Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

AQAPRE

A research work on Action Quality Assessment of Physiotherapy Rehabilitation Exercises

Pragya Madaan

M.Sc. Computer Science (Future Networked Systems)

Supervised by Prof. Aljosa Smolic

Co-supervised by Mr. Richard Blythman

A Dissertation submitted to the University of Dublin, Trinity College
in fulfilment of the requirements for the degree of
Masters in Computer Science (Future Networked Systems)

Declaration

I, Pragya Madaan declare that this thesis has not been submitted as an exercise for a degree at this or any other university and it is entirely my own work.

I agree to deposit this thesis in the University's open access institutional repository or allow the library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

PRAGYA MADAAN

Acknowledgment

I am highly grateful to the almighty for being a source of sustained intrinsic motivation and showering constant blessings upon me despite the thrilling and much demanding journey throughout my dissertation period. No words can be enough to express my deep sense of gratitude to the wonderful support network I had besides me.

I would particularly like to express my sincere thanks to my supervisor Prof. Aljosa Smolic and co-supervisor Mr. Richard Blythman. Their guidance, availability and indispensable technical knowledge has made me capable to start afresh with this emerging area of research and exponentially enhance my knowledge curve. They have been a continuous source of inspiration and motivation to me, not only for the dissertation, but throughout my academic year. I am also very thankful to Dr. John Dingliana, the Course Director and Dr. Melanie Borouche, my strand leader, for being so approachable and understanding at all times.

This indelible experience of pursuing research in a new challenging area of work whilst moving between countries, dealing with changes in physical and psychological conditions, managing through COVID circumstances is unreal without the consistent support and love from my family- mom, dad and Sakshi. Also, I am very much grateful to my friends who always stood by me just like family, throughout my journey. I feel so lucky to be a recipient of their everlasting love and endorsements.

Pragya Madaan

Abstract

While the onset of the pandemic scenario had moved the world on a precarious perch, technology has by far proved itself to be highly effective to ensure the betterment of humankind by providing cutting-edge solutions in every possible field. Now, more than ever before, it is vital to look after both physical and psychological health. Although many of the doctor's consultations have moved to virtual platforms, there do exist many more possibilities which can enhancing better ways for some forms of virtual treatments such as physiotherapy rehabilitation. One of the emerging technological areas of deep learning have been observed to assist significantly while on a lookout for solutions to such problems relating to image classification, human pose estimation etc.

Despite the ongoing work of many big technology giants on human pose estimation, there is only one well-known research published back in 2020 to utilise deep learning for action quality assessment of physiotherapy rehabilitation exercises. This dissertation has made an attempt to gain particular knowledge for advancing that research in a way to establish a possible pipeline for action quality assessment on camera-captured videos. This research is split majorly into 2 phases, namely, pose estimation (for test data preparation) and model training (for preparing the model to take test data as input).

For the same, a review of the previous work done in the field of physiotherapy, human pose estimation, deep learning is made and a framework for preparing test data by 2D pose extraction to 3D pose estimation of smartphone captured video datasets has been established by using HRNET with COCO dataset and VideoPose3D inference using Detectron library. The mapping of the obtained coordinates to 3D joints is then examined wherein it is observed that the final test data preparation can only be accomplished with licensed access to Human3.6M in order to utilise the actual inferences of VideoPose3D with Human3.6M.

On the other hand, the recent work on action quality assessment with 2D markers has been reestablished with their existing UIPRMD dataset of squats for understanding and then enhancements to the work have been made from scratch to ensure that it takes into account the 3D joint positions for action quality assessment. The obtained results have shown that the model can take 3D joint positions as an input while providing near-to-similar results as for the one with 2D markers.

Note: The reader of the research document is requested to view the document in Acrobat Reader in order to have an insight to the animated gif files which are added in the last chapters.

Pragya Madaan

Contents

List of Figures

Chapter 1

Introduction

This chapter lays the background, problem statement, motivation and objectives of the undertaken research.

1.1 Background

Physiotherapy and rehabilitation are considered essential for the postoperative recovery of individuals. However, it is infeasible and economically unjustified to offer patient access to a clinician for every single rehabilitation session [26]. Moreover, given the times when COVID-19 made it difficult to follow physio-therapeutic sessions as the term "social-distancing" evolved around the same time leading to the need to have a virtual physiotherapy session in place was recognized, and demand to develop the desired infrastructure was put forward. Accordingly, current healthcare systems around the world are organized such that an initial portion of rehabilitation programs is performed in an inpatient facility under direct supervision by a clinician, followed by a second portion performed in an outpatient setting, where patients perform a set of prescribed exercises on their own[23]. Today, tele-health is rapidly gaining popularity, and physiotherapy treatments are frequently conducted via live video streaming. By providing real-time data on the movement of the human body, smart video analysis employing deep learning models (such as human posture and shape estimate) has the potential to improve physiotherapists' effectiveness. There is very limited work being

done in this area of research. A motion capture laboratory recently collected a dataset of physiotherapy rehabilitation exercises[**?**]. Furthermore, a model that takes a sequence of joint locations as input and predicts a binary output – good or terrible performance – has been constructed just last year[23][22]. The dissertation seeks to advance on the research work carried out and build upon the existing models to improve the accuracy of detecting physio-therapeutic movements of patients and developing insights for a better-detailed report of the patient for the physiotherapist to examine. The dissertation would require training the existing models on a different dataset and evaluate the change in performance. Based on results after varying the dataset, we would examine the role of altering parameters to make the virtual physiotherapy sessions reduce the usage of advanced camera sensors thus making it economical for the patients and the physiotherapist. The technique to reduce the number of advanced sensory capture tools will make the clinical accessibility of rehabilitation centers to more patients thereby reducing the load of the centers. For the same, this project also intends to establish a deep learning based pipeline for automating the quality assessment of rehabilitation exercises (for eg. squats), in order to minimize the burden on physiotherapists even more.

1.2 Motivation

I am very closely aligned with the undertaken research and my major motivations behind pursuing this study in a novel research area is due to the following factors:

- Physiotherapy: Myself, being a sports-player, have been suffering from a chronic ankle ligament rupture which happened a few years back. Since then, I had to move multiple locations for pursuing studies, internship and job. Everytime, moving to another placed posed an additional challenge for finding a good physiotherapist in order to ensure that I'm taking utmost care of rehabilitation for my ankle. During the COVID outbreak after which I began my Masters, the possibility to even visit a physiotherapist got limited. At that time, I realised the need for better options for facilitating quality rehabilitation using technology.

- Deep Learning: Deep learning as a research area has always seemed fascinating to

me, specially due to its capabilities to provide meaningful technological solutions in every possible fields. However, being from Future Networked Systems strand, I could not really find a chance to pursue this area of research before. Therefore, I have been really interested to work on a Deep learning based research during my dissertation period. Furthermore, the problem area and objectives of my research intersect quite closely with Deep Learning as compared to any other technology.

- Action Quality Assessment: Physiotherapy exercises, if not performed correctly, can cause damage to the tendons or ligaments. Therefore, it is highly recommended to perform the same in presence of a physical therapist. Back in 2019, the rehabilitation of my ankle ligament was going quite well. Hence, I joined a gym supposing that now I'll be able to perform squats or lunges quite easily. It was just a matter of few days that I again got an ankle sprain only because I was performing squats with full enthusiasm but without any assessment from the physiotherapist. This took back my whole improvement to the point where I started from. As my dissertation research focuses on advancing the first-ever research done on Action Quality Assessment in 2020, I consider this as my chance to make some contributions in this area.

1.3 Problem Description

What all do we need to consider for developing a deep learning based pipeline for outputting movement quality of physiotherapy exercise videos?

Let us consider that we already have a pipeline, then the problem statement shrinks down to the following:
Inputs: Video of a person performing a physiotherapy exercise (let's say, squat).
Expected Output: "Good squat" or "Bad squat".

However, looking genuinely into the possible solutions, there are a lot more steps involved, which are collectively observed as the objectives of the current research.

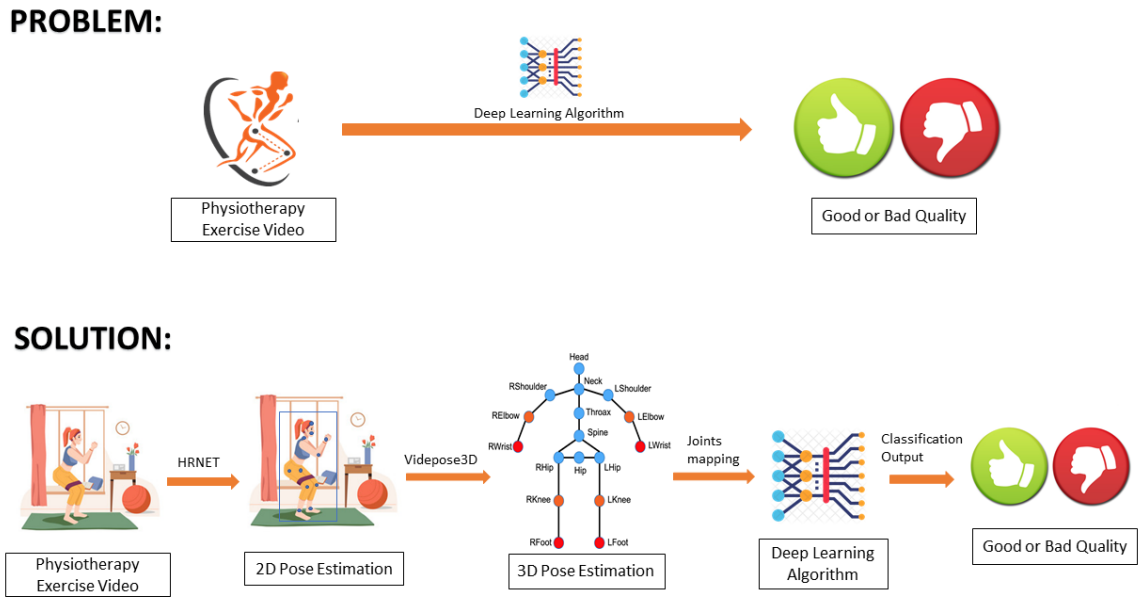


Figure 1.1: Problem Statement and possible Solution

1.4 Research Objectives

- Understanding basics of machine learning, deep learning, neural networks and human pose estimation.
- Gaining knowledge about existing human pose estimation algorithms and data sets for further utilisation in test data preparation.
- Evaluating 2D and 3D pose estimation techniques and devising a framework for camera-video based test data.
- Analysing gathered test data sets, estimating poses and comparing them.
- Re-establishing the Action Quality Assessment 2D markers based research for proper analysis.
- Gathering training dataset and converting 2D Markers to 3D joint positions.
- Enhancing the Action Quality Assessment research to take 3D joint positions as input.
- Preparing joint positions based dataset and training the model.
- Evaluating and comparing the 2D markers based model with the 3D joints based model.

Chapter 2

State of The Art

This chapter lays the current knowledge including substantive findings, as well as theoretical and methodological contributions to the topic of research.

2.1 Theory Associated with Problem Area

It is a widely acknowledged fact that sometimes post-operative recovery becomes more challenging both for physiotherapists and the patient and that too may come with a substantial cost. The solutions to address this problem using machine learning will incorporate the use of video action recognition systems which is widely applied in video indexing, intelligent surveillance, multimedia understanding, and other fields. Recently, it was greatly improved by incorporating the learning of deep information using Convolutional Neural Network (CNN). As CNN is primarily designed to extract 2D spatial features from still images and videos are naturally viewed as 3D Spatio-temporal signals, the core issue of extending the CNN from image to video is temporal information exploitation. A research was conducted by Guangle Yaoa, Tao Lei, Jiandan Zhonga [54] in this regard presenting a comprehensive review of the CNN-based action recognition methods according to these strategies.

A novel framework for the assessment of home-based rehabilitation that encompasses formulation of metrics for quantifying movement performance, scoring functions for mapping the performance metrics into numerical scores of movement quality, and deep learning-

based end-to-end models for encoding the relationship between movement data and quality scores have been proposed in the research carried out by Yalin Liao, Aleksandar Vakanski and Min Xian [23]. It can be widely noticed that efforts are in place to develop and improve the existing infrastructure for performance analysis of sports athletes. Some of the aspects of fine-tuning these methodologies have been captured in research by [59] where the main focus was to improve the estimation of athlete's pose in swimming, which was further optimized using integer linear programming. The joint detection precision of swimmers in this case study improved by 0.8%-4.8% PCK for anti-symmetrical motion and up to 1.8% PCK for symmetrical styles.

2.2 Existing Systems and Solutions

The challenge to bridge the gap in the fields of physiotherapy has been tried to resolve. The research work carried out by Yalin Liao, Aleksandar Vakanski, and Min Xian [23] closely deals with the improvement of telehealth infrastructure. The research work uses the Kalman filters, hidden Markov models, and Gaussian mixture models to develop a novel framework for computer-aided assessment of rehabilitation exercises in parallel to developing a deep Spatio-temporal NN model for outputting movement quality scores and calculating a performance metric that employs probabilistic modeling and autoencoder NNs for dimensionality reduction of rehabilitation data. However, there is some scope of improvement possible primarily in validating rehabilitation exercises performed by healthy subjects, where the measurements are acquired with an expensive optical motion capturing system. Additionally, the largest segment of the validation is based on movement data without a ground truth assessment of the movement quality by clinicians. The evaluation of the deep squat exercise in the KIMORE dataset provides a partial validation of patient data collected with a low-cost sensor.

One of the major improvements might come if a more sophisticated dataset is used. A dataset, namely Human3.6M [16], of 3.6 Million accurate 3D Human poses, acquired by recording the performance of 5 female and 6 male subjects, under 4 different viewpoints, for training realistic human sensing systems and for evaluating the next generation of human pose estimation models and algorithms. It not only provides an increased data set with

additional synchronized images, human motion capture and time of flight (depth) data, and accurate 3D body scans of all the subject actors involved. The incorporation of a new dataset to train the existing models might yield better results and hence provide a sense of direction to proceed into for better results. Another probable inclusion of the golf dataset might be useful in effectively dealing with some of the recoveries that require aerobic activity as a part of rehabilitation like Spondylolisthesis. GolfDB: A Video Database for Golf Swing Sequencing [31] which consists of 1400 high-quality golf swing videos, each labeled with event frames, bounding box, player name and sex, club type, and view type. However, the use of this dataset is limited in our project and may hamper the efficiency of the model. Hence, it becomes important to evaluate certain parameters which indicate the usefulness of training of this dataset.

One of the major limitations of earlier developed models was the use of expensive motion sensory tools to judge a person's movement. However, a study was conducted by [20] which evaluates the feasibility of quantitative movement analysis using single-camera videos which is most likely to overcome the shortcomings of previous research work carried out in these domains. Quantitative assessment of motion is critical to medical decision-making, especially in the telehealth domain. A method was proposed to predict clinically relevant motion parameters from an ordinary video of a patient. The machine learning models predict parameters include walking speed, cadence, knee flexion angle at maximum extension, and Gait Deviation Index (GDI), a comprehensive metric of gait impairment. These methods for quantifying gait pathology with commodity cameras increase access to quantitative motion analysis in clinics and at home and enable physiotherapists to evaluate the progress of the patient.

Chapter 3

Physiotherapy and Deep Learning

3.1 Physiotherapy

3.1.1 Conventional Gait Model

The conventional Gait Model (CGM) has been the mechanical model applied in doing clinical gait analysis and strength is that it is easily understandable by non-experts in the field of biomechanics. However, it has been observed that the extensive application of CGM has led to the identification of certain criticism such as there is lack of accuracy and effective alignment in the systemic positioning of the body's thigh and shank segment as per the system's wand-mounted markers [44]. The latter has resulted in larger errors while defining the coronal planes of the segments. Conventional Python CGM (PyCGM) runs through the command line or the direct calling of the functions. The credentials for the setup of the joint angle calculations can be done directly or by loading the models and executable files that contain the functions. As a result, the required data gets stored in the established Python Dictionary. There are certain steps through which the PyCGM works and this starts with passing the python terminal and the PyCGM is segmented first and later aligned with a much larger Python managed code and line of information [44]. Through static and dynamic trials, data is uploaded and managed with detailed subject specifications and the measurements. In the absence of static offsets, data is collected from static trials. There are various frames and alterations of the dynamic trial methods and iterations which are

segmented and distributed into specified cores. Each core receives static offsets and data is saved using a specific format.

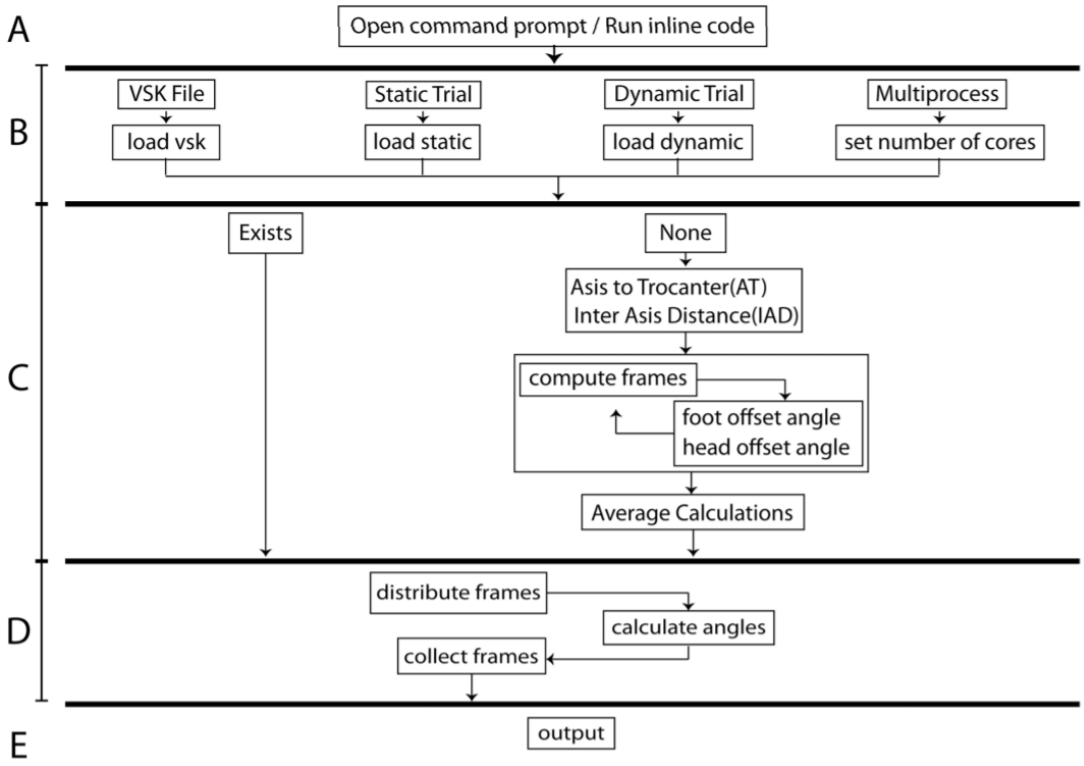


Figure 3.1: PyCGM process

3.1.2 Parallelization

Parallelization is performed in the next stage through the help of three main parts of CGM which starts with the static trial, and then moves on to the dynamic trial which is more feasible and applicable. It is finally concluded with the subject measurements for the required model. For parallelization, a static calibration is designed and developed with the estimated values that further act as the main constants in the specified and focused estimation functions [44]. The figure below depicts the process of parallelization and from the first row; the data is transferred to all the targeted cores. Specifically, in between 2 and 6, data is estimated which decreases the number of usable cores to make the systems streamlined and is achieved as per the tasks are completed and executed. This is represented in the following manner:

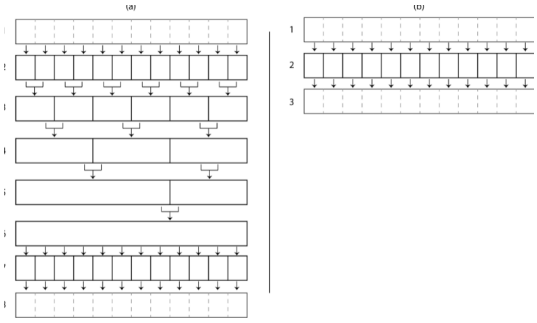


Figure 3.2: Parallelization

3.1.3 Placement of Markers

The Conventional CGM model has been upgraded and pyCGM2 is developed in which there are defined segments of the autonomy and also is possible to deter the position as well as the orientation of the segments. Three points are taken to identify a segment and assuming a ball and socket joint. It implies that one of these points is the joint within the definition of the proximal segment of the targeted model. It has been observed that certain anthropometric measurements need to be done so that location of the joint centres within the segment can be identified. It is done through the help of wand markers in the femur and tibia segments.

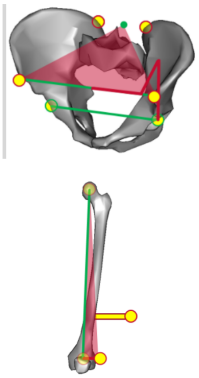


Figure 3.3: Placement of markers in different body parts

3.1.4 Python Conventional Gait Model

The paper, “The effect of subject measurement error on joint kinematics in the conventional gait model: Insights from the open-source pyCGM tool using high-performance computing methods" has focused on using Python code for the development of the CGM. It is one of the most popular and easily executable biomechanical models in the Python environment. CGM has been observed to depend on the retro-reflective markers placed along with certain anatomical landmarks [44]. The python coded or operated CGM is a free platform that has been useful in doing the computational part easily. With the paper, reproduction of joint kinematic outputs from Vicon CGM has been possible and it has been observed that in a parallel approach the integration is done with a high-performing computer [44]. It has been observed from the estimated joint angles through pyCGM that the results are in agreement with Vicon CGM outputs. The maximum difference has been identified in the joint angle difference which is less than 10 raise to power -5 degrees. The PyCGM system is represented in the following manner:

Table 2. Joint angle differences between Vicon CGM and pyCGM for the ROM file.

Lower	**X**	**Y**	**Z**	**Maximum**
R Pelvis	2.66359E-06	3.31919E-06	4.27735E-06	**0.000004**
L Hip	6.41988E-06	-1.37459E-05	2.6727E-05	**0.000027**
R Hip	5.38463E-06	-1.08932E-05	5.17191E-05	**0.000052**
L Knee	1.1362E-05	3.41015E-05	1.14514E-05	**0.000034**
R Knee	1.07576E-05	4.18676E-05	1.25827E-05	**0.000042**
L Ankle	-1.55859E-05	8.80732E-06	1.40949E-05	**0.000014**
R Ankle	-2.91199E-06	2.5783E-05	1.54074E-05	**0.000026**
L Foot Progress	5.03701E-05	4.05533E-05	8.8887E-06	**0.000050**
R Foot Progress	2.77097E-05	1.27706E-05	9.09723E-06	**0.000028**
Maximum	**0.000050**	**0.000042**	**0.000052**	
Upper	**X**	**Y**	**Z**	
L Shoulder	4.26187E-06	5.26738E-06	1.16389E-05	**0.000012**
R Shoulder	4.27861E-06	5.75079E-06	6.06542E-06	**0.000006**
L Elbow	7.35977E-06	6.552E-12	6.551E-12	**0.000007**
R Elbow	8.83274E-06	7.974E-12	7.603E-12	**0.000009**
L Wrist	3.99468E-06	3.60593E-06	9.05932E-06	**0.000009**
R Wrist	2.37015E-06	3.46521E-06	1.08998E-05	**0.000011**
R Spine	2.88501E-06	3.3882E-06	3.33372E-06	**0.000003**
R Thorax	4.75358E-06	1.69255E-06	4.253E-06	**0.000005**
R Neck	8.83274E-06	7.974E-12	7.603E-12	**0.000009**
R Head	3.3013E-05	1.40494E-05	2.51925E-05	**0.000033**
Maximum	**0.000033**	**0.000014**	**0.000025**	

Figure 3.4: PyCGM system

The required markers are allocated over the defined functions of the ASIS as well as the PSIS in order to establish a plane of the pelvis that contains the anatomical landmarks of the

model. It is also aligned with the lateral femoral epicondyle as well as on the wand of the lateral thigh. This helps the two markers to get aligned that allows the hip joint centre to come in direct alignment with the coronal plane of the model. The marker is also allocated on the subject's forefoot and another one is allocated on the posterior of the heel in order to carry out non-cognitive static trials in the model. Code style of PyCGM is developed using a Harrington method and is depicted in the following manner:

```
if vsk['Harrington'] == 'Yes':
    pw = vsk['InterAsisDistance']
    pd = vsk['PD']

    L_Xh = -0.24 * pd - 9.9
    L_Yh =  0.30 * pw + 10.9
    L_Zh = -0.33 * pw - 7.3

    R_Xh =  -0.24 * pd - 9.9
    R_Yh = ( 0.30 * pw + 10.9 ) * -1
    R_Zh =  -0.33 * pw - 7.3
```

Figure 3.5: Harrington Method to develop PyCGM Code

3.2 Deep Learning

3.2.1 Machine learning

Machine learning can be defined as a branch of AI or artificial intelligence which emphasises the use of data and algorithms for imitating the best humans learn and gradually improve their accuracy. Machine learning also focuses on identifying the correlation among big data. Machine learning operates with ordinary differential equations and partial differential equations. Machine learning is used with logistic regression as well. While using machine learning in logistic regression, the equation can be changed from P(x) = e $\hat{}$(b0 +b1x) / (1 + e(b0 + b1x)) to ln(p(x) / 1-p(x)) = b0 + b1x. In the current era, there has been increased consideration of scientific machine learning. According to [39], Physics informed neural networks or PINNs use partial differential equations in the cost functions of the neural networks in order to incorporate prior scientific knowledge. This is a major advantage of merging differential equations with the concept of machine learning.

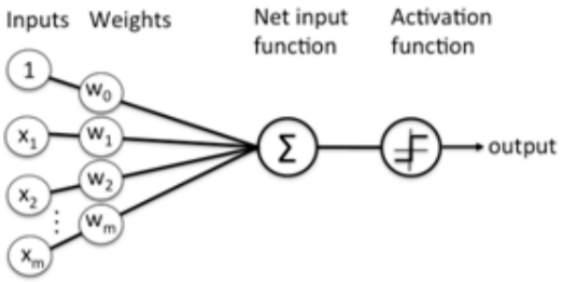


Figure 3.6: Neural Networks with machine learning, [27]

The inclusion of machine learning in businesses has significantly helped in making informed decisions in the long run. According to Lipton (2018), rapid progress in machine learning has contributed to the deployment of automated decision processes. The algorithm of machine learning takes input and predicts the corresponding output. Interpretability of machine learning is one of the major benefits. Moreover, [27] argued that neural networks also use machine learning. This network is a series of algorithms that focuses on recognising the existing connections and relationships in a provided dataset through a procedure that can replicate the human brain functions and operations. However, Machine learning algorithms have the possibility of high error.

3.2.2 Deep learning

Deep learning can be considered as a subpart of the larger machine learning environment and technology which is focused on algorithms inspired by the procedures and structural functions of the brain. These structural functions and the related connections are part of the artificial neural networks. According to [42], the deep learning approach is dependent on a neural network and it is also used in predicting different kinds of IDS attacks. An ID is an Intrusion Detection System that helps in identifying malicious activities in a network. As deep learning focuses on computational functions for representing any data or information, it is applied in IDSs.

Computational graphs of deep learning can be presented mathematically, while computing the gradients of the cost function J, as per the benchmarks and parameters, W and b, the equation will be,

For a given parameter α, we set $d\alpha^{[i]} = \frac{\partial J}{\partial \alpha^{[i]}}$ and we have at the i^{th} layer:

$$dZ^{[i]} = dA^{[i]} \star \psi'^{[i]}(Z^{[i]})$$
$$dA^{[i-1]} = {W^{[i]}}^T dZ^{[i]}$$
$$dW^{[i]} = dZ^{[i]} A^{[i-1]}$$
$$db^{[i]} = dZ^{[i]}$$

Figure 3.7: Equation of computational graph of deep learning [32]

Among the derivatives of deep learning, there is the consideration of convolutional neural networks or CNN, Multilayer perceptron or MLP and Recurrent Neural Networks or RNN. According to [4], the Multilayer perceptron is constituted of a singular output layer within the system and its interdependencies. These layers are conceptualised and made with the help of neurons which serve as the central information processing units. Deep learning is helpful in ensuring that the multilayer perceptron is effective. As per [55], the increasing computational resources, as well as RNNs and CNNs, have currently generated substantial development in the domain of deep learning. Deep learning possesses the ability to crawl and extract effective representations from existing information to create precise and accurate models. However, deep learning requires a fairly large and represented dataset to provide effective results.

3.2.3 Neural Networks

Neural networks are considered as the subset of machine learning and these are the core factors of the algorithms for deep learning. Neural networks can be known as simulated neural networks (SNNs) as well as artificial neural networks (ANNs). This system is used to mimic the way of working of the biological neurons in which the neuron is used to pass the signal from one neuron to another. This has been structured from the circuit of neurons and the composition is done with the artificial nodes. As stated by [7], a simple neural network used to have three basic layers which form its neural schema. These layers are called the input layer, followed by the hidden layer and finally the output layer. Mathematical and computational models are used in these artificial neural networks by which processing of information have been done. In most cases, the artificial neural network changes the structure depending on the internal and external information which is needed to flow through

the overall networking system [2].

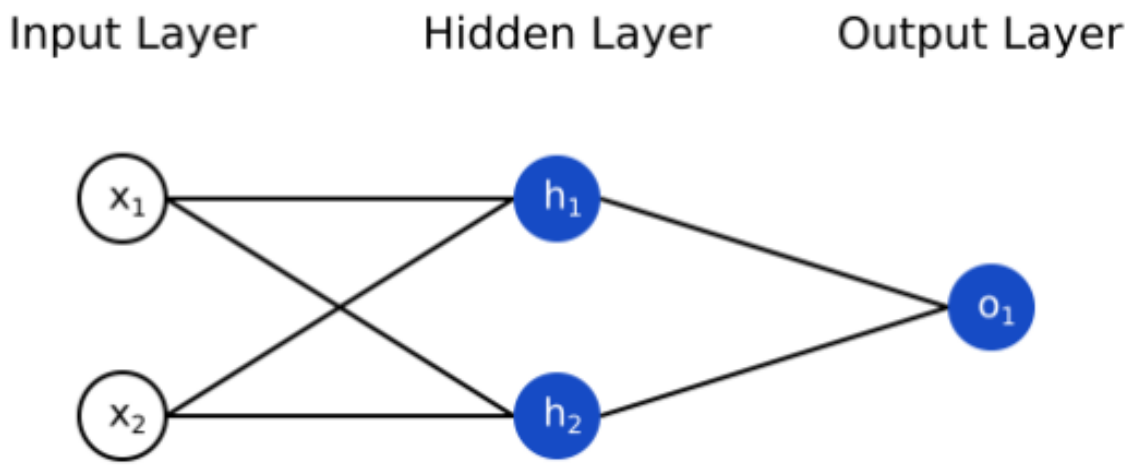


Figure 3.8: Structure of a simple neural network

The use of neural networks has been seen in different areas. In the field of functional approximation, neural networks have been used with the inclusion of modelling and time series prediction. Moreover, sequence and pattern recognition can be done with the help of the neural network for the classification process. In addition to this, sequential decision making and novelty detection can also be done [28]. The key role of neural networks is the processing of data. Clustering, filtering, separation of blind signal and compression functions can be managed by the neural network for the data processing. There are three main types of neural networks which are currently used across deep learning frameworks. These are namely the Artificial Neural Network (ANN), followed by the precisely functioning Convolution Neural Network (CNN) and finally the data pattern oriented Recurrent Neural Network (RNN). These three neural networks are managing the core process of deep learning [10]. In machine learning, there are four stages by which data processing is managed. These steps are the input, extraction of the features, classification and the output. However, the process of deep learning is slightly different. In deep learning, the neural networks manage both the feature extraction function and the classification. In this context, deep learning has three stages which start from the input design and manipulation, followed by the feature extraction and classification and finally, the output modelling or representation.

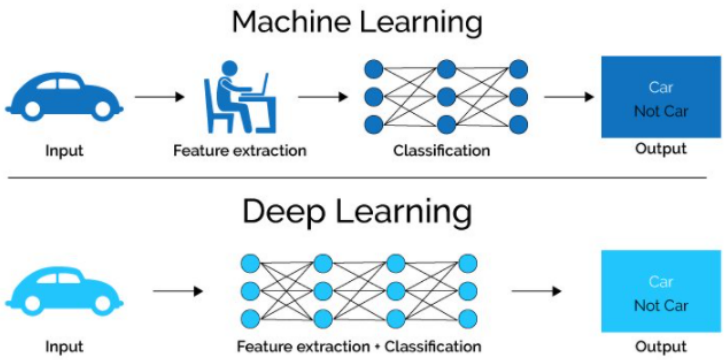


Figure 3.9: Difference between machine learning and deep learning

Artificial Neural Network (ANN)

Artificial neural networks used to have multiple perceptrons in every layer of the system. Another term in which this neural network is called is Feed-Forward neural network. This is the neural network process of input is done in a single direction [5]. Similar to the simple neural network architecture, the ANN also has three simple layers in the networking system which includes the input layer, then the hidden layer and finally the output modelling or generating layer. In the hidden layer, the data processing is happening and the output shows the result after the processing of the data taken from the input. This networking process is used for resolving the issues of the text data, image data and tabular data. The ANN is helpful for the management of any nonlinear function [41]. This is also known as the Universal Function Aproximator for its activation function. The relationship between the input and output can be learned with the ANN. Besides the advantages of the ANN, there are several disadvantages which are seen in the ANN. The first step of the ANN process is to transform the 2-dimensional image within the system into an overt and segmented 1-dimensional vector. In this process, the size of the image is used to increase drastically which can be stated as the primary limitation of the artificial neural network. Another issue of using ANN in processing image data is that this neural networking system loses the spatial feature of the image which has been imputed for the data processing.

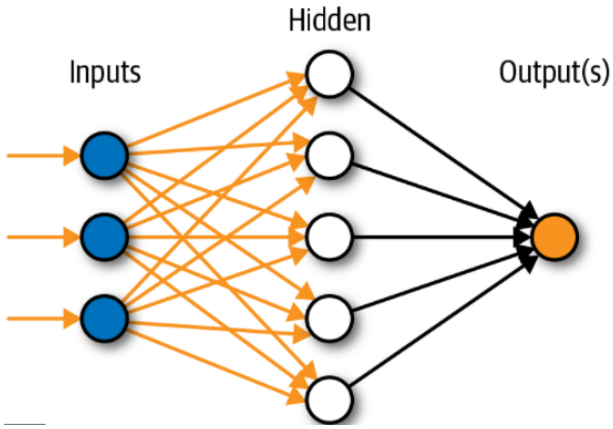


Figure 3.10: Artificial Neural Network

Convolution Neural Network (CNN)

The use of convolution neural networks is a trend in video and image processing projects. This is an algorithm of deep learning and this takes images as input and this is helpful for assigning the importance of different factors of the image and differentiating all the factors from each other. In comparison to the other algorithm for classification, the pre-processing required lower effort in the CNN [49]. The CNN building blocks are known as Kernels and these facilitate people to extract different kinds of features from the imputed image with the help of the convolution operation. In the context of analysing visual imagery, this class of deep neural networks is used. As opined by [50], the advantage of CNN is that the filters can be done automatically with the system of the CNN. The filters usually extract the image features which are relevant and right from the imputed image. In comparison to the ANN network, the spatial features of the image file that is rendered with the help of the CNN are not lost. The relationship between pixels is managed with the CNN and the objectives can be identified properly with the help of the spatial features of the images. Location of the factors and objects in the image and their relationship with the other objects of the image can be gained with the CNN [43].

The concept of sharing the parameters is followed by CNN. The feature map is produced with the help of a single filter. The difference between the ANN and CNN is showing that all the neurons are connected in the neural networking system. On the other hand, in the context of the CNN, only the last layer is connected to others. Furthermore, this has seen

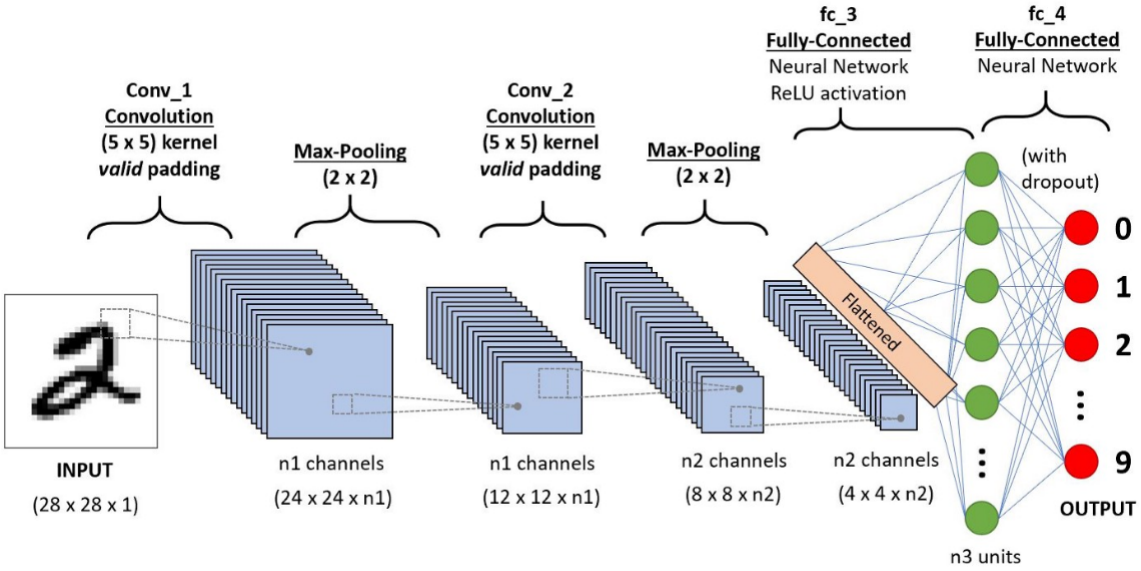


Figure 3.11: Convolution Neural Network

that convolutional neural fabrics and the CNN which are interlinked failed in the production of high-quality segmentation if lack of design has been seen in every sub-network and the fusion in the multi-scale [40].

Recurrent neural networks (RNN)

Recurrent neural networks are different from the ANN and CNN as this networking system uses time-series data and sequential data. The temporal dynamic behaviour can be exhibited with the help of this networking process. Google voice search and Apple's Siri are two main key platforms which are using this kind of networking process [38]. This algorithm of Deep learning has internal memory and this networking process can remember the input data. RNN is suitable for problems related to machine learning which are related to the use of sequential data. Focused and independent functions like the Natural language processing (NLP) and the speech recognition framework (SRF) of the RNN and their related and relevant sequential characteristics are recognized by this networking process. The patterns are used by this system for the prediction of the next scenarios as this tries to remember the input of the previous along with its internal memory [19]. In comparison to the ANN, the RNN networks used to have a loop in their hidden layer. This has changed the network from the feed forwards network and the loop of the network has been seen in the nodes of

the hidden layer. This recurrent connection in the hidden layer of the RNN helps the data processing system in ensuring that the sequential information is captured with the help of the input data [11].

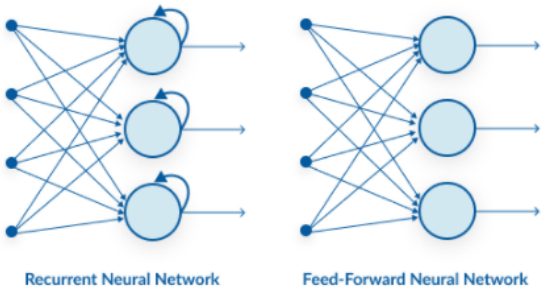


Figure 3.12: Difference between RNN and ANN

The problem related to the multiple data types like audio, text, and time series models can be solved with the help of the RNN. The advantage of the RNN is that it helps in predicting the scenario for the text from the input of the word. This can be done with the help of the Many2Many Seq2Seq model. The prediction is done with the dependency and for the ability to remember the previous data for the internal memory. In theory, it has been observed that the prediction of the next scene can be done by the RNN by focusing on the existing word that is fed in the system as well as on the previously rendered word. On the other hand, parameter sharing is another key function of RNN. The computational cost is decreased for its parameter sharing process and this provides fewer parameters. However, different disadvantages can be seen with the RNN using a system of multiple time steps and the issue of vanishing gradient that are common for different types of neural networks. The Convolutional network is successful for the modelling of temporal information which is used to tackle RNN and with the mechanism of the neural machine translation, speech generation, speech recognition and language modelling. However, in comparison to the RNN, CNN can enable the processing of multiple frames in parallel.

Long Short Term Memory (LSTM)

Long Short Term Memory is the architecture of the artificial RNN and this is widely used in deep learning. This is one of the forms of the recurrent neural network and these are more

suitable in processing, classifying and making predictions on the basis of the series data. Lag for some unknown duration can be seen in this neural networking system between the relevant events of the time series data [**?**]. The function of the LSTM is similar to the RNN. Similar to the RNN this networking system processes the passing of data and propagates the data forwards. However, the difference in the functions of the LSTM cells can be seen for the different operational procedures. This has seen that different operations are applicable in the LSTM which allow this neural networking system to forget as well as to remember the input data. Furthermore, in comparison to CNN, the functionality of LSTM is different from the working process of CNN. LSTM is mostly used for prediction making for the sequence input. On the other hand, CNN is responsible for the exploitation of the spatial correction for the data which are related to the speech input or the image input.

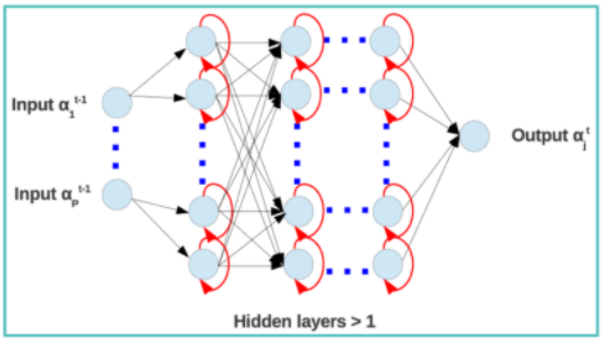


Figure 3.13: Architecture for the LSTM

Some of the use of the LSTM can be seen in robot control, speech recognition like the RNN, prediction of the time series, learning of the rhythm, learning of grammar and many other areas. Along with these, in the context of the recognition of the handwriting and the recognition of the human action LSTM can be used as one of the categories of the RNN network. This can show a more complex area of deep learning in relation to the recurrent neural network [50]

Autoencoders

Autoencoder is one of the most popular types of Artificial Neural Network (ANN) which is helpful for learning unlabeled data coding. This is known as unsupervised learning. Autoencoders learn the representation of the dataset which is known as the encoding system.

Moreover, this is related to the reduction of dimensionality. The training is given to the networking system which can ignore noise such as insignificant data [51] Furthermore, this neural network is related to the process of the raw data representation in a compressed way. An encoder is composed in the autoencoder as well a decoder sub-model is also composed in the autoencoder [8]. Inputs which are given to the encoder are compressed by the encoder in the overall autoencoder system. After the processing of the data, the decoder tries to recreate the input from the compressed version which is provided by the encoder in the overall neural system. Dimensionality is reduced with the help of the autoencoder for the captured data.

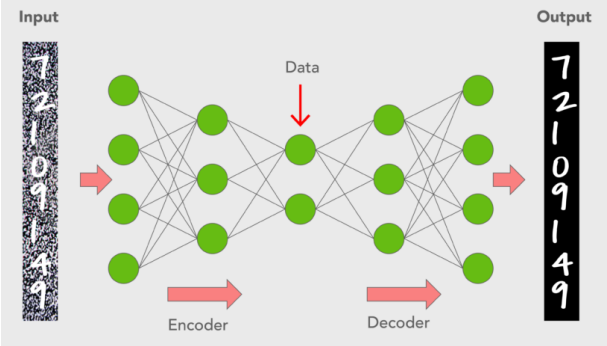


Figure 3.14: The process of working of the autoencoder

For this reason, the autoencoder is considered an unsupervised technique of learning in which people can leverage the neural network for the representation learning task. The basic three components and steps are the encoding, code and decoding of the data [6]. The codes are produced with the help of the encoder in which the input data are coded by compressing activities. After that, the decoder reconstructs the input in front of the user of the data. This neural networking process is different from the other types such as the ANN, CNN and RNN and this networking has a distinct feature of the coding and decoding for the process of the data and information [12].

Chapter 4

Algorithms and Datasets

4.1 Pose Models, Quality Assessment

4.1.1 2D Pose Estimation

The process of 2D pose estimation is the task of detecting the 2D pose of an object which involves orientation and location. It estimates the locality of the key points within the 2D space environment with respect to the targeted image or the video frame as per the requirement pr feasibility of the system. 2D pose estimation in real-time is a critical component that allows the digital systems and machines to visually depict and understand the interpreting humans and their collected process of system interactions and engagements. Hence, a multi-person 2D pose detection enables in including the body, hand, foot as well as the facial key points for the clinical suggestion. 2D pose estimation is also focused on images. in image dependent 2D Pose Estimation, it is assumed that P(x|I) = CNN(I) where CNN is a nonlinear function that returns N 2D heat maps. In this case, use of convolutional pose machines or CPMs. Image variations are also clearly depicted by the robust 2D pose estimation.

It is clear that the 2D pose estimation accompanied with the closest match from the example helps in creating the 3D pose estimation. The 2D pose estimation is predominantly used in real-life situations as well. As per the viewers of [13], addressing the chance of pedestrians

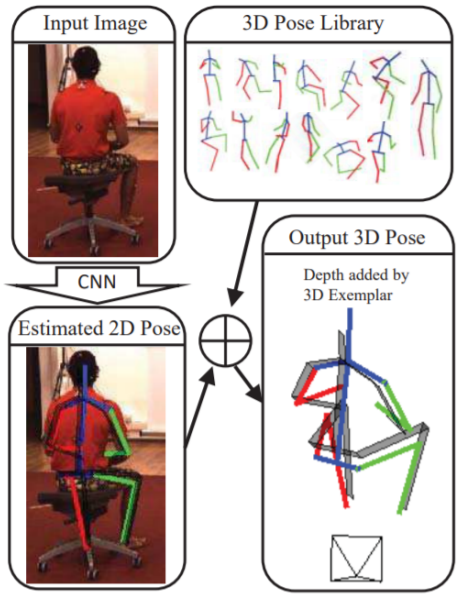


Figure 4.1: Overview of 2D Pose estimation

crossing and not crossing classification can be realised through its image dependent estimation process and techniques. In this case, adopting a sliding window approach can be useful in this case. Moreover, [14] argued that a 2D Pose estimation module includes a deep regression module. They can be useful in locating fractures in joint locations. Hourglass network architecture is applied as the 2D pose estimation module. However, the 2D pose estimation is insufficient in the presence of 3D pose estimation.

Relevance of DeepHRNET

In the discussion of 2D Pose estimation, there is a significant importance of DeepHRNET. The application of DeepHRNET is useful in enhancing human pose estimation. This is often done with a direct focus and approach on the learning dependable functions as well as high-resolution representations (GitHub, 2021). In this regard the overall target is to recover the high-resolution representations that are transferred from the low-resolution representations with the assistance of a high to low flowing resolution network with multiple ranges of linear resolutions only. During the first stage, starting from a high-resolution sub-network can be crucial. The human pose estimation system is dependent on the rapid advancement of the network structure [60]. A normal training schedule from the HRNet is significantly essential in ensuring that networks are maintained effectively. In the discussion of human

pose estimation, deep high-resolution representation is often utilised. Therefore, there is the significant importance of deepHRNET in identifying the challenges and issues prevalent with the human pose estimation. In this case, the convolutional networks have been able to achieve a highly effective performance reaching the state-of-the-art level.

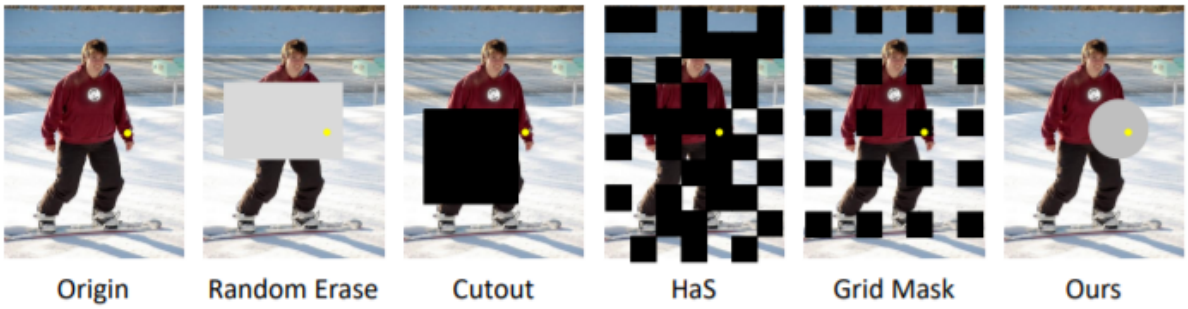


Figure 4.2: Illustration of various information dropping methods [60]

The exploitation of deep learning is possible for effectively modelling the pair wise as well as the unary components and it also imitates the iterative inference procedure. The importance of deep resolution learning lies in the fact that there is a chance of infusing deep learning in the process of simulation. As per the views of [4] deep learning frameworks like the deep high-resolution representation learning is also used in visual recognition. A high-resolution representation is needed for the task that is highly positioning sensitive. The application of DeepHRNET in human pose estimation can be possible by adopting a heat map estimation framework. Therefore, research is conducted with the specified components of the DeepHRNet for semantic human pose segmentation and its related human pose estimation. Therefore, the application of DeepHRNET is also crucial for semantic segmentation. As a result, a semantically richer and more precise visual recognition can be identified. Object detection is also possible in the presence of DeepHRNET.

Deep learning is a major contributor to the DeepHRNET as the main concept of deep learning is used in this. Deep learning is critical in this regard and one of the key aspects of this technology that is also effective in considering DeepHRNET is the application of the algorithms. As per the views of [14], using the EM algorithm for commuting a 3D skeleton by including a sparse dictionary induced from the 2D heat maps. Therefore, using the appropriate algorithms in the DeepHRNET is the most important factor for ensuring the effectiveness of human pose estimation.

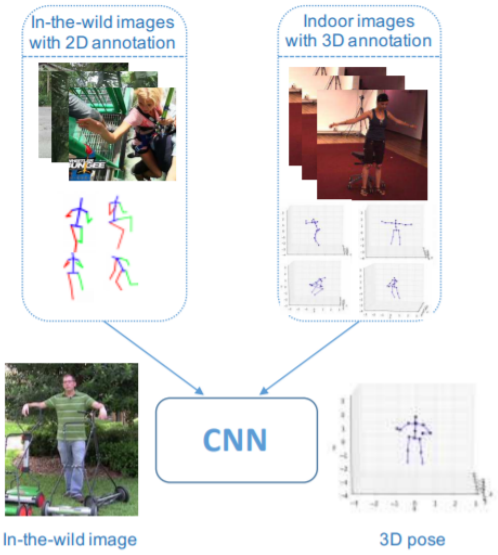


Figure 4.3: Differences of 3D and 2D annotations [14]

Discussion about DeepHRNET

Deep learning is a major contributor to the DeepHRNET which is useful in real-world situations as well as clinical approaches. the deep convolutional neural networks or DCNNs have achieved state-of-the-art results. This is done in cohort research that focuses on computer vision tasks such as object detection, image classification, video semantics and its related semantic segmentation as well as human pose estimation. In the consideration of human pose estimation, DeepHRNET mainly focuses on high-resolution presentations. This helps in finding minor differences in images without high concentration. However,[60] argued that constraints such as relationship in a human pose or the interaction between the environment and the human are significantly crucial. Humans usually use the appearances of the key points. This is majorly intuitive and has been proved to be effective in most cases. The relationship of the environment with humans helps them to locate the key points under issues and challenges such as ambiguity and occlusion.

The DeepHRNET has a major contribution to the closely related representation of learning techniques created for human pose estimation. human pose estimation aims at detecting the locations of the components of the image. In the case of the human pose estimation, the consideration of the time cost of the DeepHRNet for training is similar. In this context,

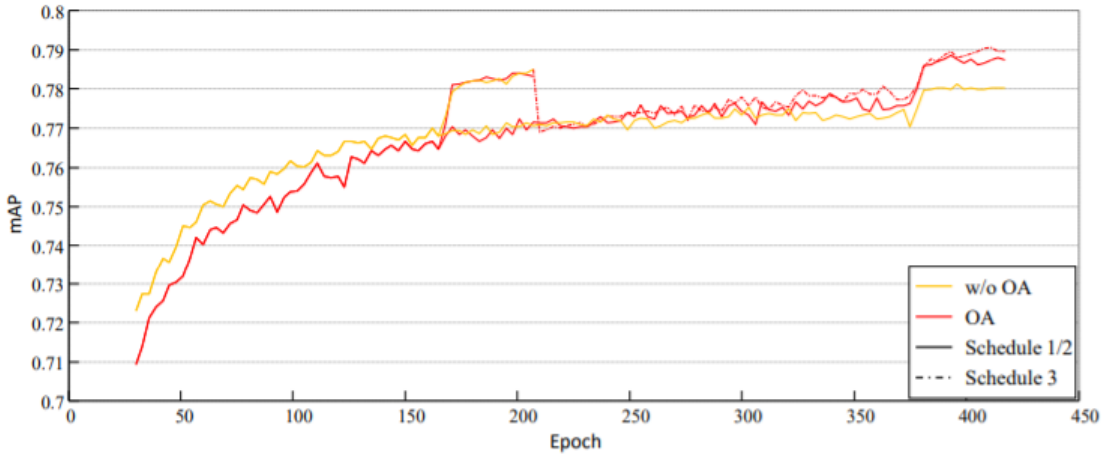


Figure 4.4: The performance of various configurations [60]

considering the complexities of the DeepHRNET is often considered. Furthermore, [14] argued that human pose estimation issues have been heavily studied for computer vision. Deep neural networks have been assisted with 2D human pose estimation, which has contributed to significant success in the current era. On the contrary, deep neural networks are trained on the datasets and they do not generalise well to other environments. There has been increased importance of 3D human pose estimation. However, DeepHRNET with 2D estimation is also effective for providing similar results to the 3D human pose estimation.

Technical architecture of DeepHRNET

The technical architecture of a model plays a major role in ensuring that deep learning is effectively used in the design of DeepHRNET. Technical architecture refers to the way in which computer systems are designed [15]. The development of the technical blueprint is possible if there is an effective technical architecture present. In the technical architecture of DeepHRNET, there is consideration of several important networks. According to [47], in the technical architecture of the DeepHRNET, there is consideration of parallel high-to-low resolution sub-networks. These subnetworks usually work repeatedly with the exchange of information. The vertical and horizontal directions are related to the multi-scale fusion that is further related to the depth of the targeted network and the measurement and tracking scale of the feature maps. Therefore, there is the significant importance of the sub-networks of the network architecture of DeepHRNET.

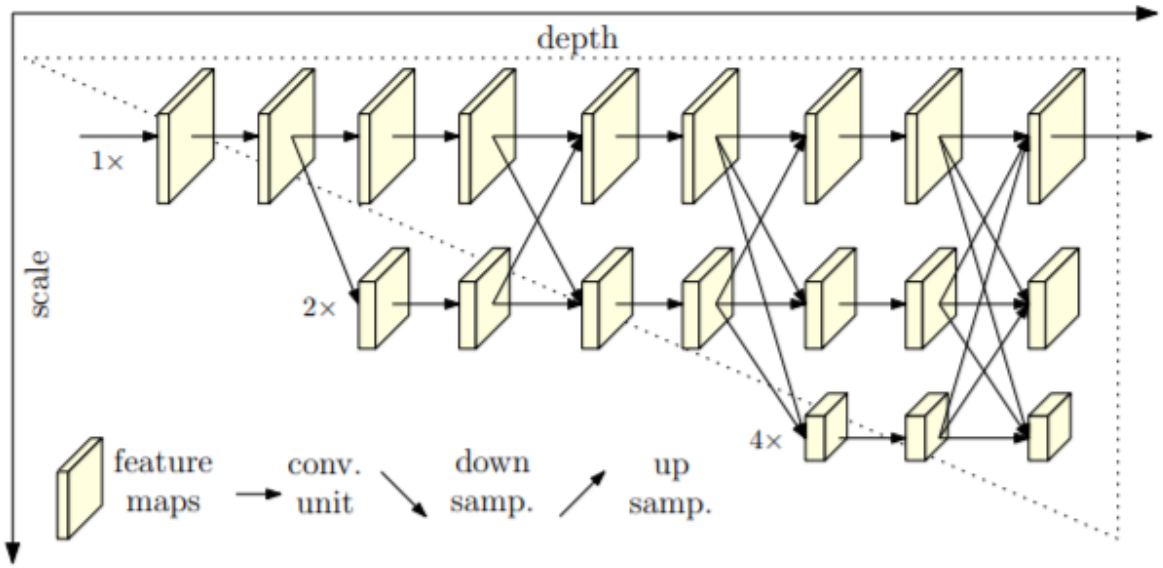


Figure 4.5: Technical architecture of DeepHRNET

In the discussion of the DeepHRNET, there is the significant importance of the novel architecture as well which is called the High-resolution Net. This is an application in maintaining a high-resolution representation in the entire process [47]. In this case, a high-resolution convolution stream is considered as the primary choice before the performance of the high- to low-resolution convolution is considered. The stages in the HRNet are mainly consisting of ascertained modularised blocks. These blocks tend to present a high-resolution image. Furthermore, it has been argued that multi-resolution sub-networks are defined and as per need, added to the main network and the whole process continues. The estimation of the key points over the high-resolution representation output is represented by the network. In the discussion of the DeepHRNET, there has been consideration of the 2D pose estimation module. In presenting more detailed network architecture for the 2D pose estimation modules, it is important to include the state-of-the-art hourglass network architecture.

The network architecture of DeepHRNET resonates with the main idea of the 2D pose estimation. This also contributes to the concept of deep learning. The use of deep learning in the present scenario helps in ensuring that the knowledge of Deep learning has been used for real-world issues and problems. Technical architecture helps in representing the idea of the DeepHRNet more clearly.

DeepHRNET implementation

The paper, "Deep High-resolution representation for Human Pose Estimation" has focused on understanding the human pose estimation issue along with a focus on gaining knowledge on the reliable high-resolution representation. The proposed network has focused on maintaining a high-resolution representation through the entire process [48]. The architecture of the proposed network is HRNet with repeated information exchange throughout the multi-resolution sub-networks. Horizontal as well as vertical directions are adjacent to the network depth and also help in scaling the feature of the maps effectively. The architecture is represented in Figure 4.5.

The approach that has been taken consists of connecting the network to high-to-low subnetworks in parallel and has helped in maintaining a high-resolution representation of the entire process. There has been repeated fusing of the representations produced through the help of high-to-low subnetworks [48]. The apThe DeepHRNET has a major contribution to the closely related representation of learning techniques created for human pose estimation. As per, human pose estimation aims at detecting the locations of the components of the image. In the case of the human pose estimation, the consideration of the time cost of the DeepHRNet for training is similar. In this context, considering the complexities of the DeepHRNET is often considered. Furthermore, [60] argued that human pose estimation issues have been heavily studied for computer vision. Deep neural networks have been assisted with 2D human pose estimation, which has contributed to significant success in the current era. On the contrary, deep neural networks are trained on the datasets and they do not generalise well to other environments. There has been increased importance of 3D human pose estimation. However, DeepHRNET with 2D estimation is also effective for providing similar results to the 3D human pose estimation.proach that has been adopted does not make use of heat map supervision and has been observed to perform better in detecting the key points. There has been the use of sequential multi-resolution sub-networks in which the existing networks for estimating poses are developed by making a connection between various multiple networks in series with varying ranges of high to low resolutions. N(sr) is the sub-network in sth stage and r is the resolution index [48]. It is represented as:

$$\mathcal{N}_{11} \rightarrow \mathcal{N}_{22} \rightarrow \mathcal{N}_{33} \rightarrow \mathcal{N}_{44}.$$

The parallel multi-resolution sub-networks are created by adding one network in the range of the high-to-low resolution and the initiation is also done from a sub-network but with a high-resolution framework. Further steps and stages are conceptualised and multi-resolution sub-networks are created with an addition of an extra lower one. This is represented in the following manner:

$$\begin{array}{ccccccc} \mathcal{N}_{11} & \rightarrow & \mathcal{N}_{21} & \rightarrow & \mathcal{N}_{31} & \rightarrow & \mathcal{N}_{41} \\ & \searrow & \mathcal{N}_{22} & \rightarrow & \mathcal{N}_{32} & \rightarrow & \mathcal{N}_{42} \\ & & & \searrow & \mathcal{N}_{33} & \rightarrow & \mathcal{N}_{43} \\ & & & & & \searrow & \mathcal{N}_{44}. \end{array}$$

There has been the use of repeated multi-scale fusion in which exchange units are introduced throughout the parallel sub-networks. One of the examples has been represented in the following manner:

$$\begin{array}{ccccccccccc} \mathcal{C}_{31}^{1} & \searrow & & \nearrow & \mathcal{C}_{31}^{2} & \searrow & & \nearrow & \mathcal{C}_{31}^{3} & \searrow & \\ \mathcal{C}_{32}^{1} & \rightarrow & \mathcal{E}_{3}^{1} & \rightarrow & \mathcal{C}_{32}^{2} & \rightarrow & \mathcal{E}_{3}^{2} & \rightarrow & \mathcal{C}_{32}^{3} & \rightarrow & \mathcal{E}_{3}^{3}, \\ \mathcal{C}_{33}^{1} & \nearrow & & \searrow & \mathcal{C}_{33}^{2} & \nearrow & & \searrow & \mathcal{C}_{33}^{3} & \nearrow & \end{array}$$

The study has engaged in doing various experiments and one of them is the COCO key-point Detection in which there has been a collection of 2000,000 images and 250,000 different instances of individuals are labelled through 17 key points. The standard evaluation metric has been represented with help of Object Key-point Similarity (OKS) in the following manner:

$$\frac{\sum_i \exp(-d_i^2/2s^2k_i^2)\delta(v_i>0)}{\sum_i \delta(v_i>0)}$$

The human detection box has been extended in height and width so that the dataset can be trained and it is done in the ratio of 4:3. The image is cropped in a manner to discard

the box from the picture framework of the concerned individual and resized to a defined resolution of 256*192 px or 384*288 px as per requirements. It has been observed that the data augmentation consists of random rotation, random scaling as well as flipping . The test result that has been generated reflects that the end output is better than the bottom-up approaches. Moreover, it has been observed that the HRNet-W32 achieves an AP of 74.9, and hence, can be said that it has outperformed all the other top-down approaches. This also validates the fact that the designed framework is efficient when compared as per the framework model size. This has been represented in the following manner:

Table 1. Comparisons on the COCO validation set. Pretrain = pretrain the backbone on the ImageNet classification task. OHKM = online hard keypoints mining [11].

Method	Backbone	Pretrain	Input size	#Params	GFLOPs	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR
8-stage Hourglass [40]	8-stage Hourglass	N	256 × 192	25.1M	14.3	66.9	–	–	–	–	–
CPN [11]	ResNet-50	Y	256 × 192	27.0M	6.20	68.6	–	–	–	–	–
CPN + OHKM [11]	ResNet-50	Y	256 × 192	27.0M	6.20	69.4	–	–	–	–	–
SimpleBaseline [72]	ResNet-50	Y	256 × 192	34.0M	8.90	70.4	88.6	78.3	67.1	77.2	76.3
SimpleBaseline [72]	ResNet-101	Y	256 × 192	53.0M	12.4	71.4	89.3	79.3	68.1	78.1	77.1
SimpleBaseline [72]	ResNet-152	Y	256 × 192	68.6M	15.7	72.0	89.3	79.8	68.7	78.9	77.8
HRNet-W32	HRNet-W32	N	256 × 192	28.5M	7.10	73.4	89.5	80.7	70.2	80.1	78.9
HRNet-W32	HRNet-W32	Y	256 × 192	28.5M	7.10	74.4	90.5	81.9	70.8	81.0	79.8
HRNet-W48	HRNet-W48	Y	256 × 192	63.6M	14.6	75.1	90.6	82.2	71.5	81.8	80.4
SimpleBaseline [72]	ResNet-152	Y	384 × 288	68.6M	35.6	74.3	89.6	81.1	70.5	79.7	79.7
HRNet-W32	HRNet-W32	Y	384 × 288	28.5M	16.0	75.8	90.6	82.7	71.9	82.8	81.0
HRNet-W48	HRNet-W48	Y	384 × 288	63.6M	32.9	**76.3**	**90.8**	**82.9**	**72.3**	**83.4**	**81.2**

Table 2. Comparisons on the COCO test-dev set. #Params and FLOPs are calculated for the pose estimation network, and those for humar detection and keypoint grouping are not included.

Method	Backbone	Input size	#Params	GFLOPs	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR
Bottom-up: keypoint detection and grouping										
OpenPose [6]	–	–	–	–	61.8	84.9	67.5	57.1	68.2	66.5
Associative Embedding [39]	–	–	–	–	65.5	86.8	72.3	60.6	72.6	70.2
PersonLab [46]	–	–	–	–	68.7	89.0	75.4	64.1	75.5	75.4
MultiPoseNet [33]	–	–	–	–	69.6	86.3	76.6	65.0	76.3	73.5
Top-down: human detection and single-person keypoint detection										
Mask-RCNN [21]	ResNet-50-FPN	–	–	–	63.1	87.3	68.7	57.8	71.4	–
G-RMI [47]	ResNet-101	353 × 257	42.6M	57.0	64.9	85.5	71.3	62.3	70.0	69.7
Integral Pose Regression [60]	ResNet-101	256 × 256	45.0M	11.0	67.8	88.2	74.8	63.9	74.0	–
G-RMI + extra data [47]	ResNet-101	353 × 257	42.6M	57.0	68.5	87.1	75.5	65.8	73.3	73.3
CPN [11]	ResNet-Inception	384 × 288	–	–	72.1	91.4	80.0	68.7	77.2	78.5
RMPE [17]	PyraNet [77]	320 × 256	28.1M	26.7	72.3	89.2	79.1	68.0	78.6	–
CFN [25]	–	–	–	–	72.6	86.1	69.7	78.3	64.1	–
CPN (ensemble) [11]	ResNet-Inception	384 × 288	–	–	73.0	91.7	80.9	69.5	78.1	79.0
SimpleBaseline [72]	ResNet-152	384 × 288	68.6M	35.6	73.7	91.9	81.1	70.3	80.0	79.0
HRNet-W32	HRNet-W32	384 × 288	28.5M	16.0	74.9	92.5	82.8	71.3	80.9	80.1
HRNet-W48	HRNet-W48	384 × 288	63.6M	32.9	**75.5**	**92.5**	**83.3**	**71.9**	**81.5**	**80.5**
HRNet-W48 + extra data	HRNet-W48	384 × 288	63.6M	32.9	**77.0**	**92.7**	**84.5**	**73.4**	**83.1**	**82.0**

4.1.2 3D Pose Estimation

The goal of 3D Pose estimation is to detect the XYZ Coordinates of a certain number of joints of the human body and is termed as key points. This is done by using the subject's image whose 3D pose estimation needs to be done. Figure 6.1 shows the 3D key points within a human body. The detection of the key points can be demonstrated with the going up and down coordinates of the spine angle [**?**]. All the detected frames are assessed and compared to single angles in neighbouring frames which results in getting vectors such as 1s and 0s [50]. The continuous stretch of 1s represents ascent and 0s are for the descent. Moreover, there is a requirement of clearing up some of the accidental detections that occur. This has been done through the help of a sliding window application within a relatively small-size (4-5) frame that can replace all existing values inside the given window with a larger or a majority value for window transaction [**?**]. The following is a representation of the first few frames after adding cleanup value:

```
array([0., 0., 0., 0., 0., 1., 0., 1., 0., 0., 0., 0., 0., 0., 0., 0., 1.,
       1., 1., 1., 1., 0., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,

array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 1.,
       1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1., 1.,
```

Actual edge frames can be detected after the cleanup process is completed and is the position when the individual finished one repetition. This is done through the addition of a convolutional filter or kernel and is how VideoPose3D works.

Working of the 3D Pose Estimation process with temporal convolution and semi-supervised learning

3D Human Pose Estimation in the video is executed with help of temporal convolutions and semi-supervised training. 2D key point detection has been used and the then pose estimation is done in 3D. It has been observed that the temporal convolutional model takes up the 2D key-point sequences from the bottom as input and results in VideoPose3D as output (top) [35]. The mechanism that has been used is that of dilated temporal convolutions

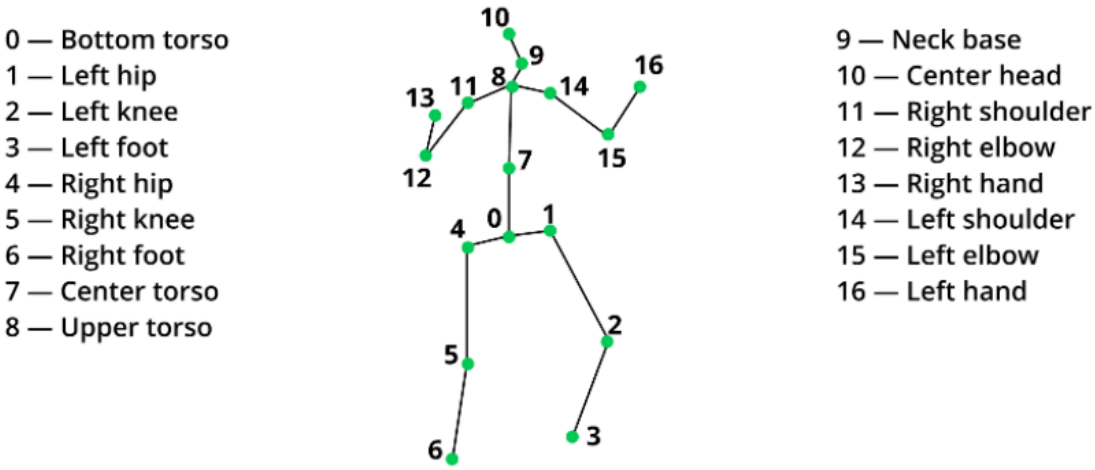


Figure 4.6: VideoPose3D Pose Estimation key points

```
# typical kernel for edge detection
kernel = np.array([10, 10, 10, 10, 10, -10, -10, -10, -10, -10])
peak_points = np.convolve(kernel, mirror_pad(pool_frames, (4, 5)), mode='valid
peak_points
```

```
array([  0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,  10.,
        20.,  30.,  40.,  50.,  40.,  30.,  20.,  10.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0., -10.,
       -20., -30., -40., -50., -40., -30., -20., -10.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,  10.,  20.,  30.,  40.,
        50.,  40.,  30.,  20.,  10.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0., -10., -20., -30.,
       -40., -50., -40., -30., -20., -10.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,  10.,  20.,  30.,  40.,  50.,  40.,
        30.,  20.,  10.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
```

Figure 4.7: Addition of kernel for edge detection

```
        30.,  20.,  10.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0., -10., -20., -30., -40., -50., -40., -30., -20., -10.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,  10.,  20.,  30.,
        40.,  50.,  40.,  30.,  20.,  10.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0., -10., -20., -30., -40., -50., -40., -30.,
       -20., -10.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,  10.,  20.,  30.,  40.,  50.,  40.,  30.,  20.,  10.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0., -10., -20., -30., -40., -50., -40., -30., -20., -10.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
         0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,  10.,
        20.,  30.,  40.,  50.,  40.,  30.,  20.,  10.,   0.])
```

to collect and collate data related to long term engagement and application. The reason behind using 3D Pose Estimation is that it is compatible with a 2D key point detector and has the capability of managing large contexts using dilated convolutions. In place of RNNs, VideoPose3D has been used so that higher accuracy is achieved along with doing the task in a simplified and effective manner. Computational complexity, as well as some parameters, gets reduced when the video pose estimation is done with 3D technique in comparison to RNN [9]. Earlier works have focused on using single-frame setting, however, in recent times that have been effort done in exploiting the temporal information from the video so that robust predictions are produced. 3D poses are inferred from Histograms of Oriented Gradients (HoG) features of Spatio-temporal volumes.

Technical Architecture of VideoPose3D

There has been the use of detected key point coordinates instead of heat maps and allows usage of 1D convolution rather than using 2D convolutions with the system over a defined coordinate time series. The figure below shows identifies that the input involves 2D key points for a respective field of 243 frames where each frame B has 4 blocks along with J= 17 points. The convolutional layers are represented in green where 2J, 3d1, 1024 denotes 2. J input channels [35]. The kernels are of size 3 and have dilation 1 along with 1024 output channels. The tensor size has been represented in parentheses that represent 1-frame prediction. For example, (243, 34) quadrant represents that there are 243 frames and 34 channels.

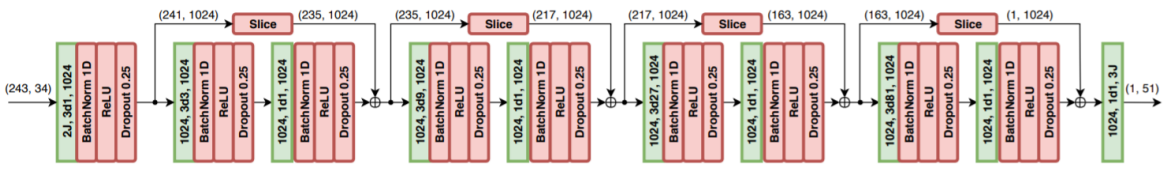


Figure 4.8: Technical Architecture of 3D Pose Estimation Process

Temporal convolution models have been used in estimating human poses as it allows parallelization across both aspects of batch as well as dimensions of duration (unit of time constraint) in comparison to RNNs. RNNs cannot get parallelized over time and the path of a

gradient in between output and input has a fixed length irrespective of sequence length. Input layer takes concatenated (x, y) coordinates of J points for each of the frames, and a temporal convolution with kernel size W and C output channels are applied [25]. The next stage is followed by B ResNet-style blocks surrounded by a skip-connection in which each block at first performs a 1D convolution with kernel size W and dilation factor D= WB̂. Batch normalization is done after convolution followed by rectified linear units and dropout. Parameters increase linearly whereas block exponentially increases the receptive field by a factor of W.

In this case, the filter hyper-parameters W and D are specified so that a tree is formed for the output frame covering all inputs. Figure 6.2 shows the instantiation of architecture for a receptive field size of 243 frames with B having 4 blocks and the W is set to be 3 along with C= 1024 output channels. The mechanism also uses a semi-supervised approach for training the data sets where labelled 3D ground-truth pose data is limited.

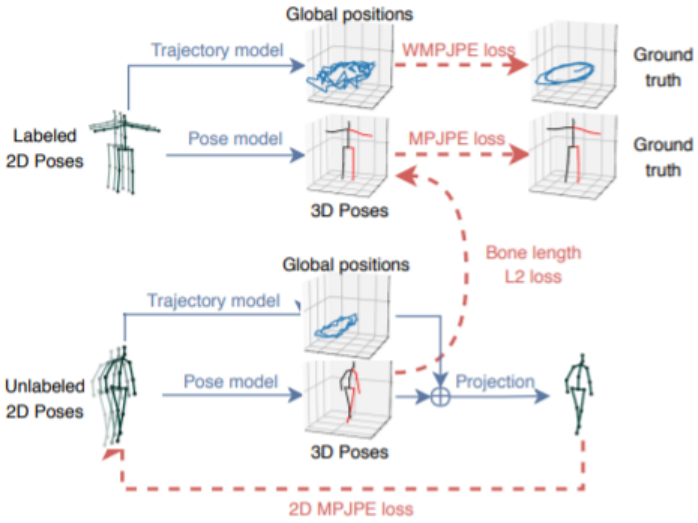


Figure 4.9: 3D pose model oriented training with a semi-supervised approach

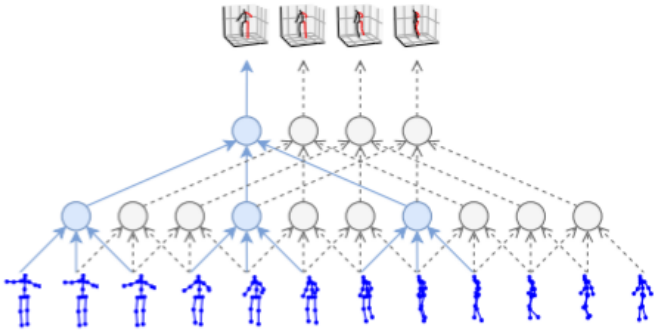
A soft-constraint is added in this case with the primary purpose to align the average lengths of the bone with a direct comparison of the labelled ones with the unlabelled predictions that helps in better orientation of the bone setup and placement in the model form factor. It has been observed that the unlabeled data is applied for implementation of an autoencoder loss and the respective 3D poses that are predicted are further projected back to the 2D pose modelling for better iteration and alignment. It has been observed that at times, issues

occur due to the subject being further away from the camera, and as a result, optimization of weighted mean per-joint position error (WMPJPE) loss function is done. The function is represented as: E= (1/y(z)) [f(x)-y] This implies the fact that each sample has been weighted with help of inverse of the ground-truth depth (y(z)) function in camera space. It has been observed that there is a Bone length L2 Loss that needs to be covered up and back up the 3D pose predictive mechanism as an offset rather than just duplicating the input data with the predicted or even the output data. A soft constraint is added to do so that the bone length is matched of the subjects of the unlabeled batch with subjects whose batch is labelled. While working with 3D Video Pose estimation, it is important that training and evaluation on 3D pose in the camera is done through rotation and translating the ground-truth pose as per camera transformation. It has been observed that for a Human3.6M, a decaying learning rate schedule is used that starts from a shrink factor of 0.95 when applied on each epoch. Temporal models with receptive fields of more than one are sensitive to the correlation of samples identified in pose sequence. The correlation is decreased in training samples by selecting training clips from different segments of video. The clip set size is specified as per the requirement of the receptive architecture of the model and as a result, the model estimates and yields a defined yet singularly independent 3D pose per training clip.

It has been observed that single frame optimization is possible through the replacement of the dilated convolutions with that of stridden ones. This is possible when the stride factor is set to a certain dilation factor. In the process of doing inferences, the entire sequence can be processed and intermediate states are reused of other 3D frames. To avoid the loss of frames to certain valid convolutions, padding is done at the input boundaries of the sequence. In order to have a stable running statistic with the architecture, the decay is done till the value reaches 0.001. In the next stage, horizontal flip augmentation is performed for training and doing time tests. On other hand, while applying a semi-supervised approach, it has been observed that the process gets efficient as the amount of labelled data gets reduced. There has been the use of the fully convolutional model in 3D video pose estimation and the architecture has been using temporal information. There has been the use of back projection in 3D pose estimation so that supervised learning is achieved in training the NN framework with labelled data. The method has been applied in cases when the labelled data is low in quantity and works well with certain unlabeled data obtained from a video.

VideoPose3D implementation

The research paper, “3D human pose estimation in video with temporal convolutions and semi-supervised training" has focused on estimation conceptualisation of the human pose in 3D based networks with dedicated temporal convolutions development environments and semi-supervised training modules. In the paper, the 3D pose estimation has been done with the help of dilated temporal convolutions of 2D key points. There has been use of back-projection so that the semi-supervised training method is made effective [35]. The start has been made with the prediction of the 2D key points for the unlabeled video and then the estimation of the 3D poses has been made [35]. The result has been observed to outperform the previous results by 6 mm mean per joint position error and has been reducing the error by 11%. It has been observed that the temporal convolutional model focuses on the 2D key-point sequences as the primary input in the process. As a result, it generates 3D pose estimate functions as the main outputs in the process. This has been represented in the following manner:



The following is the representation of the architecture and it has been observed that the input is gathered from the mapping of the 2D key points of the human poses and has a repetitive field of 243 frames in which there are 4 blocks in each frame [35]. There has been the addition of kernels in which the dilation applied is 1 and the size of the kernel is 3. It has been observed that due to the presence of valid convolutions, the residuals are sliced left and right in a symmetric manner so that the shape of the subsequent tensors is matched.

The data results of the temporal convolution model have been represented in the following manner:

There has been the use of Amsgrad and training has been done for 80 epochs. From the com-

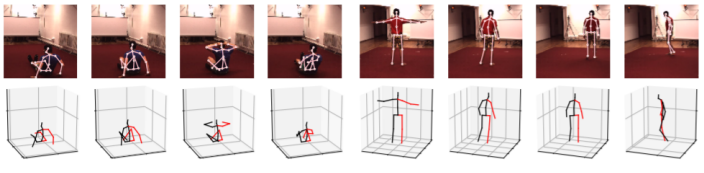
	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg
Pavlakos *et al.* [41] CVPR'17 (∗)	67.4	71.9	66.7	69.1	72.0	77.0	65.0	68.3	83.7	96.5	71.7	65.8	74.9	59.1	63.2	71.9
Tekin *et al.* [52] ICCV'17	54.2	61.4	60.2	61.2	79.4	78.3	63.1	81.6	70.1	107.3	69.3	70.3	74.3	51.8	63.2	69.7
Martinez *et al.* [34] ICCV'17 (∗)	51.8	56.2	58.1	59.0	69.5	78.4	55.2	58.1	74.0	94.6	62.3	59.1	65.1	49.5	52.4	62.9
Sun *et al.* [50] ICCV'17 (+)	52.8	54.8	54.2	54.3	61.8	67.2	53.1	53.6	71.7	86.7	61.5	53.4	61.6	47.1	53.4	59.1
Fang *et al.* [10] AAAI'18	50.1	54.3	57.0	57.1	66.6	73.3	53.4	55.7	72.8	88.6	60.3	57.7	62.7	47.5	50.6	60.4
Pavlakos *et al.* [40] CVPR'18 (+)	48.5	54.4	54.4	52.0	59.4	65.3	49.9	52.9	65.8	71.1	56.6	52.9	60.9	44.7	47.8	56.2
Yang *et al.* [56] CVPR'18 (+)	51.5	58.9	50.4	57.0	62.1	65.4	49.8	52.7	69.2	85.2	57.4	58.4	43.6	60.1	47.7	58.6
Luvizon *et al.* [33] CVPR'18 (∗)(+)	49.2	51.6	47.6	50.5	51.8	60.3	48.5	51.7	61.5	70.9	53.7	48.9	57.9	44.4	48.9	53.2
Hossain & Little [16] ECCV'18 (†)(∗)	48.4	50.7	57.2	55.2	63.1	72.6	53.0	51.7	66.1	80.9	59.0	57.3	62.4	46.6	49.6	58.3
Lee *et al.* [27] ECCV'18 (†)(∗)	**40.2**	49.2	47.8	52.6	50.1	75.0	50.2	**43.0**	**55.8**	73.9	54.1	55.6	58.2	43.3	43.3	52.8
Ours, single-frame	47.1	50.6	49.0	51.8	53.6	61.4	49.4	47.4	59.3	67.4	52.4	49.5	55.3	39.5	42.7	51.8
Ours, 243 frames, causal conv. (†)	45.9	48.5	44.3	47.8	51.9	57.8	46.2	45.6	59.9	68.5	50.6	46.4	51.0	34.5	35.4	49.0
Ours, 243 frames, full conv. (†)	45.2	**46.7**	43.3	**45.6**	**48.1**	**55.1**	44.6	44.3	57.3	**65.8**	**47.1**	**44.0**	**49.0**	32.8	**33.9**	**46.8**
Ours, 243 frames, full conv. (†)(∗)	45.1	47.4	**42.0**	46.0	49.1	56.7	**44.5**	44.4	57.2	66.1	47.5	44.8	49.2	**32.6**	34.0	47.1

(a) Protocol 1: reconstruction error (MPJPE).

	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg
Martinez *et al.* [34] ICCV'17 (∗)	39.5	43.2	46.4	47.0	51.0	56.0	41.4	40.6	56.5	69.4	49.2	45.0	49.5	38.0	43.1	47.7
Sun *et al.* [50] ICCV'17 (+)	42.1	44.3	45.0	45.4	51.5	53.0	43.2	41.3	59.3	73.3	51.0	44.0	48.0	38.3	44.8	48.3
Fang *et al.* [10] AAAI'18	38.2	41.7	43.7	44.9	48.5	55.3	40.2	38.2	54.5	64.4	47.2	44.3	47.3	36.7	41.7	45.7
Pavlakos *et al.* [40] CVPR'18 (+)	34.7	39.8	41.8	38.6	42.5	47.5	38.0	36.6	50.7	56.8	42.6	39.6	43.9	32.1	36.5	41.8
Yang *et al.* [56] CVPR'18 (+)	**26.9**	**30.9**	36.3	39.9	43.9	47.4	**28.8**	**29.4**	**36.9**	58.4	41.5	**30.5**	**29.5**	42.5	32.2	37.7
Hossain & Little [16] ECCV'18 (†)(∗)	35.7	39.3	44.6	43.0	47.2	54.0	38.3	37.5	51.6	61.3	46.5	41.4	47.3	34.2	39.4	44.1
Ours, single-frame	36.0	38.7	38.0	41.7	40.1	45.9	37.1	35.4	46.8	53.4	41.4	36.9	43.1	30.3	34.8	40.0
Ours, 243 frames, causal conv. (†)	35.1	37.7	36.1	38.8	38.5	44.7	35.4	34.7	46.7	53.9	39.6	35.4	39.4	27.3	28.6	38.1
Ours, 243 frames, full conv. (†)	34.1	36.1	34.4	**37.2**	**36.4**	**42.2**	34.4	33.6	45.0	**52.5**	**37.4**	33.8	37.8	**25.6**	**27.3**	**36.5**
Ours, 243 frames, full conv. (†)(∗)	34.2	36.8	**33.9**	37.5	37.1	43.2	34.4	33.5	45.3	52.7	37.7	34.1	38.0	25.8	27.7	36.8

(b) Protocol 2: reconstruction error after rigid alignment with the ground truth (P-MPJPE), where available.

parison of the convolutional model to that of LSTM one, it has been observed that model parameters along with the estimated result of dedicated operations like the Floating Operations (FLOPs) that are added to predict one frame at inference time for the former one [35]. At the same time, the Matrix multiplications scenarios as well as the concept of amortized cost are considered for LSTM which has a sequence of infinite length provided it is done over a hypothetical scenario. It has been observed that with help of a convolutional model in doing 3D pose estimation; the process can be parallelized for both numbers of sequences and the targeted temporal dimensions in the process. This is the advantage gained over RNN as with it only parallelization of different sequences can be done. The speed factor is also independent of the batch size as there is the use of temporal processing. The architecture that has been in use makes application of temporal information. This is followed or accompanied with dilated convolutions over pre-specified 2D key point trajectories. Back-Projection has been observed to be used when the labelled data availability is scarce in semi-supervised training methods. The figure below shows the video frames projected from the top in 2D and the way it gets transformed to 3D. Fully convolutional architecture tends to improve the 3D projection and has been used.



	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg
Single-frame	12.8	12.6	10.3	14.2	10.2	11.3	11.8	11.3	8.2	10.2	10.3	11.3	13.1	13.4	12.9	11.6
Temporal	3.0	3.1	2.2	3.4	2.3	2.7	2.7	3.1	2.1	2.9	2.3	2.4	3.7	3.1	2.8	2.8

Table 2: Velocity error over the 3D poses generated by a convolutional model that considers time and a single-frame baseline.

Method	P1	P2	Method	P1	P2
Martinez *et al.* [34] (GT)	45.5	37.1	Ours (GT)	37.2	27.2
Martinez *et al.* [34] (SH PT)	67.5	52.5	Ours (SH PT from [34])	58.6	45.0
Martinez *et al.* [34] (SH FT)	62.9	47.7	Ours (SH FT from [34])	53.4	40.1
Hossain & Little [16] (GT)	41.6	31.7	Ours (D PT)	54.8	42.0
Lee *et al.* [27] (GT)	38.4	–	Ours (D FT)	51.6	40.3
Ours (CPN PT)	52.1	40.1	Ours (CPN FT)	46.8	36.5

Model	Parameters	≈ FLOPs	MPJPE
Hossain & Little [16]	16.96M	33.88M	41.6
Ours 27f w/o dilation	29.53M	59.03M	41.1
Ours 27f	8.56M	17.09M	40.6
Ours 81f	12.75M	25.48M	38.7
Ours 243f	16.95M	33.87M	37.8

4.1.3 Action Quality Assessment Model trained using Markers

The paper "A Deep Learning Framework for Assessing Physical Rehabilitation Exercises" focuses on understanding the framework that can be designed to examine whether the patients are performing as per the prescribed rehabilitation exercises or not. It is assessed based on the movement data that are processed and captured with help of the sensory system. The deep learning-based framework has been proposed for assessing the quality of the physical rehabilitation exercises in an automated manner (LiaoVakanski and Xian, 2020). The framework has been implemented through the help of certain metrics that are used for quantification of movement performance and scoring functions are determined through which performance metrics are mapped into numerical scores. It is used to assess the quality of the movement and the segmented quality scores with the application of the deep neural network models. These can be generated by the input oriented movements through the help of supervised learning (LiaoVakanski and Xian, 2020). Similarly, the performance metric in the system is based on the log-likelihood of functions like the Gaussian mixture model. It also encodes low-dimensional data representation functions that are completely received through a set-up of specialised ANNs like the deep autoencoder networks. As a result, a Deep Spatio-temporal neural network identifies the data pattern and accordingly arrange the data into a specially encoded data structure that are also called the temporal pyramids that has the capacity to understand human movement through the help of sub-networks. It is used in processing functions and rendering activities like joint displacements of various parts in the human body and the framework is put to test through collecting data of body movements of ten rehabilitation exercises.

The following diagram helps in understanding the deep-learning framework which has been designed for the assessment of the rehabilitation exercises. Skeletal joint coordinates are gathered through the help of the sensory system and are processed through the dimension-

ality reduction technique. Performance measures are quantified and scoring is mapped to receive the movement quality scores (LiaoVakanski and Xian, 2020). This has been used in training the NN Model and then the trained NN model is used to automate the generation process of movement quality scores and is given as input.

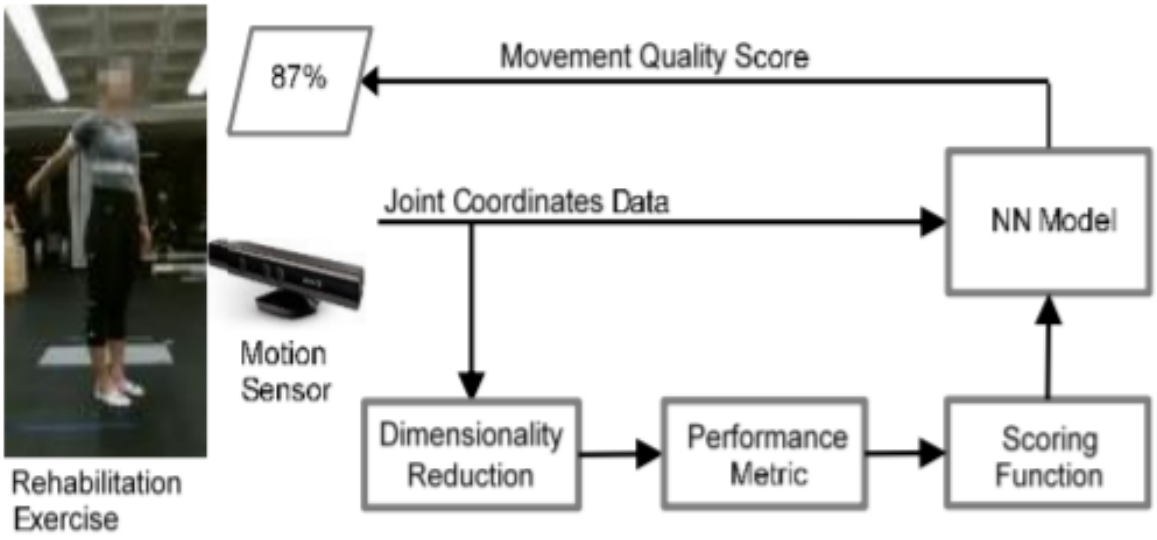


Figure 4.10: Approach for Action Quality Assessment of Marker based model

The sensory system collects the rehabilitation exercise performed by a S subject for R(s) number of times. Combined data of all R(s) repetitions of the exercise is represented through the help of delta(s). R is the total of repetitions done by the S subjects and is represented by R= summationR(s) where S starts from 1 and continues till the last subject number whose exercise movement is captured. With help of notation, X (s, r) for r-th repetition and s-th subject, there is another formulation created for exercise measurement with respect to each subject. Xs,r is a temporal sequence of T number of measurements and is represented through the help of Xs,r = (x (1) s,r; X(2) s,r;....x (r) s,r) in which the superscripts are meant to denote the index of the temporal order of joint displacement vector. Y represents the data collected from the patient group data source which is represented by Y=Y s,r where Y s,r is the data to be used in r-th repetition.

Sensory systems are also applied whose primary function is motion capturing that track in between 15 and 40 skeletal joints and it depends upon the sensor type. It has been observed that the measurement data consisted of the 3-dimensional spatial formation of 10,30 and 117 computational units. A temporal sequence of 4-dimensional vectors is selected to rep-

resent the code of the proposed network. The following figure depicts how dimensionality reduction has been done. It is evaluated that the autoencoder architecture can identify the input functions and project the time oriented movement data into a code representation function or system. Once this is taken care of, it re-projects the focused and segmented data that enhances the output.

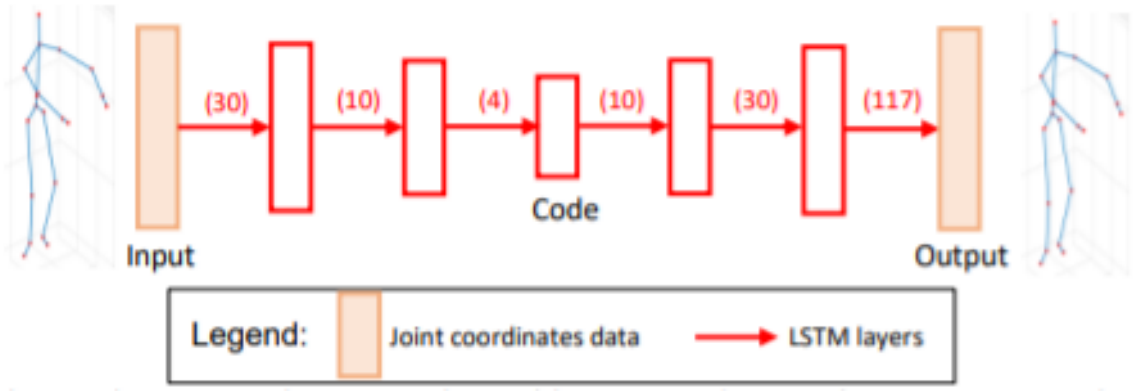


Figure 4.11: Representation of joint coordinates data as an input to LSTM layers

The multiple metrics for quantification of data collected from the function of patient performance are categorized into either a model-less or a model-based set of KPIs. It has been observed that the model-less metrics make use of distance functions that involve Euclidean, Mahalanobis distance as well as dynamic time warping (DTW) deviation amidst the flow of multiple data sequences. In this case, a metric or KPI that is designed based on the Gaussian Mixture Model (GMM) has been selected. It is declared within the system as the parametric probabilistic model for the representation of data with the combination of multiple Gaussian probability based density functions. It is represented in the following manner:

$$\mathcal{P}\left(\boldsymbol{x}_{s,r}^{(t)}|\lambda\right) = \sum_{c=1}^{C} \pi_c \mathcal{N}\left(\boldsymbol{x}_{s,r}^{(t)}|\mu_c, \Sigma_c\right),$$

The model parameters of lambda are estimated with help of the Expectation-Maximization (EM) algorithm and it has been observed that a negative log-likelihood is applied to gather the performance metrics. This is represented in the following manner:

$$\mathcal{P}(\mathbf{Y}_{s,r}|\lambda) = -\sum_{t=1}^{T} \log\left\{\sum_{c=1}^{C} \pi_C \mathcal{N}\left(\mathbf{y}_{s,r}^{(t)}|\mu_C, \Sigma_C\right)\right\}.$$

Scoring Function works towards mapping data patterns and recorded values of the major performance metrics. In return, it focuses towards the analysis of the movement quality scores in the range of 0 to 1. The sores obtained helps patients to have a self-assessment and progress towards recovery can be interpreted. Supervised training is done with help of deep NN models trained by the movement quality scores. The following is the scoring function that has been proposed:

$$\bar{x}_k = \left(1 + e^{\frac{x_k}{\mu+3\delta} - \alpha_1}\right)^{-1};$$
$$\bar{y}_k = \left(1 + e^{\frac{x_k}{(\mu+3\delta)} - \alpha_1} + \frac{y_k - x_k}{\alpha_2(\mu+3\delta)}\right)^{-1},$$

where k represents

$$k \in \underline{L},\ \mu = \frac{1}{L}\sum_{k=1}^{L}|x_k|, \delta = \sqrt{\frac{1}{L}\sum_{k=1}^{L}(|x_k| - \mu)^2}$$

In the study, there is a proposal of deep learning architecture for rehabilitation assessment. With the help of this architecture, the spatio-temporal modelling of skeletal data is done. The figure below shows the graphical representation of the NN Architecture and the model that has been used to design the spatial characteristics of human movements. The data given as input is arranged in temporal pyramid form for measuring the various scaled and layered versions of the system generated movement repetitions. Spatial dependencies in the human movement have been learned with help of one-dimensional convolutional filters.

A hierarchical model has been used to develop the NN Architecture and there is an arrangement for five recurrent sub-networks taking the various actions performed as input. This involves the actions performed by the patients with the help of their limbs, torso and joint

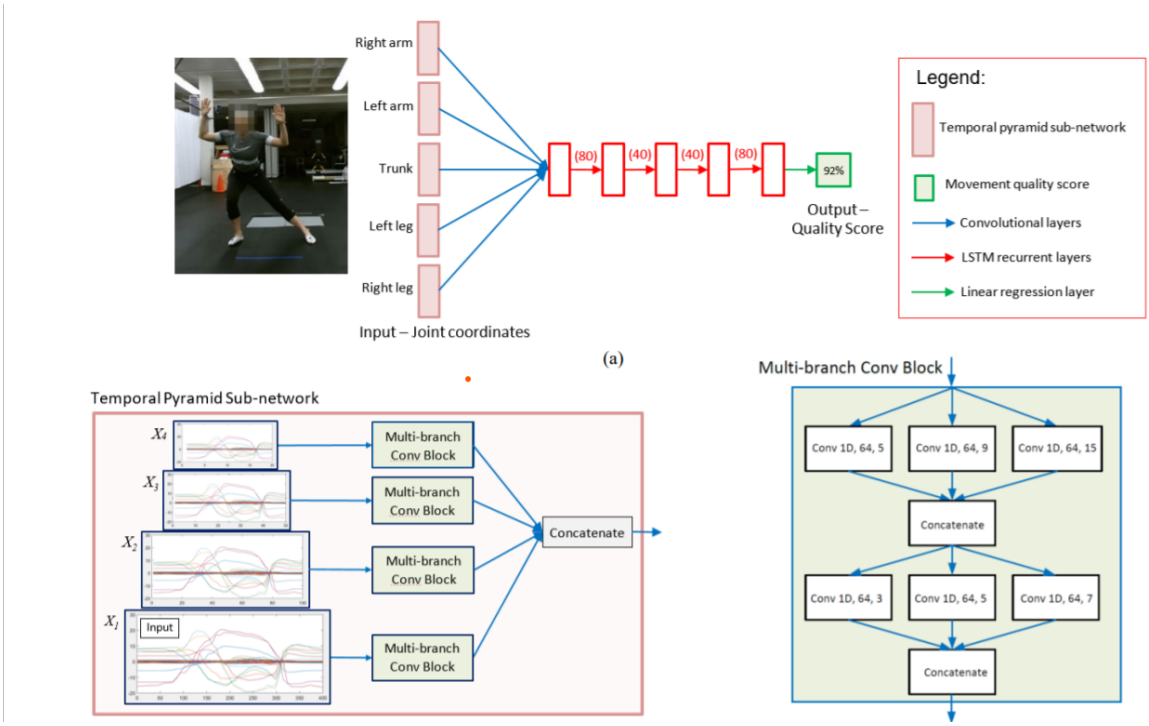


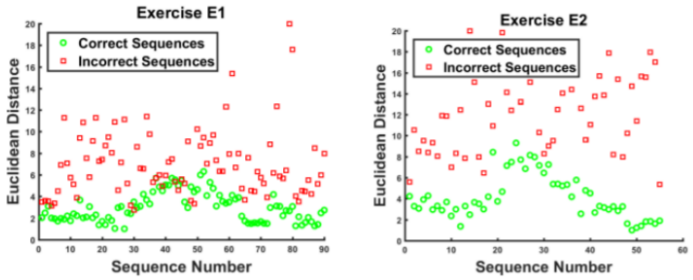
Figure 4.12: Deep Learning Architecture for Action Quality Assessment

displacements of these body parts that lead to the formation of the input data. Such hierarchical representation of network layers makes it easy for low-level spatial information to be collected from joint movements. Once this is achieved, a high-level representation of targeted body parts and their scales and limitations are made. Convolutional units in the hierarchical layers are applied in this case with the layers in succession that further use recurrent units. Temporal pyramids are used for the processing of video data through dynamic sub-sampling input videos at varying frame rates. At the same time, multi-scale video pyramids are also used to enhance the detection, localization and zeroing of human oriented actions in terms of body movement. It has been observed that the convolutional layers used in the architecture consist of two of the layers along with a dropout layer and have a rate of 0.25. It has been observed that each of these layers contain a set of three different branches which form the basis of the 1D convolutional filter that are of varying lengths. There has been the use of multiple branches so that the designed framework has the capacity to use the most suitable filter length as per the initial data collection. A recurrent representation and its digitally portioned replica of the model has been observed to consists of four layers 80,40,40 and 80 LTSM units. The results have been identified in the following manner:

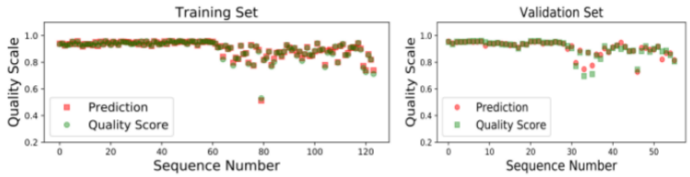
PERFORMANCE COMPARISON: AVERAGE ABSOLUTE DEVIATION PER EXERCISE

Exercise	Our approach	Co-occurrence [47]	Deep CNN	PA-LSTM [48]	Two-stream CNN [49]	Hierar. LSTM [10]	Deep LSTM
E1-E10	**0.02527**	0.02703	0.02615	0.04534	0.11044	0.08819	0.04059
E1	0.01077	**0.01052**	0.01357	0.01839	0.28798	0.03010	0.01670
E2	**0.02824**	0.02905	0.02953	0.04413	0.22349	0.07742	0.04934
E3	**0.03980**	0.05577	0.04141	0.08094	0.20493	0.13766	0.09382
E4	**0.01185**	0.01347	0.01640	0.02347	0.36033	0.03580	0.01609
E5	0.01870	0.01687	**0.01300**	0.03156	0.12332	0.06367	0.02536
E6	**0.01779**	0.01886	0.02349	0.03426	0.21119	0.04676	0.02166
E7	0.03819	**0.02733**	0.03346	0.04954	0.05016	0.19280	0.04090
E8	**0.02305**	0.02464	0.02905	0.05070	0.04337	0.07260	0.04590
E9	**0.02271**	0.02720	0.02495	0.04313	0.14411	0.06508	0.04419
E10	0.04162	0.04657	**0.03667**	0.07727	0.11044	0.16009	0.05198
Training time (in seconds)							
	177	325	**52**	598	4,668	295	410

The sequence number of exercises has been represented in the following manner:



The results of the proposed neural network which is the deep NN for assessing exercise E1 are randomly split in a ratio of 0.7/0.3. 90 correct and 90 incorrect repetitions were divided and equally distributed random splitting. The training set consisted of 124 data and it further consists of a set of validation units of 56 repetitions.



4.2 Datasets

4.2.1 COCO Dataset

The research paper aims to understand the visual scenes which is one of the important objectives of computer vision. In the research work that was carried out, the images are gathered of composite everyday scenes involving ordinary objects in their natural habitat and environment. The research paper proposes to use fully segregated instances to allow precise detector evaluation. Here, a total of 2.5 million labeled instances were used in about 328k images which drew extensive crowd worker involvement. The dataset contained 91 objects type which would easily be identifiable by a 4-year-old kid. The images were collected by following two processes. Firstly, the common object categories were selected. In the research, only "thing" categories were included and not the "stuff" categories as the research is focused more on the precise localization of object instances[46].

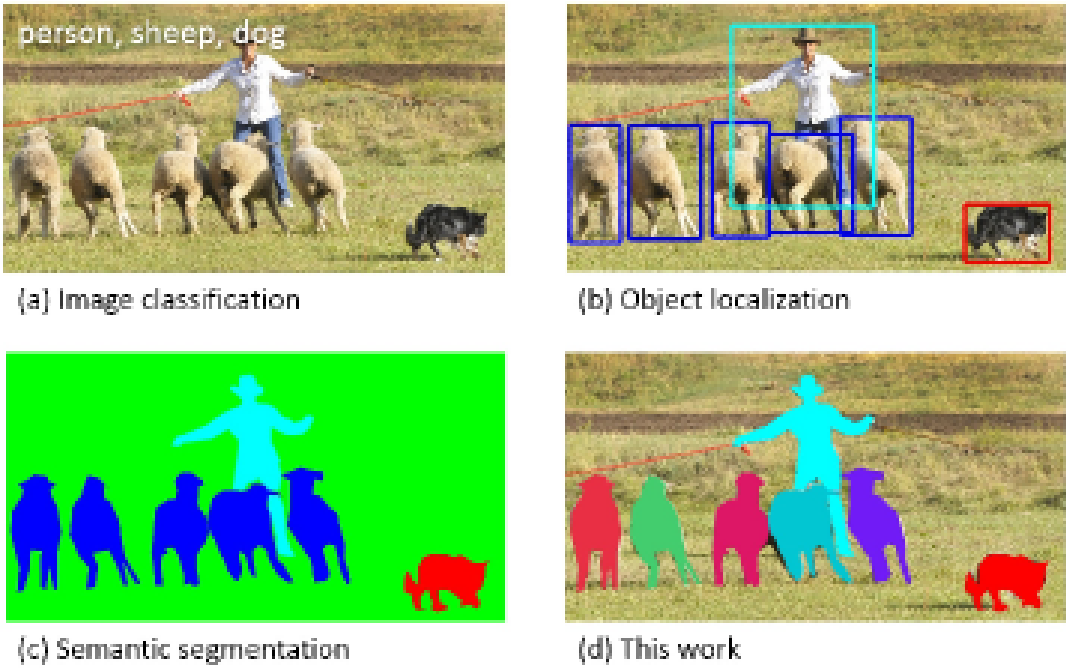


Figure 4.13: Microsoft COCO image recognition

To permit the practical assemblage of a large number of instances per category, the dataset was limited to entry-level categories only. Based on the list of object categories collected, the upcoming step followed is to gather candidate images based on three types: iconic scenic

images, iconic object images, and non-iconic images. The aim of the research was to gather a dataset in which the bulk of the pictures are non-iconic. Datasets involving a larger number of non-iconic images are more preferable to generalizing. The next step was to annotate the collected data. For all crowdsourcing tasks, Amazon's Mechanical Turk (AMT) has been used. Several steps have been followed to improve the quality of annotations. The annotators for instance spotting and category labeling stages were maximized to 8. After category labeling and instance spotting, instance segmentation of each object instance was done. A comprehensive statistical analysis of MS COCO in comparison to SUN, ImageNet, and PASCAL has been discussed, apart from an algorithmic analysis of MS COCO being highlighted. Using a Deformable Parts Model, a performance analysis was drawn based on segmentation detection results and bounding box.

Process of data collection

The images required to carry out the research work were gathered following various processes. They are:

- The selection of object categories was the first step: The categories were designed to represent a set of all categories that would be applicable to feasible applications and which happened with peak frequency to allow the collection processes of a big dataset. The research used various sources to gather entry-level object grouping of "things" where a list of grouping was first compiled by integrating various groups from PASCAL VOC and a sub-division of 1200 most commonly occurring words that indicated visually identifiable objects. A lot of children ranging between the ages of four to eight were demanded to label each of the objects they observe in outdoor and indoor environments. Finally, the co-authors voted on a scale of 1 to 5 based on their usefulness in practical applications, how commonly they occur, and their relative diversity. To ascertain backward similarity all the groupings from PASCAL VOC were also incorporated.

- The non-iconic pictures were collected using two procedures: Images were firstly gathered from sites that provided the lesser number of iconic images such as Flickr. Flickr mainly provided pictures transmitted by amateur photographers with browsable key-

words and data . The next step followed was to search for pairwise amalgamation of object groups which resulted in more non-iconic images. The searched images contained more than the two specified categories. Scene/object category pairs were also searched to further supplement the dataset of the research [45].

- Taking the help of more than 60,000 worker hours a huge multitude of object cases were organized, annotated, and gathered to manage the development of segmentation algorithms and object detection. In this way, more importance was placed on detecting non-iconic pictures of objects in their natural environment. Data statistics pointed out that the pictures contain a generous amount of contextual information.

The signification of the research

The research draws a comparison between MS COCO and other datasets such as PASCAL, SUN, ImageNet based on dataset statistics and algorithmic analysis. With more than 91 major groups and categories, the Microsoft common object in context (MS COCO) data set has more than 5000 instances in around 82 categories. Although COCO has lesser groups and categories as compared to the ImageNet dataset, it has larger instances per group. This aids in researching the complete objective models efficient of accurate 2D localization. As compared to SUN and PASCAL VOC, it has larger instances per group [18].

Figure 4.14: COCO stuff database

ImageNet was designed to apprehend more fine-grained object groups. SUN pays more attention to labeling scene types and their corresponding objects. PASCAL VOC's major im-

plementation is to identify the objects in the natural environment. Here MS COCO provides an additional feature as it provides both segmentation and detection of objects occurring in the innate environment. An important feature of the MS COCO dataset is that it finds non-iconic images containing objects. On average, the MS COCO dataset involves 7.7 instances per picture and 3.5 groups and categories. More contextual information is required to recognize the smaller objects which are hard to find [56]. For bounding-box detection analysis, DPMv5-P and DPMv5-C models were examined on both PASCAL and MS COCO. On analyzing the average performance, it was noticed that there is a similar drop in performance (by a factor of 2) in both the models for MS COCO. In some categories, the dataset of MS COCO performed better than PASCAL and in others, it performed badly [24]. Overall, the performance of MS COCO is much lower.

Methodologies used

The above-mentioned research follows a primary quantitative research analysis process. The domain set includes 272 candidates and 40 scene groups (used for solving problems on scene objects). The AMT user associates for gathering non-iconic images, instance spotting, group labeling, instance segregation, and crowd labeling are described here. For collecting high-quality non-iconic images, candidate images were collected by searching for paired object categories. An ATM filtering task was then created which allowed the ATM users to eliminate inappropriate and iconic pictures from 128 candidates.

For category labeling, workers were encouraged to annotate all groups present in the picture. This is done by dropping and dragging icons from the bottom onto respective object instances. For the spotting process, a blinking icon was initiated that included a single instance. The workers were then asked to spot 10 instances of the selected groups by indicating a single cross within the given area. Therefore, it was important to provide a “magnifying glass” attribute that doubled the power of the worker’s currently chosen area. Next for instance segmentation, the researcher has modified the source code from the Open Surfaces project. To fasten the segmentation process, a visualization of the object group symbol was added to remind workers of the groups in which the pictures are to be segmented [24]. A zoom-in functionality was also added to enable quality annotation of curved bound-

aries and objects. Multiple workers were used to ensuring higher analysis of all object instances [33]. Due to the time consumption of instance segmentation, all the instances are segmented only once.

Figure 4.15: Microsoft COCO key point detection

Thus, segmentation verification becomes important to ensure that each segmented part is of sufficiently better quality. Here, the workers were required to detect the bad quality segmentation out of the total 64 segmentations. 4 were selected to be the worst by the workers. Each segmentation was further revealed to three annotators. Further, if any of the three annotators detects that the segmentation is of poor quality, it is conveyed to two more workers. At this stage, any segmentation process that does not get 4 to 5 necessary votes, is discarded. The final step is crowd labeling which is necessary only for images containing a larger amount of object instances of a given group. This method requires the workers to "paint" all pixels in question rather than drawing appropriate polygonal structures around each of the objects.

Future scope of the research

The research based on MS COCO provides a lot of future scope. They are:

- The researcher focuses more on "things" categories for selection for the object categories[52].

A decision regarding whether to include both "things" and "stuff" categories is crucial. With respect to the research carried out The researcher in the research chose "things" categories due to the precise localization of object instances. However, the research can be carried out by including "stuff" categories in the future. The future classification of "stuff" groups would be helpful as it can provide significant contextual information.

- Various object detection algorithms take advantage of supplementary annotations, such as the position of key points on the object and the amount an instance is obstructed. In the future, the MS COCO dataset could give a good tag for other types of the labeling process, including attributes, scene types, and full sentence written description. Various annotations can be included in the future.

4.2.2 Human3.6M

The paper revolving around the usage of Human3.6M-Large Scale datasets and corresponding predictive methods is based on 3D human sensing and how it has brought a change in the concept of artificial intelligence in recent years [21]. The research described in the paper identified above elucidates how various technologies of artificial intelligence are used for identifying the presence of one or more human bodies in spite of any intentional participation of the person that is being identified. Intricate information is provided regarding how 3D poses of human bodies can be captured from real images. The paper incorporates a wide overview of how this 3D human sensing system plays a crucial role in training. 3D capturing of human bodies or other objects in motion has been in the headlines for the past few years. This system has proved to become a game-changer because the system of 3D sensing has become a scientific breakthrough as a whole. Objects and facial attributes of a human body can be recognized using this technology with the help of artificial intelligence has been discussed in the paper.[29]. The 3D human sensing technology has the capability to capture the exact breadth, length, and height. The capture of these aspects is done with proper clarity with unique advancements. The research community has brought light to various aspects of 3D human sensing where a huge set of human poses, synchronized modalities, and diverse motion are involved. It's quite understandable that 3D human sens-

ing has its clear advantage because it does not require a special costume or special market to capture objects rather the video is taken using 2D cameras. The paper particularly focuses on the Human 3.6M data set and the data set comprises more than 3 million 3D segmented poses [30]. The 3D poses are also captured by a team of professional actors.

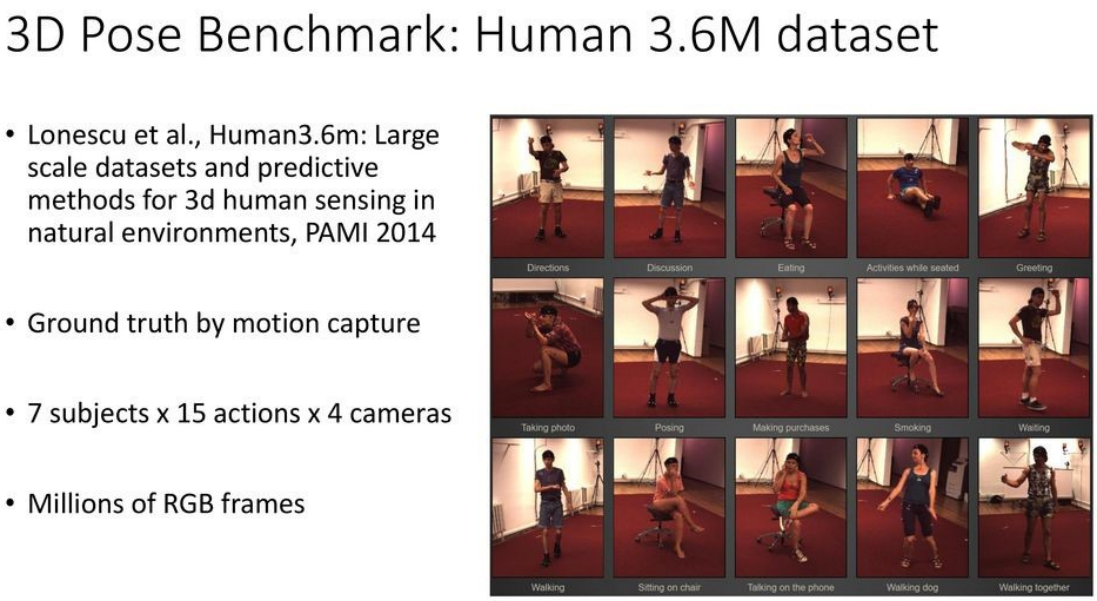


Figure 4.16: 3D Pose for Human 3.6M dataset

Process of gathering data

Information regarding the Human 3.6M has been gathered using predictive methods and a huge number of data sets. The paper has also incorporated detailed information on how different and essential human activities such as greeting, receiving a phone call, eating, clicking pictures, or any other poses or motion can be utilized and encountered using this data set. The data of the paper regarding the Human 3.6M data set has been gathered by recording and analyzing the performance of some participants [34]. The paper has taken a total of four viewpoints and opinions. Recording conditions and the composition of the data have been done by making an experimental setting. A laboratory has been developed where sensors were used to gather data. There were motion cameras, a digital camera for video, and a single time-of-flight sensor. These sensors were combined using both software and hardware synchronization. The participants who were a part of the experiment were attached

to a minor reflective marker [37]. The reflective marker played a vital role in tracking the subjects over time. This way the movements and poses of the subjects were recorded which helped in maintaining the label identity and also helped in tracking how it propagated with the passage of time as per the initial pose of the subjects labeled either automatically or in a manual way. Hence, the recording ensured that accurate parameters were only considered by using a fitting process. The fitting process is used to gather data used and detected each body labels accurately to ensure the relevance of the paper [57]. The Body Mass Index (BMI) of the 11 subjects was in the range of 17 to 29. Out of which, the participants were divided for testing and training, 2 males and females had neem chosen for testing, and others were chosen for validation and training. The way of gathering the data was done efficiently where the division was done to ensure moderate variability of different body shapes of the 11 participants. The information that was gathered was volumetric. The information was gathered like 3D body scans to get an accurate representation of the joint positions, joint angles, and so on.

Significance of the research

The paper incorporates the use of the Human 3.6M data set which is one of the well-known and biggest datasets for capturing motions. This data set is significant because this data set includes about 3.6 million poses of human bodies. The human 3.6M data set is significant for future generations as it helped in capturing high quality and high-speed motion pictures and videos as it has about four cameras which are of extremely high quality to capture videos at 50 Hertz [57]. The human 3.6M data set has become a game-changer for its uniqueness and fast capturing of high speed and high-quality pictures and videos therefore, this makes the topic enticing and worthwhile to be researched. The research topic makes it significant how 3D human sensing can facilitate the upsurge of the system of “visual human sensing”. The research has been conducted in the first place because in today’s world of extreme uncertainties, may it be due to the COVID-19 pandemic, people have understood the importance of technological advancement. Technological advancement has not left human operations stagnant. Therefore, the research also brings out the importance of the Internet of Things and Artificial intelligence. Human 3.6M data set has been applied in so many fields such as consumer electronics are some big examples. This data set is also seen to be

a part of the gaming industry. The essence of the research revolves around the advantages technological advancement carries and illustrates how different algorithms and sensors of extremely efficient and high quality can allow effective 3D human sensing.

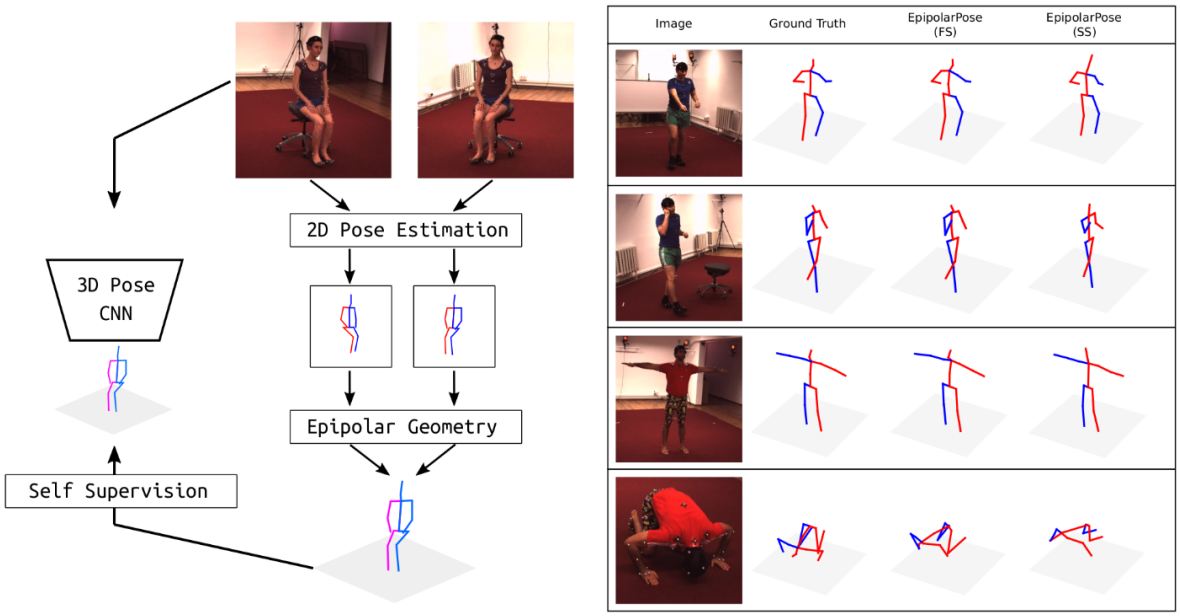


Figure 4.17: 3D Human Sensing

Methodologies

The research was initially qualitative in nature where information was collected by gathering information from journals, online articles, library resources, books, or an interview. However, with the progress of the research, the researcher had to conduct quantitative research later for making use of graphic renderings [58]). The labeling of the 3D and 2D models was also qualitative. The research was conducted by creating an experimental setting of 11 participants. The experimental setting was the first method where the performance of the 11 participants was recorded. An 'Actor and Human Set' was made to conduct the experiment therefore the data was gathered from the live population. 3D body scans of all 11 actors were collected and used in the research. For predicting the models, the 'Kernel Dependency Estimation' (KDE) was used in the research to determine and study the estimation of different poses of the 11 actors. The researcher had also used 'Fourier Kernel' as the methodology because it brought more relevance to the research which also served the purpose of the research carried out. For receiving the quantitative aspect of the research, Prediction

Experiments were carried out. Image descriptors, training set size, pose data were some aspects of the prediction experiments. Moreover, the mixed reality results were also a significant part of the research where models were incorporated into the laboratory tests and data. Using these methodologies for conducting quantitative and qualitative research gave the researcher a clear path to understand the new data set, Human 3.6M [37].

Limitation and future scope

The above-described research has developed an overview of the large data set which is Human 3.6M. This data set has wide-scale scope in the future because this data set has about 3.6 million poses which are segmented as 3D. The research has distinctive future scope because 3D human sensing or the Human 3.6M data set highly compliments the data sets which are existing and it also provides accurate and high-quality 3D models and will serve the purpose in various fields. The gaming industry was the first industry to accept this system however, the research opens up more opportunities to different industries to cater to their needs by using 3D human sensing and Human 3.6 M data set in particular [37]. However, as every coin has two sides, the research also has some limitations as the consumer electronics market still does not understand its importance therefore the research is majorly based on the 11 participants as there is not sufficient information regarding this data set and its application.

4.2.3 UIPRMD

During the procedure of recovery of multiple musculoskeletal aspects, the participation of patients who are involved in physical therapy and other forms of rehabilitation programs is significant. One of the main components of this form of treatment is a “home exercise program” (HEP). The patients are required to execute physical forms of exercises that are recommended in a home-based environment. But unfortunately, the HEP has not been successful in aiding a patient reach full recovery, although it has incurred huge expenses (Zhao et al. 2017). Reports suggested that the main reason behind the lack of success of HEP is mainly attributable to the lack of adherence to the methods which have been recommended, and also some psychological barriers like fear of getting any form of injury and anticipation of

discomfort have added to the problem. So as to support the HEP, the development of tools is being undertaken. The process in which algorithms operate and the capability to take

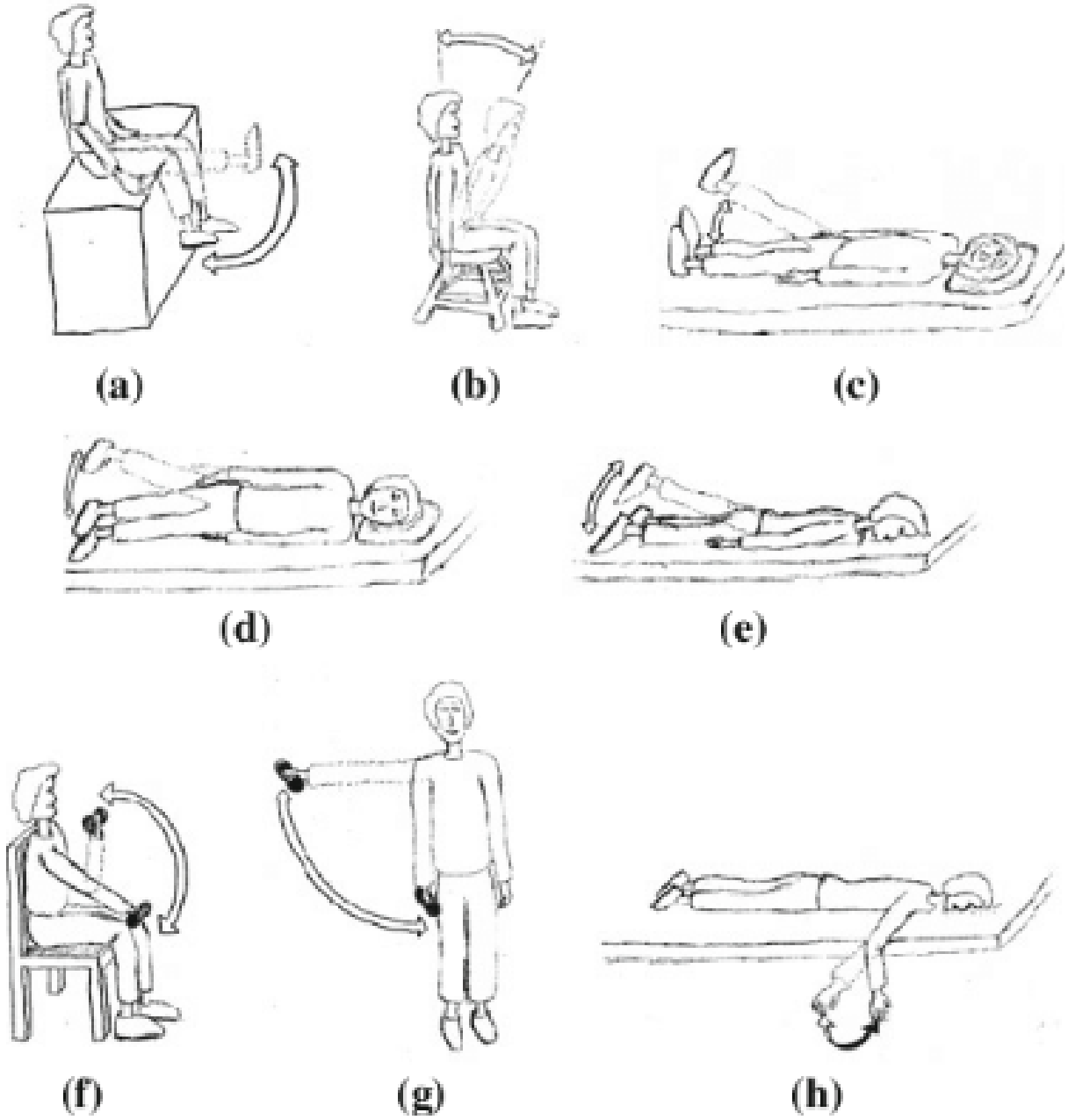


Figure 4.18: Different forms of physical rehabilitation exercise

out applicable and valuable data from the sources are very much related to the quality and quantity of the data available, this is crucial for the issues in the subject of machine learning. In recent times the companies which have access to huge amounts of data are considered to have an added advantage over their competitors. Hence a great chunk of concentrated effort on the formation of various forms of data sets for dealing with the complications that

are present in machine learning.

Significance of the research

The paper aims to have the project focusing on the progress of the new "therapy-supported too" for nursing and evaluating the "HEP episodes" on the basis of the "machine learning algorithms". The project is aimed to find out and solve the problems associated with the lack of success faced by HEP. One of the objectives is to deploy "Microsoft's Kinect sensor" so as to capture the movements of the body during the meetings and assess the performance of the patients. This would also help to determine whether the patients have been adhering to the recommendations or not. The project is based on assessing the development of a methodology which are to be used for mathematical modeling of patients and their respective movements. Definition of performance metrics is another major factor that is targeted by this model along with creating a set of movements based on therapy. On the basis of neural network architecture, the project deployed a machine learning method that has developed a model which is preliminary in nature of human motions which consists of an autoencoder and a mixture of density subnets [61].

The research community in recent times dealing with machine learning has become cautious of the provisions that are appropriate and use such data sets which are essential for improved results of the prevailing algorithms, and also assessing modified algorithms. The motivating factor behind the research was that the data which are available in the market relating to therapy movements are inadequate in scope and format. Some of the well-known examples of data sets for therapy movements like HPTE, Emopain, etc. had some lacunae each. The HPTE data set could not provide consistent body joint postures or angles, and finding out the same is not an easy task. The research work used the UI-PRMD data set. The ultimate goal of the research carried out was to assess the deviation of motion trajectories and also remodel the rehabilitation movements in general.

Process of gathering data

The research paper consists of the UI-PRMD data set and used ten different movements for the assessment. While collecting the data one of the authors of the research initially demonstrated each of the movements which are to be assessed and thereafter the subject was asked to repeat the said movements. These particular movements were selected because they managed the rehabilitation exercises in the proper manner. As a part of the rehabilitation program or physical examinations, these movements are usually used by the clinicians in situations like post-surgery recovery, upper and lower body conditions. The ultimate goal for selecting these movements was to assess the motion trajectories being performed consistently by the subjects. The utility of gathering these data sets was mainly for mathematical modeling and analyzing the movement sets. The data are gathered in segmented form, by dividing each movement into various episodes. Also, movements that are performed in incorrect and non-optimal manner are also included in the data sets[1]. Gathering data related to non-optimal performance was essential to assess the performance of the movements which were demonstrated to the patient by the professional. It is generally presumed that patients with injuries would be unable to perform these exercises in a uniform or optimal manner. Evaluation of the mathematical model to ensure the best results for correct movements and conveying the score to the patients by a system that is automatic for analyzing the movements. The collections of data were made in separate folders and were presented in ASCII text format. Each of the folders contained the files with respect to the measurements. Also, the improper movements on the basis of discrepancy in the execution of the exercises, are also contained in separate folders.

Methodologies

The methodologies used in the collection of the data were two sensory systems: Microsoft Kinect Camera and Vicon optical tracking system. The Vicon tracker which is a prolific system is designed to capture motion and analyze the same. Around eight cameras were used having characteristics of speed and resolution for the purpose of tracing a set of “retroreflective markers” ([1]). The markers were attached to planned areas on the body. Thereafter the system planned the positions of the markers and used the information for retrieving the

orientations of individual body parts.

Another method of collecting data was using a Kinect sensor which consisted of cameras that can read colors and an infrared camera which was simultaneously used for acquiring images and data. Microsoft had released a "software development kit" (SDK) which provided the libraries access streams, codes, and a skeletal tracker which had the ability to capture real-time motion. This Kinect sensor could trace the motions of 6 people and up to 25 joints of each person. Microsoft offered this sensor as a standalone unit for Windows operating system, because of its demand amongst researchers, hobbyists, and the industry.

For Vicon and Kinect the frame rate of motion capture was 100Hz and 30 Hz respectively. Also for Vicon and Kinect measurements, the angles are expressed in degrees and the cartesian position values. Because the angles were limited in a particular range, the large jumps in angle measurements were corrected with Kinect. The subjects had an average age of 29.3 years with a deviation of 5.85 years which was standard. They were mostly consistent in their performance while the data was recorded (Lessard et al. 2018). The intent was to provide basic information on the changeability of the movement and data related to the same. It was also decided that in the future the research would include investigating other metrics for explaining the steadiness of the subject's actions. Also, all the participants had given their written consent before the study was conducted.

Limitation and future scope

While summarising the whole research paper, which consisted of 10 exercises being performed by healthy subjects. The same was being recorded with motion capturing systems namely Microsoft Kinect Camera and Vicon optical tracking system. The aim of the research paper was to provide the data sets relating to the exercises involved in physical therapy and patient rehabilitation. The main purpose for the same was to assess modeling and assessment of the actions. Some of the shortcomings which are present in the research paper are that none of the movements were performed by any patients or injury-stricken persons; the subjects chosen were all healthy. Since there were no patients who were chosen as subjects for performing the movements, the actual assessment is incomplete. The goal of executing movements which are non-optimal in nature and which represent a unorthodox man-

Figure 4.19: Different exercises for fitness

ner from the correctly executed movement could not be fulfilled due to the presence of all healthy subjects. Another shortcoming would be that the movements chosen were general and had no connection with any specific condition or any specific movement which is shown by any patient. Hence the appropriate evaluation is incomplete [3]. Also, addressing problems related to machine learning requires a large set of subjects with a large set of data to be evaluated but here the subjects chosen were few in number, and the data being evaluated also lacked numerical advantage. If these shortcomings can be rectified, it can be rightly presumed that the research in the future would be capable of perfectly determining the modeling of mathematical aspects of data with the use of machine learning methods.

Chapter 5

AQAPRE Methodology and Implementation

This chapter provides an outline for the steps being followed to find a solution to the problem statement.

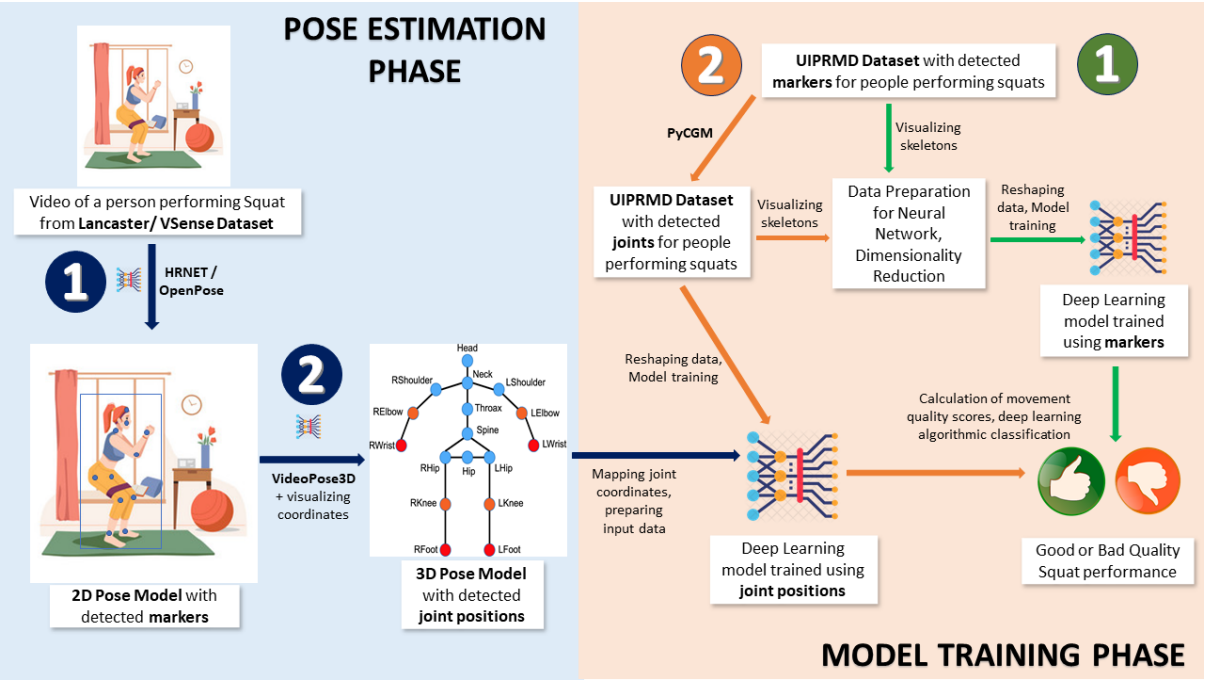


Figure 5.1: Overview of Approach

The research question that we're targeting is to find out if a person is performing a good or bad squat, given his/her video. The solution to such a problem isn't as facile as it seems to

be. For providing a better understanding of the same, the whole process has been divided into two phases, namely Pose Estimation Phase and Model Training Phase. The same has been described in detail below.

5.1 Pose Estimation

This phase majorly deals with the preparation of test data for the input to the joint positions based Action Quality Assessment Keras model that is going to be trained in the other phase.

Before moving further in the current phase of approach, it is important to gain knowledge regarding the test data. The below-mentioned datasets are utilised for preparation of test data and have been gathered from the VSense Research group at Trinity College Dublin.

- VSense Dataset: This dataset consists of videos of a researcher performing squats and it is expected that some of the squats performed by the researcher will be classified for bad quality.
- Lancaster Dataset: This dataset consists of videos of professional players performing squats and it is expected that most of the squats performed by them will be classified for good quality.

After analysing all the videos, around 11 videos from each dataset (i.e. 22 videos in total) have been gathered based on factors such as video recorded from side view or frontal view, squat performed with or without stick and dumbbell, elements placed in background etc.

Referring back to Figure 5.1, the following describes the steps followed for preparation of data:

5.1.1 Step 1: Obtaining markers from video of a person performing Squat

The input video data utilised in this step is Lancaster data and the implementation approach for this has been adapted from deepHRNET repository wherein HRNET is trained

with COCO dataset inclusive of more than 200K images with person instances labelled for 17 keypoints. For the same, COCO API is used to detect a person in the frame using Faster-RCNN network and then markers are attached upon keypoints detection.

As can be seen in the following code snippet, pytorch tensor transforms are utilised to pass an image frame from the videos to HRNET and bounding boxes are then interpreted from a single person frame. It is to be noted that pytorch tensors act in a similar manner as python numpy arrays do. These tensors possess a higher probability to get faster results as compared to numpy arrays due to their compatibility with Graphical Processing Unit (GPU). Although we can also detect multi-person bounding boxes, we have kept this research limited to videos of single person performing squats. The image is loaded and it Referring to line 7 of the code, it can be seen that a check is kept to find which prediction class of the COCO instance category names does the picture consist of. As previously described, there exist a number of COCO instance categories wherein "person" is the one the code is looking for. In this case, a check on prediction class being a "person" has been kept as shown in line 16 of the code. Further, another check on the selection of the box has been kept to ensure that the selected bounding box is big enough and actually consisting of the person.

```
1  def get_person_detection_boxes(model, img, threshold=0.5):
2      pil_image = Image.fromarray(img)  # Load the image
3      transform = transforms.Compose([transforms.ToTensor()])  # Defining
       PyTorch Transform
4      transformed_img = transform(pil_image)  # Apply the transform to
     the image
5      pred = model([transformed_img.to(CTX)])  # Pass the image to the
     model
6      # Use the first detected person
7      pred_classes = [COCO_INSTANCE_CATEGORY_NAMES[i]
8                      for i in list(pred[0]['labels'].cpu().numpy())]  #
     Get the Prediction Score
9      pred_boxes = [[(i[0], i[1]), (i[2], i[3])]
10                   for i in list(pred[0]['boxes'].cpu().detach().numpy()
     )]  # Bounding boxes
11     pred_scores = list(pred[0]['scores'].cpu().detach().numpy())
12
13     person_boxes = []
14     # Select box has score larger than threshold and is person
```

```
15      for pred_class, pred_box, pred_score in zip(pred_classes,
    pred_boxes, pred_scores):
16          if (pred_score > threshold) and (pred_class == 'person'):
17              person_boxes.append(pred_box)
18
19      return person_boxes
```

Listing 5.1: Function for obtaining single person detection boxes

Once the bounding boxes are estimated, the pose model deepHRNET [17] detects the keypoint coordinates for the person in the video.

The following code snippet depicts the transforms (eg. cropping, scaling etc.) being applied before passing the inputs to the model. The "pose model" for inference in line 20 is the pretrained deepHRNET model whose technical architecture and functioning has been described in Chapter 4.

```
1  def get_pose_estimation_prediction(pose_model, image, center, scale):
2      rotation = 0
3
4      # pose estimation transformation
5      trans = get_affine_transform(center, scale, rotation, cfg.MODEL.
    IMAGE_SIZE)
6      model_input = cv2.warpAffine(
7          image,
8          trans,
9          (int(cfg.MODEL.IMAGE_SIZE[0]), int(cfg.MODEL.IMAGE_SIZE[1])),
10         flags=cv2.INTER_LINEAR)
11     transform = transforms.Compose([
12         transforms.ToTensor(),
13         transforms.Normalize(mean=[0.485, 0.456, 0.406],
14                              std=[0.229, 0.224, 0.225]),
15     ])
16     # pose estimation inference
17     model_input = transform(model_input).unsqueeze(0)
18     # switch to evaluate mode
19     pose_model.eval()
20     with torch.no_grad():
21         # compute output heatmap
```

```
22          output = pose_model(model_input)
23          preds, _ = get_final_preds(
24              cfg,
25              output.clone().cpu().numpy(),
26              np.asarray([center]),
27              np.asarray([scale]))
28          return preds
```

Listing 5.2: Function for obtaining 2D pose estimate for given video

The outputs generated from this model are analysed in the evaluation section.

5.1.2 Step 2: Obtaining joint coordinates from a video consisting of markers along with a person performing Squat

The output from HRNET is then fed as an input to VideoPose3D trained with Human3.6M dataset. For the same, an inference script had been adapted from [36] inference files to convert the 2D pose model to 3D pose model which detects boxes, segments and keypoints using Detectron2 [53]. The output from VideoPose3D inference consisted of a .npz file which is a numpy zip archive. The same was then analysed using Python and it included 4 .npy files namely, 'boxes', 'segments', 'keypoints', 'metadata'.

The following code snippet is used to find out the data in the keypoints file. Line 4 tells the shape of keypoints array which has come out to be (148,2) meaning that there are 2 columns and 148 entries for arrays. Line 5 and 6 shows the entries for each column respectively and it is found that column 0 is empty while column 1 consists of the keypoint entries. Line 7 shows the joint coordinates entry for 140th frame of the video.

```
1 data = load('WHS50_pose.avi.npz', allow_pickle=True)
2 lst = data.files
3 print(lst)
4 print(data['keypoints'].shape)
5 print(data['keypoints'][0][0])
6 print(data['keypoints'][0][1])
7 print(data['keypoints'][[140][0]])
```

Listing 5.3: Code to describe data

```
['boxes', 'segments', 'keypoints', 'metadata']
(148, 2)


[]


[[[9.3787115e+02 9.4647473e+02 9.1492853e+02 9.5794598e+02 8.7047717e+02
   9.8375653e+02 8.3893097e+02 1.0712253e+03 9.3070160e+02 9.5937994e+02
   9.8949219e+02 9.7945477e+02 8.8194843e+02 9.7945477e+02 8.5183630e+02
   9.8519049e+02 8.2459192e+02]
  [1.9880882e+02 1.8154556e+02 1.8298416e+02 1.9737022e+02 2.0024742e+02
   2.9663400e+02 3.2972192e+02 3.7575732e+02 4.0884525e+02 3.2396750e+02
   3.4554657e+02 6.0161835e+02 6.1168860e+02 7.4691748e+02 7.8000543e+02
   8.7783057e+02 9.4256781e+02]
  [0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
   0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
   0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00 0.0000000e+00
   0.0000000e+00 0.0000000e+00]
  [2.2340438e+00 1.8431219e+00 2.1154020e+00 4.7755197e-01 1.3096163e+00
   3.8286084e-01 2.4906754e-01 6.2523216e-01 1.1647854e+00 2.7963081e-01
   3.5544416e-01 1.9028552e-01 1.8564749e-01 5.3999847e-01 4.3435523e-01
   5.0811547e-01 3.0250424e-01]]]
```

Figure 5.2: VideoPose3D inference output keypoints

Deciphering the code and printing the contents of .npy files, it was clear that the only file required to proceed further was "keypoints". However, it was observed that each frame entry had some zero filled rows in between. Going through the code again, it was found that the following part of code had added the dummy values in between (refer line 5,6):

```
1            if has_bbox:
2                kps = outputs.pred_keypoints.numpy()
3                kps_xy = kps[:, :, :2]
4                kps_prob = kps[:, :, 2:3]
5                kps_logit = np.zeros_like(kps_prob) # Dummy
6                kps = np.concatenate((kps_xy, kps_logit, kps_prob),
   axis=2)
7                kps = kps.transpose(0, 2, 1)
```

Listing 5.4: Code snippet from VideoPose inference script

This states that the keypoints obtained from kps array were given to the output as it is. Then,

```
[list([])
 array([[[9.83052429e+02, 9.90231812e+02, 9.60078491e+02, 1.00171881e+03,
          9.14130554e+02, 1.02038513e+03, 8.82541321e+02, 1.13238318e+03,
          9.44283875e+02, 1.03330798e+03, 1.05197437e+03, 9.78744812e+02,
          8.75361938e+02, 1.04048743e+03, 8.62439148e+02, 1.00171881e+03,
          8.38029236e+02],
         [2.84721985e+02, 2.73216705e+02, 2.68902222e+02, 2.94789124e+02,
          2.94789124e+02, 3.85393372e+02, 3.88269684e+02, 3.82517029e+02,
          3.99774963e+02, 3.72449890e+02, 3.72449890e+02, 6.38509888e+02,
          6.50015198e+02, 7.85202454e+02, 8.44167175e+02, 9.21827820e+02,
          9.98050476e+02],
         [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
          0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
          0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
          0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00,
          0.00000000e+00],
         [1.17840374e+00, 8.31824660e-01, 8.00455749e-01, 4.79131520e-01,
          2.08560848e+00, 2.88715035e-01, 2.90150195e-01, 4.92030412e-01,
          1.25735462e-01, 1.08417735e-01, 2.64067203e-01, 2.35259727e-01,
          1.82737052e-01, 6.20259523e-01, 3.64484012e-01, 5.53154826e-01,
          3.30659032e-01]]], dtype=float32)]
```

Figure 5.3: Analysing VideoPose3D inference output keypoints

a dummy array with 0 values was given. Next to that, a "z" coordinate with maximum probability was added. Taking this into consideration, our code removed dummy zeros and prepared a file with "X,Y,Z" columns pertaining to each of the 17 joint coordinates. Along with that, a visualisation script was prepared to ensure that the correct joint names are known for the 17 joints and the joint order is then arranged accordingly for input to the model. Array reshaping and python pandas concatenation functions were used for the same.

However, after many trials, it was found that the visualisation script was giving vague outcomes for joint positions. Hence, it became difficult to find out the correct joint names even after multiple tries. In order to proceed with a better way, Human3.6M license was required for using complete VideoPose3D repository wherein I posed many challenges to obtain a license despite much efforts.

5.2 Model Training

This section utilises a recent research work done on Action Quality Assessment last year. [23] utilised markers of squat exercises performed by 10 people (subjects) gathered from the UIPRMD dataset. Each subject performed 10 repetitions for each exercise.

5.2.1 Step 1: Model training on Markers

The model was retrained from scratch, first gathering the UIPRMD dataset, visualising it, preparing input data using segmented and unsegmented UIPRMD dataset values, using autoencoders to reduce dimensionality for the movement data and then inputting the data to Spatio-temporal deep learning model whose architecture has already been described in Chapter 4. The sensor data from Vicon sensors of UIPRMD dataset was utilised for the implementation.

5.2.2 Step 2: Model training on Joint Positions

For training the model using Joints, the UIPRMD data was passed into PyCGM to obtain the appropriate joint values for retraining the model. The output files from PyCGM consisted of the following:

- x1d: Vicon unprocessed data
- x2d: Dynamic, interactive graphics file format using XML to describe dynamic and interactive graphical objects
- c3d: contains geometric object data for a 3D model
- xcp: associated with the XLNT (EXtended Language for Windows NT) script command files.
- csv: comma separated values
- system: Vega Strike (a space simulator game) Virtual System File

- trial.enf: EndNote Filter File (used for importing bibliographic references)
- history: Eclipse IDE local history data

Post studying all these files, the PyCGM output was visualised and the following can be seen as examples of Good and Bad Squat from the data.

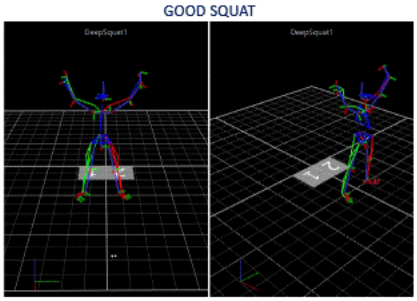


Figure 5.4: Good Squat

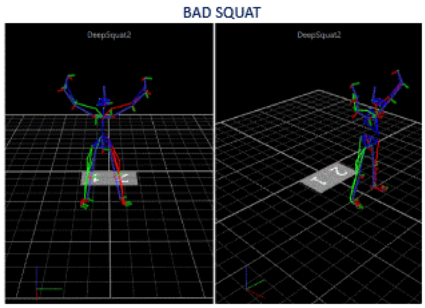


Figure 5.5: Bad Squat

A python script was then prepared to get the actual joint centers combined overall in order to send them as an input to the matlab code for data preparation. The function for the same is as follows:

```
def correctA_dataset(files):
    df= pd.DataFrame()
    j=1
    for file1 in files:
        print(file1)
        df1=pd.DataFrame()
        df1 = pd.read_csv(file1, header=None)
```

```
        N = 7
        df1.drop(index=df1.index[:N], axis=0, inplace=True)
        df1.drop(columns=df1.columns[0], axis=1, inplace=True)
        df1=df1.dropna(axis=1,how='all')
        df_split = np.array_split(df1, 10)
        print(df1)
        for i in range(0,10):
          df_split[i].to_csv("DeepSquat1_A_s"+str(j)+"e"+str(i)+".csv",
    header=False, index=False)
        j=j+1
```

Listing 5.5: Code for combining correct data for joint positions

Once the data was combined, 9 subjects with 10 repetitions each were included. This was then entered to a matlab script for normalisation, scaling and analysing distance between joint centers and preparing the data for input to the autoencoder. While processing the data, there were 10 repetitions within the dataset which were removed due to broken inputs upon visualising. The inconsistent data having measurement errors or subjects performing the exercise with one arm or leg were manually removed. Therefore, 9*10-10=80 subjects were there for training phase. Both the markers dataset and our joint dataset had the person standing-sitting-standing for one repetition of squat.

The data was then fed to an autoencoder for encoding and decoding the sequences and get reduced dimensionality. Major change to the adapted code here was understanding to give the correct inputs to the autoencoder which were found to be no. of markers= 57 (meaning joints=19) instead of no. of markers= 117 in the original keras model. After dimensionality reduction, labels were created for the input data and then the data was loaded to train the model.

As discussed in architecture of the network in Chapter 4, the data length is being reduced for sending them as an input to the sub-networks.

```
# Reduce the data length by a factor of 2, 4, and 8
# The reduced sequences will be used as inputs to the temporal pyramid
    subnetwork
train_x_2 = np.zeros((train_x.shape[0], int(train_x.shape[1]/2),
    train_x.shape[2]))
valid_x_2 = np.zeros(train_x_2.shape)
```

```
train_x_4 = np.zeros((train_x.shape[0], int(train_x.shape[1]/4),
    train_x.shape[2]))
valid_x_4 = np.zeros(train_x_4.shape)
train_x_8 = np.zeros((train_x.shape[0], int(train_x.shape[1]/8),
    train_x.shape[2]))
valid_x_8 = np.zeros(train_x_8.shape)
train_x_2 = train_x[:,::2,:]
valid_x_2 = valid_x[:,::2,:]
train_x_4 = train_x[:,::4,:]
valid_x_4 = valid_x[:,::4,:]
train_x_8 = train_x[:,::8,:]
valid_x_8 = valid_x[:,::8,:]
```

Listing 5.6: Code for reducing data length for input to temporal pyramid subnetwork

The architecture being followed takes into account five recurrent sub-networks taking as inputs joint displacements of the left arm, right arm, left leg, right leg, and torso, respectively. The values for reordering the joint positions have been calculated from the dataset post visualisation and have been entered thereafter.

```
def reorder_data(x):
    X_trunk = np.zeros((x.shape[0],x.shape[1],4))
    X_left_arm = np.zeros((x.shape[0],x.shape[1],6))
    X_right_arm = np.zeros((x.shape[0],x.shape[1],6))
    X_left_leg = np.zeros((x.shape[0],x.shape[1],7))
    X_right_leg = np.zeros((x.shape[0],x.shape[1],7))
    X_trunk =  np.concatenate((x[:,:,5:6], x[:,:,6:7], x[:,:,8:9], x
    [:,:,9:10]), axis = 2)
    X_left_arm = np.concatenate((x[:,:,27:28], x[:,:,29:30], x
    [:,:,31:32], x[:,:,33:34], x[:,:,35:36], x[:,:,37:38]), axis = 2)
    X_right_arm = np.concatenate((x[:,:,28:29], x[:,:,30:31], x
    [:,:,32:33], x[:,:,34:35], x[:,:,36:37], x[:,:,38:39]), axis = 2)
    X_left_leg = np.concatenate((x[:,:,11:12], x[:,:,13:14], x
    [:,:,15:16], x[:,:,17:18], x[:,:,19:20], x[:,:,21:22], x[:,:,23:24])
    , axis = 2)
    X_right_leg = np.concatenate((x[:,:,12:13], x[:,:,14:15], x
    [:,:,16:17], x[:,:,18:19], x[:,:,20:21], x[:,:,22:23], x[:,:,24:25])
    , axis = 2)
    x_segmented = np.concatenate((X_trunk, X_right_arm, X_left_arm,
```

```
    X_right_leg, X_left_leg),axis = -1)
    return x_segmented
```

Listing 5.7: Code for reordering data of Joints based model

The model training implementation of the pyramid network defined in Chapter 4 is as follows:

```
def TempPyramid(input_f, input_2, input_4, input_8, seq_len, n_dims):

    #### Full scale sequences
    conv1 = MultiBranchConv1D(input_f, 64, 3, 2, 2)

    #### Half scale sequences
    conv2 = MultiBranchConv1D(input_2, 64, 3, 2, 1)

    #### Quarter scale sequences
    conv3 = MultiBranchConv1D(input_4, 64, 3, 1, 1)

    #### Eighth scale sequences
    conv4 = MultiBranchConv1D(input_8, 64, 3, 1, 1)
    upsample1 = UpSampling1D(size = 2)(conv4)

    #### Recurrent layers
    x = concatenate([conv1, conv2, conv3, upsample1], axis=-1)
    return x
```

Listing 5.8: Code for defining a temporal pyramid network

In the below mentioned code snippet, it is to be noted that the dimensions of trunk, arms, legs of the joint based model have been calculated and pertain to the joint based outputs of the model. The same is then fed to the Neural Network.

```
n_dim = 30 # dimension after segmenting the data into body parts
n_dim1 = 4 # trunk dimension
n_dim2 = 6 # arms dimension
n_dim3 = 7 # legs dimension

# Build the model ...

```

```
#### Full scale sequences
seq_input = Input(shape = (timesteps, n_dim), name = 'full_scale')

seq_input_trunk = Lambda(lambda x: x[:, :, 0:4])(seq_input)
seq_input_left_arm = Lambda(lambda x: x[:, :, 4:10])(seq_input)
seq_input_right_arm = Lambda(lambda x: x[:, :, 10:16])(seq_input)
seq_input_left_leg = Lambda(lambda x: x[:, :, 16:23])(seq_input)
seq_input_right_leg = Lambda(lambda x: x[:, :, 23:30])(seq_input)

#### Half scale sequences
seq_input_2 = Input(shape=(int(timesteps/2), n_dim), name='half_scale')

seq_input_trunk_2 = Lambda(lambda x: x[:, :, 0:4])(seq_input_2)
seq_input_left_arm_2 = Lambda(lambda x: x[:, :, 4:10])(seq_input_2)
seq_input_right_arm_2 = Lambda(lambda x: x[:, :, 10:16])(seq_input_2)
seq_input_left_leg_2 = Lambda(lambda x: x[:, :, 16:23])(seq_input_2)
seq_input_right_leg_2 = Lambda(lambda x: x[:, :, 23:30])(seq_input_2)

#### Quarter scale sequences
seq_input_4 = Input(shape=(int(timesteps/4), n_dim), name='
    quarter_scale')

seq_input_trunk_4 = Lambda(lambda x: x[:, :, 0:4])(seq_input_4)
seq_input_left_arm_4 = Lambda(lambda x: x[:, :, 4:10])(seq_input_4)
seq_input_right_arm_4 = Lambda(lambda x: x[:, :, 10:16])(seq_input_4)
seq_input_left_leg_4 = Lambda(lambda x: x[:, :, 16:23])(seq_input_4)
seq_input_right_leg_4 = Lambda(lambda x: x[:, :, 23:30])(seq_input_4)

#### Eighth scale sequences
seq_input_8 = Input(shape=(int(timesteps/8), n_dim), name='eighth_scale
    ')

seq_input_trunk_8 = Lambda(lambda x: x[:, :, 0:4])(seq_input_8)
seq_input_left_arm_8 = Lambda(lambda x: x[:, :, 4:10])(seq_input_8)
seq_input_right_arm_8 = Lambda(lambda x: x[:, :, 10:16])(seq_input_8)
seq_input_left_leg_8 = Lambda(lambda x: x[:, :, 16:23])(seq_input_8)
seq_input_right_leg_8 = Lambda(lambda x: x[:, :, 23:30])(seq_input_8)

concat_trunk = TempPyramid(seq_input_trunk, seq_input_trunk_2,
```

```
    seq_input_trunk_4, seq_input_trunk_8, timesteps, n_dim1)
concat_left_arm = TempPyramid(seq_input_left_arm, seq_input_left_arm_2,
     seq_input_left_arm_4, seq_input_left_arm_8, timesteps, n_dim2)
concat_right_arm = TempPyramid(seq_input_right_arm,
    seq_input_right_arm_2, seq_input_right_arm_4, seq_input_right_arm_8,
     timesteps, n_dim2)
concat_left_leg = TempPyramid(seq_input_left_leg, seq_input_left_leg_2,
     seq_input_left_leg_4, seq_input_left_leg_8, timesteps, n_dim3)
concat_right_leg = TempPyramid(seq_input_right_leg,
    seq_input_right_leg_2, seq_input_right_leg_4, seq_input_right_leg_8,
     timesteps, n_dim3)

concat = concatenate([concat_trunk, concat_left_arm, concat_right_arm,
    concat_left_leg, concat_right_leg], axis=-1)

rec = LSTM(80, return_sequences=True)(concat)
rec1 = LSTM(40, return_sequences=True)(rec)
rec1 = LSTM(40, return_sequences=True)(rec1)
rec2 = LSTM(80)(rec1)

out = Dense(1, activation = 'sigmoid')(rec2)

from tensorflow import keras
model = Model(inputs=[seq_input, seq_input_2, seq_input_4, seq_input_8
    ], outputs=out)
opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(loss='binary_crossentropy', optimizer= opt)
```

Chapter 6

Evaluation and Results

This chapter outlines the key takeaways from the undertaken research.

6.1 Pose Estimation

Figure 6.1: 2D Pose Analysis with HRNET (stick in background)

In this animated image (Fig. 6.1), it is noted that the stick staying in the background doesn't cause any concern for the detection of the bounding box and the person's keypoints. It is

also observed that markers are re-detected in every frame, even if a person just moves a little bit to correct his standing pose (as can be seen on the right side of the image).

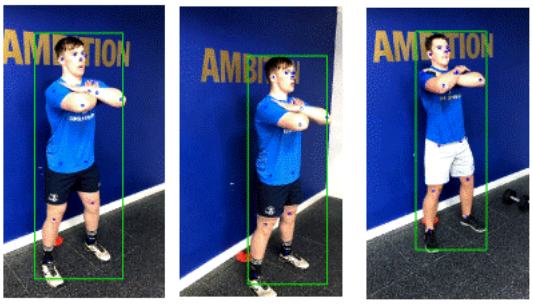


Figure 6.2: 2D pose analysis with HRNET (from different camera views)

The animated image (Fig 6.2) consists of one person performing a squats with their hands folded, captured both from side view and a relatively frontal view along with another person's video who is performing the exercise faster than these. It is observed that the markers for wrists collide for such cases which reflects precision in evaluation of HRNET. However, in such a case, the markers for chin and elbow almost collide when sitting up and down.

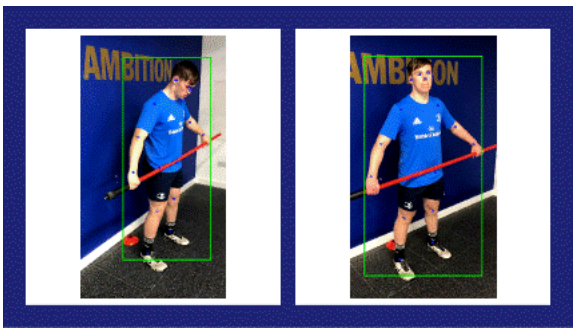


Figure 6.3: 2D pose analysis HRNET (with stick in hand)

The HRNET output of a person performing squat in this figure 6.3 shows that having a stick in hand while performing exercise doesn't correspond to incorrect keypoint detection by HRNET. Further, reflecting on the squat performed by the person on the left, it can be said that the markers are at least detected for invisible key points such as ears, elbows on the other side rather than skipping them. Moreover, it can be noticed that with the increase in speed of standing up, the key points deviate and can result in some change to the mean

values. For the person on the right side in the image, his video has been captured relatively from the front view and the difference in keypoint evaluation of the squat has been observed as the markers are better visible from this view.

Figure 6.4: 2D pose analysis HRNET (with dumbell in hand)

As observed from the figure 6.4, it can be said that having a dumbbell in hand doesn't cause much trouble in evaluating keypoints if the video is recorded from a side view. However, for the frontal view, dumbbell causes additional keypoints to be detected. It can be hereby inferred that something having more volumatic structure may cause the model to act differently.

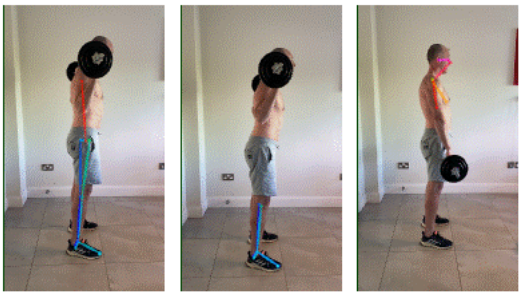


Figure 6.5: 2D pose analysis with OpenPose

The OpenPose based output of a person performing both good and bad quality squats is shown in figure 6.5. It is observed that OpenPose is good at detecting full body key points altogether, however, it acts in a restricted manner in presence of some object or when video is recorded from side view. In these figures, it is clearly reflected that only one side key points have been detected by OpenPose instead of taking into account the invisible coordinates.

Moreover, the presence of the rod in the person's hand has been significantly affecting the detections for OpenPose. Henceforth, OpenPose hasn't been included as a valuable asset to this research at any point.

6.2 Model Training

Data Preparation phase:

Techniques such as normalisation of data (splitting data into values between 0 to 1), data centering, calculation of distance between joints using functions such as log likelihood calculation etc. have been performed during "data correct" and "data incorrect" files preparation phase. The output of the following phase which produced different values for the joint-based model as shown below wherein the changed "n dim" value clearly reflects the changes incurred by inputting joint angles in place of markers as input data.

Figure 6.6: Data preparation comparisons for Marker-based and Joint-based models

Dimensionality Reduction:

The data was then passed to autoencoders which produced the results given below. The more alterations in loss value for joint positions based models indicate the limited availability of joint movement data sent to the encoder as compared to the marker based model. It is also valuable to notice the changes in encoded representation for the marker vs joint based

data. The reason behind the same is observed to be less data dimensions. However, it has been observed that this has no significant effect on the overall outputs.

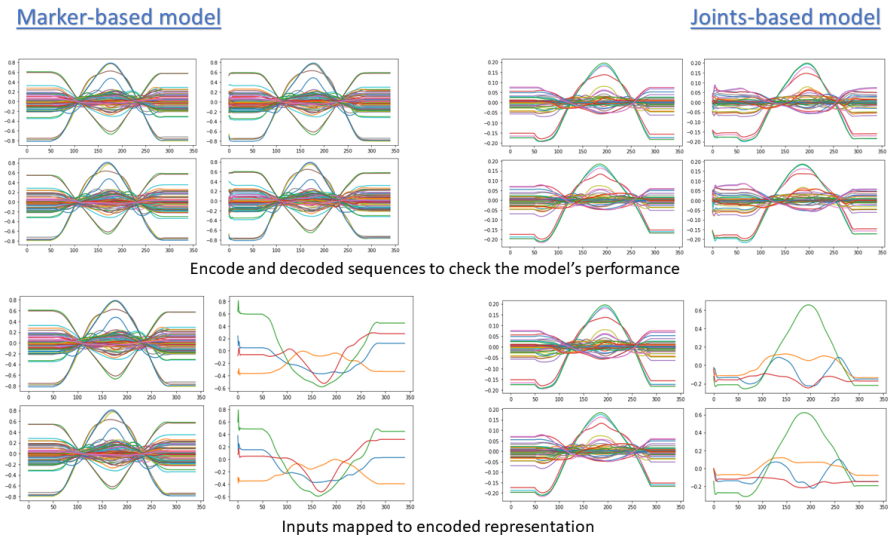


Figure 6.7: Autoencoder output comparisons for Marker-based and Joint-based models

Figure 6.8: Autoencoder output comparisons

Comparisons between model training outputs for Joint based and Marker based models:
As shown below, the training of joint based model keeping all parameters same (for eg. learning rate, optimizer, dropout layers, activation functions etc.) as the marker based model has provided the outcomes to have comparatively less training and validation loss. This indicated that the model might be overfitting on the provided data. Hence, the learning rate was

changed to 0.001 which helped to find a better learning curve and ensured that the trained model produced similar results as compared to the original marker based model. This can be seen through the values of mean absolute deviation coming out to be 0.009540543978214275 , 0.00974472374598185 for markers and joint based models respectively. Further, it can be verified by values of 0.014649246638018216, 0.013899451940783273 for root mean square deviation of markers, joint based models respectively.

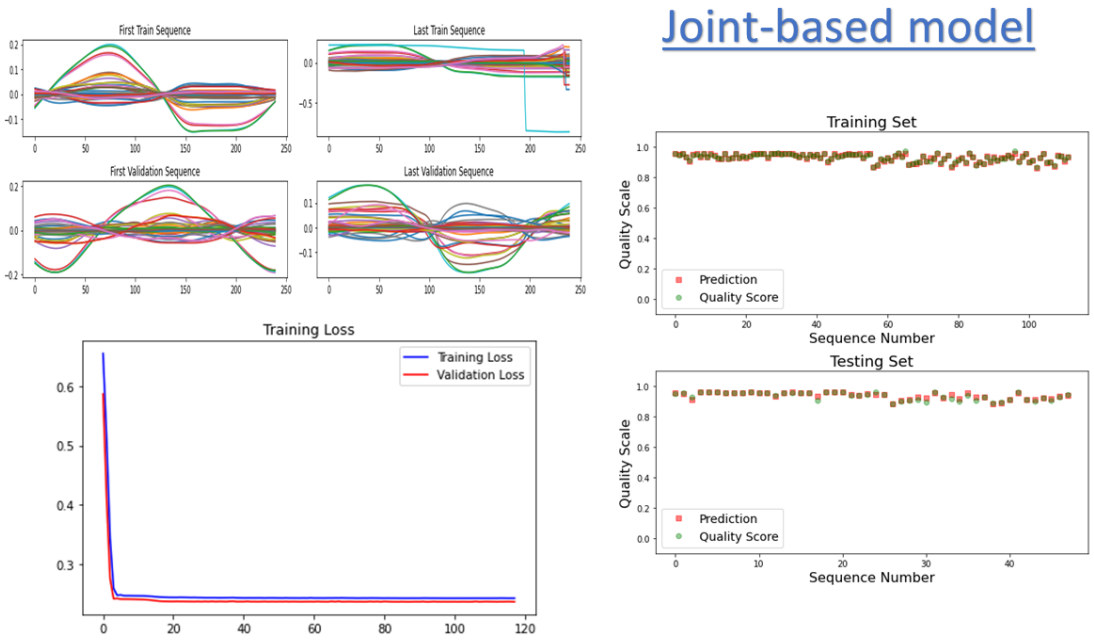


Figure 6.9: Model training outputs for Joint-based model

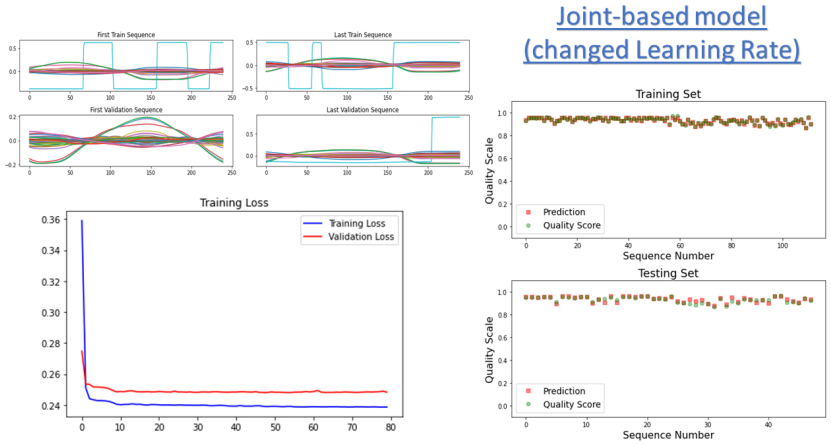


Figure 6.10: Model training comparisons for Marker-based and Joint-based models

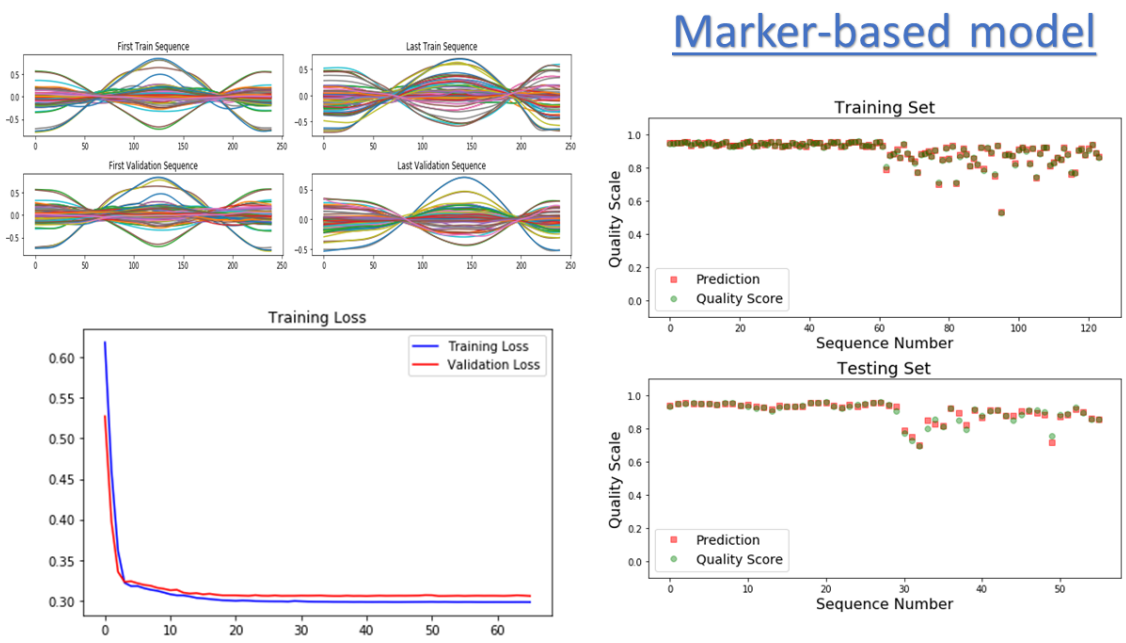


Figure 6.11: Model training outputs for marker-based model

	Markers-based model	Joint-based model	Joint based model (changed Learning Rate)
Timesteps	240	240	240
Squat repetitions	90	80	80
Subjects	10	9	9
Dimension of data sequences	117	57	57
Correct Sequences	(90, 240, 117)	(80, 240, 117)	(80, 240, 117)
Incorrect Sequences	(90, 240, 117)	(80, 240, 117)	(80, 240, 117)
Training data	(124,240,117)	(112, 240, 57)	(112, 240, 57)
Validation data	(56, 240, 117)	(48, 240, 57)	(48, 240, 57)
Training Time	0:17:52.663051	0:20:00.492997	0:09:27.867041
Learning Rate	0.0001	0.0001	0.001
Training Loss	0.2981928517260859	0.24313309788703918	0.23867109417915344
Validation Loss	0.3056857282561915	0.23716019093990326	0.2480446696281433
Mean absolute deviation	0.009540543978214275	0.006141650011936822	0.00974472374598185
Root mean square deviation	0.014649246638018216	0.009661130423161463	0.013899451940783273

Figure 6.12: Model training outputs for altered Joint-based model

Chapter 7

Conclusions and Future Work

With smartphones in reach of most people today and teleheath becoming popular, exploring possibilities to assist physiotherapists and individuals by utilising technology can be one of the major emerging fields. It is observed that deep learning has proved itself to be one of the promising technologies in many fields and have been becoming prominent in the areas of human pose estimation, keypoints detection. Review of various recent research works have revealed that even huge techgiants like Google and Facebook have been working on real-time keypoint detection or pose estimation models such as Mediapipe, PiFuHD and VideoPose3D respectively. This clearly reflects the importance of real-time estimation of human poses. Looking at the recent advances in physiotherapy exercise assessment with the help of technology, there is relatively less work done in this area. Recently, a dataset of 11 subjects performing 10 repetitions each of physiotherapy rehabilitation exercises was captured in a motion capture laboratory. The same has been utilised in a novel, first well-known research on action quality assessment of physiotherapy rehabilitation exercises. This dissertation has made efforts to analyse that research and make further advancements to the research in terms of organising a framework to analyse action quality on camera-based videos. For implementation of the same, a major requirement was to enable the model to take 3D joints as an input to the model instead of 2D markers so that the camera-video based test data can be taken as an input to the model. The undertaken research has been successful in performing such enhancement to the model by first converting the 2D markers to 3D joints for the training data, then visualising and evaluating the prepared data, reduc-

ing dimensionality and training the model on 3D joints. It is important to note that the 2D markers data had around 117 dimensions for 10 subjects performing 10 repetitions each of squats whereas the 3D joints data has resulted to have 57 dimensions with 9 subjects performing 10 repetitions each of squats. One subject is missing in the joint based data due to its missing Vicon camera values while preparation of the data. Finally, a total of 19 joint positions has been observed for training the model and the same is reordered to trunk, left arm, right arm, left leg, right leg post analysis of the data and then the model has been trained. The new model has yielded near to similar results as compared to the previous one on the model validation data. On the other hand, test data from camera-based videos has been obtained from VSense Research group. The data has been analysed on HRNET and for an example on OpenPose to see which one is more suitable to estimate 2D pose model and attach marker values. It is observed that the HRNET provides better outputs as it neglects the background or additional elements while focusing most on the person detection keypoints whereas OpenPose provides missing joint values where any such element (eg. stick) comes into picture. In order to further give this data as an input to the action quality assessment model, the output from 2D pose model has been further passed to 3D pose model wherein VideoPose3D has been utilised. However, the original VideoPose3D with Human3.6M has been inaccessible as the Human3.6M dataset is required to make inferences about 3D joint coordinates. The license for the same can be obtained only for specific research and wasn't available to use despite sending an application on my end. Therefore, an inference script for VideoPose3D with Detectron2 was utilised which takes COCO keypoints as an input and provide a Numpy Archive file as output. The keypoints archive file have been processed and 17 joints have been obtained. The mapping of the joint names corresponding to the 17 obtained joints can become scope for further research in this area. Once the mapping is obtained, next steps could be reordering the same to match with the desired inputs for Keras model and analyse the outcomes. Upon the successful completion until this point, future scope can be optimizing the training of Action Quality Assessment model, development of mobile or web apps for classification of outputs or research work pertaining to security and privacy of deep learning models.

Bibliography

[1] Vakanski A, Paul D Jun HP, and Baker R. A data set of human body movements for physical rehabilitation exercises. 0 2018.

[2] Oludare Isaac Abiodun, A. Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat Mohamed, and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4, 2018.

[3] Md. Atiqur Rahman Ahad, Anindya Das Antar, and Omar Shahid. Vision-based action understanding for assistive healthcare: A short review. 06 2019.

[4] Nadhir Ahmadlou, Mohammad, Nguyen Huu Du, Nguyen Linh, and Hedieh Sajedi. Flood susceptibility mapping and assessment using a novel deep learning model combining multilayer perceptron and autoencoder neural networks. *Journal of Flood Risk Management*, 13:1–22, 12 2020.

[5] Panagiotis G Asteris and Konstantinos G Kolovos. Data on the physical and mechanical properties of soilcrete materials modified with metakaolin. *Data in brief*, 13:487—497, August 2017.

[6] Babajide Ayinde and Jacek Zurada. Deep learning of constrained autoencoders for enhanced understanding of data. *IEEE Transactions on Neural Networks and Learning Systems*, PP:1–11, 09 2017.

[7] Harikrishnan Nellippallil Balakrishnan, Aditi Kathpalia, Snehanshu Saha, and Nithin Nagaraj. Chaosnet: A chaos based artificial neural network architecture for classification. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(11):113125, 2019.

[8] Wei Bao, Jun Yue, and Yulei Rao. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLoS ONE*, 12, 07 2017.

[9] Yu Cheng, Bo Yang, Bo Wang, and Robby T. Tan. 3d human pose estimation using spatio-temporal networks with explicit occlusion training, 2020.

[10] Rakesh Kr Mandal Dipanjan Moitra. Automated ajcc staging of non-small cell lung cancer (nsclc) using deep convolutional neural network (cnn) and recurrent neural network (rnn), 01 2019.

[11] Patrick Doetsch, Albert Zeyer, Paul Voigtlaender, Ilya Kulikov, Ralf Schlüter, and Hermann Ney. Returnn: The rwth extensible training framework for universal recurrent neural networks, 2017.

[12] A. Essien and C. (2020 Giannetti. A deep learning model for smart manufacturing using convolutional lstm neural network autoencoders. ieee transactions on industrial informatics. pages 6069–6078.

[13] Zhijie Fang and Antonio M. López. Is the pedestrian going to cross? answering by 2d pose estimation, 2018.

[14] Ye Hong, Yingjie Zhou, Qibin Li, Wenzheng Xu, and Xiujuan Zheng. A deep learning method for short-term residential load forecasting in smart grid. *IEEE Access*, PP:1–1, 03 2020.

[15] Yan L Huaiying, S. Research on technical architecture and application of big data cloud platform for electric power measurement., 01 2019.

[16] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2014.

[17] Weiqi Ji and Sili Deng. Kinet: A deep neural network representation of chemical kinetics, 08 2021.

[18] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision, 2021.

[19] Swathi Edem Kanchan M. Tarwani. Survey on recurrent neural network in natural language processing. 2017.

[20] Łukasz Kidziński, Bryan Yang, Jennifer L. Hicks, Apoorva Rajagopal, Scott L. Delp, and Michael H. Schwartz. Deep neural networks enable quantitative movement analysis using single-camera videos. *Nature Communications*, 11(1), December 2020. Funding Information: Our research was supported by the Mobilize Center, a National Institutes of Health Big Data to Knowledge (BD2K) Center of Excellence through Grant U54EB020405, and RESTORE Center, a National Institutes of Health Center through Grant P2CHD10191301. Publisher Copyright: © 2020, This is a U.S. government work and not under copyright protection in the U.S.; foreign copyright protection may apply.

[21] Nikos Kolotouros, Georgios Pavlakos, Michael J. Black, and Kostas Daniilidis. Learning to reconstruct 3d human pose and shape via model-fitting in the loop, 2019.

[22] Y. Liao, A. Vakanski, and M. Xian. A deep learning framework for assessing physical rehabilitation exercises. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 28(2):468–477, Feb. 2020.

[23] Yalin Liao, Aleksandar Vakanski, and Min Xian. A deep learning framework for assessing physical rehabilitation exercises. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 28(2):468–477, Feb 2020.

[24] Tsung-Yi Lin, M. Maire, Serge J. Belongie, James Hays, P. Perona, D. Ramanan, Piotr Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[25] Diogo Luvizon, David Picard, and Hedi Tabia. Multi-task deep learning for real-time 3d human pose estimation and action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP:1–1, 02 2020.

[26] Steven R. Machlin, Julia Chevan, William W. Yu, and Marc W. Zodet. Determinants of Utilization and Expenditures for Episodes of Ambulatory Physical Therapy Among Adults. *Physical Therapy*, 91(7):1018–1029, 07 2011.

[27] Batta Mahesh. Machine learning algorithms -a review, 01 2019.

[28] Erik Marchi, Fabio Vesperini, Stefano Squartini, and Björn Schuller. Deep recurrent neural network-based autoencoders for acoustic novelty detection. *Intell. Neuroscience*, 2017:4, January 2017.

[29] Elisabeta Marinoiu, Mihai Zanfir, Vlad Olaru, and Cristian Sminchisescu. 3d human sensing, action and emotion recognition in robot assisted therapy of children with autism. In *Proceedings - 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2018*, pages 2158–2167, United States, December 2018. IEEE Computer Society. 31st Meeting of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2018 ; Conference date: 18-06-2018 Through 22-06-2018.

[30] Julieta Martinez, Rayat Hossain, Javier Romero, and James J. Little. A simple yet effective baseline for 3d human pose estimation, 2017.

[31] William McNally, Kanav Vats, Tyler Pinto, Chris Dulhanty, John McPhee, and Alexander Wong. Golfdb: A video database for golf swing sequencing, 2019.

[32] 2020. Deep Learning's mathematics. Mebsout, I., 01 2019.

[33] Zarana Parekh, Jason Baldridge, Daniel Cer, Austin Waters, and Yinfei Yang. Crisscrossed captions: Extended intramodal and intermodal semantic similarity judgments for ms-coco, 2021.

[34] Georgios Pavlakos, Luyang Zhu, Xiaowei Zhou, and Kostas Daniilidis. Learning to estimate 3d human pose and shape from a single color image, 2018.

[35] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training, 2019.

[36] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[37] Alin-Ionut Popa, Mihai Zanfir, and Cristian Sminchisescu. Deep multitask architecture for integrated 2d and 3d human sensing, 2017.

[38] Rohit Prabhavalkar, Ouais Alsharif, Antoine Bruguier, and Lan McGraw. On the compression of recurrent neural networks with an application to lvcsr acoustic modeling for embedded speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, page 5970–5974. IEEE Press, 2016.

[39] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman. Universal differential equations for scientific machine learning, 2020.

[40] U. Raghavendra, Hamido Fujita, Sulatha V. Bhandary, Anjan Gudigar, Jen Hong Tan, and U. Rajendra Acharya. Deep convolution neural network for accurate diagnosis of glaucoma using digital fundus images. *Information Sciences*, 441:41–49, May 2018.

[41] Alireza Rostami, Mohammad Anbaz, Hamidreza Erfani, Milad Arabloo, and Alireza Bahadori. Accurate estimation of co 2 adsorption on activated carbon with multi-layer feed-forward neural network (mlfnn) algorithm. *Egyptian Journal of Petroleum*, 27, 02 2017.

[42] Sanjiban Sekhar Roy, Abhinav Mallik, Rishab Gulati, Mohammad S. Obaidat, and P. V. Krishna. A deep learning based artificial neural network approach for intrusion detection. In Debasis Giri, Ram N. Mohapatra, Heinrich Begehr, and Mohammad S. Obaidat, editors, *Mathematics and Computing*, pages 44–53, Singapore, 2017. Springer Singapore.

[43] Zuherman Rustam, Sri Hartini, Rivan Pratama, Reyhan Eddy Yunus, and Rahmat Hidayat. Analysis of architecture combining convolutional neural network (cnn) and kernel k-means clustering for lung cancer diagnosis. *International Journal on Advanced Science, Engineering and Information Technology*, 10:1200, 06 2020.

[44] Mathew Schwartz and Philippe Dixon. The effect of subject measurement error on joint kinematics in the conventional gait model: Insights from the open-source pycgm tool using high performance computing methods. *PLOS ONE*, 13:e0189984, 01 2018.

[45] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, 2018.

[46] Daniel Sonntag, Michael Barz, Jan Zacharias, Sven Stauden, Vahid Rahmani, Áron Fóthi, and András Lőrincz. Fine-tuning deep cnn models on specific ms coco categories, 2017.

[47] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5686–5696, 2019.

[48] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5686–5696, 2019.

[49] Boukaye Boubacar Traoré, Bernard Kamsu-Foguem, and Fana Tangara. Deep convolution neural network for image recognition. *Ecological Informatics*, 48:257–268, November 2018.

[50] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, D. Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43:3349–3364, 2021.

[51] Christoph Wehmeyer and Frank Noé. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *The Journal of Chemical Physics*, 148(24):241703, 2018.

[52] Shunjun Wei, Xiangfeng Zeng, Qizhe Qu, Mou Wang, Hao Su, and Jun Shi. Hrsid: A high-resolution sar images dataset for ship detection and instance segmentation. *IEEE Access*, 8:120234–120254, 2020.

[53] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

[54] Guangle Yao, Tao Lei, and Jiandan Zhong. A review of convolutional-neural-network-based action recognition. *Pattern Recognition Letters*, 118:14–22, 2019. Cooperative and Social Robots: Understanding Human Activities and Intentions.

[55] Chuanlong Yin, Yuefei Zhu, Jin long Fei, and Xin-Zheng He. A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access*, 5:21954–21961, 2017.

[56] Jun Yu, Jing Li, Zhou Yu, and Qingming Huang. Multimodal transformer with multi-view visual representation for image captioning. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(12):4467–4480, 2020.

[57] Andrei Zanfir, Elisabeta Marinoiu, and Cristian Sminchisescu. Monocular 3d pose and shape estimation of multiple people in natural scenes - the importance of multiple scene constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[58] Andrei Zanfir, Elisabeta Marinoiu, Mihai Zanfir, Alin-Ionut Popa, and Cristian Sminchisescu. Deep network for the integrated 3d sensing of multiple people in natural images. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[59] Dan Zecha, Moritz Einfalt, and Rainer Lienhart. Refining joint locations for human pose tracking in sports videos. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2524–2532, 2019.

[60] Xingyi Zhou, Qixing Huang, Xiao Sun, Xiangyang Xue, and Yichen Wei. Towards 3d human pose estimation in the wild: a weakly-supervised approach, 2017.

[61] M. A. Zulkifley, Nur Ayuni Mohamed, and Nuraisyah Hani Zulkifley. Squat angle assessment through tracking body movements. *IEEE Access*, 7:48635–48644, 2019.