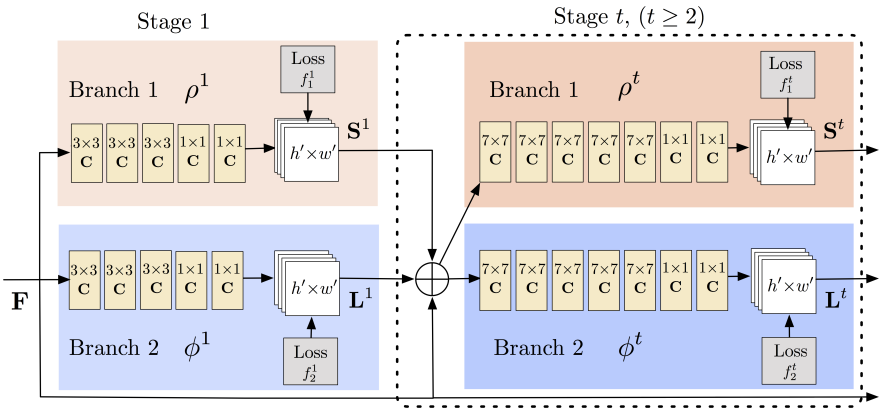


Abbildung 2.2: Mehrzweigige Struktur des Convolutional Neural Networks, welches die Confidence Maps S und die Part Affinity Fields L iterativ erzeugt. [3]

Um aus den Körperteilen und deren Beziehungen vollwertige Skelette zu gewinnen, wird mittels eines Greedy-Algorithmus ein optimal Matching generiert. Dieses optimal Matching des gewollt bipartiten Graphen ist eine Untermenge der Kanten, sodass keine zwei Kanten einen gemeinsamen Knoten haben. Daraus folgt, dass keine zwei Keypoints des selben Typs adjazent sind. Jede Zusammenhangskomponente im resultierenden Graph stellt also eine Person auf dem Bild dar. Der Greedy-Algorithmus trennt zuerst die verschiedenen Personen, indem er für jedes Knotenpaar mithilfe der Part Affinity Fields die Wahrscheinlichkeit berechnet, eine Beziehung zwischen zwei Körperteilen darzustellen. Auf den damit resultierenden Gruppierungen von Knoten wird wiederum mit den Part Affinity Fields ein bestmöglicher Teilgraph erzeugt, welcher alle Knoten zu einem Skelett verbindet. [3]

2.3 ArtTrack – Articulated Multi-person Tracking in the Wild

ArtTrack baut auf das DeeperCut-Verfahren auf. Das Alleinstellungsmerkmal dabei ist der zeitliche Verlauf der Keypoints, welcher verwendet wird, um die benötigte Rechenleistung zu reduzieren und die Robustheit zu erhöhen.

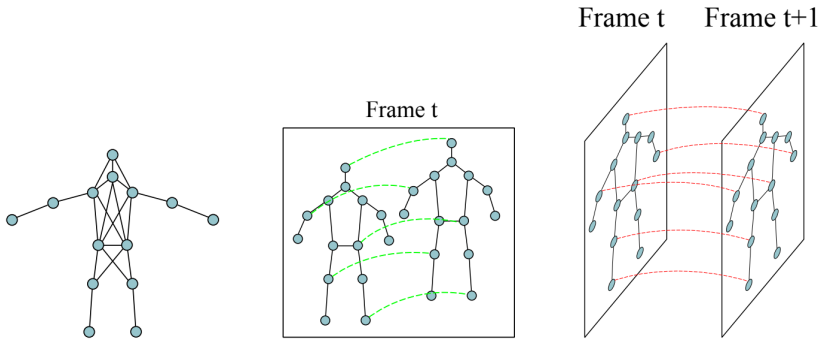


Abbildung 2.3: Verbindung der Keypoints eines Bildes mit den Keypoints des zeitlich darauf folgenden Bildes. [13]

Das Verfahren ist sowohl als Bottom-Up, wie auch in Form eines kombinierten Top-Down-/Bottom-Up-Models umgesetzt. In beiden Varianten werden Körperteil-Detektoren verwendet, um eine Menge von Keypoint-Vorschlägen $D = \{d_i\}$ zu erzeugen. Jeder Vorschlag enthält dabei die Werte $d_i = (t_i, d_i^{pos}, \pi_i, \tau_i)$, wobei t_i den Zeitpunkt im Video, d_i^{pos} die örtliche und zeitliche Position, π_i die Wahrscheinlichkeit einer korrekten Erkennung und τ_i die Art des Keypoints angibt. Im Bottom-Up-Modell wird zwischen zwei verschiedenen Möglichkeiten gewählt: Entweder werden alle Keypoints innerhalb eines Bildes verbunden oder ein reduzierter Graph, welcher Kanten zwischen voneinander unabhängigen Keypoints vernachlässigt, wird verwendet. Der Unabhängigkeitsfaktor zweier Keypoints wird dabei abhängig von den erkannten Typen τ_i und τ_j errechnet und stellt die Kostenfunktion innerhalb eines Zeitpunktes dar. Im Top-Down-/Bottom-Up-Modell werden zusätzlich zum Bottom-Up-Modell die erkannten Köpfe hervorgehoben. Die Kanten zwischen Köpfen werden ausdrücklich als „nicht zu verbinden“ gekennzeichnet. Dadurch folgt automatisch, dass alle Keypoints jeweils nur einem Kopf zugeordnet werden können, wodurch eine Trennung der Personen impliziert wird. Unabhängig vom genutzten Modell wird der Zusammenhang aufeinanderfolgender Keypoints durch eine Kostenfunktion dargestellt, welche den Euklidischen Abstand zwischen diesen Keypoints zugrunde legt. Die verschiedenen Kostenfunktionen stellen das Modell des Verfahrens dar und werden mithilfe eines CNN verwendet. [13]

2.4 Regional Multi-person Pose Estimation

Die Autoren dieses Verfahrens, welches in einem Framework namens *AlphaPose* umgesetzt wurde, haben einen problemorientierten Ansatz gewählt. Sie stellen an den bisherigen State-of-the-Art-Verfahren zwei Kritikpunkte vor: Bisherige Modelle sind stark abhängig von den erstellten Bounding-Boxen und mehrfach erkannte Personen resultieren in redundanten Skeletten, da Single Person Pose Estimator unabhängig von anderen Bounding-Boxen einzelne Skelette berechnen. Das Verfahren wird als eine Erweiterung des Top-Down-Modells vorgestellt.

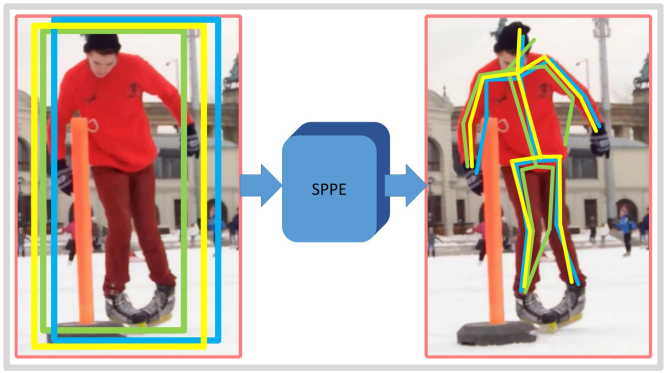


Abbildung 2.4: Eine Veranschaulichung der Probleme anderer State-of-the-Art-Verfahren: 1. Die erkannten Bounding-Boxen führen aufgrund verschiedener Größen zu unterschiedlichen Skeletten. 2. Mehrere Bounding-Boxen für eine einzige Person erkannt. [10]

Das Problem der redundanten Posen wird durch einen *Non-Maximum Supressor (NMS)* gelöst, welcher die Redundanzen eliminiert, indem er die beste Pose auswählt und ähnliche Posen iterativ löscht. Das weitere Problem der ungenauen Bounding-Boxen wird durch einen Pose-guided Proposals Generator (PGPG) gelöst. Während des Trainings werden zu jeder erkannten Bounding-Box ähnliche Boxen erstellt. Dadurch, dass man die ursprüngliche Box kennt, kann der Algorithmus die ähnlichen Bounding-Boxen dahingehend optimieren, dass unterschiedliche Bounding-Boxen zu ähnlicheren Resultaten führen. Der zuvor erwähnte NMS bietet bereits die benötigte Funktionalität, um die künstlich erzeugten redundanten Posen zu verwerfen. [10]

2.5 Pose Flow: Efficient Online Pose Tracking

Das Verfahren ist eine Verbesserung des RMPE-Verfahrens in Bezug auf das erfolgreiche Schätzen von Posen in Videos. Es handelt sich um ein Top-Down-Modell, welches zwei neuartige Aspekte liefert: *Pose Flow Building (PFB)* und *Pose Flow Non-Maximum Supression (Pose Flow NMS)*. PFB ist dafür zuständig zusammenhängende Posen über mehrere Frames hinweg zu verbinden. Der Pose Flow NMS wird im Gegensatz zu dem in ArtTrack vorgestellten NMS nicht auf einzelne Frames, sondern auf die Pose Flows angewandt. Somit erhöht sich die Stabilität des Verfahrens und es sorgt dafür, dass Keypoints, welche sonst in einzelnen Frames nicht erkannt werden könnten, erkannt werden. [34]

2.6 A Greedy Part Assignment Algorithm for Realtime Multi-person 2D Pose Estimation

Dieses Projekt legt den Schwerpunkt im Unterschied zu den State-of-the-Art-Verfahren auf den Greedy-Algorithmus zur Gruppierung der erkannten Keypoints. Die komplexe Berechnung wird vereinfacht, indem Wissen über die natürliche Körperstruktur hinzugenommen wird.

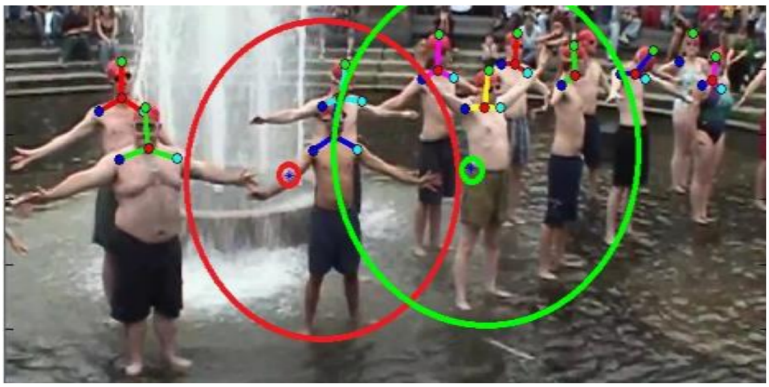


Abbildung 2.5: Eine Darstellung der wahrscheinlichsten Personencluster, welchen einzelne Keypoints zugehören können. Der rote Kreis grenzt die möglichen Cluster für den rechten Ellenbogen der dritten Person auf die zwei darin befindlichen Personen ein. Der grüne Kreis grenzt die Kandidaten für die rechte Hüfte der fünften Person ein. [33]

Um die gewünschte Verbesserung zu erzielen, werden fünf neue Ansätze verwendet. (i) Die Anzahl an Keypoints je Typ wird auf die Anzahl von Personen reduziert. (ii) Die Argumentation der Verbindungen zwischen Keypoints wird vom Kopf aus in Richtung Knöchel geführt. In je-

dem Schritt wird unabhängig von anderen Einfügeoperationen ein Keypoint nach dem anderen angehängt. (iii) Außerdem werden in jedem Schritt nur die wahrscheinlichsten Personencluster in Betracht gezogen, siehe Abbildung 2.5. (iv) Dazu wird unter anderem aus der Größe erkannter Köpfe die Körpergröße gefolgert, welche einen Hinweis auf die Wahrscheinlichkeit einer Verbindung zweier Keypoints gibt. (v) Zuletzt werden beim Zusammenfügen von Keypoints nur Teilgraphen verwendet, sodass zum Beispiel ausschließlich Kopf und Nacken als markante Keypoints für den Oberkörper verwendet werden.[33]

2.7 Towards Accurate Multi-person Pose Estimation in the Wild

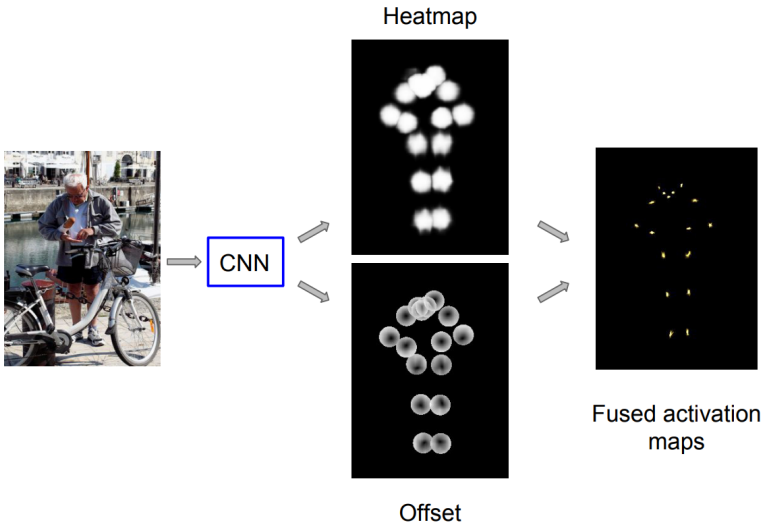


Abbildung 2.6: Dichte-Heatmap und Offsets für eine erkannte Person, sowie die daraus errechnete Activation Map.

Dieses von Google entwickelte Verfahren ist ein Top-Down-Modell. Es gliedert sich in zwei Phasen: In der ersten Phase werden Bounding-Boxen erkannt, welche wahrscheinlich Personen enthalten. Dafür wird ein *Faster RCNN Detector [28]* verwendet. Die erkannten Bounding-Boxen werden alle in das gleiche Seitenverhältnis gebracht, indem sie zuerst entweder in der Höhe oder der Breite vergrößert werden. Danach werden sie mit einem Faktor zwischen 1,0 und 1,5 vergrößert, um mehr Kontext einzubringen. Zum Schluss werden sie im Seitenverhältnis $353/257 = 1{,}37$ ausgeschnitten und auf $353 * 257$ Pixel skaliert. In der anschließenden Phase werden je 17 Keypoints in jeder Bounding-Box geschätzt, welche die enthaltene Person darstellen sollen. Für jeden Keypoint-Typ werden mittels ResNet-CNN Dichte-Heatmaps und Offsets,

wie in Abbildung 2.6 gezeigt, erzeugt. Die Resultate werden nachfolgend dazu verwendet, um Activation Maps zu generieren, welche eine genauere Position der einzelnen Keypoints darstellt. Im Gegensatz zu den anderen State-of-the-Art Top-Down-Verfahren wird hierbei NMS nicht wie gewöhnlich auf der Ebene erkannter Bounding-Boxen und deren Überlappung, sondern mithilfe der *Object Keypoint Similarity (OKS)* angewandt. Die OKS stellt Keypoints zweier erkannter Personen gegenüber und vergleicht diese miteinander. [25]

2.8 Cascaded Pyramid Network for Multi-person Pose Estimation

Dieser Ansatz verwendet ebenso ein Top-Down-Modell. Als Personen-Detektor wird der State-of-the-Art Objekt-Detektor *FPS* verwendet. Die Besonderheit des Verfahrens befindet sich jedoch in der Posenerkennung. Dazu nutzt das Verfahren ein neuartiges *Cascaded Pyramid Network (CPN)*, welches aus zwei Unternetzen besteht: GlobalNet und RefineNet. Das *GlobalNet* ist ein *Feature Pyramid Network [20]* und wird zum effektiven Auffinden jener Keypoints verwendet, welche ohne weitere Informationen „leicht“ gefunden werden können, wie unter anderem Augen. Während des Prozesses werden, wie in Abbildung 2.7 dargestellt, mehrere Ebenen mit Feature-Maps erzeugt, welche eine Repräsentation der erkannten Features darstellen. Im Gegensatz zu einem gewöhnlichen CNN ergibt sich dadurch eine stärkere Aussagekraft der erkannten Features über alle Ebenen hinweg.

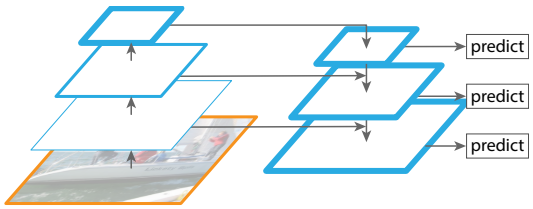


Abbildung 2.7: Ein Feature Pyramid Network, bei dem die Feature-Maps durch ihren blauen Rand eingegrenzt werden und die Stärke des Randes die Stärke der Aussagekraft des zugehörigen Features für die Objekterkennung darstellt. [20]

Die Besonderheit des Verfahrens ist das *RefineNet*, welches sich auf die Berechnung der „schweren“ Keypoints konzentriert. Unter diese Bezeichnung fallen unter anderem verdeckte Keypoints, aber auch Körperbereiche, welche nur wenige markante Stellen besitzen. Indem gleichzeitig auf alle Ebenen des dafür zuvor verwendeten GlobalNet zugegriffen wird, können verbesserte Resultate für die Objekterkennung erzeugt werden. Das in Abbildung 2.8 skizzierte Netzwerk stellt das gesamte neuronale Netz des Verfahrens dar. [20] [4]

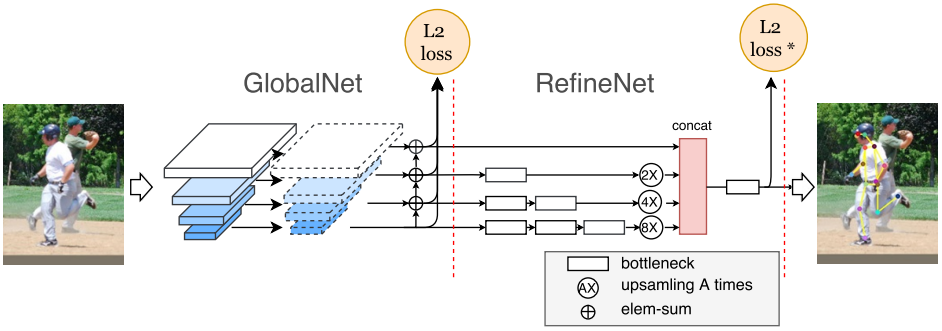


Abbildung 2.8: Struktur des Cascaded Pyramid Network mit den darin enthaltenen GlobalNet und RefineNet. [4]

2.9 Implementierungen der vorgestellten Verfahren

Nachfolgend ist eine Tabelle gegeben, welche die verfügbaren Implementierungen zu den Verfahren darstellt.

Verfahren	Original	Nachimplementierung
Deep(er)Cut	Caffe	-
Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields	Caffe	Keras/Tensorflow, PyTorch, viele weitere
ArtTrack – Articulated Multi-person Tracking in the Wild	Tensorflow	-
Regional Multi-person Pose Estimation	Tensorflow/Torch	PyTorch
Cascaded Pyramid Network	Tensorflow	PyTorch

Tabelle 2.1: Die Tabelle gibt eine Übersicht über die in diesem Kapitel vorgestellten Verfahren und existierende Implementierungen von Autoren oder Dritten.

2.10 COCO - Common Objects in Context

Der COCO-Datensatz ist eine Ansammlung von ca. 330.000 Bildern. In erster Linie handelt es sich bei COCO um einen Datensatz zur Objekt-Erkennung und Objekt-Segmentierung, wobei er sich durch etwa 1,5 Millionen enthaltene Objekte auszeichnet. Er enthält neben Menschen auch

ca. 80 weitere Objekt-Typen, sodass er breite Anwendung beim Training oder der Evaluation von verschiedensten tiefen neuronalen Netzen findet. Zugleich beinhaltet der Datensatz jedoch auch etwa 250.000 annotierte Personen inklusive ihrer Keypoints, welche für diese Arbeit essentiell sind. Die Keypoint-Annotationen zu den Bildern sind im JSON-Format gespeichert. Dazu wird eine API in MatLab, Python und Lua mitgeliefert, um auf den Daten zu operieren. Die Keypoints werden in 17 Keypoint-Typen (siehe Tabelle 3.1) unterteilt und jeweils mit $x-$ und $y-$Koordinaten sowie ihrer Sichtbarkeit v gespeichert. Die Sichtbarkeit ist dazu in drei Gruppen eingeteilt: *0 - nicht gekennzeichnet*, *1 - gekennzeichnet, aber nicht sichtbar* und *2 - gekennzeichnet und sichtbar*. Zusätzlich zu den eigentlichen Bilddaten und Annotationen zeichnet sich COCO auch durch eigene Wettbewerbe aus, in denen die verschiedenen Verfahren auf den in COCO vorhandenen Daten gemessen werden. Verfahren zur Posenschätzung können zum Beipiel an der Keypoint-Challenge teilnehmen. Die Ergebnisse dafür werden mittels einer öffentlich verfügbaren Metrik bestimmt. [21]

2.11 MPII Human Pose Dataset

Der Datensatz MPII spezialisiert sich auf die Daten zur Keypoint-Detektion. Er beinhaltet etwa 25.000 Bilder mit ca. 40.000 annotierten Personen. Die Annotationen sind in einer Matlab Struktur gespeichert, in der weitere Werte, wie z.B. Informationen zur Sichtbarkeit von Keypoints, enthalten sind. Die Keypoints ähneln denen des COCO-Datensatzes. Sie unterscheiden sich jedoch in der Gesichts- und Brustregion. MPII enthält 16 Keypoints (siehe Tabelle 3.1). Die Abbildung 2.9 visualisiert die Unterschiede der beiden vorgestellten Datensätze. [1]

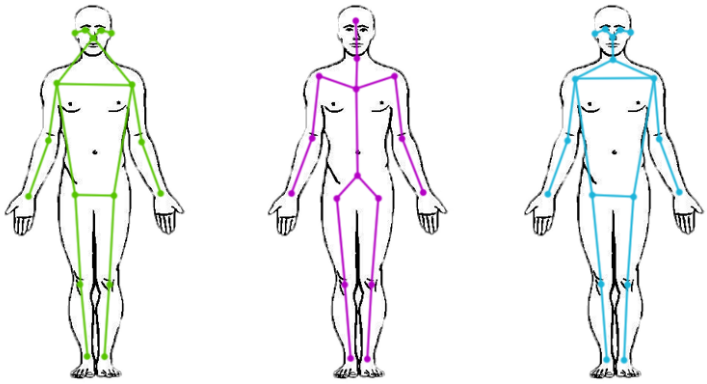


Abbildung 2.9: Eine Skizze der verwendeten Keypoint-Typen. Die grünen Keypoint-Typen sind in COCO, die lilafarbenen in MPII und die blauen in dem in Abschnitt 3.1 beschriebenen eigenen Datensatz enthalten. Körperpose aus [32].

3 Eigene Untersuchungen

In diesem Kapitel werden die für die Untersuchungen im Voraus notwendigen eigenen Arbeiten vorgestellt sowie die Ergebnisse dieser dargestellt. Darunter befindet sich ein selbsterzeugter Keypoint-Datensatz, eine Erläuterung der Anpassungen an den Verfahren und Metriken und die quantitative sowie qualitative Evaluation dieser Verfahren, vor allem hinsichtlich der Skalierbarkeit der Personenmengen auf den Bilddaten des Datensatzes. Die verwendeten Verfahren sind die bereits in Kapitel 2 vorgestellten Verfahren *Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields* [3] und *Regional Multi-person Pose Estimation* [10]. Zur Evaluation wird die an den eigenen Datensatz angepasste Keypoint-Metrik der COCO-Keypoint-Challenge [21] verwendet.

3.1 CrowdPose

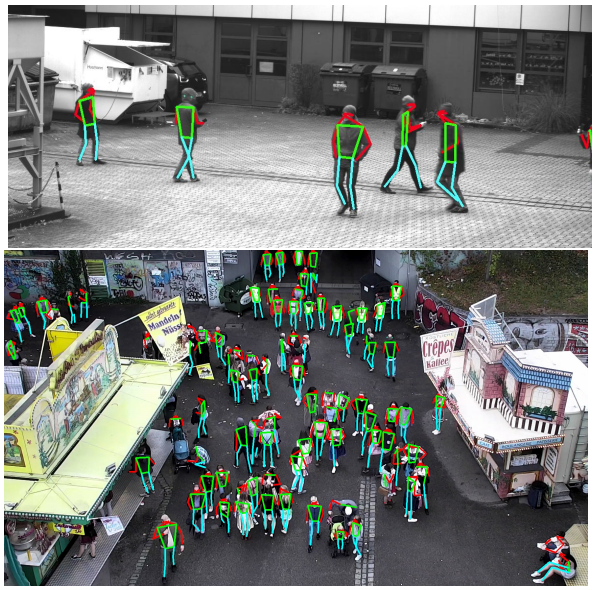


Abbildung 3.1: Darstellung der Annotationen für zwei Bilder aus dem selbst erstellten Datensatz CrowdPose. Oben sieht man eine Aufnahme am Wareneingang des IOSB und unten eine Aufnahme auf den Cannstatter Wasen. [15]

CrowdPose (Crowd-level Person Pose Estimation Dataset) ist der während dieser Arbeit entstandene Datensatz, welcher zur Evaluation von Algorithmen zur Detektion von Körperposen bezüglich der Nutzbarkeit auf Bildern mit großen Personenmengen verwendet werden kann. Annotiert wurden die Bilder mithilfe des Tools *Sloth* [7], welches vom cv:hci am KIT entwickelt wird und einer passenden vom Lehrstuhl IES am KIT entwickelten Erweiterung. Grundlage für den Datensatz ist eine vom Fraunhofer IOSB zur Verfügung gestellte Auswahl an Bildern. Der gesamte Datensatz enthält 25 teilweise partiell annotierte Bilder, von denen 19 Bilder am Wareneingang des IOSB und 6 auf dem Cannstatter Wasen aufgenommen wurden. [15]

Keypoint	COCO	MPII	CrowdPose
Kopf		✓	
Nase	✓		✓
Nacken		✓	✓
linkes Auge	✓		✓
rechtes Auge	✓		✓
linkes Ohr	✓		✓
rechtes Ohr	✓		✓
linke Schulter	✓	✓	✓
rechte Schulter	✓	✓	✓
linker Ellenbogen	✓	✓	✓
rechter Ellenbogen	✓	✓	✓
linkes Handgelenk	✓	✓	✓
rechtes Handgelenk	✓	✓	✓
Brust		✓	
Becken		✓	
linke Hüfte	✓	✓	✓
rechte Hüfte	✓	✓	✓
linkes Knie	✓	✓	✓
rechtes Knie	✓	✓	✓
linker Knöchel	✓	✓	✓
rechter Knöchel	✓	✓	✓

Tabelle 3.1: Die enthaltenen Keypoint-Typen in den beiden öffentlichen Datensätzen COCO und MPII, sowie im eigenen Datensatz CrowdPose. [21] [1]

Der Datensatz beinhaltet zu den Bilddaten noch Annotationen. Annotiert sind pro Person dabei eine ID, eine passende Bounding-Box, die Aktivität der Person (*Fallen, Springen, Anlehnen, Liegen, Rennen, Sitzen, Stehen, Laufen, Winken*) und die Positionen der einzelnen Keypoints. Die 18 verwendeten Keypoint-Typen sind in Tabelle 3.1 und Abbildung 2.9 einsehbar.

3.1.1 Charakteristika des Datensatzes

Die größte Besonderheit des Datensatzes gegenüber den öffentlichen Datensätzen ist die erhöhte Personenzahl auf den einzelnen Bildern. Insgesamt sind auf den Bildern 833 Personen gekennzeichnet. Während COCO im Mittel circa 2 Personen zeigt und kein Bild mehr als 13 Personen enthält, zeigt CrowdPose im Schnitt 33 (Standardabweichung $std = 49{,}13$ Personen) und maximal bis zu 148 Personen in einem Bild. CrowdPose lässt sich zudem in die zwei zuvor genannten Bereiche *Wareneingang IOSB* mit etwa 7 Personen pro Bild (Standardabweichung $std = 0{,}74$ Personen) und die *Cannstatter Wasen* mit etwa 116 Personen pro Bild ($std = 32{,}02$ Personen) einteilen.

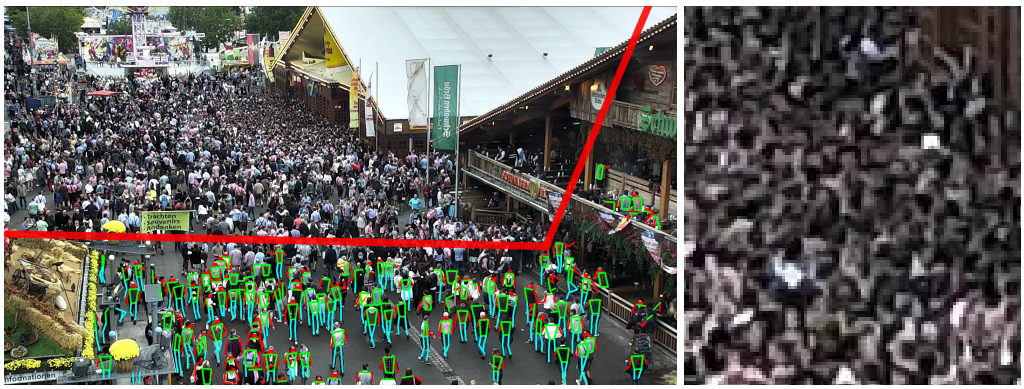


Abbildung 3.2: Links befindet sich das Bild *fruchtsaeule_beispiel_fern.jpg* von den Cannstatter Wasen und die darin annotierten Ground-Truth Daten. Der annotierte Bereich des Bildes erstreckt sich über das untere Drittel, wobei die Grenze (rote Trennlinie) auf Höhe der Hütte unten links im Bild liegt. Im rechten Bildabschnitt wurden über diese Grenze hinaus Personen auf dem Balkon bis zum mittleren Querbalken gekennzeichnet. Das rechte Bild stellt einen Ausschnitt des nicht annotierten Bildbereiches dar, welcher zeigt, dass in bestimmten Bereichen des Bildes die Annotation von Posen kaum bis gar nicht möglich ist. [15]

Die Cannstatter-Wasen-Bilder sind in Full HD (1920∗1080 Pixel) aufgelöst, während die Bilder am Wareneingang des IOSB die geringfügig höhere, aber vergleichbare Auflösung WUXGA ($1920 * 1200$ Pixel) vorweisen. Anzumerken ist hierbei, dass die Aufnahme in Darstellung 3.2 mit dem Namen *fruchtsaeule_beispiel_fern.jpg*, welche auf den Cannstatter Wasen gemacht

wurde, nur im unteren Bildbereich annotiert ist, da es im oberen Bereich des Bildes für die annotierende Person kaum möglich ist, die relevanten Keypoints zu erkennen. Die Höhe der einzelnen Personen liegt im Schnitt bei circa 201 Pixeln (std = 71 Pixel).

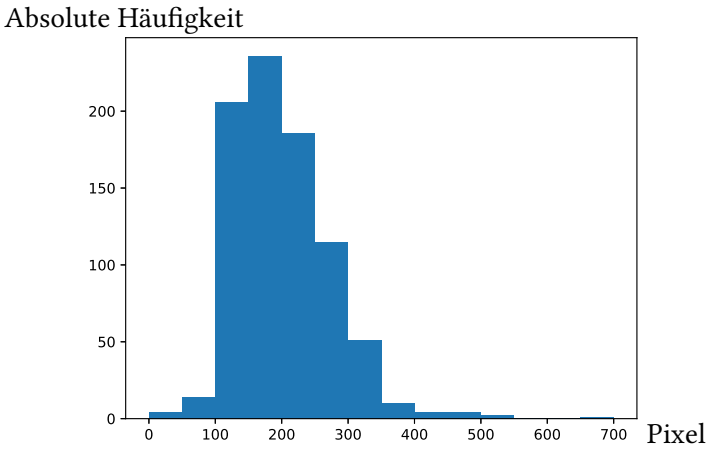
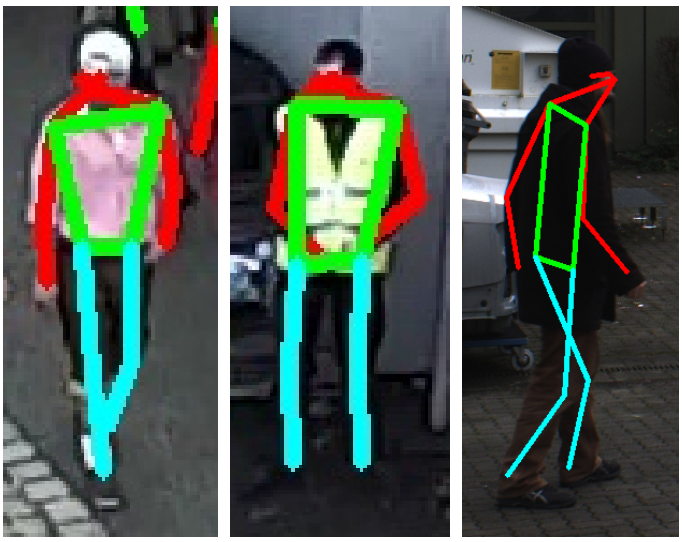


Abbildung 3.3: Ein Histogramm über die Verteilung der Körpergrößen (Höhe) im Datensatz CrowdPose. Zu sehen ist, dass die Körpergrößen nahezu normalverteilt sind.



61 ∗ 151 Pixel 61 ∗ 147 Pixel 202 ∗ 496 Pixel

Abbildung 3.4: Darstellung einzelner Personen und deren Körperposen mit der zugehörigen Auflösung der Bildausschnitte. Die linken zwei Bilder zeigen Personen von den Cannstatter Wasen, während das rechte Bild eine Person vor dem Wareneingang des IOSB zeigt. [15]

Ein weiteres Merkmal des Datensatzes ist die vorrangig durch weitere Personen erzeugte Überdeckung von Körperteilen, welche ebenfalls mit der erhöhten Personenzahl einhergeht. Nicht sichtbare Körperteile sind, sofern sie trotzdem während der Annotation lokalisiert werden können, in CrowdPose als *unsichtbar* gekennzeichnet.

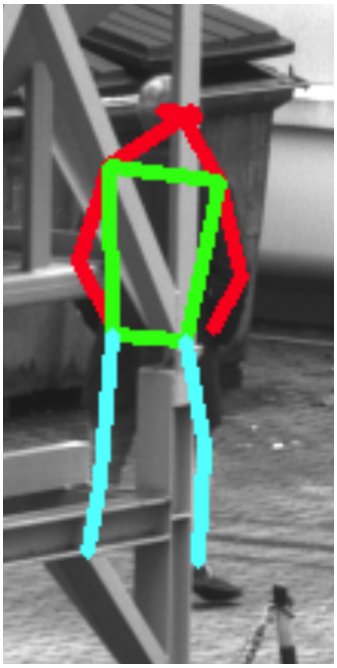


Abbildung 3.5: Ein Beispiel für unsichtbare Keypoints in CrowdPose. Einige der markierten Keypoints sind nicht sichtbar, wurden aber mithilfe des Wissens über die Körperstruktur dennoch annotiert. [15]

3.2 Verwendete Frameworks

Zur Evaluation der bisherigen State-of-the-Art-Verfahren werden zwei bereits implementierte Verfahren herangezogen. Es handelt sich dabei um die beiden Verfahren *Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields* [3] und *Regional Multi-person Pose Estimation* [10], welche unter den Namen *Openpose* und *Alphapose* als Frameworks veröffentlicht wurden.

3.2.1 Openpose

Aus dem Paper *Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields* entstand neben einigen anderen Implementierungen auch das *Openpose* genannte Framework, welches

in C++ geschrieben wurde. Für die Evaluation wurde Openpose als Plugin für das auf dem Qt-Framework basierende Programm DevEnviro verwendet. Openpose besitzt einige Parameter, welche justiert werden können, um die Ergebnisse zu verbessern oder die Geschwindigkeit des Verfahrens zu erhöhen. Dabei ist besonders der Parameter für die Auflösung relevant, welcher als Standardwert „-1x368“ beträgt. 368 stellt dabei die Höhe des Eingabebildes dar, während -1 ein Platzhalter für die Breite ist. Openpose wählt automatisch eine entsprechende Bildbreite, um das Seitenverhältnis beizubehalten. Für die Untersuchungen dieser Arbeit wurde der Standardwert für die Skalierung der Bilder beibehalten. Auch in Abschnitt 3.4.3 wurde auf diese in Openpose integrierte Möglichkeit zur Skalierung der Eingabebilder verzichtet, um eine Vergleichbarkeit zu Alphapose zu gewährleisten. [3]

Im Rahmen dieser Arbeit wurde an dem vorhandenen Plugin eine Ausgabe der Ergebnisse in Form von Bild- und Annotationsdateien ergänzt. Die ausgegebenen Bilder zeigen die von Openpose fertig gerenderten Posenskelette. Die Annotationsdateien enthalten die für eine quantitative Auswertung nötigen Keypoint-Positionen, welche mithilfe eines weiteren Python-Skripts in eine für die Anwendung der COCO-Metriken benötigte Form aufbereitet werden können. Die Prozedur 1 skizziert den für den Export implementierten Algorithmus *save2Json*.

Algorithm 1 Annotationen in JSON-Format exportieren

```
1:  procedure SAVE2JSON
2:      exportFile ← Pfad zur Export-JSON-Datei;
3:      currentKeypoints ← Keypoints des aktuelles Bildes;
4:      currentImageName ← Name des aktuellen Bildes;
5:      existingAnnotationsList ← getAnnotationsFromExistingFile(exportFile);
6:      if currentImageName ∈ existingAnnotationsList then
7:          existingAnnotationsList.replace(currentImageName, currentKeypoints);
8:      else
9:          existingAnnotationsList.append(currentImageName, currentKeypoints);
10:     exportiere existingAnnotationsList in Datei exportFile
```

3.2.2 Alphapose

Als weiteres Verfahren wurde das *Alphapose* genannte Framework untersucht. Es handelt sich dabei um die Implementierung des Ansatzes aus dem Paper *Regional Multi-person Pose Estimation*. Wie bei Openpose wurde hier ebenfalls nicht die ursprüngliche Implementierung des Papers, welche in Lua geschrieben wurde, benutzt, sondern eine Betaversion einer Portierung in der Programmiersprache Python, welche das Deep-Learning Framework PyTorch verwendet. [10]

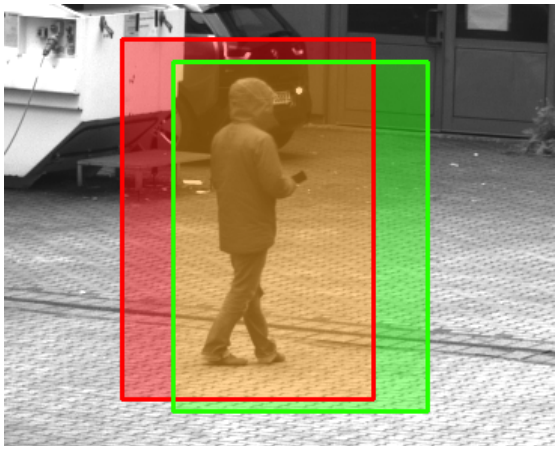
3.3 Metriken aus der COCO-Keypoint-Challenge

Für die Evaluation wurden die Keypointerkennungs-Metriken des COCO-Datensatzes mit einigen Anpassungen verwendet. Die Metriken sind Nachbildungen der Metriken *Average Precision (AP)* und *Average Recall (AR)*, welche in der Objekterkennung Anwendung finden. Folgende Gleichungen veranschaulichen die Aussage der beiden Maße. Die Average Precision gibt an, wie viele erkannte Objekte richtig erkannt wurden, während der Average Recall angibt, wie viele der zu erkennenden Objekte erkannt wurden.

$$AP = \frac{|\text{true positives}|}{|\text{true positives} + \text{false positives}|} \tag{3.1}$$

$$AR = \frac{|\text{true positives}|}{|\text{true positives} + \text{false negatives}|} \tag{3.2}$$

Grundlage für die Metriken ist ein Ähnlichkeitsmaß zwischen den Ground-Truth-Daten und den Ergebnissen. Im Fall der Objekterkennung handelt es sich bei diesem Maß um die sogenannte *Intersection over Union (IoU)*, welche die Überdeckung der Boxen aus den Ground-Truth-Daten mit denen der Ergebnisse bestimmt. Abbildung 3.6 zeigt, wie die *IoU* bestimmt werden kann.



$$IoU = \frac{\text{Schnittfläche}}{\text{Vereinigungsfläche}}$$

Abbildung 3.6: Schematische Darstellung der Berechnung der *Intersection over Union (IoU)* zwischen den Ground-Truth-Daten und Ergebnisdaten. Die rot umrandete Box stellt die Ground-Truth dar, während die grüne Box ein beispielhaftes Prädiktionsergebnis zeigt. Die *IoU* gibt das Verhältnis zwischen der gesamten Fläche zur orange schattierten Fläche an. [15]

Zur Berechnung der Metriken *AP* und *AR* werden Grenzen für die *IoU* festgelegt, ab denen es sich um *true positives* handelt. Über gestaffelten Grenzen werden Mittelwerte bestimmt, indem zum Beispiel die Metrik $AP_{@IoU=0,5:,05:0,95}$ für die *IoU*-Grenzen *{0,50; 0,55; 0,60; 0,65; 0,70; 0,75; 0,80; 0,85; 0,90; 0,95}* ausgewertet wird. Um *AP* und *AR* für die Keypointerkennung zu übernehmen, wurde die *IoU* durch ein analoges Ähnlichkeitsmaß ersetzt. Hierfür wurde in COCO die *Object Keypoint Similarity (OKS)* eingeführt. Der Einfachheit halber werden die Metriken zur Keypointerkennung, also die Metriken, welche die *OKS* nutzen, ebenfalls als *average precision (AP)* und *average recall (AR)* bezeichnet. [21]

3.3.1 Object Keypoint Similarity

Ein Objekt im COCO-Datensatz besitzt Ground-Truth-Keypoints in der Form $[x_1, y_1, v_1,..., x_{17}, y_{17}, v_{17}]$, wobei x, y die Position und v die Sichtbarkeit darstellen. Des Weiteren besitzt jedes Objekt des Datensatzes eine Größe $s = Area^2$, welche aus der Objekt-Segment-Fläche *Area* bestimmt wird. Ergebnisse müssen in gleicher Form für jeden Keypoint eine Position x, y enthalten, die Sichtbarkeit v der einzelnen Keypoints wird jedoch nicht benötigt, da sie bisher nicht in die Evaluation einfließt. Zu den Keypoints muss ein Ergebnisobjekt auch einen *Score* enthalten, welcher die Wahrscheinlichkeit einer korrekten Erkennung angibt. Die *Object Keypoint Similarity (OKS)* wird definiert als mathematische Funktion

$$OKS = \frac{\sum_i [\exp(\frac{-d_i^2}{2s^2\kappa_i^2})\delta(v_i > 0)]}{\sum_i [\delta(v_i > 0)]}, \tag{3.3}$$

wobei i als Laufvariable über alle Keypoint-Typen eines Objektes iteriert. Der Wert d_i gibt für jeden Keypoint die euklidische Distanz zwischen Ground-Truth und dem Ergebnis an, v_i ist die Sichtbarkeit aus den Ground-Truth-Daten (die Sichtbarkeit der Ergebnisdaten wird wie bereits erwähnt nicht verwendet), s ist die Objektgröße und κ_i ist eine Keypoint-abhängige Konstante, sodass $s * \kappa_i$ die Standardabweichung für die Ground-Truth-Keypoints darstellt. Perfekte Ergebnisse erhalten somit $OKS = 1$, während $OKS = 0$ für Ergebnisse mit erhöhter Abweichung von den Ground-Truth-Keypoints steht. κ_i wurde bestimmt, indem 5000 redundant annotierte Bilder ausgewählt wurden, woraufhin für jeden Keypoint-Typ i die Standardabweichung σ_i abhängig von der Objektgröße s bestimmt wurde. Für verschiedene Keypoint-Typen weichen die Werte für σ_i dabei sehr stark voneinander ab, sodass Körper-Keypoints (Schultern, Knie, usw.) im Allgemeinen ein höheres σ_i vorweisen als Gesichts-Keypoints (Augen, Nase, Ohren). Die Keypoint-Konstanten wurden schlussendlich auf $\kappa_i = 2 * \sigma_i$ festgelegt. [21]

Keypoint	Nase	Auge	Ohr	Schulter	Ellenbogen	Handgelenk	Hüfte	Knie	Knöchel
σ	0.26	0.25	0.35	0.79	0.72	0.62	1.07	0.87	0.89

Tabelle 3.2: Eine Veranschaulichung der Keypoint-Typen i zusammen mit den zugehörigen Standardabweichungen σ_i, welche abhängig von der Personengröße sind. [21]

3.3.2 Anpassung der Metriken zur Anwendung auf CrowdPose

Damit die Metriken auf CrowdPose angewandt werden können ist es notwendig, einige Veränderungen vorzunehmen. Die Metriken besitzen einen Wert für die maximale Anzahl erkannter Posen in einem Bild, welcher auf 20 Posen festgelegt wurde. Diese Grenze ist für diese Arbeit offensichtlich ungeeignet, da gerade über diese Grenze hinaus Posenschätzung in Bildern von Menschenmengen untersucht werden soll. Deshalb wird der Wert für die Evaluationen in dieser Thesis auf 200 Posen erhöht, andere Werte wurden innerhalb dieser Arbeit nicht erprobt. Während die ursprünglichen Varianten der Metriken *AP* und *AR* die Personengröße über das *Area*-Feld bestimmt, muss in dieser Arbeit auf ein anderes Maß zurückgegriffen werden. Dies liegt daran, dass das *Area*-Feld im COCO-Datensatz die Flächen der Segmentierung der einzelnen Personen angibt, CrowdPose besitzt jedoch keine Daten zur Segmentierung der Posen. Um diese Segmentierungsflächen zu ersetzen, werden deshalb die Flächen der Bounding-Boxen der einzelnen Personen genutzt. Somit gilt bei der Anwendung der Metriken auf CrowdPose nicht länger das Größenmaß $s_{COCO} = Area^2$, sondern $s_{CrowdPose} = (height * width)^2$ als Grundlage. Die neuen Metriken, welche durch die genannten Änderungen entstehen, werden von nun an als $\overline{AP}$ und $\overline{AR}$ bezeichnet. In Abschnitt 3.4.1 befinden sich hierzu Ergebnisse über die Vergleichbarkeit der Metriken nach und vor den Änderungen. Zu erwähnen ist noch, dass der eigene Datensatz CrowdPose zwar 18 annotierte Keypoint-Typen bereitstellt, das Fehlen der Standardabweichung κ des Keypoint-Typs *Nacken* zur Berechnung der Object Keypoint Similarity führte jedoch dazu, dass nur die restlichen 17 Keypoint-Typen in die Evaluation eingehen konnten.

3.4 Quantitative Auswertung

Nach der Vorstellung der verwendeten Verfahren und Metriken widmet sich dieser Abschnitt den Ergebnissen der quantitative Auswertung. Hier werden die verschiedenen Evaluationen ausführlich erklärt und die daraus resultierenden Werte interpretiert. Die quantitative Aus-

wertung stützt sich auf die in Abschnitt 3.3 beschriebenen Metriken. Mithilfe dieser Metriken werden die verschiedenen Verfahren auf den Datensätzen CrowdPose und COCO miteinander verglichen.

3.4.1 Vergleich der Metriken mit Area und Bounding-Box

Um eine Vergleichbarkeit zu den Metriken *AP* und *AR* aus dem COCO-Datensatz zu schaffen, werden die vom *Area*-Feld abhängigen Metriken mit den Bounding-Box-abhängigen Metriken $\overline{AP}$ und $\overline{AR}$ verglichen. Im ersten Schritt wurden dazu Openpose und Alphapose auf dem COCO-Validation Datensatz 2017 und CrowdPose ausgeführt. Für gewöhnlich würde man dazu den Testing-Datensatz von COCO verwenden, dieser ist jedoch ohne die Ground-Truth-Annotationen veröffentlicht worden. Die resultierenden Posen wurden danach mittels *AP*, *AR*, $\overline{AP}$ und $\overline{AR}$ ausgewertet, was in Tabelle 3.3 dargestellt ist. Die Werte für *AP* und *AR* bleiben bei CrowdPose leer, da wie in Abschnitt 3.3.2 erwähnt, die notwendigen *Area*-Werte nicht vorhanden sind.

Verfahren Metrik	Openpose COCO Val 2017	Openpose CrowdPose	Alphapose COCO Val 2017	Alphapose CrowdPose
$AP/\overline{AP}_{@OKS=0,50:0,95}$	0,29912/0,38154	−/0,00088	0,41202/0,46619	−/0,00349
$AP/\overline{AP}_{@OKS=0,50}$	0,62827/0,71057	−/0,00110	0,52165/0,52186	−/0,00356
$AP/\overline{AP}_{@OKS=0,75}$	0,25245/0,37139	−/0,00110	0,43477/0,49106	−/0,00353
$AR/\overline{AR}_{@OKS=0,50:0,95}$	0,32849/0,40806	−/0,00324	0,42581/0,47581	−/0,01297
$AR/\overline{AR}_{@OKS=0,50}$	0,64516/0,72043	−/0,00600	0,52151/0,52688	−/0,01561
$AR/\overline{AR}_{@OKS=0,75}$	0,2957/0,40323	−/0,00240	0,44086/0,49462	−/0,014406

Tabelle 3.3: Die Tabelle stellt die Ergebnisse der verschiedenen Metriken *AP*, *AR*, $\overline{AP}$ und $\overline{AR}$ angewandt auf den Resultaten von Openpose und Alphapose auf dem COCO-Validation Datensatz 2017 und CrowdPose dar. Da die Metriken *AP* und *AR* auf dem *Area*-Feld beruhen, konnten diese Metriken auf CrowdPose nicht ausgeführt werden. [3] [10] [21]

Die Verhältnisse zwischen *AP* und $\overline{AP}$, sowie *AR* und $\overline{AR}$ liegen für Openpose im Schnitt bei 0,7826 ($std = 0{,}0835$) und 0,8113 ($std = 0{,}0664$). Dies lässt darauf schließen, dass die Bounding-Boxen eine akzeptable Näherung für das *Area*-Feld ergeben. Viel besser sind die Werte jedoch bei Alphapose, dort liegen die Ergebnisse für die Relationen $AP/\overline{AP}$ und $AR/\overline{AR}$ im Schnitt bei 0,9229 ($std = 0{,}0542$) und 0,9253 ($std = 0{,}0456$). Somit lässt sich vermuten, dass die Ergebnisse der Metriken bei Alphapose eine geringere Abhängig von der korrekten Skalierung haben. Die

Werte von Alphapose und Openpose auf dem eigenen Datensatz CrowdPose korrelieren mit den allgemeinen Ergebnissen der beiden Verfahren. Schaut man sich nämlich die Ergebnisse für beide Verfahren und Datensätze an, sieht man, dass das Verhältnis zwischen Alphapose und Openpose auf CrowdPose ein Vielfaches des Verhältnisses auf dem COCO Val 2017 Datensatz ist.

3.4.2 Vergleich zwischen kleinen und großen Menschenmengen

Ein weiterer Vergleich wird ausschließlich auf dem Datensatz CrowdPose evaluiert. Hier werden die Frameworks *Alphapose* und *Openpose* auf den Cannstatter-Wasen- und den Wareneingang-IOSB-Bildern aus CrowdPose ausgeführt. Die Ergebnisse sind in Tabelle 3.4 abgebildet.

	Alle	Cannstatter Wasen	Wareneingang IOSB
Alphapose	0,00349/0,01297	0,00006/0,00359	0,01249/0,06103
Openpose	0,00088/0,00324	0,00003/0,00043	0,00231/0,01765

Tabelle 3.4: Die $\overline{AP}/\overline{AR}$ ($@OKS = 0{,}50 : 0{,}95$) der Verfahren *Alphapose* und *Openpose* auf dem Datensatz CrowdPose, getrennt nach Cannstatter Wasen und Wareneingang am Fraunhofer IOSB. [10] [3]

Diese Evaluation zeigt, dass beide Verfahren schlechte Ergebnisse für exakte Posenschätzungen bei großen Menschenmengen liefern. Im Vergleich dazu zeigen die Verfahren bei Bildern mit wenigen Personen aus dem COCO-Datensatz viel höhere Werte. Dennoch ist ein Trend erkennbar: Die Ergebnisse beider Verfahren fallen auf den Bildern vom Wareneingang des IOSB am besten aus, während die Resultate auf den Cannstatter-Wasen-Bildern in beiden Fällen am schlechtesten sind. Dass die Metriken für den kompletten Datensatz Werte dazwischen liefern, ist deshalb nicht verwunderlich. Des Weiteren lässt sich aussagen, dass die Ergebnisse von Alphapose die Ergebnisse von Openpose übertreffen, es ist aus den vorhandenen Daten jedoch nicht ersichtlich, ob eines der Verfahren besser für kleine oder große Menschenmengen geeignet ist. Im Allgemeinen sind Top-Down-Verfahren wie Alphapose jedoch besser dafür geeignet, kleinere Personen zu erkennen, während Bottom-Up-Verfahren schneller und skalierbarer bezüglich der Personenanzahl sind. [25]

3.4.3 Vergleich verschiedener Skalierungen der Eingabebilder

Um die Verfahren auf ihre Fähigkeit zur Detektion von Posen bei geringerer Pixeldichte zu vergleichen, wurden sie in einem weiteren Schritt auf dem Datensatz CrowdPose ausgewer-

tet, mit dem Zusatz, dass die Bilder auf verschiedene Größen herunterskaliert wurden. Das Seitenverhältnis der Bilder wurde dabei beibehalten. Sie werden in vier Schritten in ihrer Höhe und Breite halbiert, sodass die Bilder, welche zuvor $1920 * 1080$ Pixel hatten, in den Skalierungen $1920 * 1080$, $960 * 540$, $480 * 270$, $240 * 135$ Pixeln und die vorherigen Bilder der Größe $1920 * 1200$ Pixel analog in $1920 * 1080$, $960 * 600$, $480 * 300$, $240 * 150$ Pixeln vorliegen. Hierdurch enthalten die verschiedenen skalierten Bilder Personen in 1, 1/2, 1/4 und 1/8-facher Höhe der ursprünglichen Personen.

Skalierung	1	1/2	1/4	1/8
Alphapose	0,00348/0,01212	0,00509/0,01128	0,00506/0,00840	0,00089/0,00240
Openpose	0,00079/0,00312	0,00081/0,00336	0,00053/0,00264	0,00113/0,00168

Tabelle 3.5: Die $\overline{AP}/\overline{AR}$ ($@OKS = 0{,}50 : 0{,}95$) der Verfahren *Alphapose* und *Openpose* auf dem ganzen Datensatz CrowdPose, dessen Bilder auf das 1, 1/2, 1/4 und 1/8-fache ihrer Höhe und Breite skaliert wurden. [10] [3]

Die Tabelle 3.5 zeigt die Ergebniswerte der Untersuchung. Die $\overline{AP}$ der Verfahren zeigt hier keine besonders auffälligen Werte, beide Verfahren erzielen sogar auf den ganzen Bildern (1-fache Skalierung) schlechtere Ergebnisse, als auf den 1/2-Bildern. Openpose liefert für die $\overline{AP}$ darüber hinaus den besten Wert auf der kleinsten Skalierung der Bilder. Anders sieht das bei dem $\overline{AR}$ aus, welcher, wie in Abschnitt 3.3 beschrieben, etwas über die Anzahl der erkannten richtigen Objekte aussagt. Hier ist vor allem ab 1/4 der Bildhöhe und -breite ein klarer Abwärtstrend zu erkennen, welcher für eine schlechte Erkennungsrate bei niedrigerer Auflösung steht.

3.4.4 Zeitmessung

Ein weiteres Kriterium zur Evaluation von Posenschätzungs-Algorithmen ist die für eine Berechnung benötigte Zeit. Hierfür wurde ein Computer mit einem *Intel i7-7700k 4*4,2GHz*-Prozessor, *16 GB RAM (3200 MHz)* und einer *NVidia GeForce GTX 1080 8 GB VRAM* verwendet. Die zur Evaluation verwendeten Bilder aus dem Datensatz CrowdPose befanden sich auf einer Festplatte (5300 Umdrehungen pro Minute).

Die Zeitmessungen der Verfahren, deren Resultate in Tabelle 2 einsehbar sind, ergeben eine durchschnittliche Dauer von $0{,}34s$ ($std = 0{,}50s$) für Alphapose und $0{,}28s$ ($std = 0{,}15s$) für Openpose. Somit ist Openpose geringfügig schneller auf dem kompletten Datensatz. Dabei ist die hohe Standardabweichung bei Alphapose allerdings interessant. Schaut man sich

die Ergebnisse getrennt nach den Bildmengen des Datensatzes an, erreicht Alphapose für die Wareneingang-IOSB-Bilder im Schnitt einen Wert von 0,07s (std = 0,03s), während die Cannstatter-Wasen-Bilder circa 1,18s (std = 0,34s) benötigen. Dem gegenüber steht Openpose mit nahe beieinanderliegenden 0,22s (Wareneingang IOSB, std = 0,01s) und 0,47s (Cannstatter Wasen, std = 0,23s). Openpose benötigt folglich für die Bilder mit einer geringen Personenanzahl zwar länger, schließt Berechnungen auf Bildern mit vielen Personen aber viel schneller ab. Alphapose dagegen erreicht sehr hohe benötigte Zeiten für Bilder mit vielen Menschen, während die Posen auf den nur leicht bevölkerten Bildern in Bruchteilen einer Sekunden berechnet werden.

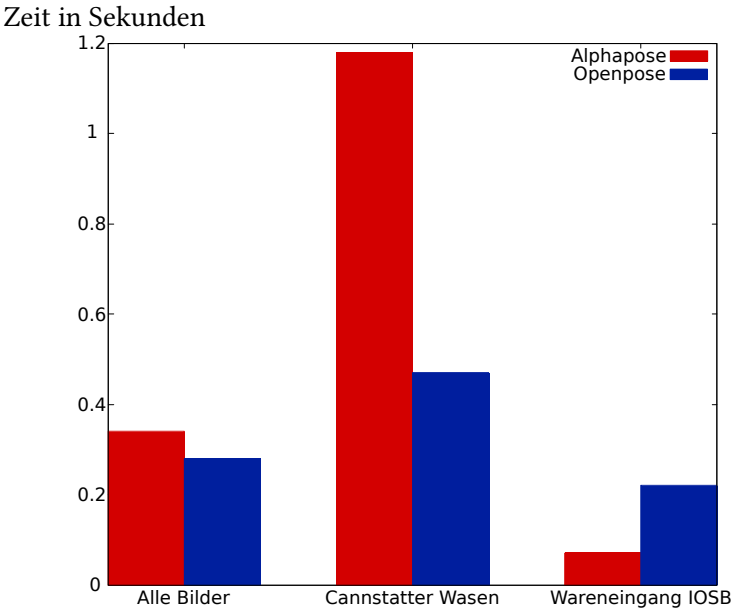


Abbildung 3.7: Darstellung der benötigen Zeiten von Alphapose und Openpose auf den Bildern des Datensatzes CrowdPose in Form eines Balkendiagramms. Die roten Säulen stellen die Alphapose- und die blauen die Openpose-Ergebnisse dar.

Die Stärke von Alphapose auf Bildern mit kleinen Personenzahlen könnte durch das Top-Down-Modell, auf welchem dieses Verfahren beruht, erklärt werden, da die Anzahl der Iterationen des Single-Person-Pose-Estimators (Kapitel 2) direkt von der Anzahl erkannter Personen abhängt. Openpose hingegen folgt dem Bottom-Up-Modell, wodurch unabhängig von der Personenzahl möglichst alle Keypoints auf einem Bild bestimmt werden.

Vergleicht man die benötigte Zeit für die verschiedenen Skalierungen aus Abschnitt 3.4.3, sieht man, dass beide Verfahren mit abnehmender Bildgröße schneller werden. Die Bilder in Originalgröße benötigen im Schnitt, wie zuvor erwähnt, circa 0,34s (Alphapose std = 0,50s)

und 0,28s (Openpose $std = 0{,}15s$). Die Ergebnisse für die Bilder mit 1/8 der ursprünglichen Bildhöhe- und breite sind bereits in etwa 0,2s (Alphapose $std = 0{,}30s$) und 0,21s (Openpose $std = 0{,}01s$) bestimmt. Hier kann auch Openpose von einer niedrigeren Anzahl erkannter Keypoints profitieren, da die die Komplexität des Optimierungsproblems (Kapitel 2) sinkt.

3.5 Qualitative Auswertung

Während die quantitative Auswertung ein sehr ernüchterndes Ergebnis liefert, zeigt die qualitative Auswertung ein anderes Bild. Im Gegensatz zu den vorherigen Ergebnissen, welche durch viele Schwierigkeiten nur begrenzt aussagefähig sind, zeigt die qualitative Auswertung, dass die Verfahren doch sehr gute Ergebnisse erzeugen können.

Wie in den Bildern aus Abbildung 3 zu sehen ist, gibt es eine beachtliche Überdeckung der Personen in den Ergebnissen. Trotz der kleinen durchschnittlichen Personengröße wird nahezu jede Person erkannt, die meisten sogar in ihrer gesamten Körperpose. Zu beachten ist bei den qualitativen Ergebnissen, inwieweit die Erkennung der Personen mit der Erkennung der Körperkeypoints zusammenhängt. Die vorher ausgeführte quantitative Auswertung lässt nur Aussagen über die korrekte Detektion von Keypoints zu, während die qualitative Untersuchung vor allem die ganzen Personen und deren erfolgreiche Erkennungsrate beschreiben kann.

Infrage zu stellen ist deshalb auch die Verwendung der Metriken, welche durch die zuvor festgelegten Keypoint-Standardabweichungen (Abschnitt 3.3.1) zur Bestimmung der Object Keypoint Similarity möglicherweise zu stark an die den COCO-Datensatz annotierenden Personen angepasst sind. Durch die eigenständige Annotation des Datensatzes CrowdPose könnten ungewollte Nebeneffekte eingetreten sein, welche sich auf die Ergebnisse der Metriken auswirken. Die Verfahren wurden in den Untersuchungen mit jenen Modellen verwendet, welche auf dem COCO-Datensatz trainiert wurden. Eine Abweichung im eigenen Annotationsverhalten zu dem der COCO-Mitarbeiter könnte der Grund für die niedrigen quantitativen Ergebniswerte sein. Ein Nachtrainieren der Verfahren zur besseren Nutzbarkeit auf CrowdPose war im Rahmen dieser Arbeit nicht vorgesehen, könnte aber für bessere Resultate auf den Metriken sorgen. Ebenfalls wäre eine Vergrößerung des Datensatzes CrowdPose eine Möglichkeit, bessere Ergebnisse zu erzielen, da man gegebenenfalls die Keypoint-Standardabweichungen neu bestimmen könnte.

Auffällig ist im Vergleich zwischen Alphapose und Openpose noch, dass Ersteres im Allgemeinen mehr Keypoints detektiert. Dies hat zur Folge, dass die Ergebnisse von Alphapose überwiegend mehr Informationen zu einzelnen Personen liefern, wodurch aber auch sehr abwegige Verbindungen, wie in dem in Abbildung 3.8 dargestellten Bild, entstehen. Die unterschiedliche Stärke der Verbindungslinien könnten ein Indiz für die durch das Verfahren

bestimmten Wahrscheinlichkeiten sein, dass es sich um eine korrekte Verbindung handelt. Somit würde das Verfahren solche Keypoints zwar verknüpfen, gleichzeitig aber aussagen, dass die Verbindungen unwahrscheinlich sind. Zur Bedeutung der Linienstärke konnte jedoch keine Erläuterung gefunden werden.

Abbildung 3.8: Darstellung der Querverbindungen zwischen Keypoints verschiedener Personen in den Ergebnissen von Alphapose nach der Anwendung auf CrowdPose. Zu sehen ist, dass eindeutig unlogische Verbindungen zwischen verschiedenen Personen geknüpft werden. [10] [15]

3.5.1 Vergleich verschiedener Skalierungen der Eingabebilder

Der Vergleich aus Abschnitt 3.4.3 zeigte, dass die $\overline{AP}$ der Verfahren zwar auf höher aufgelösten Bildern in etwa gleich gering ist wie bei niedrigerer Auflösung, ihr $\overline{AR}$ jedoch spürbar mit der Auflösung abnimmt. Die qualitative Auswertung bestätigt den sinkenden $\overline{AR}$ vor allem bei der Verwendung von Openpose, wie in den unteren beiden Bildern der Abbildung 4 beispielhaft zu sehen ist. Bei Alphapose tritt dieser Trend in einzelnen Bildbereichen zwar auch auf, fällt jedoch im Allgemeinen, wie in Abbildung 4 zu sehen ist, viel geringer aus.

4 Fazit

Dieses Kapitel fasst die gewonnenen Erkenntnisse zusammen und schließt daraufhin mit einem Ausblick auf weitere Untersuchungsansätze diese Arbeit ab.

4.1 Zusammenfassung

Zu Beginn dieser Arbeit geht Kapitel 1 auf die zum Verständnis nötigen Grundlagen des Themengebiets ein. Dazu werden neuronale Netze, besonders auch im Hinblick auf Convolutional Neural Networks, und Körperposen erläutert. In Kapitel 2 gibt diese Arbeit einen breiten Einblick in die aktuellen State-of-the-Art Deep-Learning-Verfahren zur Posenschätzung auf Bilddaten und vergleicht im darauf folgenden Kapitel 3 zwei dieser Verfahren hinsichtlich ihrer Performance auf Bildern mit großen Menschenmengen. Dazu wird auch ein Datensatz vorgestellt, welcher während der Thesis für diese Evaluation ausgearbeitet wurde.

Die quantitativen Untersuchungen sprechen für noch viel Verbesserungspotential. Die überprüften Verfahren erzielen auf CrowdPose unverhältnismäßig schlechtere Ergebnisse als auf den bekannten Datensätzen COCO [21] und MPII [1]. Dies könnte sowohl auf die Verfahren, das Training der neuronalen Netze sowie auf die für die Evaluation verwendeten Metriken zurückzuführen sein. Die qualitative Auswertung zeigt, dass die Anzahl der erkannten Personen relativ hoch ausfällt. Da die Metriken jedoch Aussagen auf der Ebene der Keypoints treffen, ist nicht sicher, inwieweit die hohe Erkennungsrate der Personen auch auf einen hohen *Average Recall* $\overline{AR}$ schließen lässt. Abschließend kann man sagen, dass die Verfahren bereits gute Ergebnisse bezüglich der Personendetektion erzielen, die Keypointschätzung jedoch noch stark zu wünschen übrig lässt.

4.2 Ausblick

Nachfolgend werden drei Ansätze für weitere Untersuchungen bezüglich einer Verbesserung der quantitativen Ergebnisse vorgestellt.

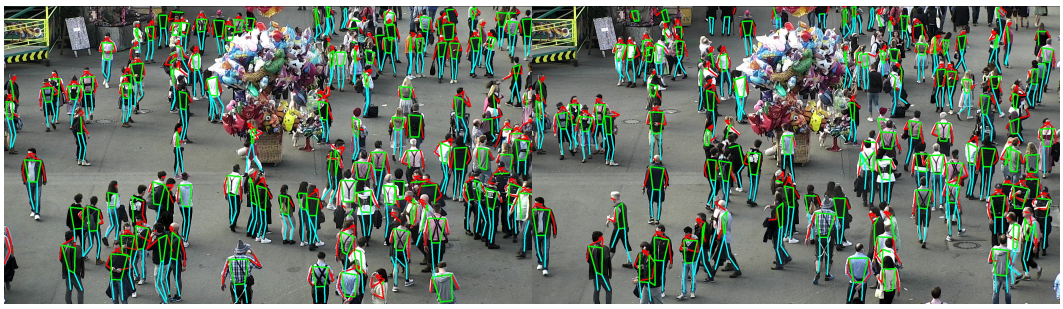
Die verwendeten Verfahren sind nur für kleine bis mittelgroße Menschenmengen bestimmt, was auch durch die quantitativen Ergebnisse belegt wurde. Ein neues Verfahren, welches eine Art Window-Slicing implementieren würde, könnte eingehende Bilddaten zuerst rastern und in einem weiteren Schritt die gerasterten Bildteile an bestehende Verfahren übergeben. Dies könnte einen Ausgangspunkt für weitreichenden Verbesserungen auf einer höheren Ebene bieten, da die aktuellen State-of-the-Art-Verfahren substituierbar wären.

Sowohl Alphapose, als auch Openpose sind abgeschlossene Frameworks, welche auf dem COCO-Datensatz trainiert wurden. Ein Nachtraining zur Verbesserung der Anwendbarkeit auf CrowdPose wäre ebenfalls ein vielversprechender Ansatz. Dies bringt allerdings einen großen Mehraufwand mit sich, da der Datensatz erheblich erweitert werden müsste, um ein sinnvolles Nachtraining in Aussicht stellen zu können. Unter Umständen könnte dieses Problem durch Verwendung von synthetischen Daten gelöst werden. Dazu könnte ein Datensatz wie *Joint Track Auto* [9] verwendet werden.

Zur Untersuchung wurden die Metriken des COCO-Datensatzes abgewandelt, sodass diese auf CrowdPose angewandt werden konnten. Trotz der Betrachtung der Änderungen der Metriken in Abschnitt 3.4.1 könnten hier unvorhergesehene Effekte die Vergleichbarkeit zwischen COCO und CrowdPose beeinträchtigen. Hier sind vor allem die Keypoint-Standardabweichungen in Abhängigkeit zur Körpergröße als Schwachstelle der Metriken anzumerken. Eine Untersuchung, inwieweit die Standardabweichungen der Keypoints in Datensätzen von der Personenanzahl auf den einzelnen Bildern abhängen, könnte Aufschluss darüber geben, ob die Skalierung der Personen ein Problem darstellt.

Anhang

1 CrowdPose



1. ACC-Export-2018 - 05-08 14.13.02-59.jpg

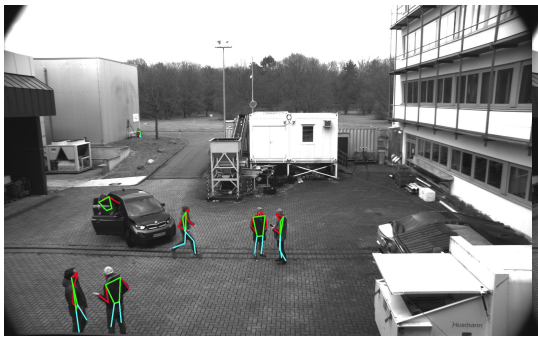
1. ACC-Export-2018 - 05-08 14.13.02-217.jpg

1. ACC-Export-2018 - 05-08 14.13.02-306.jpg

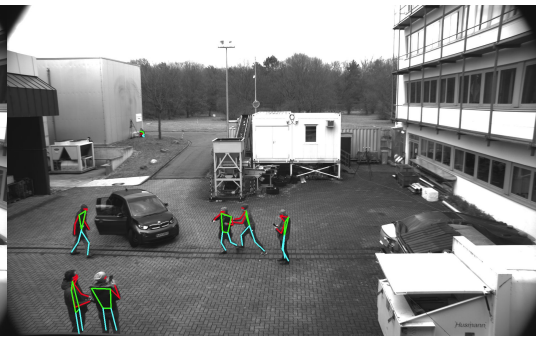
4. fruchtsaeule_beispiel_fern.jpg

5. fruchtsaeule_beispiel_nah.jpg

6. kegelenstrasse_beispiel.jpg

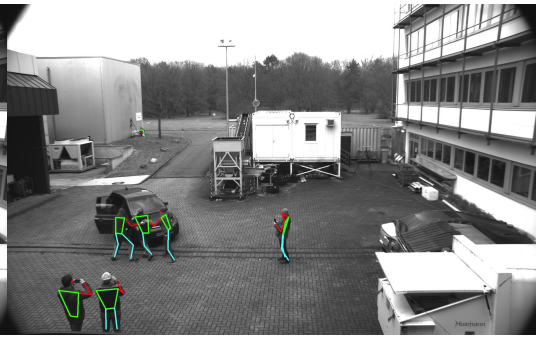


7. PtgreyMultiCam_ImageRight _1517494749371288.jpg

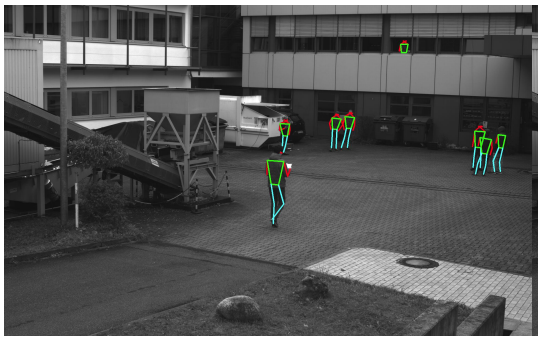


8. PtgreyMultiCam_ImageRight _1517494752196277.jpg

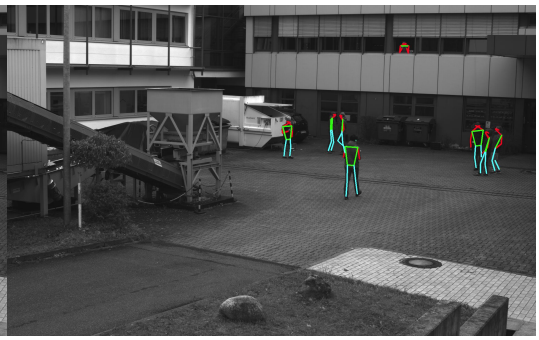
9. PtgreyMultiCam_ImageRight _1517494752278980.jpg



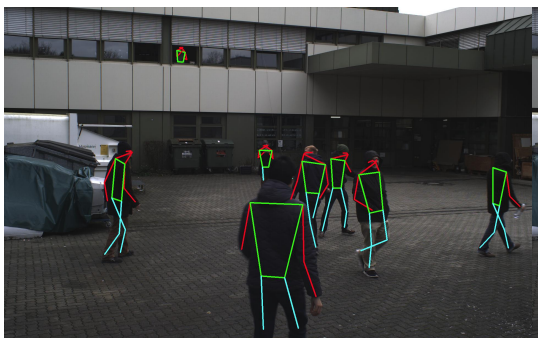
10. PtgreyMultiCam_ImageRight _1517494754148425.jpg



11. PtgreyMultiCam_ImageRight _1517495002626235.jpg

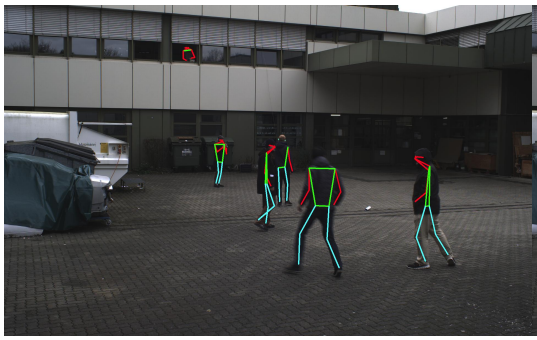


12. PtgreyMultiCam_ImageRight _1517495006001582.jpg

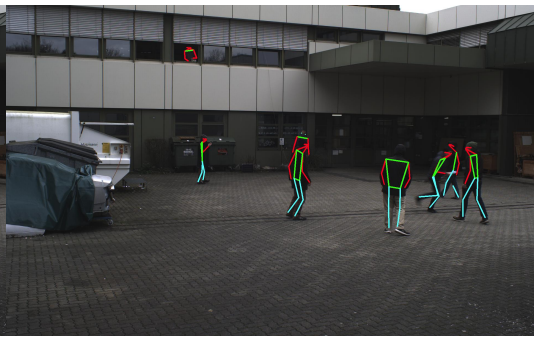


13. PtgreyMultiCam_ImageRight _1517495023952611.jpg

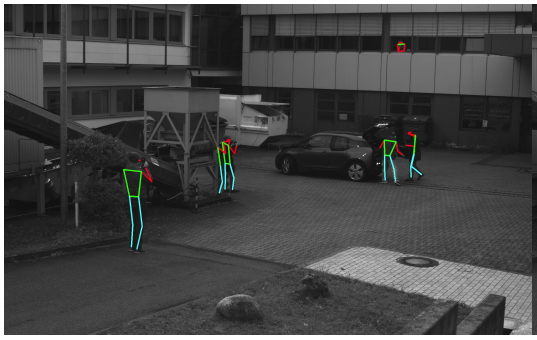
14. PtgreyMultiCam_ImageRight _1517495027602921.jpg



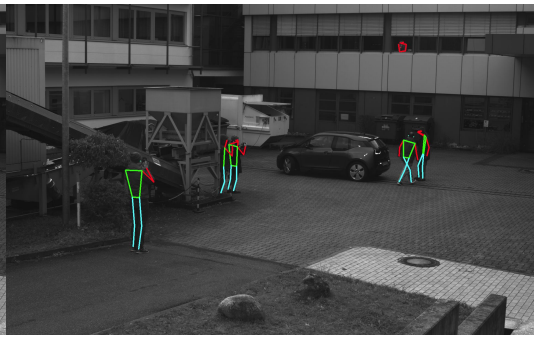
15. PtgreyMultiCam_ImageRight _1517495029518921.jpg



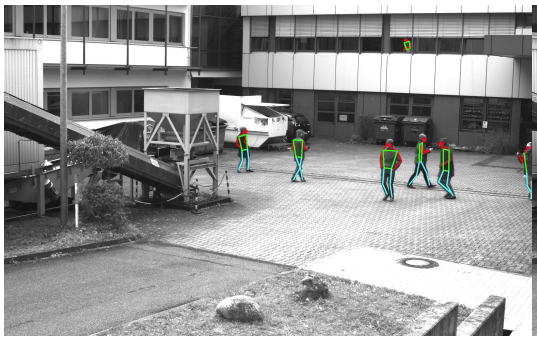
16. PtgreyMultiCam_ImageRight _1517495035415433.jpg



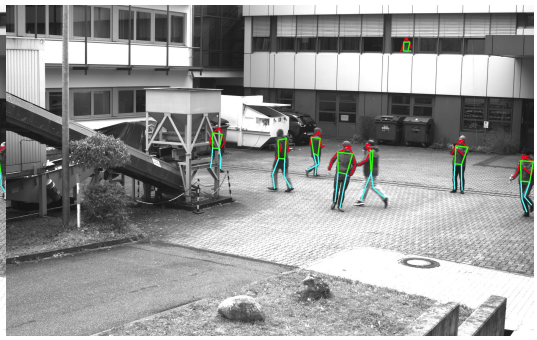
17. PtgreyMultiCam_ImageRight _1517495344444089.jpg



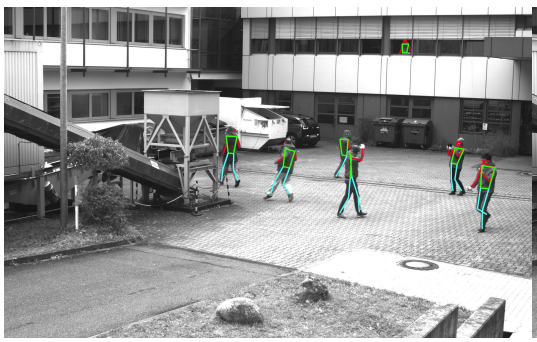
18. PtgreyMultiCam_ImageRight _1517495348178868.jpg



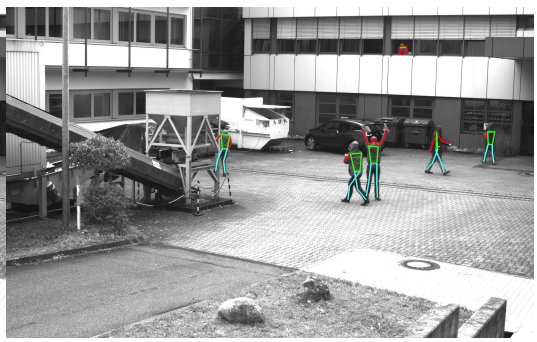
19. PtgreyMultiCam_ImageRight _1517495530213973.jpg



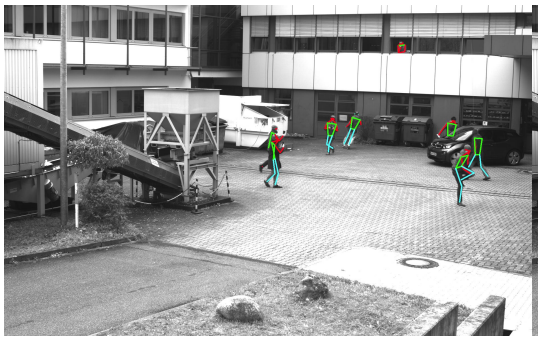
20. PtgreyMultiCam_ImageRight _1517495532495512.jpg



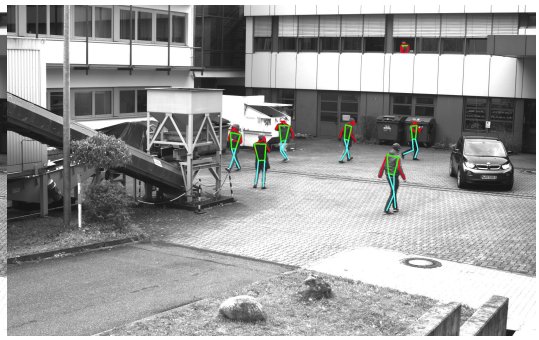
21. PtgreyMultiCam_ImageRight _1517495533448720.jpg



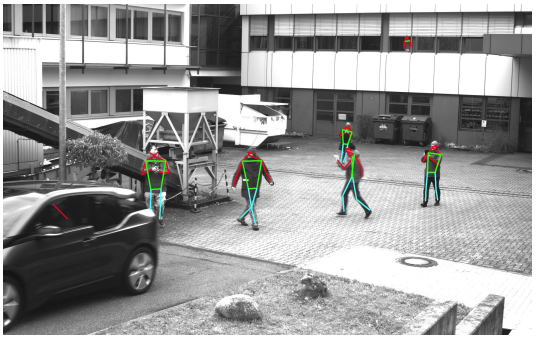
22. PtgreyMultiCam_ImageRight _1517495540465100.jpg



23. PtgreyMultiCam_ImageRight _1517495547325234.jpg



24. PtgreyMultiCam_ImageRight _1517495552607020.jpg



25. PtgreyMultiCam_ImageRight _1517495585548061.jpg

Abbildung 1: Der komplette Datensatz CrowdPose inklusive der annotierten Ground-Truth-Daten, welche mithilfe eines openCV-Skriptes in die Bilder gezeichnet wurden. Die ersten sechs Bilder (Auflösung 1920 ∗ 1080 Pixel) wurden auf der Cannstatter Wasen aufgezeichnet, die restlichen im Wareneingang des IOSB Auflösung 1920 ∗ 1200 Pixel. [15]

2 Personenhöhen der einzelnen Bilder

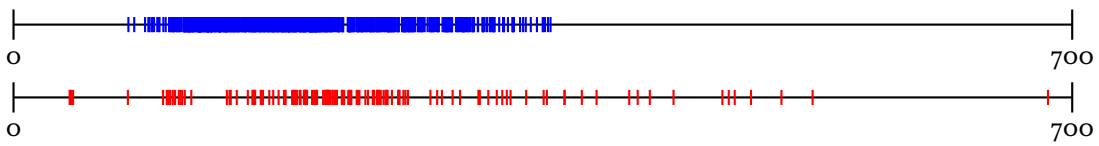


Abbildung 2: Verteilung der Personenhöhen bei Cannstatter-Wasen- (blau) und Wareneingang-IOSB-Bildern (rot) in Pixeln.

Bild	Personenhöhe	
	Durchschnitt	Standardabweichung
1	225,04	53,06
2	234,55	49,69
3	232,95	52,83
4	126,52	16,45
5	194,55	48,74
6	150,62	23,65
7	189,87	68,66
8	191,3	65,43
9	195,51	67,16
10	194,46	64,85
11	186,19	58,69
12	179,71	36,35
13	394,26	171,13
14	343,92	122,90
15	331,69	138,75
16	253,62	69,02
17	235,94	88,31
18	238,86	90,56
19	202,32	45,26
20	219,12	50,09
21	221,63	58,83
22	200,93	64,6
23	180,20	41,78
24	186,59	50,62
25	263,77	87,36
Cannstatter Wasen	194,04	41,95
Wareneingang IOSB	232,1	59,91

Tabelle 1: Diese Tabelle zeigt die durchschnittlichen Personenhöhen in Pixeln der einzelnen Bilder mit den dazugehörigen Standardabweichungen.

3 Auswahl an Ergebnissen von Alphapose und Openpose

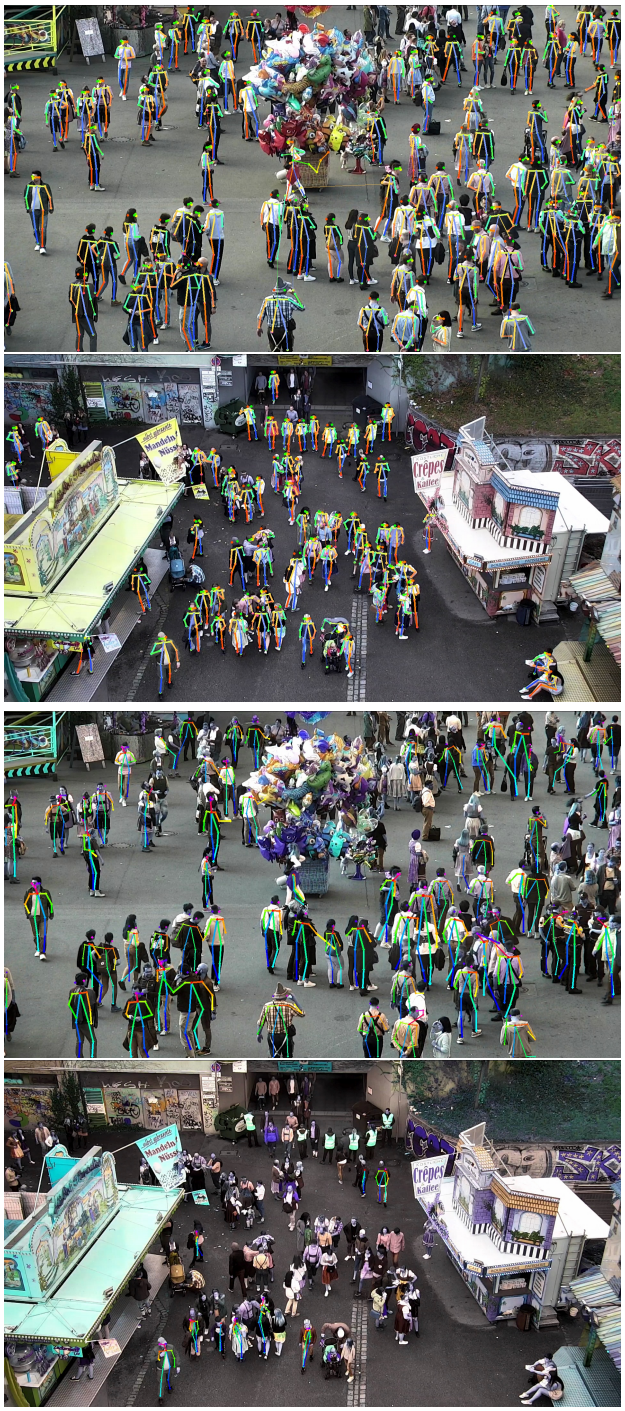


Abbildung 3: Eine Auswahl an Bildern aus dem Datensatz CrowdPose, nachdem Alphapose und Openpose darauf angewandt wurden. Die Abbildungen sind die fertig gerenderten Bilder aus den Frameworks. Die beiden oberen Bilder stellen die Ergebnisse von Alphapose, die unteren die Ergebnisse von Openpose dar. [10] [3] [15]

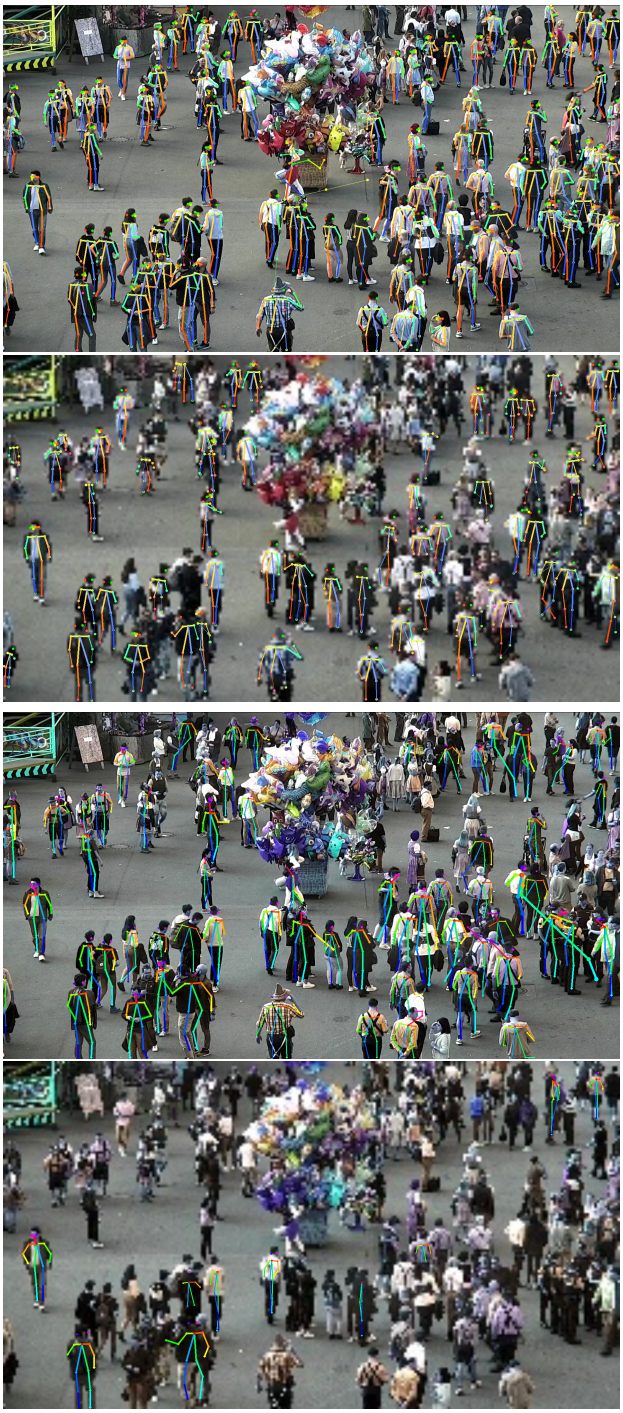


Abbildung 4: Ein Beispielbild aus dem Datensatz CrowdPose, welches mithilfe der Verfahren Alphapose (oben) und Openpose (unten) ausgewertet wurde, bei Originalgröße (jeweils oben) und 1/8 der Bildhöhe und -breite (jeweils unten). Zu erkennen ist, dass im unteren Bild die Menge der erkannten Personen viel geringer ausfällt, als im oberen Bild. Dies trat besonders bei den Untersuchungen von Openpose auf. [3] [15]

4 Zeitmessung

Bild	Auflösung	Personen	Skalierung									
			1/1		1/2		1/4		1/8			
			Alphap.	Openp.	Alphap.	Openp.	Alphap.	Openp.	Alphap.	Openp.		
1	Full HD	141	1,52	0,65	1,44	0,65	1,40	0,57	1,01	0,23		
2		124	1,38	0,61	1,36	0,54	1,32	0,51	0,92	0,23		
3		136	1,59	0,79	1,46	0,76	1,41	0,64	1,11	0,25		
4		148	0,77	0,21	0,73	0,21	0,64	0,21	0,13	0,2		
5		58	0,74	0,31	0,64	0,29	0,6	0,28	0,19	0,21		
6		90	1,05	0,23	0,92	0,23	0,73	0,21	0,20	0,21		
7	WUXGA	7	0,05	0,23	0,05	0,21	0,05	0,23	0,05	0,21		
8		7	0,05	0,22	0,07	0,22	0,05	0,21	0,05	0,21		
9		7	0,06	0,21	0,05	0,21	0,05	0,21	0,06	0,21		
10		7	0,05	0,23	0,05	0,21	0,05	0,21	0,06	0,21		
11		8	0,05	0,22	0,06	0,21	0,06	0,21	0,16	0,21		
12		7	0,05	0,21	0,06	0,22	0,05	0,21	0,07	0,21		
13		6	0,07	0,22	0,06	0,22	0,08	0,22	0,08	0,21		
14		8	0,16	0,22	0,15	0,21	0,13	0,21	0,15	0,21		
15		8	0,14	0,22	0,13	0,21	0,16	0,21	0,11	0,22		
16		8	0,05	0,21	0,07	0,21	0,06	0,21	0,07	0,21		
17		6	0,14	0,21	0,15	0,21	0,11	0,21	0,12	0,21		
18		6	0,06	0,21	0,07	0,21	0,07	0,21	0,05	0,21		
19		7	0,05	0,21	0,09	0,21	0,06	0,21	0,07	0,21		
20		8	0,06	0,22	0,05	0,21	0,06	0,21	0,04	0,21		
21		8	0,08	0,21	0,07	0,21	0,07	0,22	0,07	0,21		
22		6	0,05	0,22	0,06	0,21	0,06	0,21	0,06	0,21		
23		8	0,05	0,21	0,06	0,21	0,07	0,22	0,05	0,20		
24		7	0,09	0,21	0,07	0,21	0,11	0,21	0,07	0,21		
25		7	0,06	0,22	0,06	0,21	0,05	0,21	0,05	0,21		

Tabelle 2: Die Ergebnisse der Zeitmessung der Verfahren Alphapose und Openpose in Abhängigkeit der Skalierung der Bilder in Sekunden. [10] [3]

Literatur

[1] Mykhaylo Andriluka et al. *2D Human Pose Estimation: New Benchmark and State of the Art Analysis*. IEEE, 2014.

[2] Gary Bradski und Adrian Kaehler. *Learning OpenCV, Computer Vision with OpenCV Library; software that sees*. O'Reilly Media, 2008.

[3] Zhe Cao et al. *Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields*. CoRR, 2017.

[4] Yilun Chen et al. *Cascaded Pyramid Network for Multi-Person Pose Estimation*. CoRR, 2017.

[5] François Chollet et al. *Keras*. GitHub, 2015.

[6] Ronan Collobert, Samy Bengio und Johnny Marithoz. *Torch: A Modular Machine Learning Software Library*. Idiap, 2002.

[7] cv:hci. *A Universal Labeling Tool: Sloth*. Github, o. J.

[8] John Duchi, Elad Hazan und Yoram Singer. *Adaptive Subgradient Methods for Online Learning and Stochastic Optimization*. JMLR.org, 2011.

[9] Matteo Fabbri et al. *Learning to Detect and Track Visible and Occluded Body Joints in a Virtual World*. CoRR, 2018.

[10] Hao-Shu Fang et al. *RMPE: Regional Multi-person Pose Estimation*. CoRR, 2017.

[11] Ian Goodfellow, Yoshua Bengio und Aaron Courville. *Deep Learning*. MIT Press, 2016.

[12] Kaiming He et al. *Deep Residual Learning for Image Recognition*. CoRR, 2016.

[13] Eldar Insafutdinov et al. *ArtTrack: Articulated Multi-person Tracking in the Wild*. CoRR.

[14] Eldar Insafutdinov et al. *DeeperCut: A Deeper, Stronger, and Faster Multi-Person Pose Estimation Model*. CoRR, 2016.

[15] Fraunhofer IOSB. *IOSB Wareneingang und Cannstatter Wasen Bilder*. IOSB, 2017.

[16] Yangqing Jia et al. *Caffe: Convolutional Architecture for Fast Feature Embedding*. CoRR, 2014.

[17] Diederik P. Kingma und Jimmy Ba. *Adam: A Method for Stochastic Optimization*. CoRR, 2014.

[18] Yann LeCun. *Deep learning tutorial*. Citeseer, 2013.

[19] Yann LeCun, Yoshua Bengio und Geoffrey Hinton. *Deep learning*. Nature Publishing Group, 2015.

[20] Tsung-Yi Lin et al. *Feature Pyramid Networks for Object Detection*. CoRR, 2016.

[21] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context*. CoRR, 2014.

[22] Zachary C. Lipton et al. *Overfitting and regularization - The straight Dope 0.1 documentation*. Github, 2018.

[23] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. Github, 2015.

[24] Michael A. Nielsen. *Neural networks and deep learning*. Determination Press, 2015.

[25] George Papandreou et al. *Towards Accurate Multi-person Pose Estimation in the Wild*. CoRR, 2017.

[26] Adam Paszke et al. *PyTorch*. GitHub, 2016.

[27] Leonid Pishchulin et al. *DeepCut: Joint Subset Partition and Labeling for Multi Person Pose Estimation*. CoRR, 2015.

[28] Shaoqing Ren et al. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. Curran Associates, Inc., 2015.

[29] Günter Daniel Rey und Karl F. Wender. *Neuronale Netze: Eine Einführung in die Grundlagen, Anwendungen und Datenauswertung*. Huber Bern, 2008.

[30] Frank Seide und Amit Agarwal. *CNTK: Microsoft's Open-Source Deep-Learning Toolkit*. ACM, 2016.

[31] Theano Development Team. *Theano: A Python framework for fast computation of mathematical expressions*. CoRR, 2016.

[32] Unbekannt. *Human Body Anatomy Outline Printable for Kids - Health Token*. ClipArt Best, o. J.

[33] Srenivas Varadarajan, Parual Datta und Omesh Tickoo. *A Greedy Part Assignment Algorithm for Real-time Multi-person 2D Pose Estimation*. CoRR, 2017.

[34] Yuliang Xiu et al. *Pose Flow: Efficient Online Pose Tracking*. CoRR, 2018.

Tabellenverzeichnis

Abbildungsverzeichnis