Department of Informatics
Technical University of Munich

Master's Thesis in Informatics

Exploiting Spatial-Temporal Relationships for Occlusion-Robust 3D Human Pose Estimation

Ausnutzung räumlich-zeitlicher Beziehungen für eine verdeckungssichere 3D-Positionsschätzung von Menschen

Supervisor Prof. Dr.-Ing. habil. Alois C. Knoll

Advisor Soubarna Banik, M.Sc.
Emec Ercelik, M.Eng.

Author Patricia Gschoßmann

Date December 15, 2022 in Garching

Disclaimer

I confirm that this Master's Thesis is my own work and I have documented all sources and material used.

Garching, December 15, 2022

(Patricia Gschoßmann)

Abstract

Occlusion is one of the main sources of error in 3D human pose estimation (HPE) from monocular videos. Despite its relevance, most recent methods do not explicitly consider the effects of occlusion. This thesis addresses the problem of occlusion from two perspectives by considering both incomplete and noisy input data. To this end, three different methods for occlusion-robust 3D HPE were implemented. Central to all methods is the exploitation of spatial and temporal relationships through the combined use of Graph Convolutional Networks and Transformers. The first method serves as a baseline approach on which the other two methods are built. The second method attempts to achieve robustness to incomplete data through additional simulation of occlusion using data augmentation during training. The third method tackles the problem of noisy data by means of an auxiliary task that attempts to identify occluded joints based on the noise of the 2D input. The proposed methods are evaluated on the Human3.6M dataset, a standard benchmark for monocular 3D single-person HPE.

All methods have shown to be able to effectively capture spatial and temporal information, thereby producing similarly satisfactory results on the test dataset. In particular, both structural geometric information of the human skeleton and implicit spatial relationships beyond first- and second-order neighbors are captured, as well as long-range temporal relationships across distant frames. Furthermore, it was shown that a simple occlusion simulation strategy during training increases test-time robustness to incomplete input data through more efficient use of temporal relations, in particular by complementarily capturing long-range temporal relations at different levels. At the same time, it was also shown that estimating input noise is possible, shifting the focus from inaccurate to more accurate data in the input, but only up to a certain extent due to the constraints imposed by the dataset. While this did not improve robustness to noisy input data, it did improve generalizability to unseen data.

Zusammenfassung

Verdeckung ist eine der Hauptfehlerquellen bei der 3D-Positionsschätzung von Menschen aus monokularen Videos. Trotz ihrer Relevanz berücksichtigen die meisten neueren Methoden die Auswirkungen der Verdeckung nicht explizit. Diese Arbeit betrachtet das Problem der Verdeckung aus zwei Perspektiven, indem sie sowohl unvollständige als auch verrauschte Eingabedaten berücksichtigt. Dazu wurden drei verschiedene Methoden zur verdeckungssicheren 3D-Positionsschätzung implementiert. Im Mittelpunkt aller Methoden steht die Ausnutzung räumlicher und zeitlicher Beziehungen durch kombinierten Einsatz von Graph Convolutional Networks und Transformers. Die erste Methode dient als Basisansatz für die beiden anderen Methoden. Die zweite Methode versucht, unvollständige Daten durch zusätzliche Simulation von Verdeckungen mittels Datenerweiterung während des Trainings zu kompensieren. Die dritte Methode geht das Problem mit Hilfe einer Hilfsaufgabe an, die versucht, verdeckte Gelenke auf der Grundlage des Rauschens der 2D-Eingabe zu identifizieren. Die vorgestellten Methoden werden anhand des Human3.6M-Datensatzes evaluiert, einem Standard-Benchmark für die monokulare 3D-Positionsschätzung einzelner Personen.
Alle Methoden zeigen sich in der Lage, räumliche und zeitliche Informationen effektiv zu erfassen und zufriedenstellende Ergebnisse auf dem Testdatensatz zu erzielen. Insbesondere werden sowohl strukturelle geometrische Informationen des menschlichen Skeletts als auch implizite räumliche Beziehungen jenseits von Nachbarn erster und zweiter Ordnung erfasst, ebenso wie weitreichende zeitliche Beziehungen über entfernte Bilder hinweg. Darüber hinaus ergab sich, dass eine einfache Strategie zur Simulation von Verdeckung während des Trainings die Robustheit gegenüber unvollständigen Eingabedaten durch die effizientere Nutzung zeitlicher Beziehungen erhöht, insbesondere durch die komplementäre Erfassung weitreichender zeitlicher Beziehungen auf verschiedenen Ebenen. Gleichzeitig wurde gezeigt, dass die Schätzung von Rauschen möglich ist, wodurch sich der Fokus von ungenauen auf genauere Daten in der Eingabe verlagerte, allerdings nur bis zu einem gewissen Grad aufgrund der durch den Datensatz auferlegten Einschränkungen. Das verbessert zwar nicht die Robustheit gegenüber verrauschten Eingabedaten, wohl aber die Generalisierung auf ungesehene Daten.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Three-dimensional (3D) human pose estimation (HPE) aims to locate joints of a human body in 3D space from images or videos. It provides comprehensive geometric and motion information about the human body, enabling a wide range of potential applications [YTH20; Wan+21a]:

Human-Computer Interaction (HCI) 3D HPE can help to recognize human actions, which is crucial for computers and robots to serve and interact with people.

Augmented Reality (AR) and Virtual Reality (VR) 3D HPE can clarify the relations between human and virtual/real world to improve the interactive experience, e.g. of games controlled by poses and gestures, or of virtual try-on systems in the e-commerce fashion industry.

Sport Motion Analysis 3D HPE can assist in quantitative analysis of athlete performance (e.g. number of jumps) and provide immediate feedback for its improvement.

Video Surveillance 3D HPE can support the detection of people with abnormal behavior, which can be an important clue for video surveillance.

Autonomous Driving 3D HPE can help predict the behavior of pedestrians to ensure that self-driving cars respond appropriately.

Movies and Animation 3D HPE can serve as a cost-effective alternative to marker-based motion capture systems used for the development and animation of digital characters in the entertainment industry.

Healthcare 3D HPE can assist in physical therapy and rehabilitation training and enables remote diagnoses.

Most modern approaches adopt a two-stage pipeline, where two-dimensional (2D) keypoints are first estimated and then lifted into 3D space (i.e. 2D-to-3D lifting approach). These methods have achieved impressive results in recent years, especially due to advances in 2D HPE. However, 2D-to-3D lifting remains an inherently ill-posed problem due to (self-)occlusion and depth ambiguity in 2D representations leading to noisy 2D keypoints and multiple feasible solutions.
Studies have shown that occlusion is one of the main sources of error when estimating human poses from a single image [Sár+18; GM22]. To alleviate this problem, many researchers utilize temporal information across multiple frames. However, standard 3D HPE benchmarks do

not systematically model occlusion effects. Hence, most existing methods treat all keypoints equally, as there is no information about which ones might be unreliable. Furthermore, the ability of state-of-the-art models to handle occlusion has not been quantified and their behavior under occlusion remains largely unexplored [Sár+18; GM22].
At the same time, previous work fails to fully exploit spatial and temporal relationships in videos. Former methods [Cai+19; BGK21; XT21; Zha+19b; Doo+20] employ graph convolutional networks (GCNs) [KW17] to estimate 3D poses with a graph-based representation of human skeletons. These techniques perform well but suffer from a limited receptive field, as they restrict the graph convolution filter to operate only on neighboring nodes of first-order. Recent attempts [Zhe+21; Li+22a; Li+22b] utilize the popular Transformer [Vas+17] network, which is able to capture global correlations across long input sequences, thus mitigating the problem of receptive field. However, many of them ignore the natural distinction of spatial relations within individual frames, leaving potential improvements unused. Additionally, the self-attention mechanism cannot fully exploit the spatial relations of the human skeleton, since it "builds upon calculating the similarities of (...) joints, which (...) ignores or weakens the graph structure information" [Zha+21]. Consequently, not all sources of prior knowledge are taken into account.

1.2 Research Objectives

This thesis addresses the problem of occlusion when estimating the pose of a single person in 3D space given a sequence of images. Robust models that can cope with different degrees of occlusion while maintaining their performance are required. To this end, three different methods are designed to overcome the problem of occlusion. Central to all methods is the effective exploitation of spatial-temporal relationships through the combined use of GCNs and Transformers. GCNs are employed to leverage prior knowledge of the human skeleton through fixed adjacency matrices, while Transformers capture implicit spatial and long-range temporal relationships through the self-attention mechanism. Of the three methods presented, the first serves as a baseline approach upon which the other two build. The second method equals the first regarding the model architecture, but attempts to achieve occlusion robustness through additional simulation of occlusion using data augmentation. In contrast to this data-based approach, the third method extends the baseline method architecturally. It tackles the problem by means of an auxiliary task that attempts to identify occluded joints based on the noise of the corresponding 2D keypoints. The proposed methods will be evaluated on the Human3.6M [Ion+14] dataset, the largest publicly available and most commonly used benchmark for monocular 3D single-person HPE. Each method is analyzed individually by evaluating its overall performance on the test data and examining its effectiveness in terms of occlusion robustness through appropriate, individually designed tests. In a final analysis, all methods are compared with the current state-of-the-art.

1.3 Thesis Outline

The rest of the thesis is organized as follows:

- Chapter 2 summarizes the theoretical background of the thesis, in particular the mathematical foundations of GCNs and Transformers, as they form the basis of the proposed methods.

- Chapter 3 provides a comprehensive overview of related work on 3D human pose estimation to position the proposed methods in relation to existing approaches.
- Chapter 4 presents the three methods for occlusion-robust 3D human pose estimation in detail.
- Chapter 5 describes the experiments conducted and evaluates the results.
- Chapter 6 concludes the thesis by summarizing the findings and limitations of the proposed methods and possible directions for future work.

Chapter 2

Background

This chapter presents the theoretical background of this thesis and accompanying notations. The fundamentals of imaging geometry are presented to understand the challenge of depth ambiguity in 3D human pose estimation. The mathematical foundations of deep learning and artificial networks are explained in detail, specifically GCNs and Transformers. Also, a comprehensive overview of the 3D human pose estimation problem is provided, including its challenges and variants. Throughout the thesis, bold uppercase letters denote matrices, bold lowercase letters denote vectors, and lowercase letters denote scalars. $\boldsymbol{x}(i)$ denotes the ith entry of vector $\boldsymbol{x}$. $\boldsymbol{A}(i,j)$ denotes the entry at the intersection of the ith row and jth column of matrix $\boldsymbol{A}$. $\boldsymbol{A}^{\mathrm{T}}$ denotes the transpose of matrix $\boldsymbol{A}$.

2.1 Imaging Geometry Fundamentals

Digital images are 2D arrays (matrices), typically encoding light intensities acquired by cameras (specifically called intensity image). The following section explains the geometric principles underlying the formation of intensity images and the fundamental mathematical model of intensity cameras. This is necessary to understand the problem of 3D human pose estimation and its challenges. All definitions are taken from [TV98; HZ04; Tsa87].

2.1.1 Geometric Image Formation

Geometric image formation is the process of projecting 3D scene points into two-dimensional (2D) image plane locations. Camera models model the geometric projection performed by the sensor and determine where in the image plane the projection of a scene point will be located. The **perspective camera model** (or pinhole camera model) (Fig. 2.1) is a simple geometric model of an intensity camera. The model consists of the image plane π and the center of projection $\boldsymbol{o} \in \mathbb{R}^3$. The distance between π and $\boldsymbol{o}$ is the focal length f. The line through $\boldsymbol{o}$ and perpendicular to π is the optical axis. The intersection $\boldsymbol{c}$ between π and the optical axis is called the principal point or image center. $\boldsymbol{p}_i$ is the image, i.e. projection, of $\boldsymbol{p}$ and corresponds to the point where the straight line through $\boldsymbol{p}$ and $\boldsymbol{o}$ intersects the image plane π.

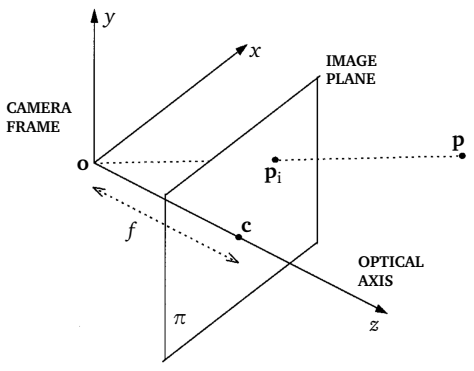


Figure 2.1: The perspective camera model (adapted from [TV98]).

Consider the 3D reference frame where $\boldsymbol{o}$ is the origin and the plane π is orthogonal to the z-axis. This reference frame is called camera coordinate system. The coordinates of point $\boldsymbol{p}_i = [x_i, y_i, z_i]^{\mathrm{T}}$, image of the 3D point $\boldsymbol{p} = [x, y, z]^{\mathrm{T}}$, are given by:

$$\begin{aligned} x_i &= f\frac{x}{z} \\ y_i &= f\frac{y}{z}. \end{aligned} \tag{2.1}$$

Note that in the camera frame, the third coordinate of an image point is always equal to the focal length, i.e. $z_i = f$. For this reason, we omit the third coordinate and write $\boldsymbol{p}_i = [x_i, y_i]^{\mathrm{T}}$ instead of $\boldsymbol{p}_i = [x_i, y_i, f]^{\mathrm{T}}$. (x_i, y_i) can also be thought of as coordinates of a new reference frame, called image coordinate system.

2.1.2 Camera Parameters

Additional knowledge about the properties of the camera is required to link the coordinates of points in 3D space with the coordinates of the corresponding points in digital images. These properties are known as the extrinsic and intrinsic camera parameters. The problem of estimating extrinsic and intrinsic parameters is called camera calibration.

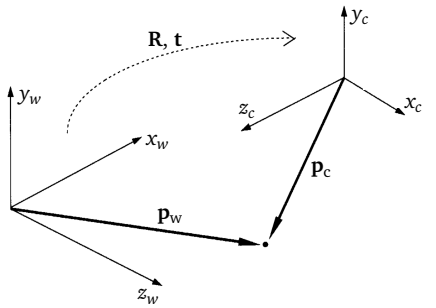


Figure 2.2: The relationship between the world and camera coordinate systems (adapted from [TV98]).

We assume that the camera coordinate system can be located with respect to some other, known reference frame (the world coordinate system). **Extrinsic parameters** define the location and orientation of the camera coordinate system with respect to the world coordinate system, i.e. they specify the transformation between the two reference frames. This transformation is captured by:

- a 3D translation vector $\boldsymbol{t}$ describing the relative positions of the origins of the two reference frames, and
- a 3×3 orthogonal[1] rotation matrix $\boldsymbol{R}$ that brings the corresponding axes of the two frames onto each other.

For a point with coordinates $\boldsymbol{p}_w$ in the world coordinates system, the corresponding coordinates $\boldsymbol{p}_c$ in the camera coordinate system are

$$\boldsymbol{p}_c = \boldsymbol{R}(\boldsymbol{p}_w + \boldsymbol{t}). \tag{2.2}$$

Fig 2.2 visualizes the relation.

We further assume that the coordinates of points in the image reference frame can be obtained from pixel coordinates available directly from the digital image (i.e. a new reference frame, called pixel coordinate system). We need to consider that:

- pixel coordinates of a digital image are discrete values, while points in the image plane are represented in continuous physical measurements (e.g. millimeters);

[1] I.e., $\boldsymbol{R}^{\mathrm{T}}\boldsymbol{R} = \boldsymbol{R}\boldsymbol{R}^{\mathrm{T}} = \boldsymbol{Id_3}$.

- digital images typically have their origin at the top-left corner, thus, image coordinates and pixel coordinates differ by a translation;
- the image coordinate system may be skewed, which means that the angle between x- and y-axis is slightly larger or smaller than 90 degrees.

Intrinsic parameters link the pixel coordinates of a point in a digital image with the corresponding coordinates in the camera reference frame. They comprise

- the focal length f specifying the perspective projection of camera coordinates to image coordinates as in Eq. 2.1[2], and
- the transformation between image coordinates and pixel coordinates.

Let (x_i, y_i) to denote the coordinates of an image point in the image reference frame and let (x_{im}, y_{im}) to denote the coordinates of the same point in the pixel coordinate system. Their relation is given by:

$$\begin{aligned} x_{im} &= \frac{1}{s_x} x_i + c_x, \\ y_{im} &= \frac{1}{s_y} y_i + c_y, \end{aligned} \tag{2.3}$$

where (c_x, c_y) are the location of the image center in pixel coordinates and (s_x, s_y) are the pixel width and height (in millimeters), respectively. If the image coordinate system is skewed, we must first transform from rectangular to skewed image plane before transforming from image to pixel coordinates, with:

$$\begin{aligned} x_i^{skew} &= x_i - y_i \cot\theta, \\ y_i^{skew} &= y_i / \sin\theta, \end{aligned} \tag{2.4}$$

where θ is the angle between x- and y-axis.

2.2 Artificial Neural Networks

Artificial neural networks (ANNs) are a class of deep learning (DL) algorithms that extract hierarchical representations (features) of the observational data [Ben09]. Lower level features (starting with the raw input) are transformed into representations at higher, more abstract levels of the hierarchy through non-linear operations [LBH15]. The key aspect is that these layers of features are not human-crafted but learned from the data using a general-purpose learning procedure [LBH15; Ben09]. This differentiates ANNs from other machine learning (ML) methods. Automatically learning features at multiple levels of abstraction allow an ANN to learn complex functions mapping the input to the output directly from data [Ben09]. This chapter first addresses the basic mathematical concepts of ANNs and then explains in detail the working principles of the two ANNs relevant to this thesis.

[2]It may be the case that physical sensors introduce non-linearity such as distortion to the projection. The intrinsic parameters do not account for lens distortion, since the perspective camera model has no lens. Radial and tangential lens distortion, if present, must be modeled separately to correctly represent a real camera. The corresponding equations can be found in the appendix.

2.2.1 Fundamentals

This section provides a brief review of the main principles of machine learning before discussing ANNs in more detail. All definitions are taken from [GBC16; PG17; RN09; Rud16; SB12; TK21; LP08; LBH15; Ben09].

Preliminaries

Machine Learning Paradigms Most ML algorithms can be broadly divided into supervised or unsupervised learning methods. This thesis only addresses the task of supervised learning, but ANNs can generally be used for both categories. However, the remainder of this chapter focuses on ANNs in the context of supervised learning.
Supervised learning algorithms use labeled datasets, i.e. each data point (example) is an input-output pair consisting of features and an associated label or target. The input is described by a vector $\boldsymbol{x} \in \mathbb{R}^n$, where each entry $\boldsymbol{x}(i)$ corresponds to a feature. The corresponding output y is generated by an unknown function $y = f^*(\boldsymbol{x})$ and dependent on the ML task. Given a *training set* of N example input-output pairs $(\boldsymbol{x}_0, y_0), \dots (\boldsymbol{x}_{N-1}, y_{N-1})$, the goal of a supervised learning algorithm is to learn a function f that approximates the true function f^*. A quantitative performance measure specific to the task evaluates the abilities of the algorithm. The performance measure is evaluated on a *test set* with examples that are distinct from the training set. f *generalizes* well from the training data if it correctly maps the new unseen examples.

Machine Learning Tasks ML is used to solve various tasks, the following of which are relevant for this thesis:

- *Regression* is the task of predicting a numerical value given some input. The ML algorithm constructs a function $f : \mathbb{R}^n \to \mathbb{R}$, $f(\boldsymbol{x}) = y$, which assigns an input $\boldsymbol{x}$ to a real-valued output y.

- *Classification* is the task of identifying which of k categories some input belongs to. The ML algorithm produces a function $f : \mathbb{R}^n \to \{1, \dots, k\}$, $f(\boldsymbol{x}) = y$, which assigns an input $\boldsymbol{x}$ to a category identified by the numeric code y. Alternatively, f outputs a probability distribution over all classes.

Components

Essentially, an ANN approximates some function $f^*(\boldsymbol{x})$ by defining a mapping $f(\boldsymbol{x}; \boldsymbol{\Theta}) \approx f^*(\boldsymbol{x})$ and learning the value of the parameters $\boldsymbol{\Theta}$ that result in the best function approximation. The behavior of neural networks is shaped by its architecture (i.e. the overall structure of the network). The exact architecture of a neural network depends on the problem specification and is configured according to the application. Several types of ANNs exist characterized by different architectures, but the basic components are the same.
Generally, ANNs are composed of multiple non-linear data-processing units (i.e. *neurons*) assembled in successive *layers*. The total number of layers L determines the *depth* of the network. Each layer can have a different number of neurons. Neurons from one layer are connected to neurons from other layers by direct links. The connection between two neurons i and j is associated with a quantity (i.e. *weight*) w_{ji} indicating the connection strength. Each neuron receives information (features) from neurons of the preceding layer, performs simple calculations, and then passes its results to neurons of the subsequent layer (see Fig. 2.3).

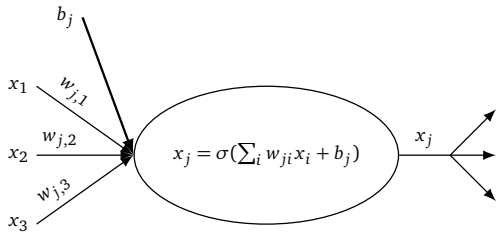


Figure 2.3: The mathematical model of a neuron.

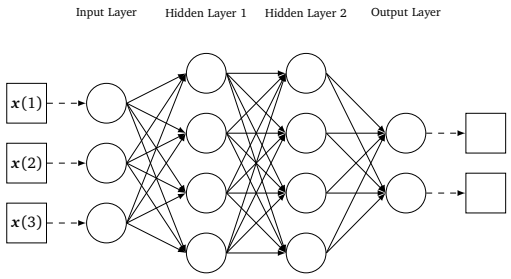


Figure 2.4: A multi-layer perceptron (MLP).

The *net input* (activation) of a neuron j is the sum of all its input values multiplied by their corresponding connection weights:

$$a_j = \sum_{i=0}^{m-1}(w_{ji}x_i) + b_j, \tag{2.5}$$

where x_i is the output value of neuron i in the previous layer and b_j is a *bias* term. The weights specify how each feature affects the output of the neuron. The bias acts on a neuron like a horizontal offset. The weights and biases are the trainable parameters of a neural network and control its behavior. The *activation function* is a non-linear function that determines the output value (response) of a neuron based on its net input / activation. It is typically applied element-wise with:

$$x_j = \sigma(a_j) \tag{2.6}$$

In modern DL, the rectified linear activation function (ReLU) [Fuk75], defined by $\sigma(z) = \max(0, z)$, is recommended by default for most ANNs.
The quintessential example of an ANN is the multi-layer perceptron (MLP) consisting of an input layer, one or more hidden layers and an output layer. In MLPs, each neuron of a layer is connected to each neuron of the following layer, i.e. the network is "fully connected". Fig. 2.4 shows a typical multi-layer perceptron neural network structure. MLPs are classic feed-forward networks (FFNs), as information flows in only one direction (forward). There are neither lateral connections between neurons within a layer nor feedback/cyclic connections where the output of a layer is fed back to previous layers.[3] An MLP as shown in Fig. 2.4 with depth L is a function f^L with:

$$f^l(\boldsymbol{x}) = \sigma^l(\boldsymbol{W}^l f^{l-1}(\boldsymbol{x}) + \boldsymbol{b}^l), \quad f^0(\boldsymbol{x}) = \boldsymbol{x}, \tag{2.7}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the input vector and σ^l is the activation function used in layer l. $\boldsymbol{W}^l \in \mathbb{R}^{n_l \times n_{l-1}}$ is the weight matrix of layer l, where n_l is the number of neurons in layer l and n_{l-1} is the number of neurons in the previous layer. $\boldsymbol{b}^l \in \mathbb{R}^{n_l}$ are the biases of layer l. $\boldsymbol{W}^l$ and $\boldsymbol{b}^l$ constitute the layer parameters. $\boldsymbol{\Theta}$ is the set of parameters of all layers. Initially, all parameters of a network are initialized randomly.

Optimization

Optimization refers to the task of finding the best function approximation f with regard to some criterion. To this end, a *loss function* $\mathcal{L}$ is defined that measures the difference (loss / error) between the true output y according to the data and the predicted (approximated) output $\hat{y}$ of the network. The choice of loss function depends on the ML task. Essentially, the

[3] Recurrent neural networks (RNNs) are ANNs in which neurons from one layer can also be connected to neurons from the same or previous layers.

goal is to find the parameters $\boldsymbol{\Theta}$ that minimize the loss of the prediction and the ground truth. The process of successively adjusting the parameters to produce more and more accurate approximations is called *learning*. *Training* a neural network means learning the parameters based on the training set.

The optimization algorithm specifies how the parameters are updated during training. ANNs are usually trained using iterative, gradient-based optimization algorithms based on *gradient decent*. The gradient of $\mathcal{L}$ with respect to $\boldsymbol{\Theta}$ measures the change in error caused by a change in the parameters. The negative gradient indicates the direction of steepest descent that takes the loss function closer to a minimum with low error.

Gradient descent updates the parameters by making repeated steps in the direction of the negative gradient until convergence:

$$\boldsymbol{\Theta}_{k+1} = \boldsymbol{\Theta}_k - \alpha \nabla_{\boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\Theta}_k), \tag{2.8}$$

where $\nabla_{\boldsymbol{\Theta}} \mathcal{L}$ denotes the average gradient over all examples in the training set and $\alpha \in (0, 1]$ is the *learning rate*. Fig. 2.5 illustrates the procedure. α is a *hyperparameter*, i.e. its value is not derived via training but set before the learning process. However, it is common to gradually decay the learning rate over time. The *backpropagation* (BP) algorithm is used to calculate the gradients. For more details refer to [GBC16; PG17].

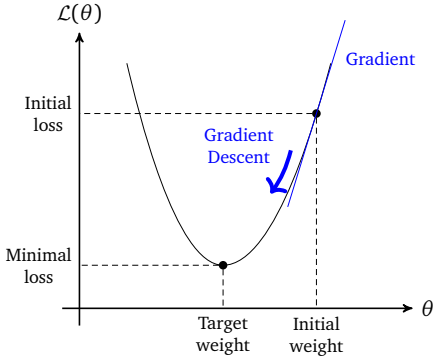


Figure 2.5: Gradient descent for a loss function $\mathcal{L}$ with a single weight coefficient θ.

In practice, an extension of the gradient descent algorithm is used, called *stochastic gradient descent* (SGD). SGD reduces the costs of computing the gradient by estimating it at each step of the algorithm using a small subset (i.e. *minibatch*) of examples $\mathbb{B} = \{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{m-1}\}$ drawn uniformly from the training set:

$$\nabla_{\boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\Theta}_k) \approx \frac{1}{m} \sum_{i=0}^{m-1} \nabla_{\boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{x}_i, y_i; \boldsymbol{\Theta}_k) = \boldsymbol{g}_k. \tag{2.9}$$

Learning with SGD can sometimes be slow as the gradient is scaled equally across all dimensions. Many different variants of the SGD-algorithm have been designed to accelerate learning. Among the most frequently used are:

Momentum Momentum [Qia99] accumulates an exponentially-decaying average of past gradients denoted as velocity $\boldsymbol{v}$ and moves in the direction of $\boldsymbol{v}$. An update step is the largest when the subsequent gradients all point to the same direction. The update rule is given by:

$$\begin{aligned} \boldsymbol{v}_k &= \beta \boldsymbol{v}_{k-1} + \alpha \boldsymbol{g}_k, \\ \boldsymbol{\Theta}_{k+1} &= \boldsymbol{\Theta}_k - \boldsymbol{v}_k. \end{aligned} \tag{2.10}$$

β is a hyperparameter (i.e. the accumulation rate) determining how much previous gradients affect the current direction (usually $\beta = 0.9$).

RMSProp RMSProp [TH12] scales the learning rate α by an exponentially-decaying average of squared gradients $\boldsymbol{s}$ to dampen the oscillation for high-variance directions. The update rule is given by:

$$\begin{aligned} \boldsymbol{s}_k &= \beta \boldsymbol{s}_{k-1} + (1-\beta)[\boldsymbol{g}_k \odot \boldsymbol{g}_k], \\ \boldsymbol{\Theta}_{k+1} &= \boldsymbol{\Theta}_k - \frac{\alpha}{\sqrt{\boldsymbol{s}_k} + \in} \boldsymbol{g}_k, \end{aligned} \tag{2.11}$$

where $\odot$ denotes the element-wise multiplication and $\in$ is a hyperparameter improving numerical stability (typically $\in = 10^{-8}$). $\boldsymbol{s_0}$ is initialized with 0.

Adam Adam [KB15] combines Momentum with RMSProp. The update rule is given by:

$$\begin{gathered} \boldsymbol{v}_k = \beta_1 \boldsymbol{v}_{k-1} + (1-\beta_1)\boldsymbol{g}_k, \quad \boldsymbol{s}_k = \beta_2 \boldsymbol{s}_{k-1} + (1-\beta_2)[\boldsymbol{g}_k \odot \boldsymbol{g}_k], \\ \hat{\boldsymbol{v}}_k = \frac{\boldsymbol{v}_k}{1-\beta_1^k}, \quad \hat{\boldsymbol{s}}_k = \frac{\boldsymbol{s}_k}{1-\beta_2^k}, \\ \boldsymbol{\Theta}_{k+1} = \boldsymbol{\Theta}_k - \frac{\alpha}{\sqrt{\hat{\boldsymbol{s}}_k} + \in} \hat{\boldsymbol{v}}_k. \end{gathered} \tag{2.12}$$

$\boldsymbol{v}_0$ and $\boldsymbol{s}_0$ are initialized with 0 and usually $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\in = 10^{-8}$.

AMSGrad Adam may fail to converge to an optimal solution for some applications. AMSGrad [JKK18] is a variant of Adam that keeps a running maximum of the squared gradients $\hat{\boldsymbol{s}}^{\max}$ instead of an exponentially-decaying average. $\hat{\boldsymbol{s}}^{\max}$ serves as a long-term memory of past, large and informative gradients. The update rule in Eq. 2.12 is extended by:

$$\begin{gathered} \hat{\boldsymbol{s}}_k^{\max} = \max(\hat{\boldsymbol{s}}_{k-1}^{\max}, \hat{\boldsymbol{s}}_k), \\ \boldsymbol{\Theta}_{k+1} = \boldsymbol{\Theta}_k - \frac{\alpha}{\sqrt{\hat{\boldsymbol{s}}_k^{\max}} + \in} \hat{\boldsymbol{v}}_k. \end{gathered} \tag{2.13}$$

$\hat{\boldsymbol{s}}_0^{\max}$ is initialized with 0.

2.2.2 Graph Convolutional Networks

Graph neural networks (GNNs) are a family of ANNs used for processing data represented as graphs. Their key element is the encoding of structural information through message passing between the nodes of a graph [Zho+20]. Several types of GNN architectures exist that differ in the implementation of message passing. Encouraged by the success of convolutional neural networks (CNNs) [LB98] in computer vision (CV), a large number of methods have been developed that generalize the notion of convolution from grid to graph data. Wu et al. [Wu+21] group these methods under the term convolutional graph neural networks (ConvGNNs).
ConvGNNs are further divided into two categories depending on the type of convolution: Spectral-based methods and spatial-based methods. Spectral-based CNNs define the convolution operator in the spectral domain based on graph Fourier transform and assume the graphs to be undirected. Spatial-based methods define convolutions in the spatial domain (i.e. vertex domain) based on the graph topology [Zha+19a; Zho+20]. Graph convolutional networks (GCNs) [KW17] bridge the gap between spectral-based and spatial-based methods [Zha+19a; Wu+21]. In the following sections, the underlying operation is explained in more detail from both perspectives.

Fundamentals

In the beginning it is necessary to define some mathematical concepts and to recap the theory of spectral-based and spatial-based ConvGNNs to understand the GCN-operation. All definitions are taken from [Wu+21; Zha+19a; Zho+20; Cao+22].

Graphs A graph is represented as $G = (V, E)$, where V is the set of nodes and E is the set of edges. $|V| = n$ is the number of nodes in the graph and $|E| = m$ is the number of edges. Let $i \in V$ to denote a node and $e_{ij} = (i, j) \in E$ to denote an edge pointing from node i to node j. The neighborhood of radius K (or K-hop neighborhood) of a node i, denoted as $N(i, K)$, is the

set of nodes with distance less than or equal to K from i. It applies: $N(i,1) = \{j \in V | (i,j) \in E\}$. The adjacency matrix $\boldsymbol{A}$ is a $n \times n$ matrix that indicates the weight of an edge (i.e. the strength of a connection). If $e_{ij} \notin E$ then $\boldsymbol{A}(i,j) = 0$. For unweighted graphs holds: If $e_{ij} \in E$ then $\boldsymbol{A}(i,j) = 1$.
A directed graph is a graph where all edges represent a one-way relationship between two nodes, directed from one node to another. The edges of an undirected graph indicate a two-way relationship between connected nodes and have no direction. Undirected graphs can be considered as a special case of directed graphs, where there is a pair of edges with opposite directions whenever two nodes are connected. A graph is undirected if and only if its adjacency matrix is symmetric.
The Laplacian matrix $\boldsymbol{L}$ is an alternative way to mathematically represent an undirected graph. It is defined as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$, where $\boldsymbol{D}$ is a diagonal matrix of node degrees (the number of edges attached to each node), i.e. $\boldsymbol{D}(i,i) = \sum_j \boldsymbol{A}(i,j)$. The corresponding symmetrically normalized Laplacian matrix is defined as $\boldsymbol{L}_{norm} = \boldsymbol{Id}_n - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$, where $\boldsymbol{Id}_n$ is an identity matrix.

Graph Signals A graph may have node attributes $\boldsymbol{X}$, where $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ is a node feature matrix whose rows represent the respective d-dimensional feature vectors of each node. In graph signal processing, a graph signal defined on the nodes is a vector $\boldsymbol{x} \in \mathbb{R}^n$ with $\boldsymbol{x}(i)$ representing the signal value on the node i. A node attribute matrix $\boldsymbol{X}$ can be considered as multiple graph signals, where the columns represent the d signals of the graph, or in other words, as a multi-channel graph signal with d channels.

Graph Fourier Transform Conventionally, signals are portrayed as a function of time []. The classical Fourier transform is a mathematical tool to depict a signal as a function of frequency. The classical Fourier transform of a 1D signal f is computed by $\hat{f}(\varepsilon) = \langle f, e^{2\pi i \varepsilon t} \rangle$, where ε is the frequency of $\hat{f}$ in the frequency domain and the complex exponential is the eigenfunction of the Laplace operator. The Laplacian matrix $\boldsymbol{L}$ is the Laplace operator defined on a graph. Hence, an eigenvector of $\boldsymbol{L}$ associated with its corresponding eigenvalue is an analog to the complex exponential at a certain frequency. Note, that the normalized Laplacian matrix $\tilde{\boldsymbol{L}}$ can also be used as the graph Laplace operator. In particular, $\boldsymbol{L}_{norm}$ can be decomposed into $\boldsymbol{L}_{norm} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^{\mathrm{T}}$, where $\boldsymbol{U} \in \mathbb{R}^{n \times n}$ is the matrix of eigenvectors ordered by eigenvalues and $\boldsymbol{\Lambda}$ is the diagonal matrix of eigenvalues. The lth column of $\boldsymbol{U}$ is the eigenvector $\boldsymbol{u}_l$ and $\boldsymbol{\Lambda}(l,l)$ is the corresponding eigenvalue λ_l. Then, the graph Fourier transform $\mathcal{F}$ of a graph signal $\boldsymbol{x}$ can be computed as:

$$\begin{aligned} \mathcal{F}(\boldsymbol{x}) &= \boldsymbol{U}^{\mathrm{T}}\boldsymbol{x} = \hat{\boldsymbol{x}}, \\ \hat{\boldsymbol{x}}(\lambda_l) &= \langle \boldsymbol{x}, \boldsymbol{u}_l \rangle = \sum_{i=0}^{n-1} \boldsymbol{x}(i)\boldsymbol{u}_l^{\mathrm{T}}(i) \end{aligned} \tag{2.14}$$

Analogously to the classical Fourier transform, the graph Fourier transform projects a graph signal from the vertex domain into the spectral domain - the orthonormal space, where the basis is formed by eigenvectors of the normalized graph Laplacian. The inverse graph Fourier transform $\mathcal{F}^{-1}$ of the transformed graph signal $\hat{\boldsymbol{x}}$ in the spectral domain can be written as:

$$\begin{aligned} \mathcal{F}^{-1}(\hat{\boldsymbol{x}}) &= \boldsymbol{U}\hat{\boldsymbol{x}} = \boldsymbol{x}, \\ \boldsymbol{x}(i) &= \sum_{l=0}^{n-1} \hat{\boldsymbol{x}}(\lambda_l)\boldsymbol{u}_l(i) \end{aligned} \tag{2.15}$$

Graph Filtering In signal processing, filtering is a localized operation on signals used to remove unwanted features from the signal by amplifying or attenuating the strength of frequency components. Filtering can be performed in both the time and frequency domain. Analogously, one can localize a graph signal in its vertex or spectral domain.
Filtering in the time domain is expressed as the convolution of the input signal with a filter signal. However, graph convolution in the vertex domain is not as straightforward as classical signal convolution in the time domain, due to the irregular structure of graphs.

Spectral graph filtering therefore makes use of the convolution theorem [Mal09] to compute the graph convolution in the spectral domain. The theorem states that the Fourier transform of a convolution of two signals in the time domain is equivalent to the element-wise multiplication of the spectral representations of the signals. Applied to graphs, this means for an input graph signal $\boldsymbol{x}$ and a filter graph signal $\boldsymbol{g} \in \mathbb{R}^n$ defined in the vertex domain:

$$\boldsymbol{x}_{out} = \boldsymbol{x} *_G \boldsymbol{g} = \mathcal{F}^{-1}(\mathcal{F}(\boldsymbol{x}) \odot \mathcal{F}(\boldsymbol{g})) = \boldsymbol{U}(\boldsymbol{U}^{\mathrm{T}}\boldsymbol{x} \odot \boldsymbol{U}^{\mathrm{T}}\boldsymbol{g}), \tag{2.16}$$

where $*_G$ denotes the convolution on graphs. If we denote $\boldsymbol{g}_\theta = diag(\hat{\boldsymbol{g}})$, Eq. 2.16 can be rewritten as:

$$\boldsymbol{x}_{out} = \boldsymbol{x} *_G \boldsymbol{g} = \boldsymbol{U}(\boldsymbol{U}^{\mathrm{T}}\boldsymbol{x} \odot \hat{\boldsymbol{g}}) = \boldsymbol{U}\begin{bmatrix} \hat{\boldsymbol{g}}(\lambda_1) & & 0 \\ & \ddots & \\ 0 & & \hat{\boldsymbol{g}}(\lambda_n) \end{bmatrix}\boldsymbol{U}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{U}\boldsymbol{g}_\theta\boldsymbol{U}^{\mathrm{T}}\boldsymbol{x}. \tag{2.17}$$

All spectral-based ConvGNNs follow this definition of graph filtering, but differ in the choice of filter $\hat{\boldsymbol{g}}$ defined in the spectral domain.

Spatial graph filtering performs graph convolution in the vertex domain as the aggregation of information from neighboring nodes. Essentially, the convolved output signal of the graph at node v is the weighted average of the signal values of the node itself and its neighbors. In mathematical terms, this means for an input graph signal $\boldsymbol{x}$:

$$\boldsymbol{x}_{out}(i) = w_{i,i}\boldsymbol{x}(i) + \sum_{j \in N(i,K)} w_{i,j}\boldsymbol{x}(j), \quad \forall i \in V, \tag{2.18}$$

where the parameters $\{w_{i,j}\}$ are the weights used to combine the signal values of node i and node j. Spatial graph filtering is localized in the vertex domain, i.e. local features are extracted regardless of graph size.[4] Naturally, this does not apply to spectral graph filtering, as the filter $\hat{\boldsymbol{g}}$ is domain-dependent. However, Shuman et al. [**Shuman2013**] show that when using Kth-order polynomials as filters one can interpret Eq. 2.17 in the vertex domain according to Eq. 2.18.

Spectral-based Perspective of Graph Convolutional Networks

Now that all the necessary terms have been defined, the GCN-operation can be derived from a spectral-based perspective.

The first spectral-based ConvGNN designed by Bruna et al. [Bru+13] processes data according to Eq. 2.17 and considers $\hat{\boldsymbol{g}}$ as a vector of learnable parameters $\boldsymbol{\theta} \in \mathbb{R}^n$ in the spectral domain, i.e. $\boldsymbol{g}_\theta = diag(\boldsymbol{\theta})$. This causes several problems: First, evaluating Eq. 2.17 is computationally expensive as it requires to compute the eigenvalue decomposition of $\boldsymbol{L}_{norm}$, which

[4] Analogous: 2D convolution on images is translation-invariant, meaning that identical features are recognized independently of their spatial locations.

has a time complexity of $\mathcal{O}(n^3)$. Furthermore, the multiplication with the eigenvector matrix $\boldsymbol{U}$ has a time complexity of $\mathcal{O}(n^2)$. Second, there are n parameters to be learned for each convolution operation. Third, the non-parametric filter[5] $\hat{\boldsymbol{g}}$ is not localized in the vertex domain. Defferrard et al. [DBV16] propose Chebyshev Spectral CNN (ChebNet) that defines $\hat{\boldsymbol{g}}$ as a K-polynomial filter, i.e. $\hat{\boldsymbol{g}}(\lambda_l) = \sum_{k=0}^{K} \theta_k \lambda_l^k$, to localize the graph convolution in the vertex domain and to reduce the number of learnable parameters to $\mathcal{O}(K) = \mathcal{O}(1)$. ChebNet further reduces the complexity by approximating $\boldsymbol{g}_\theta$ through Chebyshev polynomials $T_K(x)$ of order K, i.e. $\boldsymbol{g}_\theta \approx \sum_{k=0}^{K} \theta_k T_k(\tilde{\boldsymbol{\Lambda}})$, where $\tilde{\boldsymbol{\Lambda}} = 2\boldsymbol{\Lambda}/\lambda_{max} - \boldsymbol{Id}_n$ are the scaled eigenvalues within $[-1, 1]$. The Chebyshev polynomials are defined recursively as $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$ with $T_0(x) = 1$ and $T_1(x) = x$. As a result, the graph convolution of a graph signal $\boldsymbol{x}$ with a filter $\boldsymbol{g}$ is approximated as:

$$\boldsymbol{x} *_G \boldsymbol{g} \approx \boldsymbol{U}\left(\sum_{k=0}^{K} \theta_k T_k(\tilde{\boldsymbol{\Lambda}})\right)\boldsymbol{U}^{\mathrm{T}}\boldsymbol{x}. \tag{2.19}$$

One can prove that $T_i(\tilde{\boldsymbol{L}}) = \boldsymbol{U}T_i(\tilde{\boldsymbol{\Lambda}})\boldsymbol{U}^{\mathrm{T}}$, where $\tilde{\boldsymbol{L}} = 2\boldsymbol{L}_{norm}/\lambda_{max} - \boldsymbol{Id}_n$. Eq. 2.19 can be rewritten as:

$$\boldsymbol{x} *_G \boldsymbol{g} \approx \sum_{k=0}^{K} \theta_k T_k(\tilde{\boldsymbol{L}})\boldsymbol{x}. \tag{2.20}$$

Graph convolutional network (GCN) proposed by Kipf et al. [KW17] simplifies Eq. 2.20 by truncating the Chebyshev polynomial to first-order (i.e. $K = 1$) and assuming $\lambda_{max} = 2$:

$$\boldsymbol{x} *_G \boldsymbol{g} \approx \theta_0\boldsymbol{x} + \theta_1(\boldsymbol{L}_{norm} - \boldsymbol{Id}_n)\boldsymbol{x} = \theta_0\boldsymbol{x} - \theta_1\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{x}. \tag{2.21}$$

GCN further assumes $\theta = \theta_0 = -\theta_1$ to restrain the number of learnable parameters per convolution to one, leading to the following approximation:

$$\boldsymbol{x} *_G \boldsymbol{g} \approx \theta(\boldsymbol{Id}_n + \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}})\boldsymbol{x}. \tag{2.22}$$

Using $\boldsymbol{Id}_n + \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$ empirically causes numerical instabilities and exploding/vanishing gradients. GCN applies a renormalization trick to circumvent this problem, which replaces $\boldsymbol{Id}_n + \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$ by $\hat{\boldsymbol{A}} = \tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}$ with $\tilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{Id}_n$ and $\tilde{\boldsymbol{D}}(i,i) = \sum_j \tilde{\boldsymbol{A}}(i,j)$.
Eq. 2.22 can be generalized to a multi-channel input graph signal $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ with d channels:

$$\boldsymbol{h} = \tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{X}\boldsymbol{\theta} = \hat{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{\theta}, \tag{2.23}$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ is a learnable filter vector and $\boldsymbol{h} \in \mathbb{R}^n$ is the single-channel output graph signal. To obtain multi-channel outputs, GCN applies multiple filters and Eq. 2.23 becomes:

$$\boldsymbol{H} = \tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{X}\boldsymbol{\Theta} = \hat{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{\Theta}, \tag{2.24}$$

where $\boldsymbol{\Theta} \in \mathbb{R}^{d \times c}$ contains c filters and $\boldsymbol{H} \in \mathbb{R}^{n \times c}$ is the convolved multi-channel output graph signal with c channels.

Spatial-based Perspective of Graph Convolutional Networks

The GCN-operation in Eq. 2.23 can be expressed as follows:

$$\boldsymbol{H}(i) = \boldsymbol{\Theta}^{\mathrm{T}}\Big(\sum_{j \in N(i,1) \cup \{i\}} \hat{\boldsymbol{A}}(i,j)\boldsymbol{X}(j)\Big), \quad \forall i \in V, \tag{2.25}$$

which is essentially equivalent to aggregating node feature vectors from direct neighborhoods according to spatial graph filtering. Therefore, GCNs can also be considered as a spatial-based method.

[5] Non-parametric filters are filters whose parameters are all free [DBV16].

2.2.3 Transformer

The Transformer [Vas+17] is a type of ANN for processing data represented as a sequence. Its key concept is the self-attention mechanism, which allows the entries of the sequence to be processed all at once rather than sequentially. In this way, the Transformer is able to capture global information of the entire sequence and find long-range dependencies between the entries regardless of their distance. The self-attention mechanism is explained in more detail below, followed by an architectural overview of the Transformer. All definitions are taken from [Han+23; Liu+21; Kha+22; Vas+17].

Self-Attention

Self-attention is a technique for highlighting important aspects of the input data and suppressing unimportant parts. It is a sequence-to-sequence operation, meaning it takes a sequence of entries as input and produces a sequence of entries as output. The goal is to capture the interaction among all entries by encoding each entry in terms of the global contextual information.

The general self-attention function on a sequence can be described as mapping a query and a set of key-value pairs to an output. The output is computed as a weighted sum of the values. The weight (i.e. score) assigned to each value is computed by a compatibility function (usually the dot-product) of the query with the corresponding key.

Before self-attention is applied, each entry (also referred to as a token) in the input sequence is embedded into a vector $\boldsymbol{x}_i$ with dimension d_m. Let $\boldsymbol{X} = [\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n-1}]^{\mathrm{T}} \in \mathbb{R}^{n \times d_m}$ to denote an embedded input sequence with n entries. First, the input sequence is projected to three different matrices, namely, the queries $\boldsymbol{Q} \in \mathbb{R}^{n \times d_k}$, the keys $\boldsymbol{K} \in \mathbb{R}^{n \times d_k}$ and the values $\boldsymbol{V} \in \mathbb{R}^{n \times d_v}$:

$$\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}^Q, \qquad \boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}^K, \qquad \boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}^V, \tag{2.26}$$

where $\boldsymbol{W}^Q \in \mathbb{R}^{d_m \times d_k}$, $\boldsymbol{W}^K \in \mathbb{R}^{d_m \times d_k}$ and $\boldsymbol{W}^V \in \mathbb{R}^{d_m \times d_v}$ are learnable weight matrices. Subsequently, the output $\boldsymbol{Z} \in \mathbb{R}^{n \times d_v}$ of the attention function is calculated as follows:

1. Compute the scores by matching the queries against the database of keys: $\boldsymbol{S} = \boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}$, where $\boldsymbol{S} \in \mathbb{R}^{n \times n}$. These scores determine the degree of attention that should be given to other entries when encoding the entry at a certain position.

2. Normalize the scores for stable gradients during training: $\boldsymbol{S}_n = \boldsymbol{S}/\sqrt{d_k}$.

3. Translate the scores into a probability distribution: $\boldsymbol{P} = \mathrm{softmax}(\boldsymbol{S}_n)$.

4. Calculate the output by assigning the probabilities to the corresponding value elements: $\boldsymbol{Z} = \boldsymbol{P}\boldsymbol{V}$.

The above steps can be summarized as:

$$\mathrm{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{V} = \boldsymbol{Z}. \tag{2.27}$$

Eq. 2.27 calculates what is referred to as *scaled dot-product self-attention*. The mechanism is illustrated step by step in Fig. 2.6 on the left.

Self-attention is limited in that it is unable to focus on one or more specific entries without simultaneously influencing the attention on other equally important entries. This is due to the restricted representation subspace of the input. *Multi-head self-attention (MHSA)* is able to jointly attend to information from different representation subspaces at different positions by using several (i.e. h) independent self-attention blocks, called heads. Specifically, each head has its own set of learnable weight matrices $\{\boldsymbol{W}^{Q_i}, \boldsymbol{W}^{K_i}, \boldsymbol{W}^{V_i}\}$, where $i = 0, \ldots, (h-1)$, to project the input into different representation subspaces due to random initialization. The outputs of the h attention heads are concatenated into a single matrix $\boldsymbol{Z}' = [\boldsymbol{Z}_0, \boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_{h-1}] \in \mathbb{R}^{n \times (h \cdot d_v)}$ and eventually $\boldsymbol{Z}'$ is projected to the final output $\boldsymbol{Z}_{out} \in \mathbb{R}^{n \times d_m}$. These steps can be formulated as:

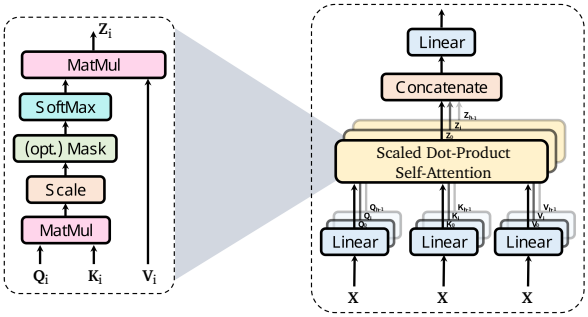


Figure 2.6: Left: Scaled dot-product self-attention. Right: Multi-head self-attention. The figure is adapted from [Liu+21]).

$$
\begin{aligned}
&\boldsymbol{Q}_i = \boldsymbol{X}\boldsymbol{W}^{Q_i}, \qquad \boldsymbol{K}_i = \boldsymbol{X}\boldsymbol{W}^{K_i}, \qquad \boldsymbol{V}_i = \boldsymbol{X}\boldsymbol{W}^{V_i}, \\
&\boldsymbol{Z}_i = \text{Attention}(\boldsymbol{Q}_i, \boldsymbol{K}_i, \boldsymbol{V}_i), \\
&\text{MultiHead}(\boldsymbol{Q}', \boldsymbol{K}', \boldsymbol{V}') = \text{Concat}(\boldsymbol{Z}_0, \ldots, \boldsymbol{Z}_{h-1})\boldsymbol{W}^O = \boldsymbol{Z}'\boldsymbol{W}^O = \boldsymbol{Z}_{out},
\end{aligned}
\tag{2.28}
$$

where $\boldsymbol{Q}'$ (and similarly $\boldsymbol{K}'$ and $\boldsymbol{V}'$) is the concatenation of $\{\boldsymbol{Q}_i\}_{i=0}^{h-1} \in \mathbb{R}^{n \times (h \cdot d_k)}$ and $\boldsymbol{W}^O \in \mathbb{R}^{(h \cdot d_v) \times d_m}$ denotes the learnable output projection matrix. Usually, $d_m/h = d_k = d_v$. The entire process of MHSA is visualized in Fig. 2.6 on the right.

Self-attention by itself ignores the sequential order of the input, since it acts on the input embeddings simultaneously and identically. This means, if the input sequence is permuted, the output sequence will be exactly the same except that it is also permuted (i.e. self-attention is permutation invariant) []. To make use of the order of the sequence, *positional encodings* with dimension d_m are added to the input embedding to capture the relative position of each token in the sequence. There are many different options for positional encodings, learned or predefined. A typical choice are sine and cosine functions with different frequencies:

$$
\begin{aligned}
\text{PE}(pos, i) &= \begin{cases} \sin(pos \cdot \omega_k), & \text{if } i = 2k \\ \cos(pos \cdot \omega_k), & \text{if } i = 2k+1, \end{cases} \\
\omega_k &= \frac{1}{10000^{2k/d_m}}, \quad k = 1, \ldots, d_m/2,
\end{aligned}
\tag{2.29}
$$

where pos denotes the position of the token in the sequence and i represents the current dimension of the positional encoding.[6]

Original Transformer

The Transformer was originally proposed by Vasmani et al. [Vas+17] for the field of natural language processing (NLP). The original architecture (see Fig. 2.7) follows an encoder-decoder structure with two components. The first component is a stack of $N = 6$ identical

[6] Note that k starts at 1, such that $pos = 1$ corresponds to the token at index 0 in the sequence and $i = 1$ corresponds to the dimension at index 0 of the positional encoding.

blocks (encoders). Each block contains two sublayers, the first is an MHSA mechanism and the second is a simple fully-connected FFN. The self-attention layer is responsible for feature aggregation while the feed-forward layer performs feature transformation/extraction. The second component also consists of $N = 6$ identical blocks (decoders). In addition to the two sublayers in each encoder, each decoder has a third sublayer that performs MHSA over the output of the stack of encoders. All sublayers in both encoder and decoder apply residual connections [He+16] alongside layer normalization [BKH16] to enhance the scalability and achieve higher performance. The output of each sublayer is thus LayerNorm(x+Sublayer(x)), where Sublayer(x) is the function implemented by the sublayer itself. To facilitate residual connections, all sublayers, as well as the embedding layer, produce outputs of dimension d_m. Finally, the embedded sequences mapped to the output sequence with a linear layer followed by a softmax layer.

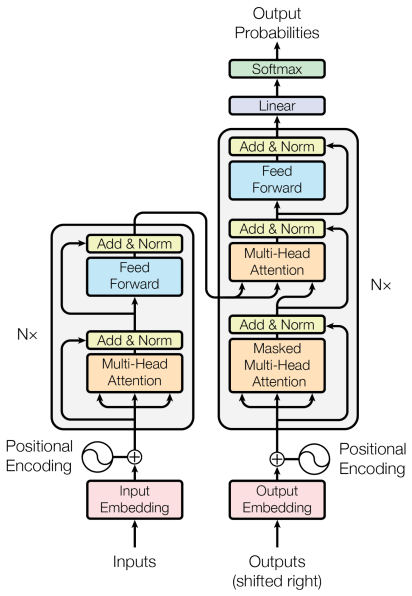


Figure 2.7: The Transformer architecture (taken from [Vas+17]).

Transformer in Vision

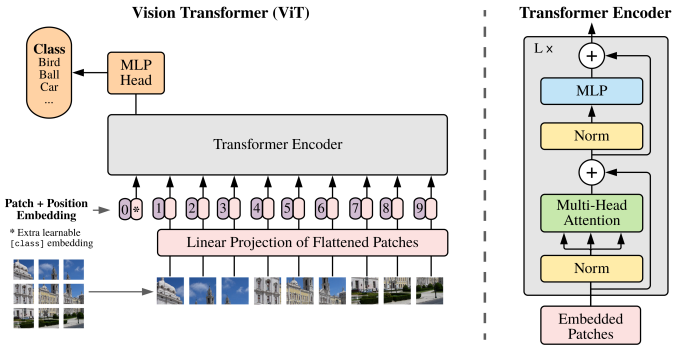


Figure 2.8: The Vision Transformer (ViT) architecture (taken from [Dos+21]).

The Vision Transformer (ViT) proposed by Dosovitskiy et al. [Dos+21] is the first pure Transformer architecture that has been used in CV, specifically for the task of image classification. ViT operates on a sequence of non-overlapping image patches (see Fig. 2.8) and consists of a stack of Transformer encoders (with minimal changes) whose output is followed by a multi-layer perceptron (MLP). Note that most Transformer-based methods in CV tasks utilize only Transformer encoders and omit the decoders. In the remainder of this thesis, the term Transformer encoder refers to the encoder in ViT, unless otherwise specified.

2.3 3D Human Pose Estimation

Human pose estimation (HPE) is the process of estimating the configuration (pose) of human body parts from a single image or video. The goal is to determine the location of keypoints, which usually correspond to the joints of the human body. We distinguish between 2D and

3D pose estimation according to the spatial dimension of the output results (see Fig. 2.9). The following sections introduce the fundamentals of 3D HPE as well as its variants and challenges. All definitions are taken from [Wan+21a; Zhe+20; YTH20; Ji+20; Liu+22].

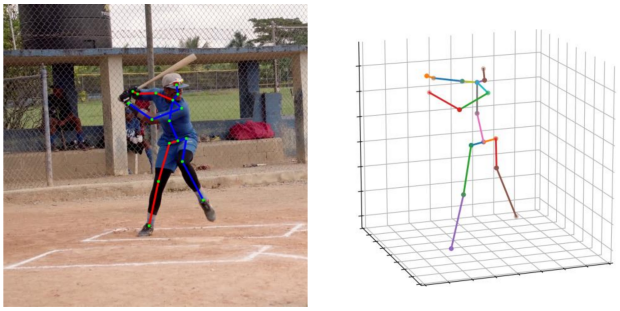


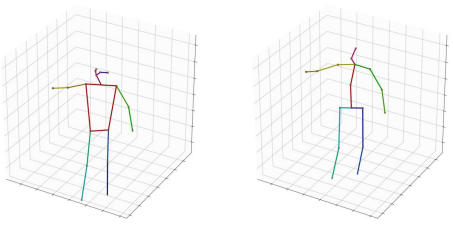
Figure 2.9: The human pose estimation problem (adapted from [Cai+19]).

2.3.1 Human Body Modeling

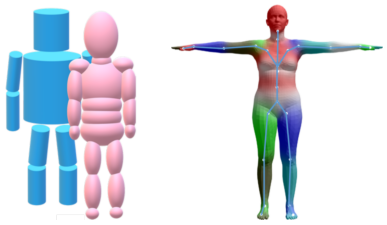
Modeling the human body is a key component of HPE. Different methods adopt different models based on the requirements for their specific application to describe the pose of a human body. In 3D HPE usually one of the following two types is used:

Kinematic Model Kinematic models, also known as stick-figures or skeleton-based models, follow the skeletal structure of the human body. They represent a human body as a set of joint positions (typically between 10 and 30) and, optionally, the corresponding limb orientations (see Fig. 2.10a). The relations between different body parts are intuitively captured in a simple and flexible way.

Volumetric Model Volumetric models provide more comprehensive information about the human body. They capture not only the positions but also the shape and appearance of the human body by using geometric shapes or polygon meshes (see Fig. 2.10b). Geometric shapes for modeling body parts include cylinders, ellipses and conics. Mesh-based representations model the body proportions and deformation using shape and pose parameters, respectively. Volumetric models are typically used for human mesh reconstruction, an extension of 3D HPE that aims at restoring not only the body pose but also the body shape.



(a) Kinematic models (adapted from [Isk+19]).



(b) Volumetric models (adapted from [Liu+22]).

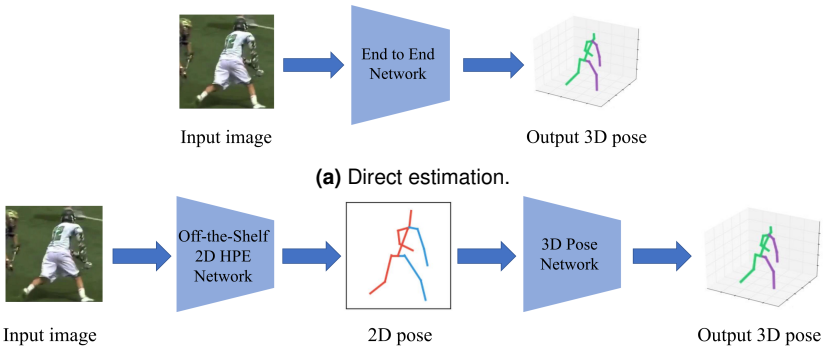
Figure 2.10: Common representations of the human body.

2.3.2 Methods

Existing approaches to 3D HPE are usually categorized as follows:

Direct Estimation Direct methods estimate the 3D pose from 2D image features in an end-to-end framework without intermediate pose representations (see Fig. 2.11 top) The 3D pose can be inferred in two ways: 1) Detection-based methods generate a volumetric heatmap per joint. The location of each joint is determined by the local maximum of its heatmap. 2) Regression-based methods directly predict the 3D coordinates of each joint.

2D-to-3D Lifting 2D-to-3D lifting is a two-stage pipeline that first performs 2D pose estimation to predict 2D joint positions in the input images and then lifts these intermediate 2D poses into 3D space (see Fig. 2.11 bottom). It builds on and benefits from the reliable and robust performance of state-of-the-art 2D pose detectors. Most modern approaches adopt this two-stage pipeline as it generally outperforms direct methods. They usually use an off-the-shelf 2D pose detector (e.g. High-Resolution Networks (HRNet) [Wan+21b], Cascaded Pyramid Networks (CPN) [Che+18], Stacked Hourglass Networks (SH) [NYD16]) to obtain the 2D poses and mainly focus on regressing the 3D poses from these 2D keypoints.



(a) Direct estimation.

(b) 2D-to-3D lifting.

Figure 2.11: Approaches to 3D HPE (adapted from [Zhe+20]).

2.3.3 Challenges

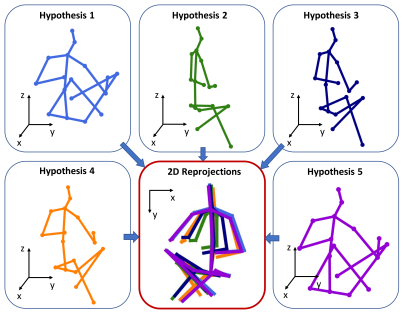
3D HPE presents some unique challenges, the most important of which are the following:

Depth Ambiguity Depth ambiguity occurs due to the loss of depth information when projection from 3D to 2D space (see Sec. 2.1.1). As a result, a single 2D pose may correspond to multiple 3D poses (see Fig. 2.12a). This is particularly challenging in the pipeline of 2D-to-3D lifting. Multiple feasible solutions exist, which makes lifting 2D poses to 3D an inherently ill-posed problem.

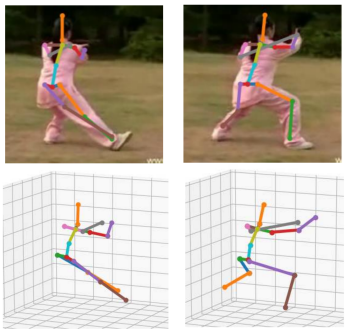
(Self-)Occlusion Occlusion means that a joint is not visible in an image because it is hidden by objects or other people in the scene. Self-occlusion occurs when a joint is hidden by other body parts of the owner of the joint. Even very accurate 2D pose detectors may fail to estimate the correct positions of the joints in such cases, resulting in noisy or missing 2D detections (see Fig. 2.12b top). 2D-to-3D lifting methods that rely on full and correct 2D poses as input become prone to errors (see Fig. 2.12b bottom).

Insufficient Training Data It is easy to construct large in-the-wild datasets for 2D HPE by manually annotating the 2D poses of people in the image. In contrast, collecting accurate 3D pose annotations for 3D HPE datasets requires marker-based motion capture

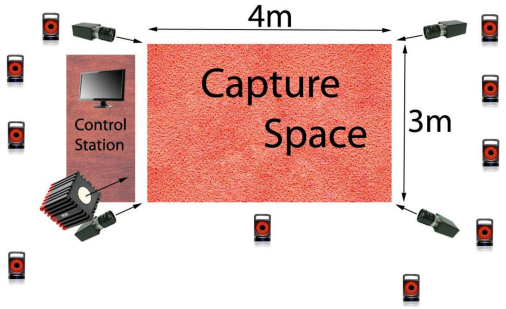
systems due to depth ambiguity. This makes the acquisition of a large scale in-the-wild dataset with 3D annotations very resource-intensive. Most existing datasets (e.g. Human3.6M [Ion+14], HumanEva [SBB10]) have been obtained under constrained conditions with limited generalizability (see Fig. 2.12c). The variability of human visual appearance (physique, clothing) and complex environments (background variations, lightning conditions, viewing angles) cannot be captured.



(a) Depth Ambiguity (taken from [LL19]).



(b) Self-occlusion (adapted from [Che+19]).



(c) Constrained data acquisition (adapted from [Ion+14]).

Figure 2.12: Challenges of 3D HPE.

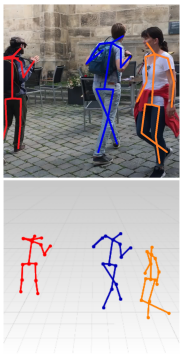
2.3.4 Variants

Several factors affect the 3D HPE pipeline, the combination of which results in many frameworks that handle different applications. These factors include the number of people, the number of images and the number of viewpoints in a given input. The following 3D HPE variants emerge:

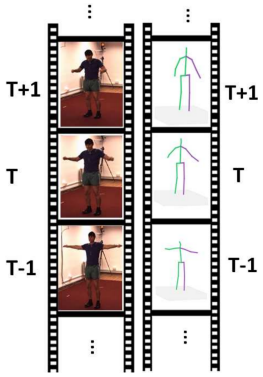
Single-Person vs. Multi-Person We differentiate between single-person pose estimation and multi-person pose estimation depending on the number of people in a given input (see Fig. 2.13a). Unlike in single-person pose estimation, the number of people in multi-person pose estimation is not known in advance. Inter-person occlusion introduces additional challenges.

Single-Frame vs. Multi-Frame We further distinguish between single-frame and multi-frame methods depending on the number of images used as input (see Fig. 2.13b). Multi-frame methods estimate the 3D pose from a sequence of consecutive images (a video) rather than from a single image. Exploiting temporal information across multiple frames can help mitigate the problem of (self-)occlusion, as occluded body parts in one frame may become visible in other frames.

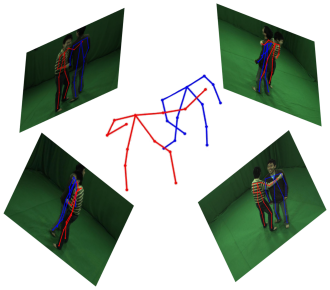
Monocular vs. Multi-View 3D HPE can be further classified into monocular and multi-view methods according to the number of viewpoints in an input (see Fig. 2.13c). Monocular systems capture a scene from only one viewpoint, whereas in multi-view systems, it is captured by multiple cameras from different viewpoints. Multiple views can help with (self-)occlusion and reduce depth ambiguity. However, aggregating information from multiple cameras is complicated and usually requires a calibrated and synchronized setup.



(a) Multi-person 3D HPE (adapted from [Che+21b]).



(b) Multi-frame 3D HPE (adapted from [Cai+19]).



(c) Multi-view 3D HPE (taken from [Li+19]).

Figure 2.13: Variants of 3D HPE.

Chapter 3

Related Work

The following section summarizes recent methods on monocular 3D single-person HPE from a single image or video to identify the current state of the art and its limitations. It focuses on GCN- and Transformer-based approaches as well as methods that explicitly deal with occlusion. All methods mentioned adopt a 2D-to-3D lifting pipeline.

3.1 GCN-based Approaches

A human body can be naturally represented as an undirected graph where the joints are the nodes and the bones are the edges. GCNs can extract useful information with relatively little compute and parameters. Therefore, it is natural to apply GCNs to estimate 3D poses from 2D poses [Ci+19]. The initial feature vectors of the nodes correspond to the 2D location of the respective joints. The final feature vectors are the estimated 3D locations. Many researchers have achieved promising results in this way.

Semantic Graph Convolutional Networks Zhao et al. [Zha+19b] propose a novel graph neural network architecture for regression tasks with graph-structured data such as 3D HPE. Their work addresses two limitations of standard GCN operations:

1. The way the weight matrix is shared across nodes: Conventional GCNs learn a shared transformation matrix for all nodes to handle various numbers of neighboring nodes.

2. The small receptive field: Conventional GCNs restrict the convolutional filters to operate on a one-ring neighborhood around each node.

They present Semantic Graph Convolution (SemGConv), an improved graph convolution operation, to tackle the problem of weight sharing by learning additional weight vectors $\boldsymbol{m}_i \in \mathbb{R}^n$ for each node i. The learned weights (combined with the adjacency matrix) indicate the local semantic relationships of neighboring nodes implied in the graph. Specifically, SemGConv learns a set of weight matrices $\boldsymbol{M}_d \in \mathbb{R}^{n \times n}$ for each channel d of output node features. The proposed operation is illustrated in Fig. 3.1b. They further suggest the use of non-local layers [Wan+18] to tackle the problem of limited receptive field and to capture global relationships between nodes.

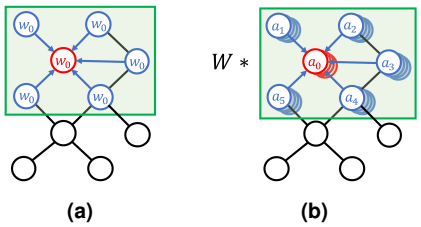


Figure 3.1: (a) Conventional graph convolution vs. (b) Semantic Graph Convolution. The figure is adapted from Figure 1 in [Zha+19b].

The authors introduce Semantic Graph Convolutional Networks (SemGCN), which consists of alternating SemGConv and non-local layers. SemGCN predicts a 3D pose from 2D joints as well as image features based on a single RGB image. The model is able to learn semantic information such as local and global node relationships that are not explicitly represented in the graph (e.g. how one joint influences other body parts).

Pre-Aggregation Graph Convolution Liu et al. [Liu+20a] study the effect of different weight sharing techniques and their impact on the task of 3D HPE. Specifically, they design five different weight sharing strategies (see Fig. 3.2): Full-sharing corresponds to the conventional GCN-operation and shares the transformation matrix among all nodes. Pre-aggregation first transforms each neighboring node feature using different weights before aggregating them to obtain the new target node feature. Post-aggregation aggregates the neighborhood information first and then transforms the resulting feature using a different weight matrix depending on the target node. No-sharing combines pre- and post-aggregation and assigns a different weight matrix between any pair of input and output nodes. Convolution-style shares weights according to the displacement between two nodes on the graph, i.e. the relationships between a node and (1) itself, (2) a neighboring body joint farther from the body center and (3) a neighboring body joint closer to the body center are modeled separately. The authors also suggest to decouple the self-connections in the graph by using a separate weight to compute the self-information transformation, leading to three more variants for full-sharing, pre-aggregation and post-aggregation. Eventually, they train a simple GCN-based model for 2D-to-3D pose lifting for comparison and conclude that among all variants, pre-aggregation is the optimal weight sharing method for 3D HPE and that it is always beneficial to decouple self-connections.

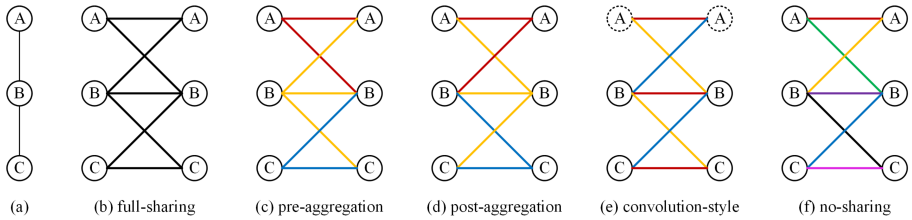


Figure 3.2: The five weight sharing methods by Liu et al. (taken from [Liu+20a]). Different colors encode different weights.

Spatial-Temporal Graph Convolutional Networks Cai et al. [Cai+19] utilize GCNs to exploit spatial and temporal relationships for 3D pose estimation from a series of 2D poses. They propose a spatial-temporal graph structure formulated on a sequence of skeletal forms as shown in Fig. 3.3. The graph topology is formed with joints as the graph nodes connected by two types of edges: Spatial edges, corresponding to the natural connectivity of joints, and temporal edges, connecting the same joint across consecutive frames.

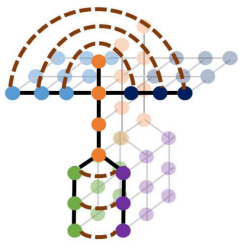


Figure 3.3: The spatial-temporal graph structure (taken from [Cai+19]).

Like Zhao et al. [Zha+19b], the authors also address the problem of weight sharing in conventional GCNs. For this purpose, the nodes are divided into six classes based on intuitive interpretations (see Fig. 3.4): 1) the center node itself; 2) a physically-connected neighboring node closer to the root node; 3) a physically-connected neighboring node farther from the root node; 4) an indirect “symmetrically-related” neighboring node; 5) a time-forward neighboring node; and 6) a time-backward neighboring node. They employ a

non-uniform graph convolutional strategy that learns different convolutional filter weights for different neighborhood nodes according to their semantic meaning. However, the modified operation applies only one single learnable mask to all channels, whereas SemGConv learns channel-wise different weights. The authors further design a "local-to-global" hierarchical network architecture to process information at multiple resolutions through successive graph pooling and upsampling layers. The proposed graph pooling operation gradually clusters the entire skeleton per frame (see Fig. 3.5). The upsampling procedure is performed in reverse order, duplicating the features of nodes in the coarser graph to the corresponding child nodes in the finer scale. A pose refinement process is introduced as a last step to further improve the estimation accuracy. The module takes as input the estimated 3D pose in two representations and outputs the confidence values for the two results. The refined 3D pose is computed as the confidence-weighted sum of the two estimation results.

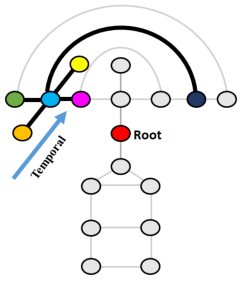


Figure 3.4: Classification of neighboring nodes based on their semantic meaning (taken from [Cai+19]).

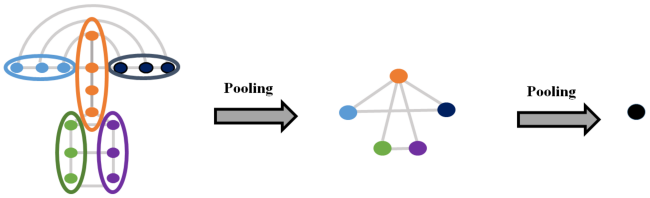


Figure 3.5: Hierarchical graph pooling (taken from [Cai+19]). The temporal links remain the same.

Graph Stacked Hourglass Networks Xu et al. [XT21] do not attempt to develop an improved graph convolution operation, but rather consider how GCNs can be effectively integrated into the architecture. They claim that sequentially connecting graph convolutional layers does not take advantage of the different model depths (see Fig. 3.6 top). The extracted features are oversimplified limiting the expressiveness and performance of existing architectures. The authors design Graph Stacked Hourglass Networks for 2D-to-3D human pose estimation from a single image. The proposed architecture (see Fig. 3.6 bottom) is able to extract multi-scale and multi-level features through repeated encoder-decoder modules (called Graph Hourglass modules). Each module processes features at different scales using novel skeletal pooling and unpooling operations on the skeleton structure (see Fig. 3.7). In this way the model is able to learn both local and global information, similar to the model of Cai et al. [Cai+19]. The outputs of the hourglass modules are further processed as intermediate features to learn multi-level features at each depth level of the model.

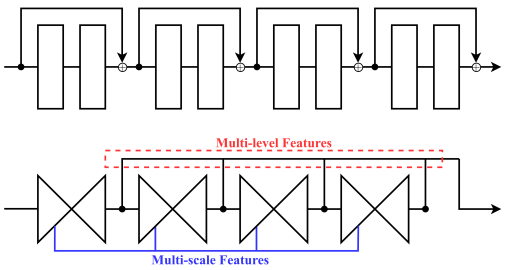


Figure 3.6: Other architectures (top) vs. Graph Stacked Hourglass Networks (bottom) (taken from [XT21]).

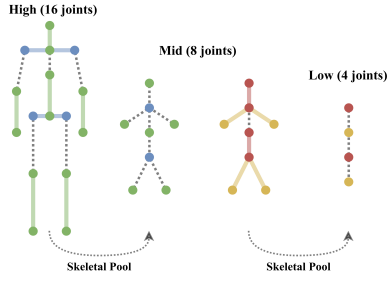


Figure 3.7: Skeletal pooling (taken from [XT21]). Skeletal unpooling is done in reverse order.

U-shaped Conditional Directed Graph Convolutional Networks Hu et al. [Hu+21] claim that modeling the natural hierarchy of human skeletons is beneficial for 3D pose estimation. They argue that undirected graphs fail to reflect the anatomical characteristics of the human skeleton because the hierarchical order between joints is not explicitly modeled. They suggest to represent the human skeleton as a directed graph, with joints as nodes and bones as edges directed from parent joints to child joints (see Fig. 3.8). The directions of edges explicitly reflect the hierarchical relationship between the nodes. Similar to Cai et al. [Cai+19], they construct a spatial-temporal directed graph on a sequence of directed graphs to additionally capture temporal relationships. They further condition the graph topology on the input poses, since non-local dependencies can vary greatly for different poses. They develop spatial-temporal directed graph convolution (ST-DGConv) to exploit the spatial relations between joints through prior knowledge of the hierarchical structure of the human skeleton. They additionally present spatial-temporal conditional directed graph convolution (ST-ConvDGConv) to aggregate the spatial information both locally and non-locally by adopting appropriate connections for different poses. Both operations aggregate temporal information by 1D convolution along the temporal dimension.

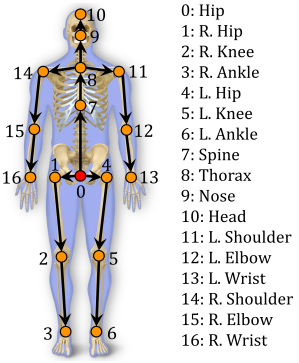


Figure 3.8: The directed skeleton graph (taken from [Hu+21]).

They construct U-shaped Conditional Directed Graph Convolutional Networks (U-CondDGCN) for multi-frame 2D-to-3D pose estimation using both proposed operations. The model of Cai et al. [Cai+19] and U-CondDGCN both operate on multiple 2D poses, but differ in that the former outputs only a single 3D pose, while the latter outputs a sequence of 3D poses. Temporal downsampling is used to reduce the temporal resolution for a larger temporal receptive field. Temporal upsampling is used to recover higher temporal resolution.

Skeletal Graph Neural Networks Zeng et al. [Zen+21] observe that while the average prediction accuracy in monocular 2D-to-3D pose estimation has been improved significantly over the years, the performance on certain poses is far from satisfactory. These poses are characterized by depth ambiguity, self-occlusion or complexity and occur, for example, while sitting (down). They also include poses that rarely occur in the training data set. The authors collectively refer to poses with high prediction errors as hard poses. The authors attribute the failure of existing GNN-based solutions for hard poses to the following two reasons:

1. Existing work does not consider that the aggregation of neighboring nodes contributes both useful information and unwanted noise. Distant nodes can provide useful information, but the further away the nodes are, the more likely unwanted noise is introduced into the aggregation procedure.

2. The relationship between body joints varies for different poses (e.g. hand and foot joints are closely related when running, but not when sitting). A static skeleton graph cannot capture this information across all poses.

They propose a novel skeletal GNN learning solution for robust 3D pose estimation of hard poses by overcoming these two limitations. A hierarchical channel-squeezing fusion (HCSF) layer effectively extracts relevant information while suppressing irrelevant noises by distinguishing short-range and long-range context (see Fig. 3.9b) Short-range features contain the most essential context of the target node. Their whole information is kept without squeezing them. Long-range features are compressed and fused to remove irrelevant information while keeping their essential components. A temporal-aware learning method constructs dynamic

graphs to capture action-specific poses. The edges of the dynamic graph not only match the fixed predefined topology of the human skeleton, but also evolve adaptively based on the node features. Temporal information is integrated into the process to cope with the change of dynamic graphs over time and to make it robust against single-frame outliers.

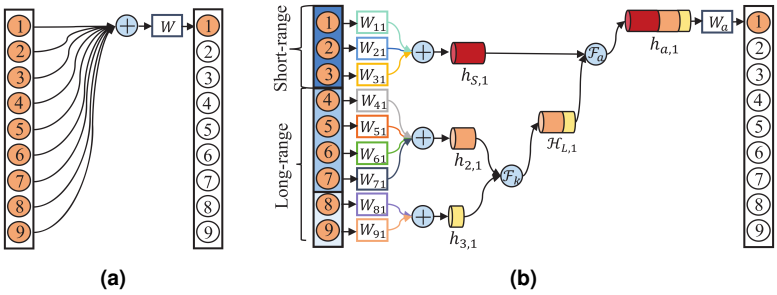


Figure 3.9: (a) Graph convolution vs. (b) Hierarchical Channel-Squeezing Fusion. The figure is adapted from Figure 5 in [Zen+21].

3.2 Transformer-based Approaches

The Transformer is able to efficiently capture global correlations over long input sequences through its self-attention mechanism. This makes it particularly suitable for sequence data problems and therefore naturally extendable to 2D-to-3D HPE from video [Zhe+21]. By definition, each frame (i.e. an entire 2D pose) of an input sequence of frames is considered as an individual token. With its strong modeling capabilities, the Transformer provides the opportunity to learn stronger temporal representations across arbitrary frames. Research in 3D HPE has successfully taken advantage of this property, outperforming purely GCN-based approaches that suffer from the small receptive field problem.

PoseFormer Zheng et al. [Zhe+21] propose PoseFormer, the first purely Transformer-based approach for 2D-to-3D lifting HPE from video. The authors not only aim to exploit temporal correlations across frames, but also to comprehensively model spatial (joint-to-joint) relationships within each frame. To capture both spatial and temporal elements, they utilize two separate transformer modules for both dimensions. The architecture of both modules follow the general ViT pipeline. As shown in Fig. 3.10, PoseFormer consists of three modules in total: Spatial transformer module, temporal transformer module and regression head module. PoseFormer takes a sequence of detected 2D poses and outputs the 3D pose for the center frame. First, the spatial transformer module extracts a high-dimensional latent feature embedding for each 2D pose separately. The input 2D pose is processed as a sequence of joints, with each 2D joint coordinate considered as a token. It thereby encodes local relationships between joints within a single frame.

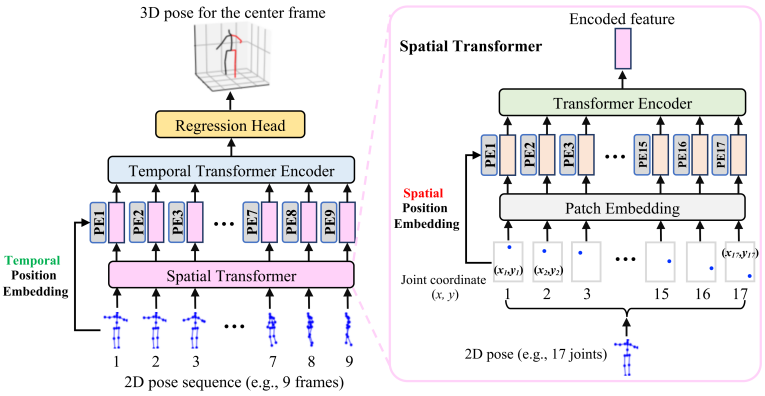


Figure 3.10: The PoseFormer architecture (taken from [Zhe+21]).

The strength of the connections between joints is determined by the self-attention mechanism of the Transformer, rather than by a predefined adjacency matrix as in typical GCN-based formulations. The module essentially forms a fully-connected graph on the joints, where the edge weights (i.e. the neighborhood aggregation function) are computed using input-conditioned, multi-headed attention scoring. It is able to flexibly adjust the relative importance of the joints to each other with each input pose. Next, the temporal transformer module captures global dependencies between frames throughout the entire sequence, regardless of their distance. To do so, it analyzes the relationship between all spatial feature representations in the sequence. The regression head applies a weighted mean operation (with learned weights) on the frame dimension to reduce the sequence output of the temporal transformer to a single frame representation. Finally, it generates an accurate 3D pose estimation of the center frame through a linear layer.

CrossFormer Hassanin et al. [Has+22] criticize the use of standard Transformer encoders in existing 3D HPE methods due to the non-local nature of the self-attention operation. As low-score relationships between joints are neglected, partially visible or occluded joints may not be represent properly and therefore cannot be reflected correctly in 3D space. The authors aim to improve the locality and inter-feature representation of the Transformer by integrating locality and rich inter-features interaction, while retaining the key advantages of the original Transformer (i.e. capacity to handle large number of tokens).
To this end, they design CrossFormer, a novel Transformer architecture for 3D pose estimation that provides rich representations of body joints to capture subtle changes across frames (i.e. inter-feature representation). Fig. 3.11 shows a conceptual illustration of the proposed framework. CrossFormer extends the architecture of PoseFormer [Zhe+21] by integrating cross-feature (joints) and cross-frame interaction modules into the vanilla spatial and temporal transformer module, respectively. Cross-Joint Interaction (CJI) resolves the locality issue of the spatial transformer module through improved attention scoring. Through non-local operations, it accounts for the locality of human body parts and their local interactions as much as encoding their non-local interactions (i.e. long-range dependencies). It further enriches the representation of joints with low attention scores through explicit encoding of the interaction between joints of a body part across channels. Cross-Frame Interaction (CFI) explicitly encodes the interaction of joints across subsequent frames for richer feature representations. It uses bilinear pooling to learn pairwise interactions between the same joint across different frames instead of traditional softmax as in vanilla self-attention. Attention is expanded to all channels instead of merging information across channel dimensions. This facilitates the discriminability of each frame and helps to model fine-grained temporal dynamics of the body parts.

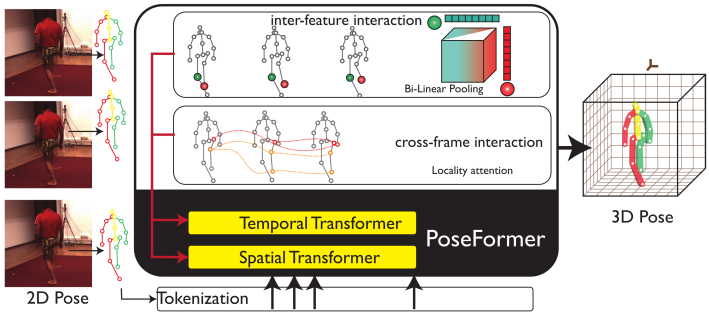


Figure 3.11: The CrossFormer architecture (taken from [Has+22]).

StridedTransformer Full-length sequences of popular HPE datasets contain significant redundancy for multi-frame pose estimation as nearby poses are highly similar. Li et al. [Li+22a] claim that existing methods do not sufficiently exploit this redundancy when estimating a 3D pose from multiple 2D poses. They further address two limitations of vanilla Transformer encoders (VTEs) for multi-frame 3D HPE:

1. The time and memory complexity of the attention operation grows quadratically with the input length, making it very expensive to process long sequences. While multiple frames are important to improve the estimation accuracy, the temporal receptive field must not become too large, especially for real-time applications.

2. While the VTE architecture is very good at capturing global dependencies over long sequences, it is less capable of extracting fine-grained local feature patterns from full-length sequences.

The authors propose to mitigate these issues by gradually merging nearby poses to shrink the sequence length until one representation of the target pose is acquired. They present a modified Transformer module called Strided Transformer Encoder (STE), which replaces the fully-connected layers in the feed-forward network with strided convolutions to progressively reduce the sequence length.
StridedTransformer is an improved Transformer-based architecture with strided convolutions to effectively lift a long sequence of 2D poses to a single 3D pose. It consists of a VTE followed by the proposed STE and is able to capture both global and local information in a hierarchical fashion. The complete model is illustrated in Fig. 3.12. First, the VTE captures long-range information in the entire sequence. It is supervised by the full 3D pose sequence to enforce temporal smoothness. Then, the STE progressively shrinks the sequence length and aggregates information in a hierarchical local-to-global fashion to generate one target pose representation. Its final output is supervised by the single 3D center pose to learn a specific representation for the target frame. This full-to-single supervision scheme is adopted to incorporate both full sequence and single target frame scales constraints into the framework. It further refines intermediate predictions to produce more accurate estimations rather than using a single component with a single output. The authors then additionally apply the same refinement module as Cai et al. [Cai+19].

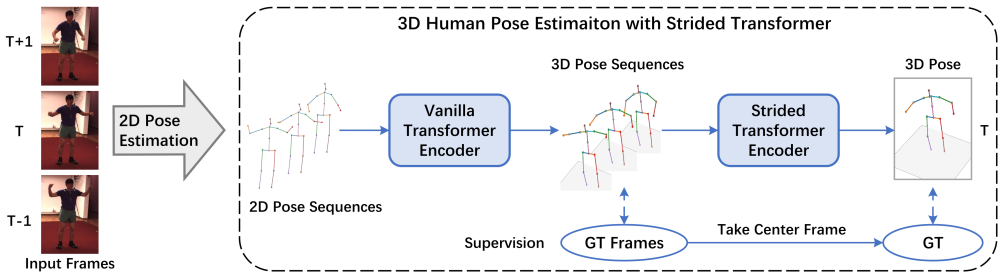


Figure 3.12: The StridedTransformer architecture (taken from [Li+22a]).

Pre-trained Spatial Temporal Many-to-One Model Shan et al. [Sha+22] propose a Pre-trained Spatial Temporal Many-to-One (P-STMO) model for 2D-to-3D HPE from video. The authors divide the extraction of spatial and temporal information into two stages, to reduce the complexity of this task: Pre-training (Stage I) and fine-tuning (Stage II). The model is supposed to capture 2D spatial temporal dependencies in Stage I and extract 3D spatial and temporal features in Stage II. The authors additionally adopt a temporal downsampling strategy (TDS) on the input side to reduce data redundancy while enlarging the temporal receptive field.
Fig. 3.13 depicts an overview of the proposed framework. Stage I consists of a self-supervised spatial temporal pre-training task, called masked pose modeling (MPM). Selected joints in the input sequence are masked randomly in spatial and temporal dimension. Spatial masking affects only a small subset of joints within a frame, while temporal masking affects all joints

(i.e. the entire pose). The coordinates of the masked joints are replaced by learnable vectors, as opposed to the very common approach of setting them to (0, 0) (center of the image). This is to avoid confusion and to better distinguish between masked and unmasked joints. A general form of denoising auto-encoder aims to recover the original 2D poses. It consists of a spatial-temporal encoder (SEM+TEM) that maps the masked input to the latent space, and a decoder that recovers the original 2D poses from latent representations. The spatial encoding module (SEM) captures spatial relationships between all joints within a single frame. A simple MLP block forms the backbone of SEM to ensure the scalability of the network when using multiple frames as inputs. Fully masked poses are not considered as input, which further reduces the computational complexity of the encoder. Any remaining partially masked 2D pose in the input sequence is sent independently to the MLP block, whose weights are shared across all frames. The temporal encoding module (TEM) is a standard Transformer architecture that models temporal dependencies between frames. Self-attention enables the network to capture non-local self-attending associations between frames that are far apart. The complete STMO model is a combination of the pre-trained encoder (SEM+TEM) with a many-to-one frame aggregator that combines information from multiple frames. In Stage II, the entire pipeline is trained to predict the 3D pose of the target (middle) frame from a sequence of 2D poses. Essentially, the pre-trained encoder is loaded and fine-tuned, while the rest of the pipeline is trained for the first time.

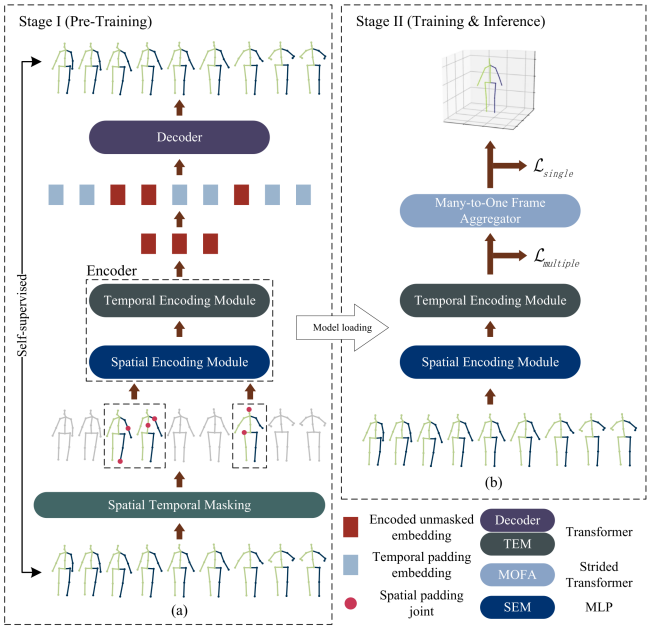


Figure 3.13: The P-STMO architecture (taken from [Sha+22]).

Multi-Hypothesis Transformer Li et al. [Li+22b] address the problem of ambiguity in monocular 3D HPE in a different way. Their work is based on the fact that 2D-to-3D lifting from monocular input is an inverse problem where multiple feasible solutions (i.e. hypotheses) exist. They propose Multi-Hypothesis Transformer (MHFormer) that learns spatio-temporal representations of diverse pose hypotheses to estimate the 3D pose of the center frame given a sequence of 2D poses. The model follows a three-stage pipeline that starts from generating multiple initial representations and gradually communicates across them to synthesize a more accurate prediction, as shown in Fig. 3.14. Essentially, MHFormer conducts a one-to-

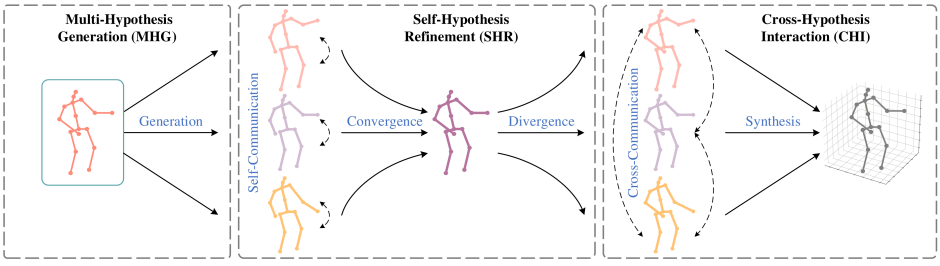


Figure 3.14: The three-stage pipeline of MHFormer (taken from [Li+22b]).

many mapping first and then a many-to-one mapping, to enrich the diversity among features of different hypotheses. The proposed model is built upon three key components:
Multi-Hypothesis Generation (MHG) models the intrinsic structure information of human joints in the spatial domain. It utilizes a cascaded Transformer-based architecture to output multiple features in different depths of the latent space. These multi-level features contain diverse semantic information and are therefore considered as initial representations of multiple hypotheses.
Following MHG, Self-Hypothesis Refinement (SHR) refines every single-hypothesis feature for temporal consistencies. First, a multi-hypothesis self-attention (MH-SA) module establishes self-hypothesis communication for intra-hypothesis message passing. It models single-hypothesis dependencies independently for feature enhancement. Second, a hypothesis-mixing multi-layer perceptron (HM-MLP) combines different hypotheses into a single convergent representation and then partitioning this representation into multiple divergent hypotheses. The whole process explores the relations among channels of different hypotheses to form refined hypothesis representations.
Subsequently, Cross-Hypothesis Interaction (CHI) models interactions among multi-hypothesis features in the temporal domain. A multi-hypothesis cross-attention (MH-CA) module builds cross-hypothesis communication for inter-hypothesis message passing. It captures mutual multi-hypothesis correlations to improve interaction modeling. Finally, a hypothesis-mixing MLP aggregates the different hypotheses to synthesize the final 3D pose.

JointFormer Lutz et al. [Lut+22] assume that enabling the network to compensate for its own uncertainty improves estimation results. To this end, they propose to predict the error associated with each joint. The true error is defined as the absolute difference between the predicted and the ground-truth pose. They present a novel single-frame 2D-3D lifting pipeline for human pose estimation, shown in Fig. 3.15. The network consists of two modules: The Joint Transformer module is a stack of Transformer encoders that estimates the 3D pose and the prediction error from a single 2D pose. The network is trained using intermediate supervision by applying a 3D pose regression head after each individual transformer encoder in the stack. This allows the network to learn initial estimates that get further refined by each Transformer encoder in the stack. Each Transformer encoder predicts its own error per joint and per coordinate through a second regression head. The error prediction forces the network to implicitly learn its own uncertainty and additionally stabilizes the training. The Refinement Transformer module refines the 3D pose predictions of the Joint Transformer by using the intermediate 3D prediction, the input 2D joint and prediction error for each joint. It has a similar architecture to the Joint Transformer with fewer Transformer encoders in the stack and a smaller hidden dimension. No intermediate supervision is applied. Following

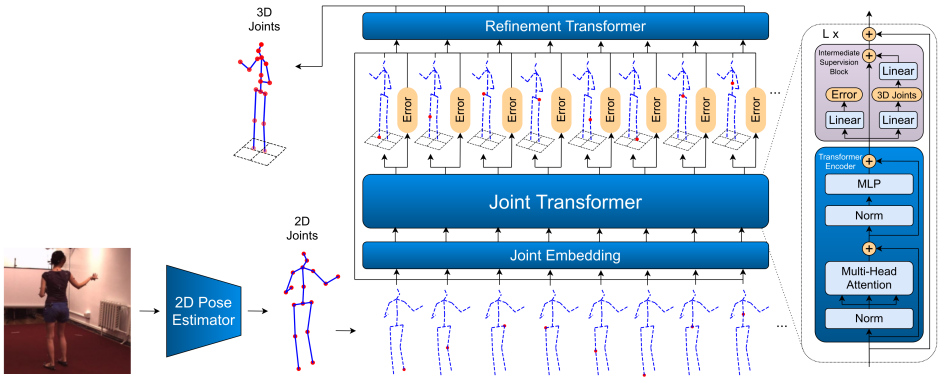


Figure 3.15: The JointFormer architecture (taken from [Lut+22]).

the stacked Transformer encoders, a linear layer regresses the final 3D pose. The authors further claim that the position of joints is already encoded implicitly as the input order of joints never changes. This makes additional encodings redundant. They omit any explicit positional encoding in both modules and use a 1D convolution with a filter size of 1 over all joints to expand the inputs to the corresponding hidden dimensions.

Mixed Spatio-Temporal Encoder Zhang et al. [Zha+22] claim that previous methods cannot efficiently model the solid inter-frame correspondence of each joint, leading to insufficient learning of spatial-temporal correlation. They suggest to learn the motion trajectories of the different body joints separately, as they vary from frame to frame. They propose Mixed Spatio-Temporal Encoder (MixSTE), a novel transformer-based seq2seq approach for 3D pose estimation from monocular video. Fig. 3.16 shows an overview of the framework. Like U-CondDGCN [Hu+21], MixSTE follows a seq2seq pipeline, i.e. it predicts the 3D poses of all input frames at once, not just the middle frame. This helps maintain global sequence coherence between input and output sequences to avoid excessive smoothness. It also increases efficiency by reducing redundant calculations. The proposed model has a periodic design with a stack architecture consisting of alternating Spatial Transformer Blocks (STBs) and Temporal Transformer Blocks (TTBs) to achieve better spatio-temporal feature encoding. STBs apply self-attention on joints to learn inter-joint spatial correlation in each frame. TTBs apply self-attention on frames to learn the global temporal motion of each joint. Opposed to previous methods, the temporal module does not treat each frame as a token, but models the motion trajectory of each joint separately to use each joint trajectory as an individual token. This separation of different joints in the temporal dimension enables the model to better capture temporal correlations. It also reduces the model dimension, allowing longer sequences to be processed. MixSTE alternately stacks STBs and TTBs for multiple loops while preserving the feature dimension to ensure that spatial-temporal correlation learning focuses on the same joint. The spatial and temporal position embedding is applied only in the first stack.

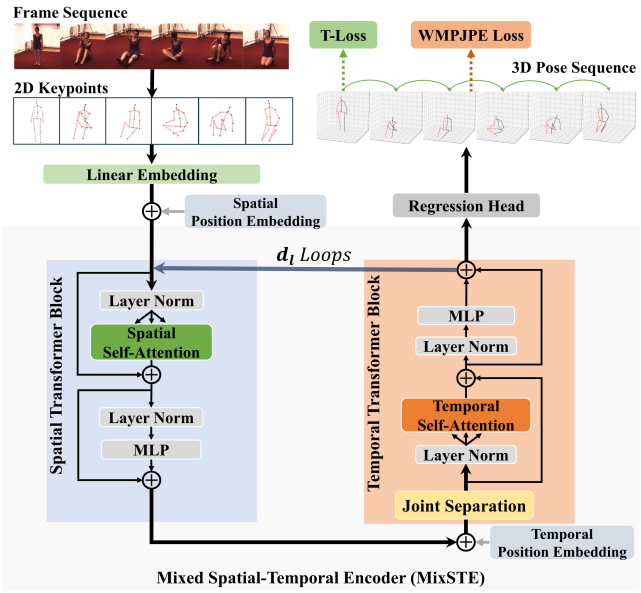


Figure 3.16: The MixSTE architecture (taken from [Zha+22]).

3.3 Combined Approaches

Transformer-based architectures have become state of the art for 3D HPE due to the strong modeling capabilities of the self-attention mechanism. However, the performance gain comes at the cost of ever-increasing computational requirements and complex models, as Zheng et al. [Zhe+22] observe. Moreover, substantial structural priors and geometric information provided by the human skeleton are ignored. Driven by the strengths and weaknesses of pure graph- and transformer-based approaches, other researchers create novel combined methods to benefit from both architectures.

Graph Transformer Encoder-Decoder with Atrous Convolution Zhu et al. [Zhu+21] propose a novel model named Graph Transformer Encoder-Decoder with Atrous Convolution (PoseGTAC) to effectively extract multi-scale context and long-range information. PoseGTAC consists of two alternately stacked key components. Graph Atrous Convolution (GAC) is an enhanced graph convolution that captures the multi-scale context of higher-order neighbors based on the local physical connectivity of the human skeleton. It obtains a larger receptive field by performing multiple graph convolutions with different dilation factors in parallel (see Fig. 3.17). The dilation factor is defined as the distance to the root node. Graph Transformer Layer (GTL) applies self-attention on the graph to determine implicit relationships between joints. A second global attention matrix encourages unconstrained learning. Graph pooling and unpooling are additionally incorporated into the pipeline to facilitate the interaction of local and global information (e.g. part-scale and body-scale).

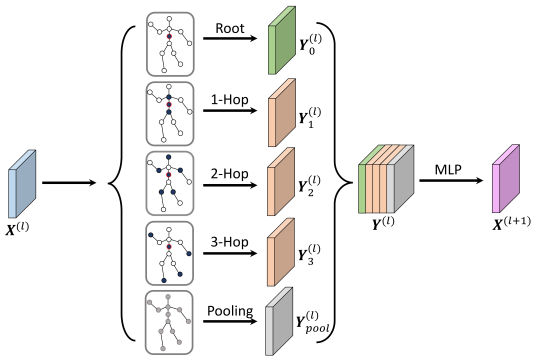


Figure 3.17: Graph Atrous Convolution (taken from [Zhu+21]).

Graph Transformer Networks Zheng et al. [Zhe+22] present Graph Transformer Networks (GTRS), a lightweight pose-based method for efficient human mesh reconstruction based on a single 2D pose. They propose graph transformer, a novel operation that combines GCNs with Transformers and improves the structured and implicit correlations based on the human kinematic information. It harnesses GCNs to form strong representations from inherent structural priors. Then follows a customized transformer block to model global dependencies: The embedding dimension is set to a smaller number and the MLP layer is replaced by an SE block [Hu+20] to reduce computational complexity. Each block starts with a graph convolution to inject structural priors before performing the self-attention. The authors design a pose analysis module (PAM) that employs multiple graph transformer blocks in a parallel fashion to model joint correlations in a lightweight manner (see Fig. 3.18). It utilizes fixed and learnable adjacency matrices to simultaneously explore diverse structured and implicit human joint correlations. A fusion block, which is a convolutional layer, fuses all paralleled features together to form an intermediate pose feature. An intermediate 3D pose regression head supervises PAM with the ground truth 3D pose. Following PAM, a mesh regression module (MRM) combines the extracted pose feature with the mesh template to reconstruct the final human mesh. MRM will not be discussed further, as it is beyond the scope of the thesis.

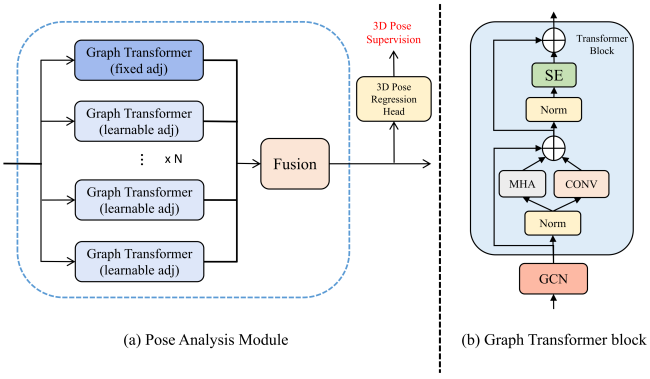


Figure 3.18: The Pose Analysis Module (PAM) architecture (taken from [Zhe+22]).

Graph Convolution Transformer Zhao et al. [Zha+21] claim that the self-attention mechanism cannot fully exploit the spatial relations of the human body, since it builds upon cal-

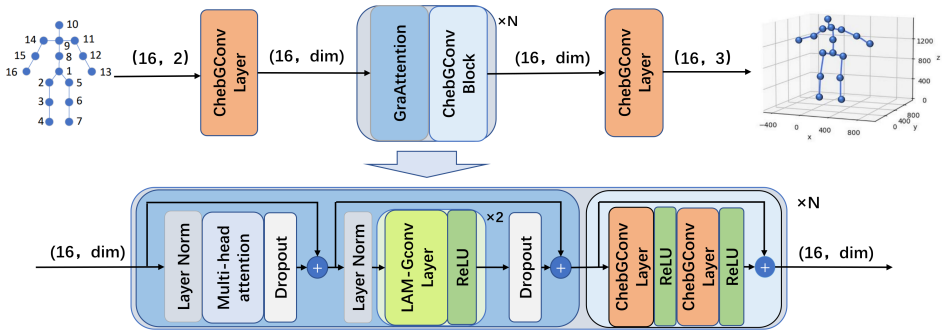


Figure 3.19: The GraFormer architecture (taken from [Zha+21]).

culating similarities between joints, which ignores or weakens the graph structure information. They present Graph Convolution Transformer (GraFormer), a novel architecture for 3D human pose estimation from a single image that combines self-attention with graph convolutions. Fig. 3.19 visualizes the proposed model. It comprises two repeatedly stacked core modules, GraAttention and ChebGConv. GraAttention is an improved Transformer encoder, where the MLP is replaced with graph convolution filters on a learnable adjacency matrix (called LAM-Gconv). The self-attention module facilitates the interaction among 2D joints based on the coordinate values. Compared to the MLP used in standard Transformers that treat the joints as a fully connected graph, LAM-Gconv updates a joint not only based on the scores of all other joints, but also based on the topology of a (learned) graph for smarter interaction. GraAttention is able to learn robust features by capturing all 2D joint relations in the global receptive field, without losing the details of a graph structure. ChebGConv is an improved graph convolution operation that uses the fixed adjacency matrix obtained from the human skeleton. Unlike vanilla graph convolutions that only model the apparent relationship of joints, ChebGConv enables 2D joints to interact in the high-order sphere to find hidden implicit relations. It is able to fuse information among the top K neighbors of a joint, which additionally brings a larger receptive field. The fixed adjacency matrix further strengthens the geometric information of the human skeleton.

3.4 Occlusion-related Approaches

Occlusion is one of the key problems in 3D HPE from monocular videos. Despite its relevance, there is little recent work that explicitly handles the problem of occlusion.

Occlusion-Aware Regression Forests Rafi et al. [RGL15] propose to incorporate semantic knowledge about occluding objects to improve pose estimation of occluded joints. To this end, they design a regression framework for 3D HPE from depth data that is based on regression forests. The proposed model, called Occlusion-Aware Regression Forests (OARF), predicts the 3D coordinates of each joint directly from a single depth image, following the direct estimation approach. OARF additionally predicts the class label of an occluding object at test time and uses this additional knowledge as context to improve the pose estimation of occluded joints.

Temporal Dilated Convolutions with Occlusion Guidance Most of the existing 2D pose detectors also generate a confidence score for each joint in addition to the geometric joint position. The confidence score decreases in case of occlusion, representing the loss of quality of the estimated keypoint. Ghafoor et al. [GM22] propose an occlusion guidance mechanism

based on these values to provide information about the visibility of joints. To this end, they present an occlusion guided framework for 3D HPE that is designed from dilated temporal convolutional networks (TCNs) to efficiently utilize temporal information. The model takes as input a sequence of estimated 2D poses with an occlusion guidance matrix and outputs a 3D pose corresponding to the center frame. The occlusion guidance matrix encodes the quality of keypoint estimates through one indicator variable each for the x- and y-coordinates of a 2D keypoint. A value of 1 indicates a high confidence score, a value of 0 indicates a low confidence score. Both indicator variables and corresponding coordinates are coupled together, as shown in Fig. 3.20, to effectively double the input dimensionality.

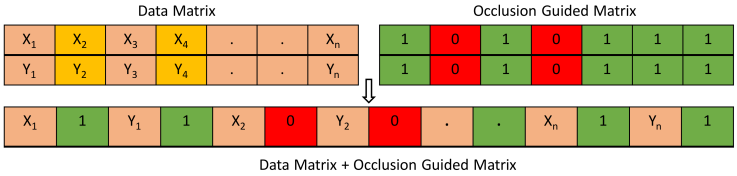


Figure 3.20: The occlusion guidance mechanism by Ghafoor et al. (taken from [GM22]).

Occlusion-Aware Temporal Convolutional Networks Cheng et al. [Che+19] propose an occlusion-aware framework that attempts to identify occluded keypoints. It follows the 2D-to-3D lifting approach, but integrates the estimation of 2D keypoints into the pipeline to train it in an end-to-end fashion. In total, the model consists of three components. Given an input video, the authors apply a state-of-the-art detector on each frame, to detect human individuals using bounding box. The first network estimates the 2D locations of the joints for each person in the form of confidence heatmaps. Each frame is processed independently. Subsequently, the confidence heatmaps are combined with an optical-flow consistency constraint to evaluate joint visibility and to filter out unreliable estimates of occluded joints. The potentially incomplete 2D keypoints are fed to the second and third networks, both of which utilize temporal convolutions to enforce temporal smoothness. The second network, a 2D temporal convolutional network (TCN), thereby aims to improve the accuracy of the 2D keypoint estimates. It does not intend to fill in the missing keypoints. Finally, the third network, a 3D TCN, outputs the 3D poses of all input frames. The authors additionally adopt the concept of adversarial learning by incorporating a discriminator into the pipeline to ensure anthropometric validity of each estimated 3D pose.
Training the occlusion-aware networks requires pairs of a ground truth 3D pose and a 2D pose with occlusion labels. As no such dataset is available, the authors introduce a novel Cylinder Man Model to approximate the occupation of body parts in 3D space (see Fig. 3.21). Given a 3D pose, the model is orthogonally projected onto a 2D plane to determine self-occluded joints. The occlusion labels are obtained by checking if the point in question is located within the projected 2D rectangle of a cylinder. To model occlusion by other persons or objects, some keypoints in the input are randomly masked. The Cylinder Man Model is further used for pose regularization of occluded keypoints when 3D ground-truths are not available.

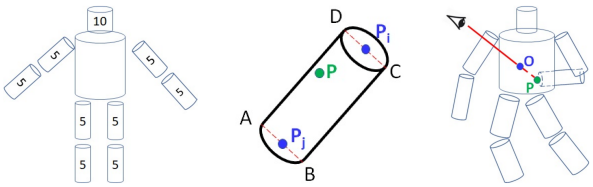


Figure 3.21: The Cylinder Man Model (taken from [Che+19]).

Spatio-Temporal Networks with Explicit Occlusion Training Cheng et al. [Che+20] perform diverse data augmentation to deal with occlusion. To this end, they introduce a spatio-temporal framework for robust 3D HPE from video. The authors explicitly mask some keypoints during training by setting the corresponding heatmaps to zero to simulate occlusion.

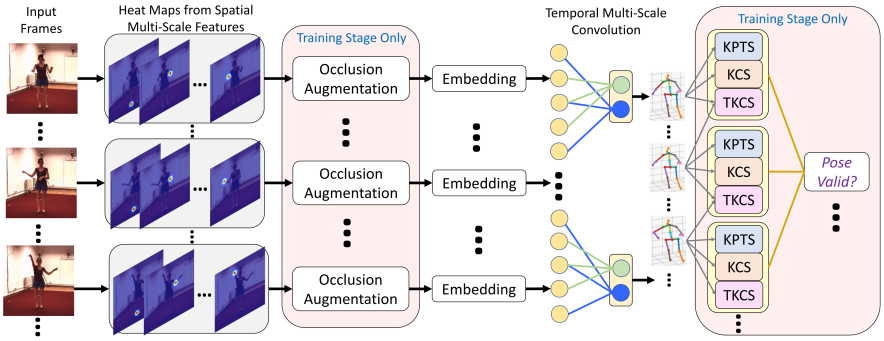


Figure 3.22: The complete framework proposed by Cheng et al. (taken from [Che+20]).

Different cases, from light to heavy occlusion, are simulated to make the model robust to various degrees of occlusion: Frame-wise occlusion randomly masks multiple frames by setting all their heatmaps to zero. Point-wise occlusion randomly masks certain keypoints by setting their corresponding heatmaps to zero. Areal occlusion specifies a partial virtual occluder area. All heatmaps of keypoints located within this area are set to zero. The authors further apply random noise to the heatmaps of the input sequence, since the output of 2D pose estimators is not strictly Gaussian distributed. In addition, the keypoints are randomly shifted or randomly swapped symmetrically to further improve robustness in case of wrong detection.

The proposed model extracts multi-scale spatial and temporal features for 2D joints in each individual frame or across frames, respectively, to deal with various scenarios (small/large target person, fast/slow motion). First, an off-the-shelf 2D pose detector generates confidence heatmaps to indicating the 2D locations of joints in the input. The estimated heatmaps encode multi-scale spatial features to provide more accurate 2D pose estimations. The model tries to maintain as much of these spatial information as possible by processing not only the peak values but the entire heatmaps. Next, multi-scale temporal features are extracted by multiple temporal convolutional networks (TCNs) with different strides and concatenated for the final pose estimation. A novel spatio-temporal discriminator based on body structures and limb motions determines whether the estimated 3D pose sequence represents a valid human motion. It considers not only the validity of a pose in individual frames, but also transitions between frames. Fig. 3.22 visualizes the entire pipeline.

Chapter 4

Methods and Materials

This thesis aims to approach the problem of occlusion in 3D HPE as comprehensively as possible by considering two perspectives. To this end, three different methods for occlusion-robust 3D HPE were developed. The first is a baseline approach aimed at occlusion robustness through a thoughtful model design. The other two approaches handle the problem of occlusion more explicitly by appropriate data augmentation and the help of an auxiliary task. The second approach addresses occlusion that causes some data points in the input pose to be missing while the third approach targets occlusion that results in noisy data points.

4.1 Spatial-Temporal Baseline Approach

The spatial-temporal model is a baseline approach for occlusion-robust 3D HPE using GCNs and Transformers. Central to this method is the differentiation between spatial and temporal context through separate modules. It is assumed that for occlusion robustness both dimensions must be modeled without neglecting substantial structural and geometric information provided by the human skeleton. The model aims to overcome occlusion by exploiting temporal information through self-attention and capturing spatial information through graph convolutions. It is based on GraFormer [Zha+21] and StridedTransformer [Li+22a], with some adaptions. Fig. 4.1 shows an architectural overview of the entire pipeline.

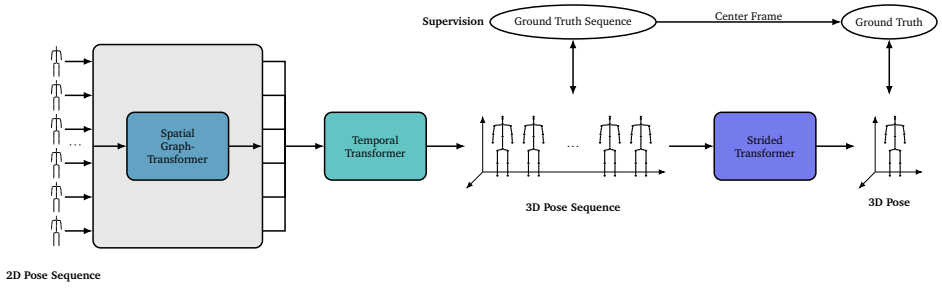


Figure 4.1: Overview of the proposed spatial-temporal baseline model for predicting the 3D pose of the target (center) frame (based on Fig. 3 from [Li+22a]).

It consists of the following components:

Spatial Graph-Transformer (SGT) The first module combines graph convolutions and self-attention to model the spatial relations of the joints within a frame separately. It utilizes GraAttention and ChebGConv layers from GraFormer [Zha+21]. First, the ChebGConv layer embeds the coordinates of each joint to a higher dimension. The input 2D pose $\boldsymbol{x}_t \in \mathbb{R}^{1\times(J\cdot 2)}$ with J joints at frame t is transformed to embedded pose features $\boldsymbol{P}_t^0 \in \mathbb{R}^{J\times d_s}$. Next, a stack of

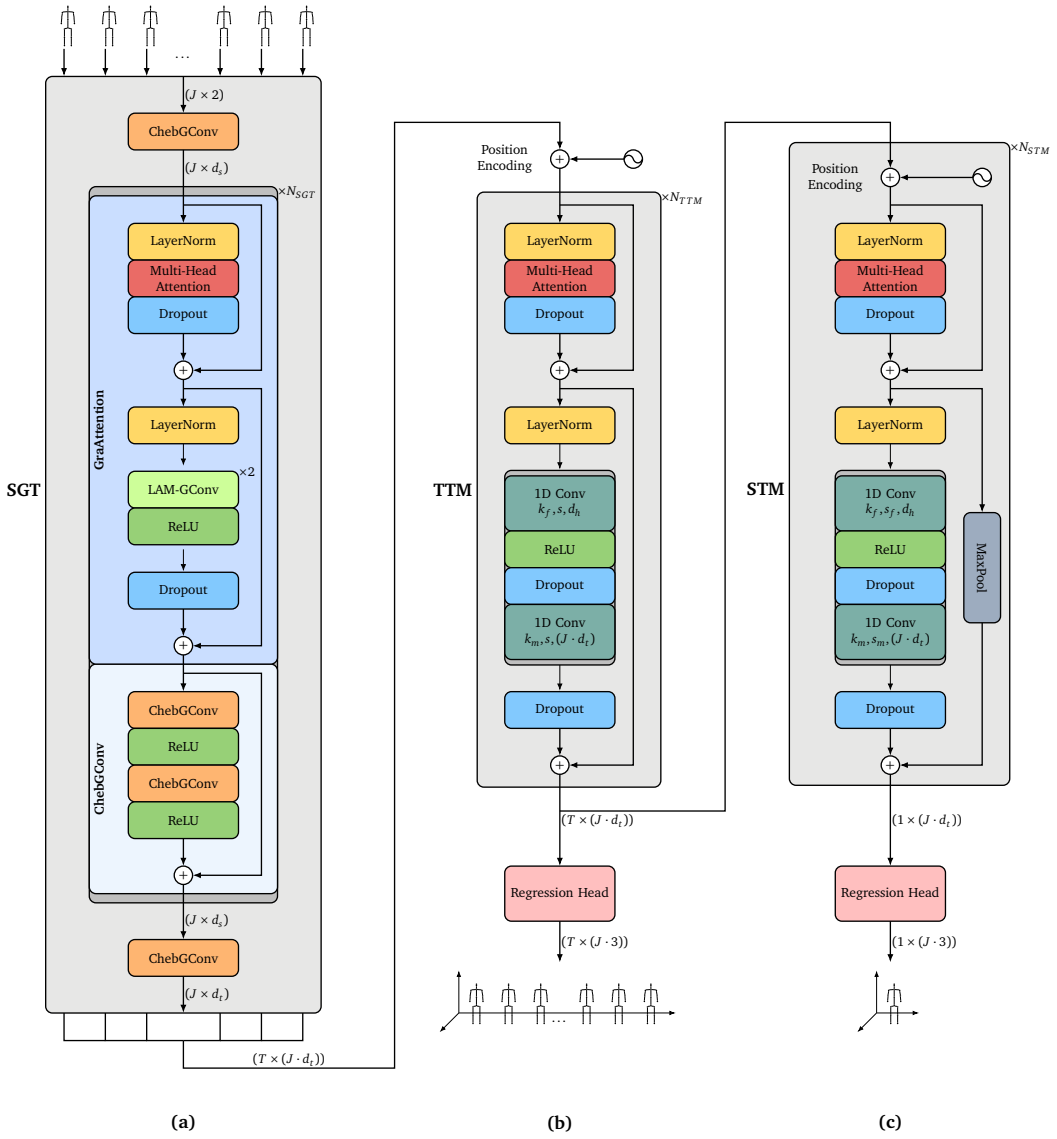


Figure 4.2: The network architecture of the spatial-temporal baseline model consisting of (a) a Spatial Graph-Transformer (SGT) module, (b) a Temporal Transformer Module (TTM) and (c) a Strided Transformer Module (STM).

N_{SGT} identical layers follows, each consisting of one GraAttention and one ChebGConv block. All layers preserve the feature embedding dimension d_s. $\boldsymbol{P}_t^i$ denotes the output of the ith layer, with $i \in \{0, \ldots, N_{SGT} - 1\}$. Lastly, another ChebGConv layer reduces the d_s-dimensional embedding of each joint to d_p (with $d_s > d_p$) for the following modules to ensure that the number of model parameters does not become too large. $\tilde{\boldsymbol{P}}_t \in \mathbb{R}^{J \times d_p}$ denotes the final output of the SGT for frame t. The component is visualized in Fig. 4.2a. See Sec. 3.3 and the original GraFormer paper [Zha+21] for a detailed description of the GraAttention and ChebGConv layers.
For the next layers, the output $\tilde{\boldsymbol{P}}_t$ of each frame t is flattened to a vector $\boldsymbol{p}_t \in \mathbb{R}^{1 \times (J \cdot d_p)}$. The vectors $\{\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_{T-1}\}$ from the T input frames are then concatenated as $\boldsymbol{Z}_0 \in \mathbb{R}^{T \times (J \cdot d_p)}$.

Temporal Transformer Module (TTM) The second module is a stack of N_{TTM} modified Transformer encoders that process the entire sequence of frames to exploit temporal context. Inspired by the Strided Transformer Encoder (STE) from [Li+22a], the modified encoders replace the standard fully-connected layers in the FFN by 1D convolutions to better integrate

local information across all frames. The decisive factor is that a stride of $s = 1$ is used in all convolutional layers to process the entire sequence without shrinking it. The module is built upon the flattened and concatenated outputs of the SGT and takes $\boldsymbol{Z}_0 \in \mathbb{R}^{T\times(J\cdot d_p)}$ as input. First, a learnable temporal position encoding $\boldsymbol{E}_1 \in \mathbb{R}^{T\times(J\cdot d_p)}$ is added to the input to retain frame position information. The resulting features are then passed to the encoders to produce the final output $\boldsymbol{Z}_1 \in \mathbb{R}^{T\times(J\cdot d_p)}$. The component is visualized in Fig. 4.2b.

Strided Transformer Module (STM) The third module consists of a stack of N_{STM} Strided Transformer Encoders (STEs) from [Li+22a] that exploit temporal features on a multi-scale basis. It takes $\boldsymbol{Z}_1$ as input and progressively shrinks the entire sequence via strided convolutions. Different learnable position encodings $\boldsymbol{E}_2^i \in \mathbb{R}^{L_i\times(J\cdot d_p)}$ are applied before each encoder i due to the shrinking input sequence. L_i denotes the length of the input sequence for encoder $i \in \{0,\dots,N_{STM}-1\}$. The final output $\boldsymbol{z}_2 \in \mathbb{R}^{1\times(J\cdot d_p)}$ encodes the 3D pose of the center frame. The component is visualized in Fig. 4.2c. See Sec. 3.2 and the original StridedTransformer paper [Zha+21] for more details about this module.

Regression Head Both full sequence and single target frame constraints are incorporated into the framework as in the full-to-single supervision scheme of StridedTransformer [Li+22a]. To this end, two separate regression heads, each with batch normalization [IS15] and a 1D convolutional layer, are applied to the outputs of the TTM and the STM, respectively. The results $\tilde{\boldsymbol{X}} \in \mathbb{R}^{T\times(J\cdot 3)}$ and $\tilde{\boldsymbol{x}} \in \mathbb{R}^{1\times(J\cdot 3)}$ correspond to the 3D pose predictions of the entire sequence and the target frame, respectively.

Pose Refinement Finally, the pose refinement module of Cai et al. [Cai+19] is applied to further enhance the estimation results. It is a simple two-layer fully-connected network that takes as input the estimated target 3D pose in two representations and outputs the confidence values for the two results. The first representation are root-relative 3D coordinates of the joints in the camera coordinate system, i.e. $\tilde{\boldsymbol{x}}$. The second representation is based on the concatenation of the predicted depths of each joint and its corresponding 2D coordinates of the input pose. The depth values in the second representation are directly obtained from the first representation. The refined 3D pose $\tilde{\boldsymbol{x}}^r$ is computed as the confidence-weighted sum of the two estimation results.

4.2 Data-driven Approach

The second approach is a data-driven solution to the occlusion problem that extends the existing baseline with occlusion-based data augmentation at training-time for increased test-time occlusion robustness. Occlusion naturally occurs in all 3D HPE datasets. However, the baseline method treats all keypoints equally, regardless of the fact that some may be affected by occlusion. This is because standard benchmarks do not systematically model occlusion effects, i.e. there is no information on which body parts are occluded. The data-driven approach circumvents this problem by additionally introducing synthetic occlusion to the available data (i.e. occlusion augmentation). Joints are marked as (artificially) occluded by setting their 2D coordinates to 0. The network is forced to rely on other relevant features to predict a correct 3D pose from an incomplete sequence of 2D poses.
The occlusion augmentation applied is essentially an improved method of explicit occlusion training presented by Cheng et al. [Che+20], where several joints are randomly masked to simulate the occluded condition. An advanced joint-occlusion procedure is designed to simulate different degrees of occlusion: Given a sequence of input frames, several frames are masked by occluding various random joints. The total number of occluded frames q_f

per sequence is within $[0, q_F]$ and is determined randomly using a uniform distribution. In real-world scenarios, joints may not be detected for longer periods of time due to occlusion. Therefore, the frames are occluded in subsets consisting of q_t consecutive frames to mimic occlusion in reality. q_f and q_t determine the number of subsets to be occluded ($= \lfloor q_f / q_t \rfloor$). First, the middle frames of each subset are determined using a uniform distribution. Then, starting from the selected frames, the subsets are formed using a heuristic. Note that this can lead to overlapping of the subsets if the drawn frames are close to each other. So, in practice, a subset may consist of fewer frames than q_t. q_s joints per frame are occluded. Each joint j is occluded with probability p_j based on a categorical distribution P prior given. The same joints are occluded in each frame of a subset, but the occluded joints may vary from subset to subset.

4.3 Model-driven Approach

The third approach makes use of an auxiliary method to improve predictions under occlusion by extracting additional information from the input 2D poses. Specifically, the goal of the auxiliary task is to identify noisy and thus unreliable keypoints in the input. This is in the spirit of predicting which of the joints are occluded, since occlusion leads to noisy 2D detections.[1] To this end, artificial target labels indicating noise are created by measuring the accuracy of the 2D keypoint detections with the 2D ground truth according to the head-normalized percentage of correct keypoints (PCKh) metric from [And+14]. PCKh is based on the distance between two keypoints in 2D normalized by the true head size h:

$$\begin{aligned} \text{dist}(\boldsymbol{x}_j, \tilde{\boldsymbol{x}}_j) &= \frac{1}{h} \left\| \boldsymbol{x}_j - \tilde{\boldsymbol{x}}_j \right\|_2, \\ h &= \left\| \boldsymbol{x}_j(i_h) - \boldsymbol{x}_j(i_n) \right\|_2, \end{aligned} \tag{4.1}$$

where $\boldsymbol{x}_j, \tilde{\boldsymbol{x}}_j \in \mathbb{R}^2$ are the predicted and true 2D location of joint j, respectively, i_h is the index of the head joint and i_n is the index of the neck joint. If the predicted location is within a certain threshold n of the true location, the keypoint of joint j is classified as accurate (i.e. non-noisy, correct). The joint is likely to be not occluded. Otherwise, the keypoint is labeled as incorrect (i.e. noisy, incorrect). The joint is likely be occluded.

Like the data-driven method, the model-driven approach builds upon the existing baseline, with some architectural extensions (see Fig. 4.3):

Auxiliary Spatial Graph-Transformer (ASGT) The first module is an adapted version of the SGT module that includes an additional auxiliary branch after the last GraAttention and ChebGConv block. This branch comprises a feed-forward network with two linear layers, ReLU [Fuk75] activation and dropout [Sri+14]. While a ChebGConv layer reduces $\boldsymbol{P}_t^{N_0} \in \mathbb{R}^{J \times d_s}$ to a d_p-dimensional pose embedding $\tilde{\boldsymbol{P}}_t$, the auxiliary branch simultaneously estimates the noise. It takes $\boldsymbol{P}_t^{N_0}$ as input and outputs a feature embedding $\tilde{\boldsymbol{N}}_t \in \mathbb{R}^{J \times d_n}$ that corresponds to the encoded noise of each joint. Next, the embedded noise features $\tilde{\boldsymbol{N}}_t$ are concatenated with the embedded pose features $\tilde{\boldsymbol{P}}_t$. Eventually, the concatenated features $\boldsymbol{Y}_t \in \mathbb{R}^{J \times (d_p + d_n)}$ are reduced back to dimension d_p to produce to the final output $\tilde{\boldsymbol{Y}}_t \in \mathbb{R}^{J \times d_p}$ of the ASGT for frame t. As with the baseline, the outputs of all frames are flattened and concatenated before being passed to the next modules.

[1] Note, however, that noisy keypoints are not necessarily caused by occlusion. Other reasons include ambiguous appearances, truncation, or complex poses.

Classification Head A classification head consisting of one linear layer, batch normalization and Sigmoid activation is applied on the noise embedding from the ASGT to produce the final noise predictions. The MLP reduces the d_n-dimensional feature embedding $\tilde{\boldsymbol{N}}_t$ of each frame t to $\tilde{\boldsymbol{n}}_t \in \mathbb{R}^J$. $\boldsymbol{n}_t(j)$ represents the estimated probability that the keypoint of joint j in frame t is noisy.

The other components (TTM, STM, regression heads and pose refinement) remain unchanged and are identical to the modules of the same name in the baseline (see Fig. 4.2).

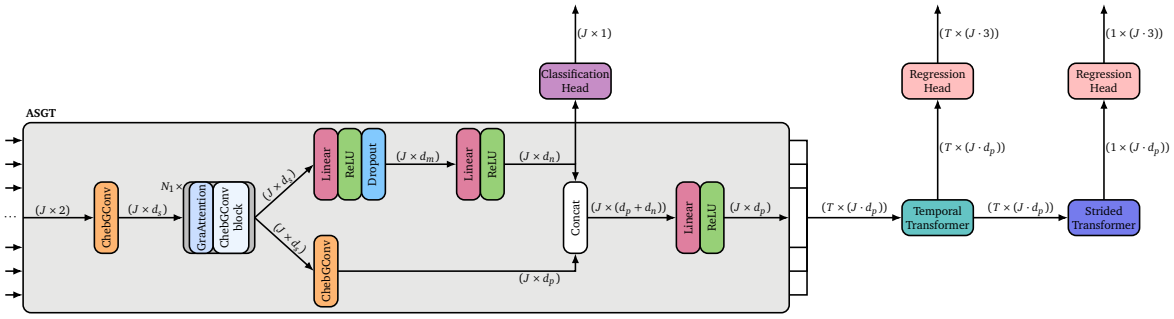


Figure 4.3: The architecture of the model-driven approach with a detailed view of the Auxiliary Spatial Graph-Transformer (ASGT) module.

Chapter 5

Experiments and Results

The following chapter presents and evaluates the experiments and results of the baseline, the data-driven approach and the model-driven approach individually. After general information on the experimental setup, individual details about the training process of a method are provided. Then, training results are presented and analyzed in detail, including an individual occlusion robustness analysis to test the effectiveness of each approach and a separate, comprehensive discussion to examine corresponding strengths and weaknesses. Lastly, all approaches are compared with each other and with the state-of-the-art, followed by a final discussion.

5.1 Experimental Setup

5.1.1 Dataset

The proposed methods will be evaluated on the Human3.6M [Ion+14] dataset, the largest publicly available and most commonly used benchmark for monocular 3D single-person HPE. The dataset contains 3.6 million accurate 3D human poses acquired by a marker-based motion capture system and synchronized with high-resolution videos from 4 different calibrated cameras at 50 Hz. 2D ground truth poses are implicitly provided through the camera calibration information. The movements include 15 everyday activities performed by 11 professional actors in an indoor environment and are captured

5.1.2 Implementation Details

The human poses are represented as a 17-joint skeleton, as depicted in Fig. 5.1, i.e. $J = 17$. The 2D input poses are in the image coordinate system and normalized to the range $[-1, +1]$ while maintaining the image aspect ratio. The 3D output poses are in the camera coordinate system to make the 2D-to-3D joint prediction task coherent across different camera views [BGK21; Zha+19b]. Furthermore, it is common to specify the 3D positions of the joints relative to a fixed global space with respect to the root joint. Therefore, the 3D poses are zero-centered by subtracting the predefined root, i.e. the pelvis joint.

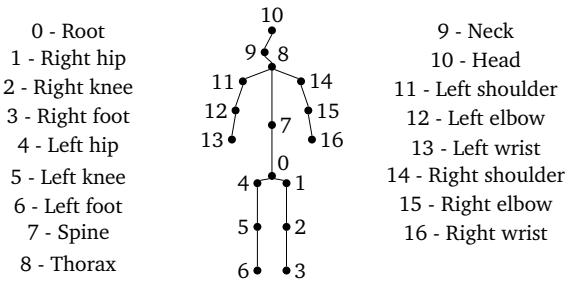


Figure 5.1: The 17 keypoints and their specification.

The input sequence length is set to $T = 81$ and contains estimated 2D keypoints from Cascaded Pyramid Networks (CPN) [Che+18] to make the proposed methods comparable to other approaches. The Spatial Graph-Transformer (SGT) contains $N_{SGT} = 2$ layers with embedding dimensions $d_s = 64$ and $d_p = 16$, as well as $h_1 = 4$ attention heads. The Temporal Transformer Module (TTM) consists of $N_{TTM} = 1$ modified Transformer encoder with strided factor $s = 1$. The Strided Transformer Module (STM) consists of $N_{STM} = 3$ STEs with strided factors $s_f = 1$ and $s_m = \{9, 3, 3\}$ resulting in shrinking output sequences of length $L_1 = 9$ vs. $L_2 = 3$ and $L_3 = 1$. Both the TTM and the STM use kernels of size $k_f = 1$ and $k_m = 3$ and $d_h = 512$ number of hidden units in the convolutional FFN, as well as $h_2 = 8$ attention heads. The dropout rate is set to 0.1 in all modules.
In the data-driven approach, joints are occluded with equal probability, i.e. P is a uniform distribution with $p_j = \frac{1}{J}$. The number of occluded joints per frame is set to $q_s = 1$. The subset size is set to $q_t = 6$. The maximum number of occluded frames is set to $q_F = \lfloor T/2 \rfloor = 40$. In the model-driven approach, the intermediate and final noise embedding dimensions are set to $d_m = 32$ and $d_n = 8$, respectively. The error threshold for creating artificial target labels is set to $n = 0.2$.
Following previous work [Liu+20b; Pav+19; Che+21a], five subjects (S1, S5, S6, S7, S8) are used for training and two subjects (S9, S11) are used for evaluation. All experiments are conducted in the PyTorch [Pas+19] framework.

5.1.3 Evaluation Details

Two standard protocols are used to evaluate the performance of the models:
The mean per joint position error (MPJPE) is the most widely used evaluation metric in 3D HPE. It is defined as the Euclidean distance between the true and estimated 3D pose averaged over all joints in one frame. Let J to denote the number of joints and $\boldsymbol{X}_f, \tilde{\boldsymbol{X}}_f \in \mathbb{R}^{J\times 3}$ to denote the true and estimated pose of frame f, respectively. The error at frame f is computed as:

$$\mathrm{E}_{\mathrm{MPJPE}}\left(\boldsymbol{X}_f, \tilde{\boldsymbol{X}}_f\right) = \frac{1}{J}\sum_{j=0}^{J-1} \left\| \boldsymbol{X}_f(j) - \tilde{\boldsymbol{X}}_f(j) \right\|_2 . \tag{5.1}$$

The poses are aligned at the root (pelvis) joints before calculating the error. MPJPE is given in millimeters and is referred to as protocol #1. The lower it is the better the performance.
The Procrustes Analysis MPJPE (P-MPJPE) is the MPJPE after rigidly aligning the predicted 3D pose to the ground truth by the Procrustes method [Gow75], i.e. using translation, rotation and scale. It is referred to as protocol #2.

To draw conclusions about the achieved performance, the learned adjacency matrices and attention maps of all modules are examined in detail. Different from previous work [Zhe+21; Zha+22], the attention output is averaged across all actions, since it is not defined when the subjects have to execute the current action (e.g. subject A may be sitting down at different time steps than subject B). Moreover, the original videos in which a subject performs an action are longer than the temporal receptive field of the models. Therefore, it does not make sense to visualize the attention maps exemplarily for only one subject and one action, because events of interest shift from input sequence to input sequence.

The binary classification task of noise prediction is evaluated based on accuracy (ACC), true

negative rate (TNR), precision and recall:

$$\begin{aligned} \mathrm{ACC}_t &= \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}, \quad \mathrm{TNR}_t = \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FN}}, \\ \mathrm{Precision}_t &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \quad \mathrm{Recall}_t = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \end{aligned} \tag{5.2}$$

where $t = 0.5$ is the classification threshold for mapping the predicted values to one of the two classes (i.e. accurate or inaccurate). TP, TN denote the number of true positives and true negatives, respectively, and FP, FN denote the number of false positives and false negatives, respectively. In addition, the average precision (AP) is taken into account:

$$\begin{aligned} \mathrm{AP} &= \sum_{i=0}^{|T|-1} T(i) \cdot \left(\mathrm{Recall}_{T(i)} - \mathrm{Recall}_{T(i+1)}\right) \cdot \mathrm{Precision}_{T(i)}, \\ \mathrm{Recall}_{T(|T|)} &= 0, \quad \mathrm{Precision}_{T(|T|)} = 1, \end{aligned} \tag{5.3}$$

where $T = \{0, 0.1, \ldots, 1.0\}$ is a list of thresholds and $T(i)$ is the threshold at index i.

5.1.4 Occlusion Robustness Analysis

Various experiments are conducted to test the models' occlusion robustness for different degrees of occlusion. In particular, occlusion robustness is tested for each joint individually.

Noisy Keypoints

To test the models' ability to handle noisy keypoints, it is analyzed whether the 3D reconstruction error for particularly noisy joints is significantly bigger compared to joints with less noise. The root joint is excluded from this analysis, as its 3D position is set to 0. The analysis includes a comparison of the average MPJPE per joint with noisy CPN [Che+18] detected keypoints, Mask R-CNN [He+17] detections and 2D ground truth as input. Note that Mask R-CNN detected keypoints are on average noisier than CPN detections. The comparison is made considering the 2D error per joint averaged over all actions (see Tab. 5.1). The error is measured as the head-normalized distance between 2D ground truth and predictions (see Eq. 4.1).

CPN [Che+18]	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	L shoulder	L elbow	L wrist	R shoulder	R elbow	R wrist	Avg.
Dist. [px]	0.1987	0.2654	0.2295	0.2756	0.3007	0.3013	0.3362	0.2248	0.2668	0.1928	0.1400	0.2470	0.3432	0.3334	0.2337	0.3291	0.3531	0.2689
PCKh@0.2 [%]	76.2	50.3	72.6	62.0	38.2	47.4	38.4	66.6	46.2	78.0	92.7	57.9	35.3	60.6	64.1	41.7	53.5	57.8
Mask R-CNN [He+17]	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	L shoulder	L elbow	L wrist	R shoulder	R elbow	R wrist	Avg.
Dist. [px]	0.2359	0.2964	0.2730	0.3466	0.3337	0.4090	0.4340	0.2586	0.2582	0.1964	0.1706	0.2836	0.3680	0.4486	0.2670	0.3566	0.4754	0.3183
PCKh@0.2 [%]	67.3	47.9	65.8	55.0	31.9	23.9	26.5	57.7	53.5	80.0	90.1	50.0	44.4	47.3	57.5	50.2	43.2	52.5

Table 5.1: Average head-normalized distance from 2D ground truth (see Eq. 4.1) and PCKh@0.2 per joint for CPN [Che+18] and Mask R-CNN [He+17] on Human3.6M [Ion+14].

Missing Keypoints

To test the models' ability to deal with missing keypoints, the impact of masking a joint on the models' performance is analyzed. Each joint is tested individually, i.e. $J = 17$ test runs are performed, masking one joint at a time for q_f frames per sequence. Frames are occluded in subsets consisting of q_t frames. The synthetic occlusion of the test set is deterministic, i.e. the same parameter specification always produces the same occluded test set to ensure comparability across different test cases.

5.2 Effectiveness of Exploiting Spatial-Temporal Relationships

The following section presents and evaluates the results of the baseline approach. The goal is to determine the advantages of exploiting spatial-temporal relations over purely spatial (e.g. GraFormer [Zha+21]) or purely temporal relations (e.g. StridedTransformer [Li+22a]). The evaluation includes an analysis of the training results as well as an individual occlusion robustness analysis that assesses the baseline's abilities to deal with noisy and missing keypoints.

5.2.1 Training Details

Strategy

The baseline model is trained on the original dataset without explicitly modeling occlusion, i.e. no occlusion augmentation is applied. The SGT, TTM and STM are trained prior to the pose refinement layers using transfer learning [PNS21] and fine-tuning. To this end, a small GraFormer [Zha+21] with a modified architecture is trained separately on the task of 3D HPE from a single image. This model has essentially the same architecture as the SGT with an additional regression head (see Fig. A.1). Training the baseline model is then divided into three phases: First, the pre-trained layers (without the regression head) are loaded into the baseline as the SGT to transfer the gained knowledge about 3D HPE from images to 3D HPE from videos. The remaining layers are trained while the pre-trained parameters (layers of the SGT) are frozen to speed up the learning process (transfer learning, phase 1). Second, the layer parameters of the SGT are unfrozen again and the module is jointly trained with the rest of the network to fine-tune the higher-order feature representations in the top layers (fine-tuning, phase 2). Third, the pose refinement module is added and the entire pipeline is trained in an end-to-end manner (pose refinement, phase 3).

Supervision

The modified GraFormer is supervised using the mean squared error (MSE):

$$\mathcal{L}_{MSE} = \frac{1}{m}\sum_{i=0}^{m-1} \|\boldsymbol{J}_i - \tilde{\boldsymbol{J}}_i\|_2^2, \tag{5.4}$$

where m denotes the total number of training samples and $\boldsymbol{J}_i, \tilde{\boldsymbol{J}}_i \in \mathbb{R}^{J\times 3}$ are the true and predicted 3D coordinates for sample i, respectively.
During transfer learning and fine-tuning, the baseline is supervised at both full sequence and single target frame scale using the intermediate and final results and the MPJPE as a loss function. The entire network is trained in an end-to-end manner with the total loss:

$$\begin{aligned}\mathcal{L} &= \frac{1}{m}\sum_{i=0}^{m-1}(\lambda_1\mathcal{L}_1 + \lambda_2\mathcal{L}_2) \\ &= \frac{1}{m}\sum_{i=0}^{m-1}\left[\lambda_1\frac{1}{T}\left(\sum_{t=0}^{T-1}\mathrm{E}_{MPJPE}\left(\boldsymbol{X}_i(t), \tilde{\boldsymbol{X}}_i(t)\right)\right) + \lambda_2\,\mathrm{E}_{MPJPE}\left(\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_i\right)\right],\end{aligned} \tag{5.5}$$

where $\mathcal{L}_1$, $\mathcal{L}_2$ denote the loss functions for supervision at full sequence and single target frame scale, respectively, and T denotes the number of input frames per sample. $\boldsymbol{X}_i, \tilde{\boldsymbol{X}}_i \in \mathbb{R}^{T\times(J\cdot 3)}$ are the true and predicted 3D pose sequence, respectively, and $\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_i \in \mathbb{R}^{1\times(J\cdot 3)}$ are the true and predicted 3D pose of the target frame, respectively.

When applying the pose refinement, the entire pipeline is supervised on the single target frame scale using:

$$\mathcal{L} = \frac{1}{m} \sum_{i=0}^{m-1} \mathrm{E}_{MPJPE}(\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_i^r), \tag{5.6}$$

with $\tilde{\boldsymbol{x}}_i^r \in \mathbb{R}^{1\times(J\cdot 3)}$ as the refined 3D pose prediction of the target frame.

Process

Pre-Training The modified GraFormer is trained for 47 epochs using the Adam optimizer and a minibatch size of $m' = 64$. The learning rate is initially set to 1×10^{-3} and multiplied by 0.9 every 50000 steps. Horizontal pose flipping as data augmentation is applied during both training and evaluation. The training is performed on a single NVIDIA GeForce RTX 3070 Laptop GPU.

Transfer Learning, Fine-Tuning and Pose Refinement The baseline model is trained for a total of 20 epochs using the AMSGrad optimizer and a minibatch size of $m = 256$. Transfer learning amounts to 7 epochs, fine-tuning to 6 epochs and pose refinement to 7 epochs. The learning rate is reset to 1×10^{-3} at the beginning of each training phase and exponentially decayed by a factor of 0.95 after each epoch and 0.5 after every fifth epoch. The weighting factors λ_1 and λ_2 are both set to 1. Horizontal pose flipping as data augmentation is applied during both training and evaluation. The training is conducted on a single NVIDIA GeForce RTX 3090 GPU.

5.2.2 Quantitative and Qualitative Analysis

Pre-Training

Pre-training the modified GraFormer yields a preliminary average MPJPE of 54.9 mm for single-frame 3D HPE.

Training and Validation

The further progression of performance during training the baseline model is visualized in detail in Fig 5.2. The figure shows the development of the loss and the MPJPE at a single target frame scale on the train and validation sets. The three phases transfer learning, fine-tuning and pose refinement are highlighted in different colors. At the beginning of the training process during transfer learning (phase 1), both loss and MPJPE drop very quickly, as expected. Before the performance starts to stagnate, the layer parameters of the spatial module are unfrozen to move into the fine-tuning phase (phase 2). It can be observed that this (in combination with resetting the learning rate) leads to a small increase in the loss at first, but in the long run allows to overcome local minima and to further reduce the loss. The MPJPE starts to fluctuate slightly from this stage onward, but decreases overall. Finally, loss and MPJPE are again significantly reduced by applying the pose refinement (phase 3). The baseline model achieves an average error of 45.0 mm under protocol #1 on the validation set as the final result, reducing the MPJPE of the modified GraFormer by approximately 10 mm through the use of temporal information. The average error under protocol #2 is 36.5 mm.

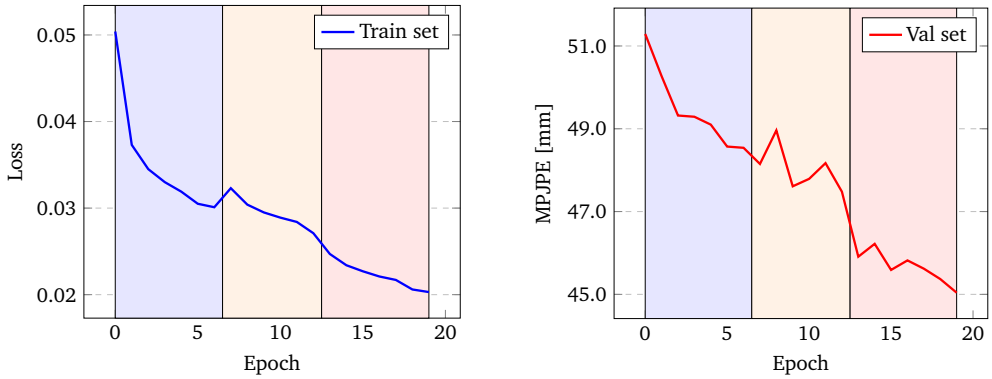


Figure 5.2: Loss and MPJPE progression during training of the baseline approach. The metrics are given only at the single target frame scale to visualize all three phases (from transfer learning and fine-tuning to pose refinement) in one graph. Background colors highlight the different phases.

Adjacency Matrix Visualization

Fig 5.3 shows the learned adjacency matrices of the two LAM-GConv layers in the SGT module. As expected, self-connections are the strongest in both layers. Besides, the first layer mainly aggregates information among neighboring joints of first- and second-order (sometimes also third-order), e.g.:

- The neck is closely related to the head (0.99) (first-order neighbor) as well as the spine (0.66) (second-order neighbors).
- The thorax is closely related to the shoulders (0.75, 0.9) (first-order neighbors) as well as the elbows (0.42, 0.49) (second-order neighbors). It is also weakly related to the wrists (0.25, 0.26) (third-order neighbors).
- The root is closely related to the hips (1.29, 1.27) (first-order neighbors) as well as the knees (0.47, 0.58) (second-order neighbors) and the neck (0.46) (third-order neighbor).

It is noticeable that the middle joints (root, spine, thorax, neck, head) often have stronger relations with the joints of the right arm than to their left counterparts. This can possibly be attributed to a larger number of right-handers in the dataset, as there are generally significantly more right-handers than left-handers.[1] Right-handers articulate predominantly with their right arm; actions s.a. talking on the phone, greeting and smoking are characterized by this limb and the rest of the body pose depends heavily on it. In contrast, the root has stronger relations with the joints of the left leg than to their right counterparts (knees: 0.58 vs. 0.47, feet: 0.15 vs. 0.0). This could be due to the fact that right-handedness often manifests itself not only in a dominant right arm, but also in a dominant right leg. In actions less characterized by strong foot activity (e.g. posing, smoking, waiting or when making a phone call), the dominant leg becomes the standing leg. Weight is shifted to the right leg, which is moved less as a result. The left leg, on the other hand, serves as a support, e.g. by being bent or stretched away, making it decisive for the (lower body) pose.
The second layer increases the radius of implicit relations between joints and additionally strengthens relations with more distant neighbors. It additionally captures implicit relationships of symmetric joints (vertically and horizontally):

- The right hip is closely related to the right shoulder (0.8) and the right elbow (0.41).

[1] Unfortunately, there is no information available about the actors' handedness. However, a manual review of the dataset reinforced the impression that most of the actors are right-handed.

- The left hip is closely related to the left shoulder (1.15) and the left elbow (0.63).
- The right knee is weakly related to the right shoulder (0.28).
- The left knee is weakly related to the left shoulder (0.33).

Interestingly, these relationships were established from joints of the lower limbs (legs) to joints of the upper limbs (arms). Conversely, the arms are only weakly related to the legs; the strongest relations are from the right elbow to the left hip (0.35) and from the left elbow to the right hip (0.22). Furthermore, the left shoulder is closely related to the right shoulder (0.45), but not the other way around (the right shoulder is barely related to the left shoulder, 0.01). This indicates that poses are mainly determined by the movement of (right) the upper body. It is further noticeable that the relations between joints of the left body parts are stronger than the relations between joints of the right body parts. Under the hypothesis that the majority of actors are right-handed, right arm movements predominate in many actions. This could make it more difficult for the right leg to establish particularly close relationships with the right arm, as its position can vary greatly. In contrast, the left arm is often moved less, making it easier for the left leg to establish a close relationship with it. The spine, thorax, neck and head establish strong connections among themselves. In turn, they also seem to focus on the left and right joints in a complementary manner, e.g. thorax and head are strongly related to the left shoulder, elbow, hip and foot, while spine and neck are strongly related to the right counterparts. It is further noticeable that many joints build strong implicit relations with the neck and the head (e.g. root, hips, elbows).

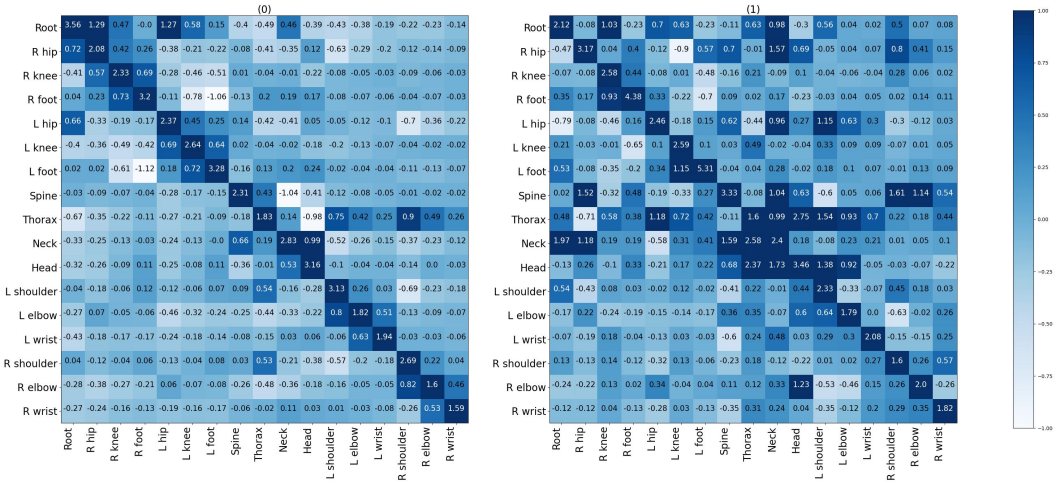


Figure 5.3: The learned adjacency matrices of the two LAM-GConv layers in the Spatial Graph-Transformer (SGT) module of the baseline approach. Dark blue indicates a strong relation.

Attention Visualization

Fig 5.4 visualizes the multi-head attention maps of the two MHSA layers in the Spatial Graph-Transformer (SGT). The first layer reveals strong implicit connections to the lower body limbs. Head 0 shows almost equal attention of all joints to each other, with the upper body focusing on the root and hip joints and the lower body focusing on the torso (thorax, neck, head, shoulders). Head 1 reveals strong symmetrical relations between left and right lower body joints, especially between the feet (0.84, 0.85) and the knees (0.48, 0.49). This intuitively makes sense, since the legs are often in a similar position even across different actions, e.g. greeting, smoking, waiting. In contrast, this observation cannot be made for the upper body

(elbows and wrists), since the position of one arm hardly gives any information about the position of the other arm (e.g. when talking on the phone, smoking, greeting). Heads 2 and 3 indicate strong implicit relations of all joints to the feet, with the latter additionally revealing relationships with hips, thorax, head and shoulders in a complementary manner.
The second layer further discloses strong implicit connections to the upper body excluding the upper limbs. Head 0 reveals strong implicit relations with head, neck and thorax along with symmetric relations between hips, knees and feet in a complementary manner. Heads 1 and 2 emphasize almost all joints equally, focusing on the lower and upper body connections, respectively, making the layers complement each other well. Head 3 shows strong connections to the feet, specifically strong symmetrical relationships between left and right feet (0.93, 0.98).

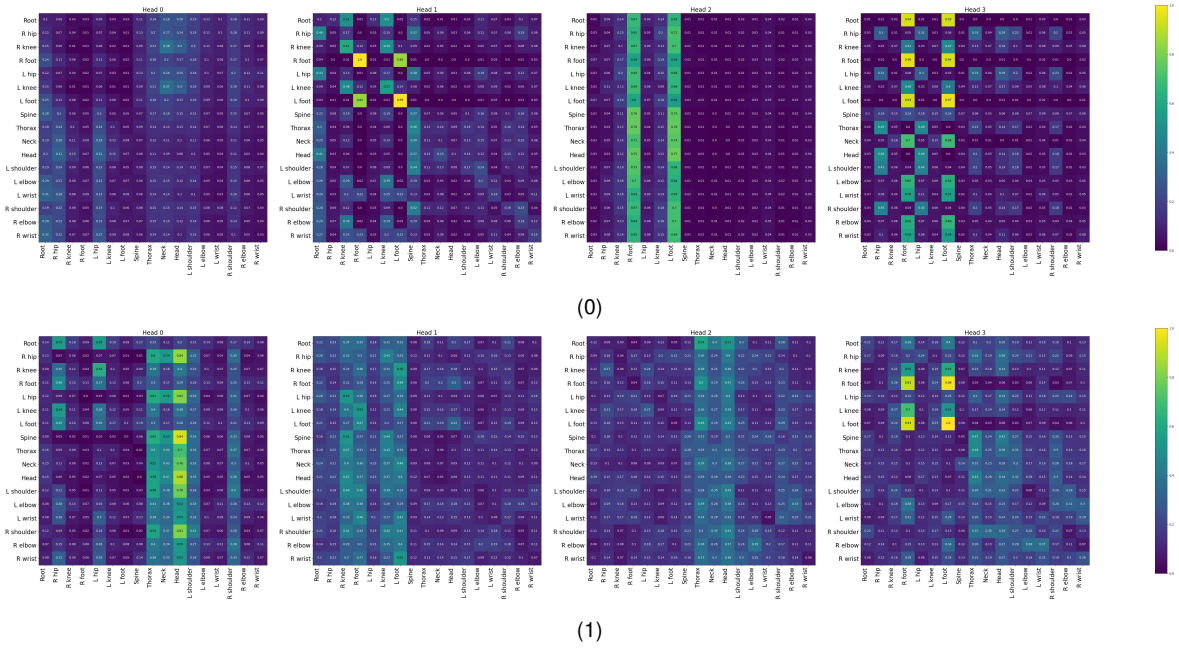


Figure 5.4: The multi-head attention maps ($h_1 = 4$) of the two MHSA layers in the Spatial Graph-Transformer (SGT) module of the baseline approach. The attention output is averaged across all actions and normalized from 0 to 1.

Fig 5.5 visualizes the multi-head attention maps of the MHSA layer in the Temporal Transformer Module (TTM). Supervision at full sequence scale enforces temporal consistency across all frames, i.e. coherent movements along preceding and subsequent frames. This is evident in heads 0, 3 and 4, which reveal strong relations between highly similar adjacent frames, creating diagonal attention highlights. There, each frame focuses on approximately 10 to 20 surrounding frames, with the number being larger for the first and last 20 to 30 frames than for the middle ones. The remaining heads show almost uniform relations across all frames, with increased attention for the middle frames (frames 30 to 50) from the frames 9/18/27, 54/63/72 and 36/40/45. Conversely, those query frames receive generally little attention resulting in a grid structured attention map. This lattice structure can also be seen in heads 0, 3, and 4, where the attention gaps are particularly visible. These highs and lows of attention at certain frames, which appear as rows and columns of decreasing width toward the middle of the sequence, are likely due to the following Strided Transformer Module (STM).
Fig 5.6 visualizes the multi-head attention maps of the first MHSA layer in the Strided Transformer Module (STM). Supervision at single target frame scale forces the module to selectively identify important frames that are close to the target frame (at the center). Thus, as

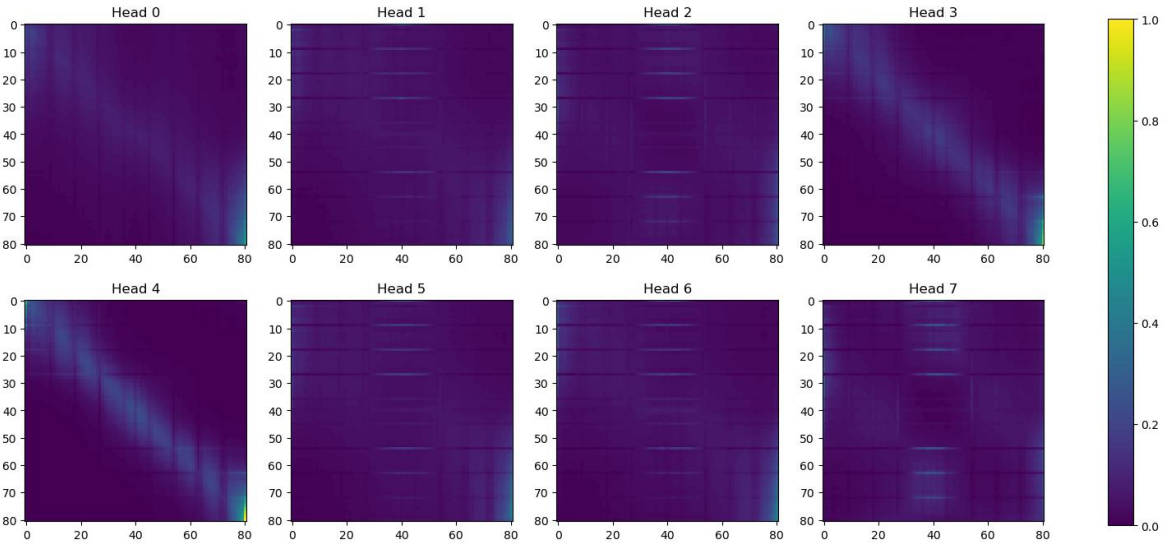


Figure 5.5: The multi-head attention maps ($h_2 = 8$) from the Temporal Transformer Module (TTM) of the baseline approach. The attention output is averaged across all actions and normalized from 0 to 1.

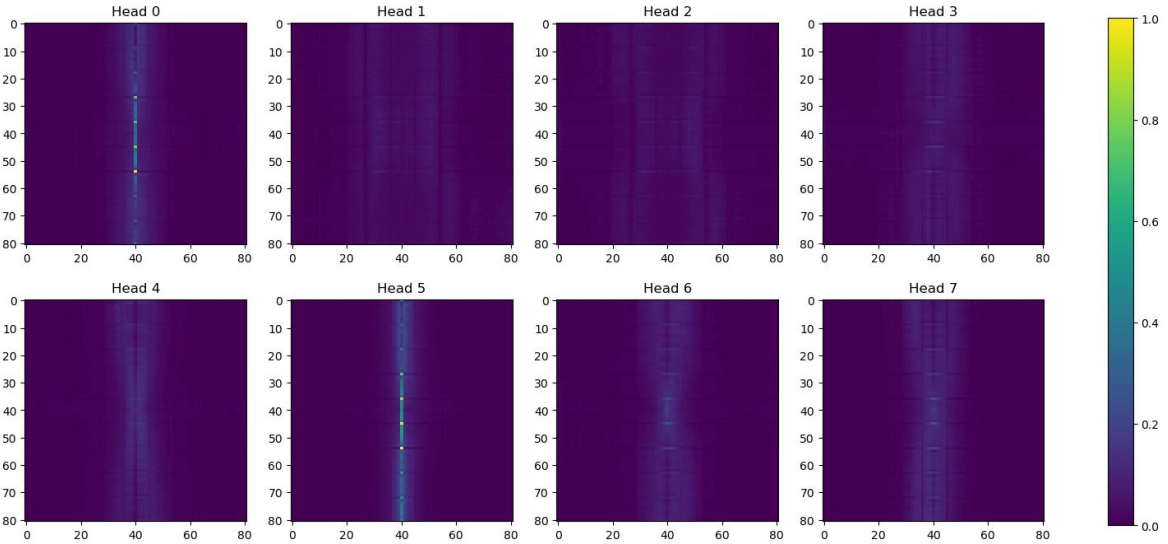


Figure 5.6: The multi-head attention maps ($h_2 = 8$) from the Strided Transformer Module (STM) of the baseline approach (first layer). The attention output is averaged across all actions and normalized from 0 to 1.

expected, the greatest focus is on the middle frames, with each head considering a different amount of frames around the center. Heads 0 and 5 reveal particularly strong relations to the frames at the immediate center (frames 37 to 43), while heads 3, 4, 6 and 7 extend the relationships to surrounding frames (frames 30 to 50) and heads 1 and 2 further increase the radius of attention (frames 20 to 60). The first and last 20 frames receive the least attention, as this is where the distance to the target frame is greatest. Overall, the middle frames near the target focus heavily on the center of the sequence, while more distant frames are less emphasized as little information is needed here from far preceding or subsequent frames. In contrast, the frames at the beginning and end of the sequence have a larger receptive field that includes frames slightly farther from the center. This has already been observed in the attention maps of the TTM. In the case of the STM, an hourglass pattern emerges. Interestingly, the first and last 20 to 35 frames do not focus directly on the center, but rather on the surrounding frames. This is indicated by a darker column in the middle of the attention

maps, which becomes lighter again toward the center, especially visible in heads 0, 4, 6 and 7. In addition, a grid structure is evident in all heads, as before with highs and lows of attention at the frames 9/18/27, 54/63/72 and 36/40/45. In heads 0 and 5, the peaks stand out as conspicuous dots in the said frames. The total number of peaks (9) corresponds to the length of the sequence after shrinking through the first STM layer. This makes clear that the lattice structure is caused by shrinking the sequence. The effect is transferred to the TTM through backpropagation, which results in the pattern appearing at the same frames in the TTM and STM.

5.2.3 Occlusion Robustness Analysis

The motivation for analyzing the occlusion robustness of the baseline approach is to obtain results that are comparable to the occlusion robustness results of the other models presented (see Sec. 4.2 and 4.3). To this end, the baseline's capabilities to deal with noisy as well as missing keypoints are analyzed.

Noisy Keypoints

Fig 5.7 compares the average MPJPE per joint with noisy CPN [Che+18] detected keypoints, Mask R-CNN [He+17] detections and 2D ground truth (GT) as input. In general, the 2D error of particularly mobile joints such as wrists, elbows and feet is the largest for all detectors. Correspondingly, this is where the 3D error is the greatest as well. By contrast, the 3D error of more static joints such as hip joints and the spine is the smallest error. The model gives on average the best result for true 2D coordinates as input (39.8 mm). However, given the noise of CPN detections, it would have been expected that the model achieves an even lower error. Most negatively, the model produces a lower error for the 3D head position when using the noisy CPN keypoint prediction as input rather than the true 2D coordinates (41.1 mm vs. 42.8 mm). This reveals a slight over-fitting of the head keypoints detected by CPN, which explains the comparatively high MPJPE for GT data. A better generalizability to unseen data from different 2D detectors is necessary.
The average 3D pose estimation results of the model are worse for Mask R-CNN detections (54.7 mm) than for less noisy CPN detections (45.0 mm). The performance of the model deteriorates by more than 20 % for Mask R-CNN detections compared to CPN detections, even though Mask R-CNN detects only about 5 % less correct keypoints compared to CPN (57.8 % vs. 52.5 %). However, the underlying error of the 2D detections is almost 20 % lower for CPN than for Mask R-CNN (0.2689 px vs. 0.3183 px). Accordingly, the performance drop from 45.0 mm to 54.7 mm is still within reasonable limits.

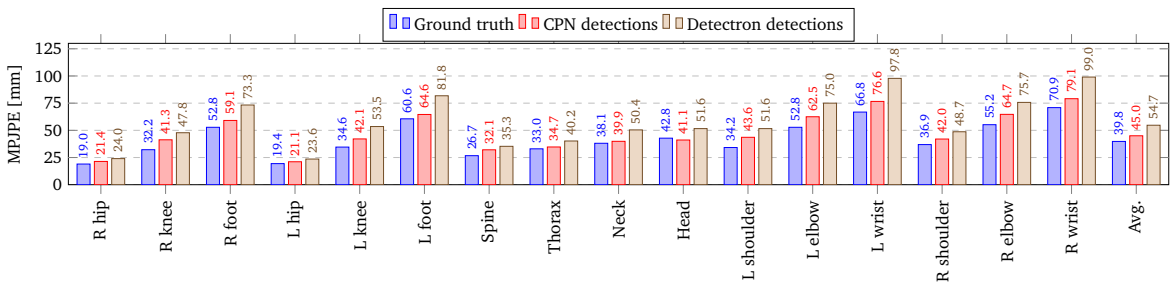


Figure 5.7: Average MPJPE comparison per joint of the baseline approach on Human3.6M [Ion+14].

Examining each joint individually reveals the following insights: In general, it can be observed that the MPJPE of the joints in the right arm (elbow, wrist) is greater than the MPJPE of the joints in the left arm. This is consistent with the corresponding 2D errors (see Tab. 5.1). Under the hypothesis that the majority of actors in the dataset are right-handed, it is more difficult to correctly localize the joints of the right arm because they are moved a lot. This leads to a higher 2D error and consequently to a higher 3D error. In contrast, the MPJPE of the joints in the right leg (knee, foot) is smaller than the MPJPE of the joints in the left leg. The same applies to the corresponding 2D errors. This may also be due to more actors in the dataset being right-handed, since right-handedness often also manifests itself in a dominant right leg, as mentioned before (see Sec. 5.2.2: Adjacency Matrix Visualization). During actions s.a. posing and waiting, weight is shifted to the right leg, which is thus moved less, making it is easier to locate. The left leg, on the other hand, is bent, stretched away, or similar and therefore more difficult to localize correctly.
Furthermore, the estimation results for different 2D detectors deteriorate roughly in line with the input, i.e. if a 2D keypoint is worse by a certain percentage, its estimated 3D location is also worse by approximately the same percentage. Consider the right hip joint: The estimated 2D keypoint of Mask R-CNN is on average 11.7 % noisier than that of CPN (0.2964 px vs. 0.2654 px). The 3D pose estimation deteriorates similarly by 12.1 % (24.0 mm vs. 21.4 mm). Note that this is not always directly transferable, as all joints are related, so the 2D error of other joints affect the 3D pose estimation of the target joint accordingly. Nonetheless, the model seems to be especially robust against noise at spine, knees and wrists:

Right Knee The estimated 2D keypoint of Mask R-CNN is on average 19.0 % noisier than that of CPN (0.2295 px vs. 0.2730 px). The 3D pose estimation deteriorates by 15.7 % (41.3 mm vs. 47.8 mm).

Left Knee The estimated 2D keypoint of Mask R-CNN is on average 35.7 % noisier than that of CPN (0.3013 px vs. 0.4090 px). The 3D pose estimation deteriorates by 27.1 % (42.1 mm vs. 53.5 mm).

Spine The estimated 2D keypoint of Mask R-CNN is on average 15.0 % noisier than that of CPN (0.2248 px vs. 0.2586 px). The 3D pose estimation deteriorates by 10.0 % (32.1 mm vs. 35.3 mm).

Left Wrist The estimated 2D keypoint of Mask R-CNN is on average 34.6 % noisier than that of CPN (0.3334 px vs. 0.4486 px). The 3D pose estimation deteriorates by 27.7 % (76.6 mm vs. 97.8 mm).

Right Wrist The estimated 2D keypoint of Mask R-CNN is on average 34.6 % noisier than that of CPN (0.3531 px vs. 0.4754 px). The 3D pose estimation deteriorates by 25.2 % (79.1 mm vs. 99.0 mm).

In contrast, the model appears to be particularly sensitive to noise at the neck and elbows:

Neck The estimated 2D keypoint of Mask R-CNN is on average 1.9 % noisier than that of CPN (0.1928 px vs. 0.1964 px). The 3D pose estimation deteriorates by 26.3 % (39.9 mm vs. 50.4 mm).

Left Elbow The estimated 2D keypoint of Mask R-CNN is on average 7.2 % noisier than that of CPN (0.3432 px vs. 0.3680 px). The 3D pose estimation deteriorates by 20.0 % (62.5 mm vs. 75.0 mm).

Right Elbow The estimated 2D keypoint of Mask R-CNN is on average 8.4 % noisier than that of CPN (0.3291 px vs. 0.3566 px). The 3D pose estimation deteriorates by 17.0 % (64.7 mm vs. 75.7 mm).

While the model may be able to derive the positions of spine, knees and wrists from related joints, it may have difficulties doing the same for neck and elbows. Some of these observations are consistent with the relationships identified in the previous analysis of the adjacency matrices and attention maps. For example, spine, wrists and knees are strongly connected to neighboring joints and symmetrical counterparts, which may allow the model to better handle noisy inputs, while the elbows do not have a particularly striking spatial relationship with other joints, making it difficult to deal with the noise. The neck joint, however, is closely related to the head and thorax joint. One would think that these relationships would enable the model to better cope with noisy neck joints. However, looking at the thorax joint reveals that the estimated 2D keypoint of Mask R-CNN is on average 3.3 % less noisy than that of CPN (0.2668 px vs. 0.2582 px). Still, the 3D pose estimation deteriorates by 15.9 % (34.7 mm vs. 40.2 mm). This could be due to the particularly strong relationships with the neighboring shoulders, whose 2D error is significantly larger for Mask R-CNN than for CPN (approx. 15 %), consequently affecting the 3D estimation results of the thorax and the neck as well.

Missing Keypoints

Tab. 5.2 reports the baseline's performance under test-time occlusion by keypoint masking. Recall that the model was trained using the original dataset, i.e. it has never seen artificially occluded data. Therefore, a very simple masking strategy is adopted with a subset size of $q_t = 1$ and the number of occluded frames set to $q_f = 30$. This results in 37 % of the frames to be affected by occlusion, thus joints that are occluded in one frame are very likely to also be occluded in a directly adjacent frame. However, since only one joint per affected frame is masked, only a maximum of $\frac{q_f}{T \cdot J} = \frac{30}{81 \cdot 17} \approx 2.2\,\%$ of all input joints is occluded. Nevertheless, the baseline's performance deteriorates on average by a factor of two (ΔMPJPE = 45.3 mm). A particularly high increase in the error occurs when masking the root joint. This is especially evident when considering the average MPJPE without the root, which at 82.0 mm is 8.3 mm lower than the average MPJPE considering all joints (90.3 mm), possibly due to the relative representation of each joint to the root. The error also increases significantly when masking the neck or head joint. This could be attributed to the strong relationships with these joints, as observed in the previous analysis of the adjacency matrices and attention maps. Similar observations can be made for the thorax. Here, however, the error increase may not be as high, because a missing thorax joint can be compensated by strong relations with the neighboring shoulders. Neck and head, in contrast, are autonomous joints whose relationships are not so easily compensated. In contrast, masked feet or knees may be compensated by the strong connections to their symmetrical counterparts. In other cases (hips, spine, shoulders, elbows), a similar error increase ($\approx$ 40 mm) could be due to the fact that none of the close spatial relationships recorded is particularly useful in this type of occlusion. However, this does not explain the small error increase for wrists, since they are not closely connected with their symmetrical counterparts. Considering the previous analysis (see Fig. 5.7), it is noticeable that the error increase with masking of joints is particularly low ($<$ 25 mm) when the general 3D error per joint is relatively high ($>$ 70 mm), e.g. for the wrists and feet. Note that masking a joint leads primarily to problems with localization of the masked joint (and thus has an effect on the localization of neighboring joints). It may be that the error of these joints increases only slightly proportionally in their absence. Since the model already has problems

	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	L shoulder	L elbow	L wrist	R shoulder	R elbow	R wrist	Avg.
MPJPE [mm]	223.3	85.4	66.4	62.0	85.4	67.5	60.7	85.8	91.4	115.3	108.6	86.9	83.1	68.8	87.4	87.2	69.5	90.3
ΔMPJPE [mm]	178.3	40.4	21.4	17.0	40.4	22.5	15.7	40.8	46.4	70.3	63.6	41.9	38.1	23.8	42.4	42.2	24.5	45.3

Table 5.2: Average MPJPE and error increase of the baseline approach for occlusion of a given joint across $q_f = 30$ random frames on Human3.6M [Ion+14]. The subset size q_t is set to 1.

locating these joints correctly when using the original data, their absence does not affect the overall error as much as that of other joints.
Furthermore, it can be observed that performance usually deteriorates more under occlusion of right limb joints than under occlusion of corresponding left limb joints. This is consistent with the observation that stronger spatial relationships were often established with joints of the right part of the body. The absence of information on which the model relies more heavily has a greater negative impact on performance.

5.2.4 Discussion

Overall, the baseline model is able to exploit spatial and temporal information effectively, achieving satisfactory results on the test set. The ability of the self-attention mechanism to capture long-range dependencies was demonstrated on both spatial and temporal scales. In particular, the model was shown to be able to capture spatial relationships that extend beyond neighboring second- and third-order nodes as well as global temporal relationships across distant frames. At the same time, close relationships were also established with directly adjacent joints or frames, showing that local dependencies play a role in both spatial and temporal contexts and should not be neglected. Indeed, the attention maps of the STM (see Fig. 5.6) reveal that the first and last 20 frames receive little attention, suggesting that a smaller temporal receptive field might also suffice. However, to confirm this, further experiments would have to be performed with input sequences of different sizes, which is beyond the scope of this thesis. Most of the implicit spatial relationships established are intuitively explainable, demonstrating the usefulness of considering geometric information provided by the human skeleton.
However, the occlusion robustness analysis did not reveal any profound ability to handle noisy or missing keypoints particularly well. Generally, the average performance under occlusion of any kind needs improvement. Even more, the model needs to generalize better to unseen data from different 2D detectors. Spatial dependencies have been shown to affect the per-joint error both positively and negatively, depending on the underlying 2D error of closely related joints. The large performance difference between masking different joints also show how much the model relies on the learned spatial relationships. While this again demonstrates the model's ability to comprehensively capture spatial relationships, it also indicates that the model does not make optimal use of temporal relationships.

5.3 Effectiveness of Occlusion-based Data Augmentation

The following section presents and evaluates the results of the data-driven approach. The goal is to determine the benefits of occlusion-based data augmentation during training compared to training on the original data without explicitly modeling occlusion. The evaluation includes an analysis of the training results as well as an individual occlusion robustness analysis that assesses the model's ability to handle missing keypoints compared to the baseline method.

5.3.1 Training Details

Strategy

The same training strategy is used as for the baseline, i.e. the modules SGT, TTM and STM are trained prior to the pose refinement layers using transfer learning and fine-tuning. In

addition, parts of the input are masked by setting the coordinates of the affected keypoints to 0 to simulate occlusion. The synthetic occlusion extends over up to 6 consecutive frames. The goal is to compensate for the missing keypoints by exploiting temporal features. To this end, the same pre-trained model that was trained on the task of 3D HPE from a single frame without occlusion-augmented data is used for transfer learning. Then, training the main model takes place in three steps: First, the learned parameters of the SGT are frozen while the remaining layers are trained on the occlusion-augmented data (phase 1). Second, the layer parameters of the SGT are unfrozen again and fine-tuned on the occlusion-augmented data (phase 2). Third, the pose refinement module is added (phase 3).

Supervision

There are no architectural changes to the baseline model in this approach, thus the same loss functions from Eq. 5.5 and 5.6 are used for supervision.

Process

Pre-Training This step is omitted because the same pre-trained model from the baseline is used.

Transfer Learning, Fine-Tuning and Pose Refinement Each of these training phases is performed under the same setup as in the baseline. The model is trained for a total of 19 epochs, with 6 epochs each for transfer learning and fine-tuning and 7 epochs for pose refinement.

5.3.2 Quantitative and Qualitative Analysis

Training and Validation

Fig 5.8 shows the development of the loss and the MPJPE at a single target frame scale during training on the train and validation sets. The three phases transfer learning, fine-tuning and pose refinement are highlighted in different colors. Note that the loss is calculated on the occlusion-augmented train set. However, the MPJPE is calculated on the unaltered validation set, since the goal of occlusion-based data augmentation is to improve the model's

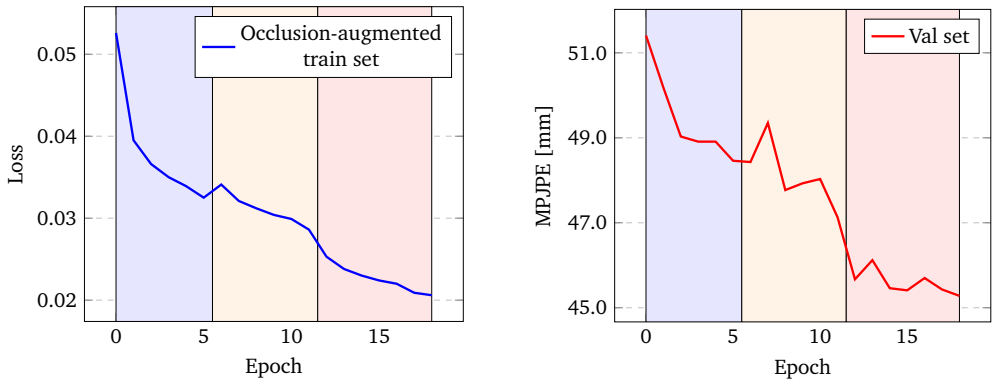


Figure 5.8: Loss and MPJPE progression during training of the data-driven approach. Note that the loss is calculated on the occlusion-augmented train set. The metrics are given only at the single target frame scale to visualize all three phases (from transfer learning and fine-tuning to pose refinement) in one graph. Background colors highlight the different phases.

occlusion robustness on the original data. Generally, it can be observed that the model is able to generalize reasonably well from occlusion-augmented training data to original (i.e. not occlusion-augmented) test data, with some performance fluctuations largely due to resetting learning rates as different training phases transition. The model achieves an average MPJPE of 45.3 mm on the validation set as the final result. The average error under protocol #2 is 36.4 mm. Thus, the model can keep up quite well with the performance of the baseline (MPJPE: 45.0 mm, P-MPJPE: 36.5 mm).

Adjacency Matrix Visualization

Fig 5.9 shows the learned adjacency matrices of the two LAM-GConv layers in the Spatial Graph-Transformer (SGT). In both layers, generally similar connections were established between the joints as in the baseline (see Fig. 5.3), with emphasis on self-connections in the first layer and more distant neighbors as well as symmetrical counterparts in the second layer. The direction of the relationships usually coincides (positive or negative), only the strength of the relationships differs slightly. This can be attributed to the fact that the same pretrained model was used for transfer learning in both approaches. Nevertheless, there are a few striking differences regarding the strength of certain connections. In the first layer of the data-driven approach, it is noticeable that almost all self-connections are stronger compared to the baseline, especially those of the thorax (from 1.83 to 2.11) and the neck (from 2.83 to 3.05). In contrast, the strength of relationships with first- and second-order neighbors tends to decrease. This affects, for example, the connections from neck to spine (from 0.66 to 0.54), from root to neck (from 0.46 to 0.35) and from shoulders to thorax (left: from 0.54 to 0.41, right: from 0.53 to 0.43) (also vice versa from thorax to shoulders). This overall development makes intuitively sense, as related nodes could be masked, forcing joints to rely more on their own features. Note, however, that the relationships of the root to both legs increase slightly, as well as the relationship of the head to the neck (from 0.53 to 0.61). This could be because the hip joints are in close proximity to the root, close to 0, making their occlusion less significant, which in turn affects the knees and feet as well. In the case of the head, there is also the fact that the neck is its only direct neighbor, making it dependent on this relationship. The neck, on the other hand, can rely on the shoulders and spine instead. In the second layer, the following developments, among others, are emerging: For the thorax, there is a tendency for its relationships to related joints to increase in strength (e.g. left

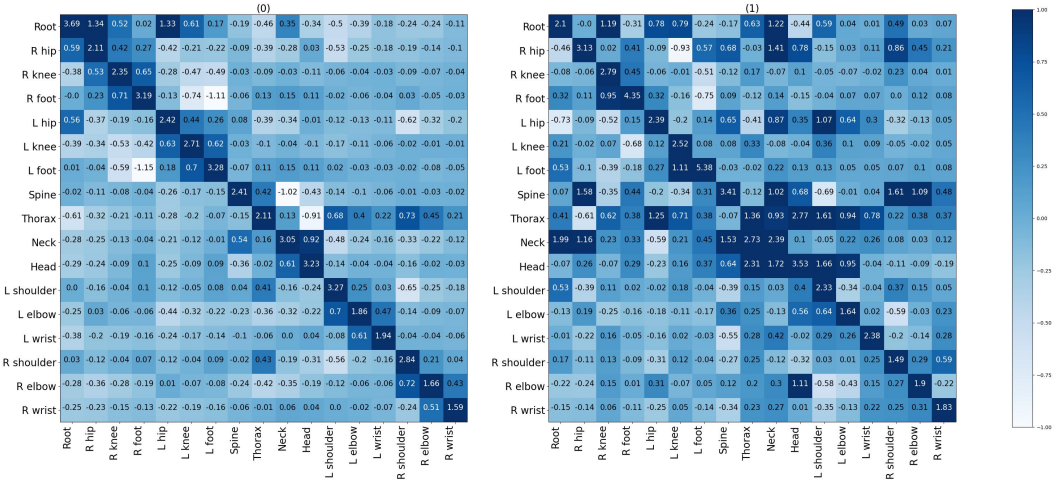


Figure 5.9: The learned adjacency matrices of the two LAM-GConv layers in the Spatial Graph-Transformer (SGT) module of the data-driven approach. Dark blue indicates a strong relation.

shoulder: from 1.54 to 1.61, right elbow: from 0.18 to 0.38), while the one to itself decreases in strength (from 1.6 to 1.36). The relationships of the right leg to the right arm are stronger (e.g. hip - shoulder: from 0.8 to 0.86), while the relationships of the left leg to the left arm are weaker (e.g. hip - shoulder: from 1.15 to 1.07). At the same time, the root tends to further strengthen its already emphasized relationships (e.g. neck: from 0.98 to 1.22, right knee: 1.03 to 1.19, left knee: from 0.63 to 0.79). However, no precise pattern can be discerned as to which relationships are affected by which trend (amplification or attenuation).

Attention Visualization

Fig 5.10 visualizes the multi-head attention maps of the two MHSA layers in the Spatial Graph-Transformer (SGT). Again, the attention maps strongly resemble those of the baseline (see Fig. 5.4), as the same pre-trained model was used for transfer learning in both approaches. There are no significant deviations from the baseline with respect to the nature of the relationships. However, as with the adjacency matrices, there are a few differences in the strength of certain relationships, but this time mainly in the second rather than the first layer. Here, too, it can be observed that attention to other joints tends to decrease. In head 0, this mainly affects the head, but also the thorax, the neck and even the hips. In head 1, the trend is most evident at the feet and knees. In head 2, the thorax and neck are again particularly affected. This development is also visible in the first layer, but not to this extent. For example, in head 1 at the root or in head 3 at the hips and feet. At the same time, attention to the knees increases in both heads. Overall, the model is quite capable of maintaining and capturing meaningful spatial relations despite artificial occlusion of the data. This argues against the claim of Shan et al. [Sha+22] that confusion with the root joint could occur if the joints are masked by setting their keypoint coordinates to 0.

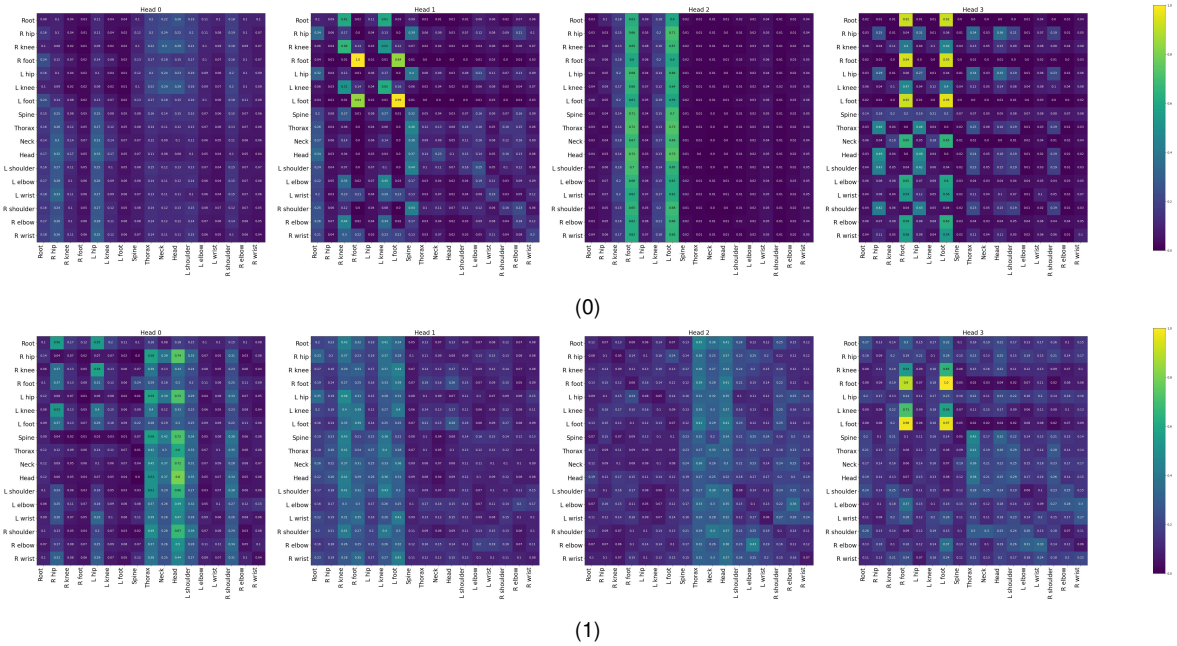


Figure 5.10: The multi-head attention maps ($h_1 = 4$) of the two MHSA layers in the Spatial Graph-Transformer (SGT) module of the data-driven approach. The attention output is averaged across all actions and normalized from 0 to 1.

Fig 5.11 visualizes the multi-head attention maps of the MHSA layer in the Temporal Transformer Module (TTM). The strength of the emphasized relations is generally much higher than in the baseline, showing how much the model relies on temporal relationships. Typical

for supervision at full sequence scale, diagonal attention highlights can be seen, this time not only in two (see head 3 and 4 in Fig. 5.5) but in 4 heads (heads 1, 3, 4 and 5), albeit with varying degrees of intensity. Toward the middle of the sequence, the number of surrounding frames in focus decreases (as in the baseline, while attention strength increases, which is especially visible in heads 3 and 4. Note that the diagonal attention highlights of these two heads are also curved toward the center rather than straight. This means that each frame no longer focuses symmetrically on surrounding frames, but tends more toward the target frame. For frames in the first half of the sequence, subsequent frames receive more attention; for frames in the second half of the sequence, preceding frames receive more attention. In head 4, for example, frame 70 no longer focuses on the frames 60 to 80, but on 55 to 75. Frame 20, on the other hand, no longer focuses on frames 10 to 30, but on 15 to 35. In the middle of the sequence, attention is again distributed symmetrically. Furthermore, heads 0 and 7 reveal particularly interesting relationships. Here, the focus is not on directly adjacent frames, but on the frames surrounding the directly adjacent ones at a distance of about 15 to 30 frames, creating two diagonals around the main diagonal. In head 7, for example, frame 70 no longer focuses on 55 to 75 (as in head 4), but on 35 to 45 and 75 to 81. Frame 20, on the other hand, no longer focuses on 15 to 35, but on 0 to 10 and 40 to 50. Thus, the heads work complementarily to the others, efficiently exploiting the temporal context. Note that this pattern is already slightly emerging in head 7 of the baseline. In heads 0, 1, 2, 6, and 7, the frames 9/18/27 and 54/63/72 particularly emphasize the middle frames (frames 30 to 50). Conversely, these query frames receive generally (i.e. in almost all heads, from all frames) little attention creating the familiar grid-like pattern of attention. It is striking that this lattice structure is hardly visible in head 5, despite recognizable highlighting of adjacent frames. The attention map is much smoother. The occlusion augmentation might force the model to pay equal attention to the entire sequence, as key-frames could be occluded.

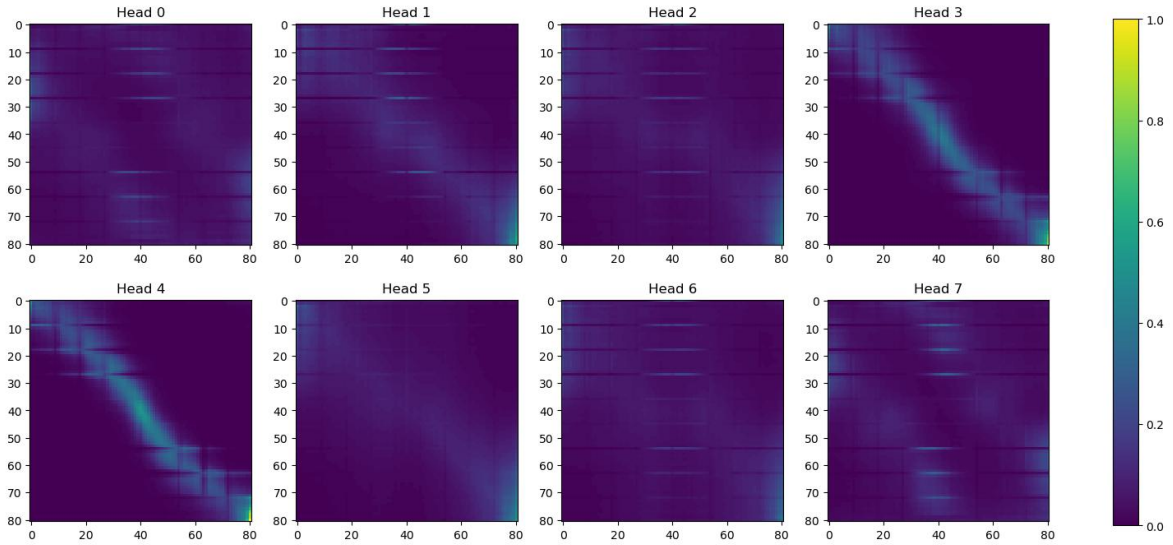


Figure 5.11: The multi-head attention maps ($h_2 = 8$) from the Temporal Transformer Module (TTM) of the data-driven approach. The attention output is averaged across all actions and normalized from 0 to 1.

Fig 5.12 visualizes the multi-head attention maps of the first MHSA layer in the Strided Transformer Module (STM). Again, it is noticeable that important relationships are much more emphasized than in the baseline. Attention is mainly focused on the middle frames (30 to 50), similar to the baseline, as can be seen especially for head 0, 1, 4 and 5. However, the attention maps do not show the typical hourglass pattern that is clearly evident in almost all heads in the baseline. In addition, heads 2 and 7 show that the model makes greater

use of more distant frames than the baseline. Generally, each head captures the temporal relationships in a different way. The attention maps of heads 4 and 5 show especially interesting dependencies not seen before: While the first 40 frames focus mainly on the frames 40 to 50, just after the target frame, the last 40 frames focus on the frames 30 to 40, just before the target frame. This effect is also emerging in head 1 and is slightly reminiscent of heads 0 and 7 of the TTM. However, unlike usual, the attention strength of the query frames in the middle part of the sequence is lower and more widely distributed. The target frame focuses on surrounding frames rather than on itself, resulting in a visual interruption in the yellowed colored mark that visualizes strong attention. Head 0 does not show this interruption, but even here the frames do not focus evenly on the middle part of the sequence. Instead, the first and last 20 frames tend to focus on frames 25 to 45 (especially 33 to 40), while the frames 20 to 60 tend to focus on frames 30 to 50 (especially 38 to 43). Attention is focused more on the adjacent frames to the left of the target. No relationship between two particular frames is clearly emphasized, which otherwise stands out as a bright point in the attention map. Nevertheless, the relationships between frames 9/18/27, 54/63/72, 36/45 and the middle part of the sequence are particularly striking. As in the TTM, these query frames receive little attention, again creating a grid-structured attention map, particularly visible in heads 2, 3, 6, and 7. This lattice structure is barely visible in head 4, similar to the attention map of head 5 of the TTM, but not to the same extent. Furthermore, heads 3 and 6 mainly capture relationships to the first half of the sequence, while heads 2 and 7 mainly capture relationships to the second half of the sequence, with the attention of the middle frames in all four heads tending toward the center. The attention maps of heads 2 and 7 look like reflections of the attention maps of head 3 and 6. This shows once again that the model aims to better capture the temporal context through complementary interaction of the heads in order to overcome occlusion. In addition, the ability of the self-attention to capture long-range temporal relationships is demonstrated particularly well.

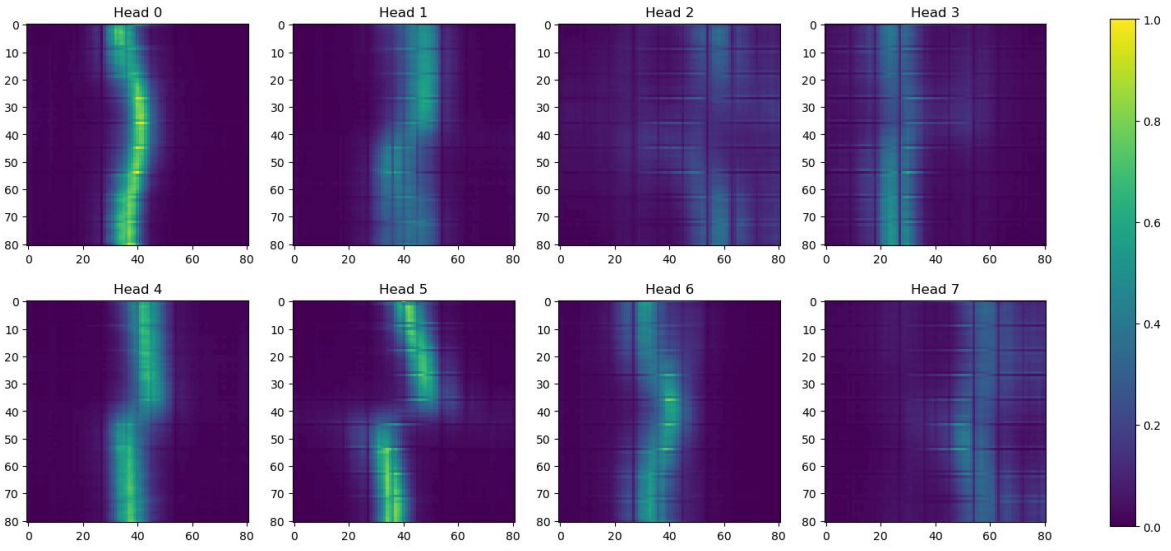


Figure 5.12: The multi-head attention maps ($h_2 = 8$) from the Strided Transformer Module (STM) of the data-driven approach (first layer). The attention output is averaged across all actions and normalized from 0 to 1.

5.3.3 Occlusion Robustness Analysis

The motivation of this approach is to develop a model that can maintain the baseline's performance while being more robust to incomplete keypoint detections. To this end, the model's capability to deal with missing keypoints is analyzed.

Missing Keypoints

Tab. 5.3 reports the model's performance under test-time occlusion by keypoint masking for different degrees of occlusion. In the different test cases, the severity of occlusion increases with the number of frames per subset, i.e. with increasing duration of occlusion. The model shows extreme improvement over the baseline in terms of occlusion robustness to missing keypoints. While the baseline's performance already deteriorates on average by a factor of two (ΔMPJPE $= 45.3\,$mm) under mild occlusion ($q_t = 1$ and $q_f = 30$, see Tab. 5.2), the data-driven model is able to maintain its performance considerably well with little degradation (ΔMPJPE $= 0.7$mm, Tab. 5.3b). Recall that the model was trained using occlusion-

	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	L shoulder	L elbow	L wrist	R shoulder	R elbow	R wrist	Avg.
MPJPE [mm]	47.6	45.5	46.7	45.9	45.4	46.4	45.9	45.6	45.3	45.8	45.6	45.6	45.9	46.2	45.6	45.8	46.4	46.0
ΔMPJPE [mm]	2.3	0.2	1.4	0.6	0.1	1.1	0.6	0.3	0.0	0.4	0.3	0.3	0.6	0.9	0.3	0.5	1.1	0.7

(a) The subset size q_t is set to 1. The number of occluded frames q_f is set to 30.

	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	L shoulder	L elbow	L wrist	R shoulder	R elbow	R wrist	Avg.
MPJPE [mm]	47.7	45.6	46.9	46.3	45.6	46.6	46.3	45.8	45.5	46.0	45.8	45.8	46.1	46.5	45.8	46.0	46.7	46.2
ΔMPJPE [mm]	2.4	0.3	1.6	1.0	0.3	1.3	1.0	0.5	0.2	0.7	0.5	0.5	0.8	1.2	0.5	0.7	1.4	0.9

(b) The subset size q_t is set to 6. The number of occluded frames q_f is set to 30.

	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	L shoulder	L elbow	L wrist	R shoulder	R elbow	R wrist	Avg.
MPJPE [mm]	47.4	45.6	46.7	46.2	45.6	46.4	46.2	45.7	45.5	45.9	45.7	45.7	46.0	46.4	45.7	46.0	46.5	46.1
ΔMPJPE [mm]	2.1	0.3	1.4	0.9	0.3	1.1	0.9	0.4	0.2	0.6	0.4	0.4	0.7	1.1	0.4	0.7	1.2	0.8

(c) The subset size q_t is set to 8. The number of occluded frames q_f is set to 24.

	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	L shoulder	L elbow	L wrist	R shoulder	R elbow	R wrist	Avg.
MPJPE [mm]	48.2	45.9	47.5	47.0	45.9	47.2	47.0	46.0	45.8	46.3	46.0	46.0	46.4	47.1	46.0	46.4	47.3	46.6
ΔMPJPE [mm]	2.9	0.6	2.2	1.7	0.6	1.9	1.7	0.7	0.5	1.0	0.7	0.7	1.1	1.8	0.7	1.1	2.0	1.3

(d) The subset size q_t is set to 10. The number of occluded frames q_f is set to 30.

	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	L shoulder	L elbow	L wrist	R shoulder	R elbow	R wrist	Avg.
MPJPE [mm]	47.9	45.9	47.2	46.9	45.8	46.9	46.9	46.0	45.8	46.2	45.9	46.0	46.3	46.9	46.0	46.3	47.0	46.5
ΔMPJPE [mm]	2.6	0.6	1.9	1.6	0.5	1.6	1.6	0.7	0.5	0.9	0.6	0.7	1.0	1.6	0.7	1.0	1.7	1.2

(e) The subset size q_t is set to 12. The number of occluded frames q_f is set to 24.

	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	L shoulder	L elbow	L wrist	R shoulder	R elbow	R wrist	Avg.
MPJPE [mm]	49.0	46.4	48.4	48.3	46.3	48.1	48.3	46.4	46.2	46.9	46.4	46.4	47.0	47.9	46.5	47.0	48.2	47.3
ΔMPJPE [mm]	3.7	1.1	3.1	3.0	1.0	2.8	3.0	1.1	0.9	1.6	1.1	1.1	1.7	2.6	1.2	1.7	2.9	2.0

(f) The subset size q_t is set to 15. The number of occluded frames q_f is set to 30.

	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	L shoulder	L elbow	L wrist	R shoulder	R elbow	R wrist	Avg.
MPJPE [mm]	52.0	48.4	51.9	53.3	48.2	51.6	53.1	48.0	48.0	49.2	48.1	48.1	49.3	50.7	48.3	49.3	51.2	49.9
ΔMPJPE [mm]	6.7	3.1	6.6	8.0	2.9	6.3	7.8	2.7	2.7	3.9	2.8	2.8	4.0	5.4	3.0	4.0	5.9	4.6

(g) The subset size q_t is set to 30. The number of occluded frames q_f is set to 30.

Table 5.3: Average MPJPE and error increase of the data-driven approach at different degrees of occlusion of a given joint on Human3.6M [Ion+14]. During training, occlusion augmentation was applied with $q_t = 6$ and $p_j = \frac{1}{J}$. The model achieves an average MPJPE of $45.3\,$mm on the original test set.

augmented data, in which frames are occluded in subsets consisting of $q_t = 6$ consecutive frames, i.e. the model was already exposed to occlusion of a certain degree. However, the average MPJPE increase is within an acceptable range (ΔMPJPE ≤ 2.0 mm) even when the model is exposed to more severe occlusion that it has not yet experienced (Tab. 5.3d – 5.3f). Longer periods of occlusion ($q_t = 30$), however, do lead to significant drops in performance, but the average MPJPE still remains below 50 mm. The greatest improvement over the baseline is seen in the occlusion of the root. While the occlusion of the root causes by far the largest error increase in the baseline, it barely affects the performance of the data-driven model in comparison (ΔMPJPE $= 178.3$ mm vs. ΔMPJPE $= 2.4$ mm for $q_t = 6$ and $q_f = 30$). Nevertheless, the root remains the joint that, if occluded, causes the greatest drop in performance, along with knees, feet, elbows and wrists (i.e. the limb joints). The error increase when occluding these joints is up to 3.7 mm for moderate occlusion (Tab. 5.3d – 5.3f) and 8.0 mm for heavy occlusion (Tab. 5.3g). Even under mild occlusion, there is a drop in performance (albeit not quite as severe) (Tab. 5.3a). Localization is particularly difficult in the case of (long-term) occlusion because the position of these joints changes considerably across frames, i.e. the joints move a lot. In contrast, shoulders, hips, spine, thorax and head cause on average the smallest drop in performance. The error increase when occluding these joints is at most 1.6 mm for moderate occlusion and 3.9 mm for heavy occlusion. For these joints, the position in adjacent frames does not change to the same extent as for limb joints, allowing localization to some degree in the case of long-term occlusion by exploiting temporal relations. Only the absence of the neck joint cannot be compensated in this way, although it also belongs to the joints with little movement. However, there are many joints (e.g. head, thorax, spine, hips and root) that have a particularly close spatial relationship to the neck, such that its occlusion has a strong impact on performance despite little movement.
Despite similar spatial relations, the subdivision of joints into severe vs. mild performance drops when occluded differs greatly from that of the baseline. There applies: Joints that strongly affect the performance are root, neck and head; joints that (comparatively) hardly affect the performance are feet, knees, elbows, wrists and shoulders. These differences illustrate the close interaction of spatial and temporal dependencies in the data-driven approach. Recall that stronger spatial relationships were more often established with joints of the right part of the body than with their corresponding left counterparts. Accordingly, it can be observed that occlusion of the right joints generally has a greater impact on performance than occlusion of the corresponding left joints (up to 0.5 mm worse). As in the baseline, the absence of information on which the model relies more heavily has a greater negative impact on performance (although the extent is not comparable to that of the baseline due to increased occlusion robustness).

5.3.4 Ablation Studies

In the following, the effects of different occlusion augmentation strategies are investigated, more specifically different subset sizes q_t and different occlusion probabilities p_j per joint. In addition to the variant presented so far, three other alternatives were trained. To this end, an advanced joint weighting procedure is designed by dividing the joints into 6 different groups: Joints that, if occluded, have a greater impact on the performance of the baseline are prioritized higher. The occlusion probability of joints with higher prior-

Grouping and Weights:
1. Root (11.9)
2. Head, Neck (4.7)
3. R/L shoulder, R/L elbow, Thorax (3.1)
4. R/L hip, Spine (2.6)
5. R/L knee, R/L wrist (1.6)
6. R/L foot (1)

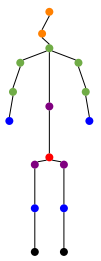


Figure 5.13: The grouping of joints based on their impact on the baseline's performance under occlusion (see Tab. 5.2).

ity is increased so that they are more likely to be selected during the joint-occlusion procedure. This is in line with the assumption that training the model more frequently on hard cases increases occlusion robustness and the overall performance. The joints are grouped based on the occlusion robustness results for the baseline shown in Tab. 5.2. The elaborated grouping with the corresponding weights is shown in Fig. 5.13. Using the PyTorch [Pas+19] framework, a probability distribution P is formed based on the elaborated weighting, which is then used to select the joint to be occluded. The effect of non-uniform occlusion is compared with its uniform variant, where $p_j = \frac{1}{J}$. It is further examined how the occlusion of subsets consisting of multiple consecutive frames compares to the occlusion of a single frame, i.e. when the subset size q_t is set to 1.

Tab. 5.4 reports the performance without test-time occlusion for each possible configuration. To ensure comparability, the models were trained for the same number of epochs in all cases. In general, the performance difference between the individual configurations is very small. The best results are obtained using the standard uniform weighting of joints in combination with occluding subsets of consecutive frames (row 3). Contrary to the assumptions, occluding joints with non-uniform probabilities leads to worse performance. The model achieves a 0.2 mm lower average MPJPE (row 4), indicating that non-uniform occlusion augmentation is ineffective. However, the elaborated grouping and / or weighting could also be not appropriate. Adjusting the weights (increase / decrease the occlusion probability of individual joints) or changing the overall prioritization could improve the effect of non-uniform occlusion augmentation. One problem with the current grouping could be that critical joints whose 3D errors are already among the highest (e.g. feet, wrists) are among the least frequently occluded. Conversely, they occur most frequently in the train set. Rather than improving the model's localization capabilities for critical joints by partially occluding other joints, localization of critical joints is only made more difficult. An alternative grouping could instead be based on the MPJPE per joint (see Tab. 5.7). However, further experiments would have to be conducted to test this hypothesis, which is beyond the scope of this thesis.

Consecutive Subsets	Non-Uniform Joint Probability	MPJPE [mm]
✗	✗	45.4
✗	✓	45.4
✓	✗	45.3
✓	✓	45.5

Table 5.4: Ablation study on different occlusion augmentation strategies. The evaluation is performed on Human3.6M [Ion+14].

When masking frames individually, the difference in performance between the occlusion probabilities (uniform vs. non-uniform; row 1 and 2) is < 0.1 mm, i.e. negligible. This may be attributed to the fact that Human3.6M [Ion+14] runs at a frame rate of 50 Hz, so different occlusion probabilities hardly matter when masking a single image. Considering only the results under uniform occlusion probability (row 1 and 3), the model's performance is 0.1 mm better when occluding subsets of consecutive frames in contrast to occluding single frames. While this is not a large difference per se, masking subsets shows a larger positive effect on robustness against heavy occlusion (compare Tab. 5.3g and Tab. A.1d in the appendix). This is a clear advantage, since occlusion usually occurs over longer periods of time rather than in few independent frames. However, it should be noted that even training under occlusion of single frames significantly improves the model's robustness to missing keypoints compared to the baseline (ΔMPJPE $= 1.4$ mm vs. ΔMPJPE $= 45.3$mm for $q_t = 6$ and $q_f = 30$).

5.3.5 Discussion

Like the baseline, the data-driven model is able to achieve satisfactory performance on the test dataset through the efficient use of spatial and temporal information, showing that train-

ing under synthetic occlusion does not affect performance on the original data. The model captures spatial relations in almost the same way as the baseline, whereas significant differences are observed in the use of temporal relations. This indicates that the model indeed compensates for the missing data by exploiting temporal features. In particular, the model captures temporal relationships at different levels (heads) in a complementary manner, as shown by the different attention maps of the STE (see Fig. 5.12). Thus, it compensates for missing keypoints in one part of the sequence by focusing on another part of the sequence. Accordingly, there is no drop in performance if no artificial occlusion is present. In this way, the model is able to maintain the baseline's performance on the original data. The model is further able to capture particularly long-ranging relationships, indicating a much more efficient use of the temporal receptive field than in the baseline. Even the distant first and last 20 frames are included in the estimation of the target pose (see Fig. 5.12), which illustrates the capabilities of the self-attention mechanism particularly well. Missing keypoints near the center are compensated by considering more distant frames. Overall, the model shows extreme improvements over the baseline in terms of robustness to missing keypoints and is able to maintain its performance even under heavy occlusion. This demonstrates the effectiveness of occlusion-augmented training. However, an ablation study of the occlusion augmentation strategy during training revealed that occlusion of multiple consecutive images is crucial for occlusion robustness. At the same time, it was not found to play a critical role in this ability whether the joints are uniformly occluded or based on their impact on the baseline's performance when occluded.

5.4 Effectiveness of Input Noise Estimation

The following section presents and evaluates the results of the model-driven approach. The focus is on whether the prediction of noise can be learned and, if so, whether the model can use the noise estimates to improve pose estimation. The evaluation includes an analysis of the training results as well as an individual occlusion robustness analysis that assesses the model's ability to handle noisy keypoints compared to the baseline method.

5.4.1 Training Details

Strategy

The model is trained on the original dataset without explicitly modeling occlusion, i.e. no occlusion augmentation is applied. In addition to estimating the 3D pose, the model attempts to identify noisy keypoints in the input. The targets for supervising the noise predictions are generated using the head-normalized distance between 2D keypoints and 2D ground truth (GT) (see Eq. 4.1). Keypoints with a distance to the 2D GT > 0.2 px are marked as inaccurate / noisy. Similar to the baseline, the modules ASGT, TTM and STM are trained prior to the pose refinement layers using transfer learning and fine-tuning, i.e. training takes place in three steps. To this end, the modified GraFormer used in the baseline is further extended with an auxiliary branch and a classification head to identify noisy keypoints in addition to estimating the 3D pose of a frame (see Fig. A.2). The auxiliary architecture thus corresponds to that of the ASGT (excluding the classification and regression heads). The auxiliary model itself is also trained by transfer learning and fine-tuning using the pre-trained layers of the modified GraFormer. The pre-trained and new layers are trained with different learning rates α_p and α_n, respectively, where $\alpha_n \gg \alpha_p$ to reduce the effects on pose estimation. All layer parameters (without the classification and regression heads) are then loaded into the main

model as part of the ASGT. Note that the linear layer that further transforms the concatenated pose and noise features is not part of the auxiliary model. This layer has to be trained from scratch in the next stage along side the remaining layers of the TTM and STM, while the pre-trained layer parameters are frozen (phase 1). Afterwards, all layer parameters are unfrozen again to fine-tune the entire network (phase 2). Finally, the entire pipeline including the pose refinement module is trained (phase 2).

Supervision

Noise estimations are supervised using binary cross entropy (BCE) [Goo56] with the artificial noise labels as targets:

$$\mathrm{BCE}(\boldsymbol{n}, \tilde{\boldsymbol{n}}) = -\frac{1}{J}\sum_{j=0}^{J-1} \boldsymbol{n}(j)\log(\tilde{\boldsymbol{n}}(j)) + (1-\boldsymbol{n}(j))\log(1-\tilde{\boldsymbol{n}}(j)), \tag{5.7}$$

where J denotes the number of joints and $\boldsymbol{n}, \tilde{\boldsymbol{n}} \in \mathbb{R}^J$ denote the true and estimated noise of the joints, respectively. Positive examples are weighted with w_{pos} to penalize false positives (i.e. keypoints incorrectly marked as accurate / non-noisy) and to account for unbalanced classes. The complete loss function of the auxiliary model is given by:

$$\mathcal{L} = \lambda_p \mathcal{L}_{MSE} + \lambda_n \frac{1}{m}\sum_{i=0}^{m-1} \mathrm{BCE}((\boldsymbol{n}_i, \tilde{\boldsymbol{n}}_i)), \tag{5.8}$$

where m denotes the total number of training samples and λ_p, λ_n are weighting factors. The total loss for training the entire network during transfer learning and fine-tuning is formulated as:

$$\begin{aligned}\mathcal{L} &= \lambda_0\mathcal{L}_0 + \lambda_1\mathcal{L}_1 + \lambda_2\mathcal{L}_2 \\ &= \lambda_0 \frac{1}{T}\left(\sum_{t=0}^{T-1} \mathrm{BCE}(\boldsymbol{n}_t, \tilde{\boldsymbol{n}}_t)\right) + \lambda_1\mathcal{L}_1 + \lambda_2\mathcal{L}_2,\end{aligned} \tag{5.9}$$

where $\mathcal{L}_0$ is the loss function for supervising the noise predictions and $\mathcal{L}_1$, $\mathcal{L}_2$ are the loss functions for supervision at full sequence and single target frame scale, respectively. T denotes the number of input frames per sample and λ_0 is another weighting factor. Pose refinement is done in the same way as in the baseline, therefore the loss function for supervision the entire pipeline remains unchanged (see Eq. 5.6).

Process

Pre-Training The auxiliary model is trained for 22 epochs with initial learning rates $\alpha_p = 10^{-6}$ and $\alpha_n = 10^{-3}$. w_{pos} is set to $\frac{N_1}{N_1+N_2} \approx 0.25$, where N_1, N_2 are the number of noisy and correct keypoints and in the training set, respectively. The weighting factors are set to $\lambda_p = 10$ and $\lambda_n = 1$. Optimizer, minibatch size, learning rate decay and device are the same as in the pre-training phase of the baseline.

Transfer Learning, Fine-Tuning and Pose Refinement Each of these training phases is performed under the same setup as in the baseline. The model is trained for a total of 16 epochs, where transfer learning amounts to 5 epochs, fine-tuning to 6 epochs and pose refinement to 5 epochs. The weighting factors λ_1 and λ_2 are both set to 10. The weighting factor λ_0 is set to 1. w_{pos} is set to $\frac{N_1}{N_1+N_2} \approx 0.25$ as in the pre-training of the auxiliary model.

5.4.2 Quantitative and Qualitative Analysis

Pre-Training

Pre-training the auxiliary model yields a preliminary average MPJPE of 55.5 mm for single-frame 3D HPE. On the task of noise prediction, the auxiliary model achieves on average an accuracy of 62.79 % with an average precision (AP) of 74.57 % and a true negative rate (TNR) of 64.88 %. Tab. 5.5 reports the accuracy of the noise estimation by head-normalized distance from the 2D ground truth (GT) (see Eq. 4.1). For noisy keypoints (distance from 2D GT > 0.2 px) applies: The larger the distance to the 2D GT, the easier it is for the model to identify noisy keypoints as such, i.e. the accuracy of the noise estimation increases. In contrast, the smaller the distance to the 2D GT, the harder it is for the model to distinguish noisy keypoints from correct ones, i.e. the accuracy of the noise estimation decreases. This intuitively makes sense and confirms that the model is able to extract information about the noise. Nevertheless, the average accuracy of 62.79 % is quite low, mainly due to the lack of ability to classify accurate keypoints (distance from 2D GT < 0.2 px) correctly. The model particularly struggles with keypoints that are just on the borderline of being classified as noisy (distance from 2D GT $\in [0.1, 0.2)$ px). Here the accuracy is only at 49.97 %. Basically, the model just randomly guesses whether a keypoint is noisy or not, with equal chances for both outcomes. The accuracy increases for keypoints that are clearly non-noisy (distance from 2D GT $\in [0.0, 0.1)$ px), but still remains low (62.77 %). Note, however, that the main goal for the task of noise prediction is to identify inaccurate keypoints as such, i.e. we aim for a high specificity / TNR. False positives (keypoints incorrectly classified as accurate) should be minimized. This also means that false negatives (keypoints incorrectly classified as inaccurate) are more likely to be tolerated. For this reason, the value of w_{pos} is kept low (≈ 0.25) to penalize false positives more than false negatives. A further challenge, and also another reason for the penalty, is the imbalance of accurate (non-noisy) and inaccurate (noisy) keypoints in the dataset, as well as a different distribution of noise in the train and test set. For PCKh@0.2, 75.19 % of the keypoints in the train set are labeled as correct, while 24.81 % of the keypoints are labeled as incorrect. In the test set, 57.69 % are correct and 42.31 % are incorrect. This makes it difficult to identify accurate and inaccurate keypoints alike.

Distance [px]	$[0.0, 0.1)$	$[0.1, 0.2)$	$[0.2, 0.3)$	$[0.3, 0.4)$	$[0.4, 0.5)$	$[0.5, 0.6)$	$[0.6, 0.7)$	$[0.7, 0.8)$	$[0.8, 0.9)$	$[0.9, 1.0)$	≥ 1.0
Accuracy [%]	62.77	49.97	62.38	75.66	87.04	94.99	97.60	98.29	97.48	95.66	94.22

Table 5.5: Average accuracy of the auxiliary model for predicting input noise on Human3.6M [Ion+14] by head-normalized distance from 2D ground truth (GT) (see Eq. 4.1). Keypoints with a distance to the 2D GT < 0.2 px are classified as accurate / non-noisy.

Training and Validation

Fig 5.14 shows the further progression of the loss and the MPJPE at a single target frame scale during training on the train and validation sets. The three phases transfer learning, fine-tuning and pose refinement are highlighted in different colors. Again, some performance fluctuations are evident over the course of training, but the MPJPE decreases overall. Thus, no significant differences from the learning curves of the baseline and data-driven approach are apparent. However, with an average MPJPE of 45.2 mm on the validation set as the final result, the model achieves a similar performance to the baseline (45.0 mm) and data-driven methods (45.3 mm), while being trained 4 and 3 epochs less, respectively. This suggests that the extracted features related to input noise may have helped to find meaningful spatial relationships and thus accelerate training. The average error under protocol #2 is 36.6 mm.

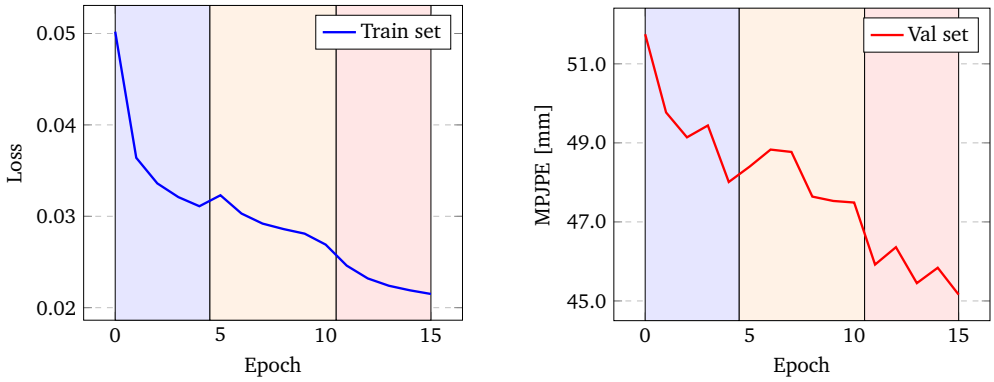


Figure 5.14: Loss and MPJPE progression during training of the model-driven approach. The metrics are given only at the single target frame scale to visualize all three phases (from transfer learning and fine-tuning to pose refinement) in one graph. Background colors highlight the different phases.

On the task of noise prediction, the model achieves on average an accuracy of 63.78 % with an AP of 76.20 % and a TNR of 41.91 %. Thus, the model was able to slightly improve the accuracy (ΔACC = 0.99 %) and AP (ΔAP = 1.63 %) of the noise estimates, but in doing so, the TNR decreases by 22.97 %. This means noisy keypoints are detected as such less frequently, despite the high penalty for false positives due to the low value of w_{pos}, but accurate keypoints are detected more frequently. Tab. 5.6 reports the accuracy of the noise estimation by head-normalized distance from the 2D GT, as for the pre-trained auxiliary model. As indicated by ACC, AP and TNR, the model was able to slightly improve the classification of accurate (non-noisy) keypoints compared to the auxiliary model. The accuracy increases by 3.77 % for keypoints with a distance from 2D GT $\in [0.0, 0.1)$ px and by 5.49 % for keypoints with a distance from 2D GT $\in [0.1, 0.2)$ px. However, this comes at the price of a lower accuracy for the classification of inaccurate (noisy) keypoints, which is reflected above all in the reduced TNR. The accuracy decreases especially for keypoints that are just on the borderline of being classified as noisy (distance from 2D GT $\in [0.2, 0.3)$ px), namely by 4.06 %. Note, however, that the model's main difficulty remains the classification of keypoints with a distance from 2D $\in [0.1, 0.2)$ px, i.e. accuracy is lowest there, indicating that identifying noisy keypoints continues to have priority over identifying non-noisy keypoints. In contrast, keypoints with a distance from 2D GT $\in [0.4, 0.7)$ px are also identified with an average of about 2 % lower accuracy. However, for particularly noisy keypoints with a distance from 2D GT ≥ 0.7 px, the accuracy of the auxiliary model could either be maintained or improved. Noise detection thus benefits from the incorporation of temporal relationships, as clear cases of accurate or inaccurate keypoints can be better classified. Less clear cases, on the other hand, can hardly be classified. Thus, the effects of the constraints imposed by the dataset are still highly noticeable.

Distance [px]	[0.0, 0.1)	[0.1, 0.2)	[0.2, 0.3)	[0.3, 0.4)	[0.4, 0.5)	[0.5, 0.6)	[0.6, 0.7)	[0.7, 0.8)	[0.8, 0.9)	[0.9, 1.0)	≥ 1.0
Accuracy [%]	66.54	55.46	58.32	73.93	85.19	92.01	95.92	97.53	98.18	97.74	96.12

Table 5.6: Average accuracy of the model-driven approach for predicting input noise on Human3.6M [Ion+14] by head-normalized distance from 2D ground truth (GT) (see Eq. 4.1). Keypoints with a distance to the 2D GT < 0.2 px are classified as accurate / non-noisy.

Adjacency Matrix Visualization

Fig. 5.15 visualizes the learned adjacency matrices of the two LAM-GConv layers in the Auxiliary Spatial Graph-Transformer (ASGT). As with the data-driven approach, the connections

established strongly resemble those of the baseline (see Fig. 5.3). This is because the pre-trained auxiliary model is based on the modified GraFormer model used for transfer learning in the baseline. Due to the low learning rate α_p and the relatively high weighting factor λ_p, the pre-trained weights change little by the parallel noise estimation. Nevertheless, there are a few differences regarding the strength of certain connections, mainly involving the thorax: In the first layer, the connection strength of the thorax to the left and right arms is much more balanced. There is no longer a tendency toward the right arm, especially the shoulder. This could be due to the fact that the average 2D detections of left and right arm joints are not significantly different from each other (see Tab. 5.1). On average, the joints of the right arm are detected only approximately 2 % better than those of the left arm. In addition, the 2D detections of the thorax are relatively poor compared to those of the shoulders and wrists, with the average PCKh@0.2 being approximately 15.0 % and 11.0 % worse, respectively. Moreover, while the self-connection of nearly all joints are close to the same as in the baseline (±0.07), that of the thorax is slightly stronger (+0.18). This may be caused by many of the other joints reducing the strength of their connection with the thorax, as seen in the subsequent MHSA layers of the spatial module (see Fig. 5.16), which are analyzed in detail in Sec. 5.4.2. In the second layer, the relationships between head, spine and neck increases in strength, while their relationships with the thorax decreases in strength (e.g. head - neck: from 1.73 to 2.21, head - thorax: from 2.37 to 2.19). In contrast, the thorax weakens its connection to head (from 2.75 to 2.59) and neck (from 0.99 to 0.83) (its connection to the spine remains weak). This may be attributed to the fact that the detected keypoints of head, neck and spine are significantly better than those of the thorax, with the PCKh@0.2 being 20 to 45 percent higher. Accordingly, the relationships of these joints to the arms also change: The thorax strengthens its connection to the left shoulder (from 1.54 to 1.91) and the elbows (left: from 0.91 to 1.01, right: from 0.18 to 0.38). In contrast, the connection between head and left elbow weakens (from 0.92 to 0.77), as does the connection between spine and right elbow (from 1.14 to 0.96). Furthermore, the connection between right hip and right shoulder is weaker (from 0.8 to 0.67), while the connection between left hip and left shoulder remains strong (1.14). The tendency for relationships between joints of the left half of the body to be stronger than relationships between joints of the right half of the body continues to strengthen. This could be due to the fact that the 2D keypoints of the right half of the body, in particular those of the right leg, are more accurate than those of the left half of the body. Thus, the joints on the right weaken their relationships to certain neighbors to

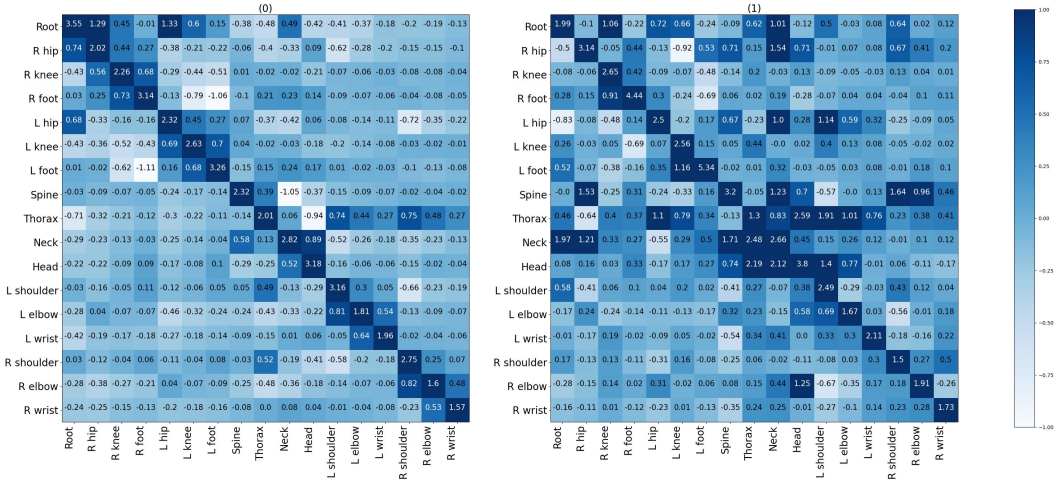


Figure 5.15: The learned adjacency matrices of the two LAM-GConv layers in the Auxiliary Spatial Graph-Transformer (ASGT) module of the model-driven approach. Dark blue indicates a strong relation.

avoid negative influence on their own keypoints. In contrast, the joints on the left retain their strong relationships to compensate for any noise of their own keypoints.

Attention Visualization

Fig 5.16 visualizes the multi-head attention maps of the two MHSA layers in the Auxiliary Spatial Graph-Transformer (ASGT). Here, too, there are strong similarities to the baseline (see Fig. 5.4), as the pre-trained auxiliary model is based on the modified GraFormer model used for transfer learning in the baseline. However, as with the adjacency matrices, the strength of certain connections differs: In head 1 of the first layer, it is noticeable that the relationships of the upper body joints to the root intensify, particularly evident at the head (from 0.41 to 0.51) and the right elbow (from 0.29 to 0.4). The relationships between hip joints and root also increase (right: from 0.46 to 0.52, left: from 0.43 to 0.49). With a PCKh@0.2 = 76.2 %, the CPN-detected keypoints of the root are relatively accurate compared to those of the other joints (see Tab. 5.1). In head 3, the trend is that the attention shifts from hips to shoulders, head and thorax, which could be due to poor 2D detections of both hip joints (right: PCKh@0.2 = 50.3 %, left: PCKh@0.2 = 38.2 %). Similar is also observable in head 0 of the second layer, where the focus shifts from hips, thorax and neck to the head, which is now the center of attention. Note that the head is the joint with the most accurate 2D detections, with a PCKh@0.2 = 92.7 %, which may explain the particularly close relationships of the other joints to it. In head 1, the relationships are weaker overall, particularly evident in the relationships to the right and the left foot, as well as the right knee. The same applies to head 3, indicated by weaker connections from the feet to themselves (left: from 1.0 to 0.75, right: from 0.91 to 0.7) as well as to each other (left to right: from 0.93 to 0.84, right to left: from 0.98 to 0.84). In head 2, the trend is that attention shifts from the poorly detected thorax (PCKh@0.2 = 46.2 %) to the head, the joints with the most accurate detections. Overall, attention is more focused toward the center of the body, indicating the influence of noise prediction, as this is where the most accurate keypoints are located.

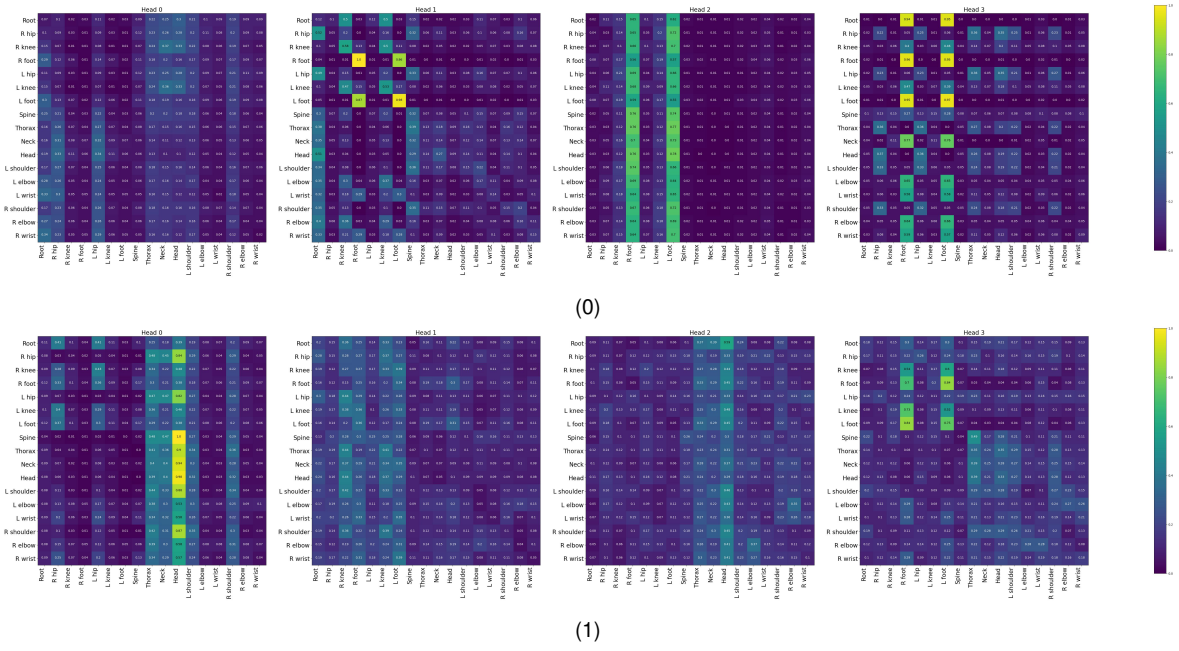


Figure 5.16: The multi-head attention maps ($h_1 = 4$) of the two MHSA layers in the Auxiliary Spatial Graph-Transformer (ASGT) module of the model-driven approach. The attention output is averaged across all actions and normalized from 0 to 1.

Fig 5.17 visualizes the multi-head attention maps of the MHSA layer in the Temporal Transformer Module (TTM). The effect of supervision at full sequence scale is again apparent as diagonal attention highlights, indicating that a particularly strong focus is placed on immediate preceding and subsequent frames, thereby achieving temporal consistency. This is especially evident in head 2, but also in heads 3, 6, and 7, and even slightly in head 1. In contrast to the baseline, the diagonal does not narrow towards the center and remains essentially the same width, i.e. each frame focuses on approximately 20 to 25 surrounding frames throughout the entire sequence. The frames in the middle of the sequence are not biased toward the center, where the target frame is located. Even the target frame itself (frame 40) considers the same number of adjacent frames. This general observation suggests that estimating noise at the spatial level extracts information that supports temporal consistency. Inaccurate keypoints in the target frame itself (or in its immediate vicinity) cause the surrounding frames to be considered with the same importance to overcome the noise. The hypothesis is supported by the fact that the strength of the emphasized relationships is generally much stronger than in the baseline. In head 4 and 5, the effects of supervision at full sequence scale show up again, but in the form of two diagonals. Each frame does not focus on directly adjacent frames, but rather on preceding and subsequent frames that are approximately 15 to 30 frames further away. For example, while frame 20 focuses on frames 10 to 30 in head 2, it focuses on frames 0 to 5 and 35 to 50 in head 5. Frame 60, on the other hand, no longer focuses on frames 50 to 70, but on frames 30 to 45 and 75 to 81. Thus, head 4 and 5 examine temporal dependencies at a different level that complements head 2, 3, 6 and 7. This pattern already emerges in head 7 of the baseline (see Fig. 5.5), though not to the same extent, and in head 0 and 7 of the data-driven approach (see Fig. 5.11). There, the frames 9/18/27 and 54/63/72 particularly emphasize the frames close to the target, i.e. frames 30 to 50. However, in the model-driven approach, all relationships of frames 0 to 30 and 50 to 81 are generally much more emphasized those of the middle frames. This affects not only their relationships with the frames near the target, but also their relationships with the frames at the beginning and end of the sequence. Thus, in the attention map of head 5, not only the top and bottom center are highlighted, but also the top left and bottom right. This again suggests a positive interaction between noise prediction and supervision at full sequence scale to achieve temporal consistency. Nevertheless, the grid structure familiar from the baseline

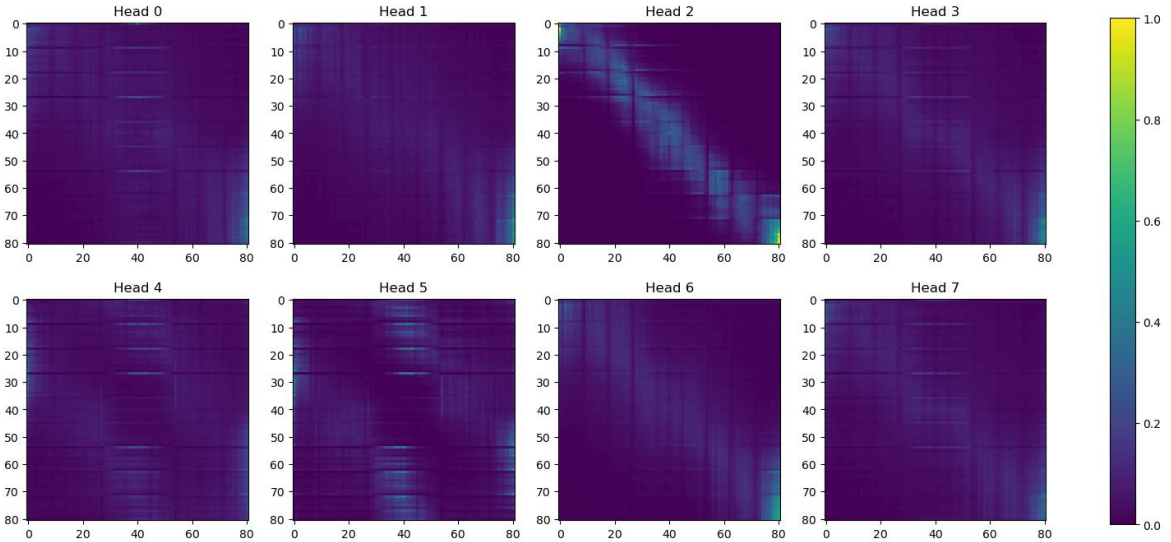


Figure 5.17: The multi-head attention maps ($h_2 = 8$) from the Temporal Transformer Module (TTM) of the model-driven approach. The attention output is averaged across all actions and normalized from 0 to 1.

and data-driven approach is also unmistakably present in the model-driven approach. Here, too, the pattern emerges from the lack of or emphasized attention on the frames 9/18/27, 54/63/72 and 36/40/45 and is caused by shrinking the sequence using strided convolutions. However, on closer inspection it is also noticeable that the attention maps of the baseline and the data-driven model are much smoother than those of the model-driven approach. In the first two approaches, each frame has a certain receptive field consisting of preceding and subsequent frames, all of which are accounted for (except for the keyframes that create the grid structure). In the model-driven approach, each frame as well has a certain receptive field consisting of preceding and subsequent frames, but not each of them is considered. Instead, frames within the receptive field are alternately emphasized resulting in less smooth attention highlights. This may be caused by adjacent frames containing very similar poses, so inaccurate keypoints in one frame are likely to be inaccurate in immediately preceding and subsequent frames as well. Thus, the model alternately amplifies and attenuates frames depending on whether they contain more or less inaccurate keypoints, demonstrating the effect of noise prediction.

Fig 5.18 visualizes the multi-head attention maps of the first MHSA layer in the Strided Transformer Module (STM). The attention maps are again very reminiscent of those in the baseline (see Fig. 5.6), as all frames have a particularly strong focus on the middle of the sequence, specifically on frames 30 to 50, which are closest to the target. Heads 2 and 3 immediately catch the eye, as there particularly strong focus is placed on the target frame, resulting in highlighted peaks of attention in the center column, just like heads 0 and 5 of the baseline. This behavior is also visible in head 1. Moreover, head 2 has by far the smallest temporal receptive field considering only frames 37 to 43, while the temporal receptive field of head 1 extends to frames 35 to 45 and those of the remaining heads further to frames 30 to 50. The hourglass pattern known from the baseline is also slightly implied in most of the heads, especially in head 7. In contrast to the middle frames near the target, frames in the beginning and end of the sequence have a larger receptive field that accounts for more distant frames, as more information from far preceding or subsequent frames is necessary to correctly estimate the center pose. Still, none of the heads seem to focus on frames that are farther than 10 frames from the target. Heads 1 and 2 of the baseline, on the other hand, expand their temporal receptive field to the frames 20 to 60. On closer inspection, however,

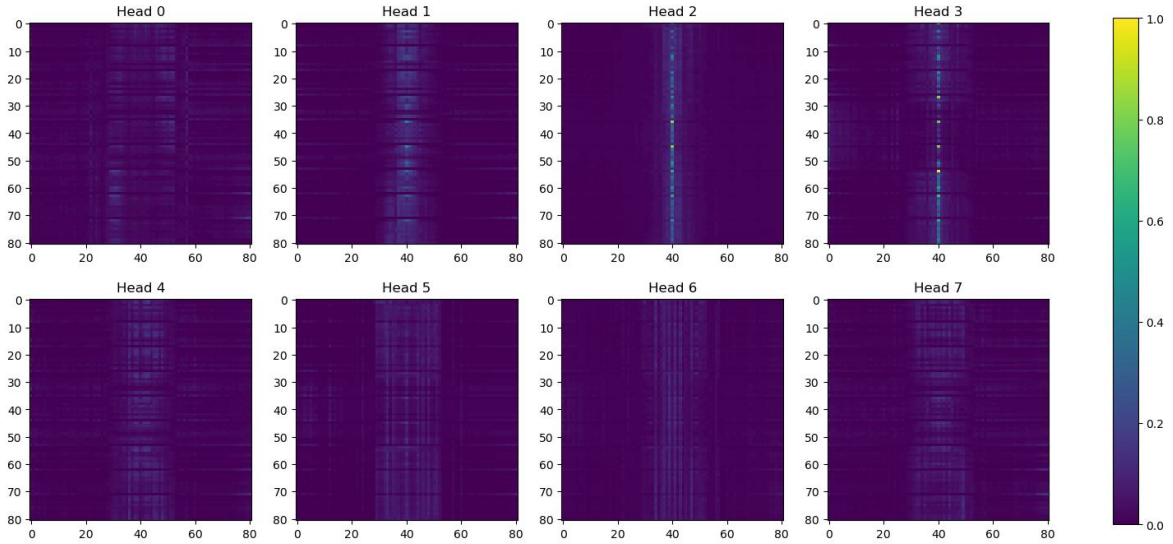


Figure 5.18: The multi-head attention maps ($h_2 = 8$) from the Strided Transformer Module (STM) (first layer). The attention output is averaged across all actions and normalized from 0 to 1.

it can be seen that some heads in the model-driven approach even consider frames at the very beginning or end of the sequence (although not to the same extent as those in the middle), e.g. heads 3, 5 and 6. It is striking, however, that this does not apply uniformly to all frames of the entire sequence, but only to individual frames, which makes the corresponding rows particularly prominent. In addition, not all frames in the middle of the sequence are emphasized but rather individual frames. The behavior of heads 2 and 3, where the target frame is particularly highlighted, is applied to nearby frames, causing the corresponding columns to be particularly prominent. The highlighted rows and columns create a grid-like pattern that is reminiscent of that of the baseline and data-driven approach, but much more tightly meshed. No longer are only frames 9/18/27, 54/63/72 and 36/47/45 particularly emphasized. Similar to the TTM, predicting input noise could result in frames of the entire sequence being considered, as keypoints in and around the target frame could be noisy. The spacing between the highlighted rows and columns could be due to the fact that adjacent frames contain highly similar poses, so inaccurate keypoints in one frame are likely to be inaccurate in the immediately preceding and subsequent frames as well. As already mentioned, this also leads to significantly more structured attention maps in the TTM. This effect might be reinforced by the STM due to backpropagation. Meanwhile, head 4 reveals relationships not visible in the other heads. Opposed to the other head, the focus is not evenly placed on the frames near the target. Instead, the first part of the sequence focuses on the frames 45 to 50, just after the target, and the second part of the sequence focuses on the frames 30 to 35, just before the target. The pattern is reminiscent of head 5 in the TTM and may have originated there.

5.4.3 Occlusion Robustness Analysis

The motivation of this approach is to develop a model that outperforms the baseline's performance by being more robust to input noise. To this end, the model's capability to deal with noisy keypoints is analyzed.

Noisy Keypoints

Fig 5.19 compares the average MPJPE per joint with noisy CPN [Che+18] detected keypoints, Mask R-CNN [He+17] detections and 2D ground truth (GT) as input. As in the baseline, especially mobile joints such as wrists, elbows and feet are generally the main source of error (MPJPE $>$ 60 mm for CPN detections), whereas more static joints such as hips, spine and thorax cause the smallest errors (MPJPE $<$ 35 mm for CPN detections). However, the model-driven approach outperforms the baseline by 1.6 mm for 2D GT input and by 0.5 mm for Mask R-CNN detections, suggesting better generalizability. Recall that the average error at the head in the baseline was lower for CPN detections than for 2D GT. This is no longer the case with the model-driven approach, even though the average error at the head is 0.5 mm smaller for CPN detections, supporting the claim of better generalizability. Significant improvement over the baseline is also especially observable at the left elbow, with lower error for both CPN and Mask R-CNN detections (ΔMPJPE $= -1.5$ mm and ΔMPJPE $= -1.6$ mm, respectively). Note that the majority of 2D detections of the left elbow is classified as not accurate. In fact, for CPN detections, the left elbow is the joint with the fewest accurate 2D detections (PCKh@0.2 $= 35.3\,\%$). Possible causes for the improved 3D position may be that the left elbow shifts its attention from other, less accurate joints to more accurate ones, thus being less affected by noise. For example, the spatial attention maps (Fig. 5.16) show that the joint reduces its attention on the thorax (from 0.39 to 0.25 in layer 1, head 2), while at the same time increasing its attention on root (from 0.26 to 0.35 in layer 0, head 1) and head (from 0.38

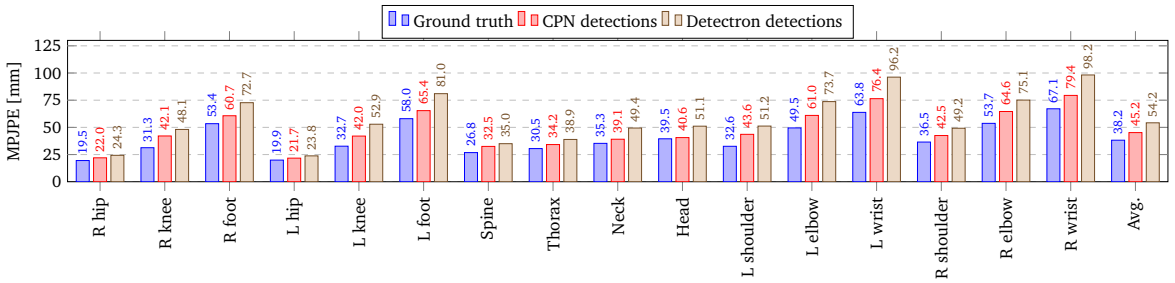


Figure 5.19: Average MPJPE comparison per joint of the model-driven approach on Human3.6M [Ion+14].

to 0.43 in layer 1, head 2). It further reduces the strength of its connection to the thorax (from 0.35 to 0.23) as well as to itself (from 1.79 to 1.67) in the second adjacency matrix (Fig. 5.15). However, this does not always necessarily lead to an improved localization in 3D. Recall, for example, that when comparing model-driven and baseline approach at a spatial level, the right foot pays much less attention to the left foot (from 0.98 to 0.84 in Fig. 5.16, layer 1, head 3), as its keypoints are strongly affected by noise. The same is observable for the left foot itself (from 1.0 to 0.75). Nevertheless, the estimated 3D localization of both joints deteriorates compared to the baseline. In addition, the error at the left foot for CPN detections deteriorates by "only" 0.8 mm, while that at the right foot deteriorates by 1.6 mm. The general deterioration can be attributed to the fact that both joints continue to have very strong, unique relationships to each other and to themselves compared to the other joints (layer 0, head 1 and layer 1, head 3), although they are not among the most reliable keypoints, especially the left foot (PCKh@0.2 = 38.4 %). The significantly higher deterioration of the right foot can be attributed to the fact that it not only reduces its attention to the left foot, but also to itself. Even more, while its attention on the left foot is significantly lower in absolute terms than before (from 0.98 to 0.84), in relative terms it is now significantly more in focus than the right foot itself (which is now at 0.7), thus paying 20 % more attention to the noisy keypoints of the left foot. The same, mirrored development is observed for the left foot, i.e. the focus is more on its symmetrical counterpart than on itself, but unlike before, this is not a problem as the right foot is more accurate. As a result, the 3D estimates of the left foot deteriorates less than those of the right foot. It should be noted, however, that overall the error at the left foot is still 4.7 mm greater than the error at the right foot.
Similar can be observed for other joints, namely a shift of attention from other, less accurate joints to more accurate ones, e.g. from the thorax to the head, without improving the estimated 3D position. As with the feet, this is due to the interaction of all joints, because when less inaccurate joints are considered, the relationships with other joints consequently becomes more prominent if they are not adjusted accordingly. For example, the spatial attention maps (Fig. 5.16) show that the right knee places clearly less focus on the inaccurate left hip (PCKh@0.2 = 38.2 %) (from 0.63 to 0.43 in layer 1, head 0). Nevertheless, its error deteriorates by 0.8 mm, since the already particularly strong connections to the feet (right: 0.66 and left: 0.7 in layer 0, head 2) are much more significant now. This is problematic in that the estimated 3D location of both feet are significantly less accurate (left: ΔMPJPE = 0.8 mm, right: Δ MPJPE = 1.6 mm). Furthermore, it can also be observed that while inaccurate joints receive generally less attention, improving the error of some joints, this trend is also true for the corresponding symmetrical relationships. Thus, the left hip also focuses less on the right knee (from 0.53 to 0.44 in layer 1, head 1), despite its PCKh@0.2 = 72.6 %. Although the inaccurately detected thorax is also less taken into account (from 0.61 to 0.47 in layer 1, head 0), the error of the left hip still deteriorates by 0.6 mm, since its relation to the right knee is more essential than its relation to the thorax due to their proximity and similar nature.

A general comparison between the performance under CPN and Mask R-CNN detections reveals that the model appears to be particularly robust against noise at the same joints as the baseline, i.e. knees, wrists and spine. At these joints, the difference between the 2D detections of CPN and Mask R-CNN is significantly larger compared to the corresponding estimated 3D pose, i.e. the model is still able to localize the joint relatively accurately despite less accurate 2D detections. Also, note that while some of the model's 3D estimates of certain joints are slightly worse compared to the baseline, the model's robustness against noise is consistently greater:

Right Knee The estimated 2D keypoint of Mask R-CNN is on average 19.0 % noisier than that of CPN (0.2295 px vs. 0.2730 px). The 3D pose estimation of the model-driven approach deteriorates by only 14.3 % (42.1 mm, vs. 48.1 mm), that of the baseline by 15.7 % (41.3 mm vs. 47.8 mm).

Left Knee The estimated 2D keypoint of Mask R-CNN is on average 35.7 % noisier than that of CPN (0.3334 px vs. 0.4486 px). The 3D pose estimation of the model-driven approach deteriorates by only 26.0 % (42.0 mm vs. 52.9 mm), that of the baseline by 27.1 % (42.1 mm vs. 53.5 mm).

Spine The estimated 2D keypoint of Mask R-CNN is on average 15.0 % noisier than that of CPN (0.2248 px vs. 0.2586 px). The 3D pose estimation of the model-driven approach deteriorates by only 7.7 % (32.5 mm vs. 35.0 mm), that of the baseline by 10.0 % (32.1 mm vs. 35.3 mm).

Left Wrist The estimated 2D keypoint of Mask R-CNN is on average 34.6 % noisier than that of CPN (0.3334 px vs. 0.4486 px). The 3D pose estimation of the model-driven approach deteriorates by only 25.9 % (76.4 mm vs. 96.2 mm), that of the baseline by 27.7 % (76.6 mm vs. 97.8 mm).

Right Wrist The estimated 2D keypoint of Mask R-CNN is on average 34.6 % noisier than that of CPN (0.3531 px vs. 0.4754 px). The 3D pose estimation of the model-driven approach deteriorates by only 23.7 % (79.4 mm vs. 98.2 mm), that of the baseline by 25.2 % (79.1 mm vs. 99.0 mm).

Meanwhile, the model's sensitivity to noise is consistent with that of the baseline:

Neck The estimated 2D keypoint of Mask R-CNN is on average 1.9 % noisier than that of CPN (0.1928 px vs. 0.1964 px). The 3D pose estimations of both the model-driven and baseline approach deteriorate by 26.3 % (39.1 mm, vs. 49.4 mm and 39.9 mm vs. 50.4 mm, respectively).

Left Elbow The estimated 2D keypoint of Mask R-CNN is on average 7.2 % noisier than that of CPN (0.3432 px vs. 0.3680 px). The 3D pose estimation of the model-driven approach deteriorates by 20.8 % (61.0 mm vs. 73.7 mm), that of the baseline by 20.8 % (62.5 mm vs. 75.0 mm).

Right Elbow The estimated 2D keypoint of Mask R-CNN is on average 8.4 % noisier than that of CPN (0.3291 px vs. 0.3566 px). The 3D pose estimation of the model-driven approach deteriorates by 16.3 % (64.6 mm vs. 75.1 mm), that of the baseline by 17.0 % (64.7 mm vs. 75.7 mm).

Thus, contrary to expectations, the model is not necessarily better at localizing noisy keypoints, but rather reinforces its already existing occlusion robustness by focusing less on noisy and more on more reliable keypoints, resulting in better generalizability.

5.4.4 Ablation Studies

In the following, different design choices for incorporating the pre-trained auxiliary model into the ASGT module for transfer learning are investigated. The variant already presented concatenates the pre-trained pose and noise embeddings directly, without additional pre-processing. In addition, two other variants were investigated (see Fig. A.3 in the appendix). The first alternative uses one extra layer to transform the noise embedding prior to concatenation with the pose embedding, while the second alternative transforms both pose and noise embedding prior to concatenation by two extra layers. Note that these extra layers are not part of the auxiliary model and are only trained during transfer learning, fine-tuning and pose refinement (phase 1 - 3). In theory, this allows the model to bring the separately extracted features on one level before concatenation, if necessary. Tab. 5.7 reports the final results of all three variants. It is easy to see that the additional processing degrades performance in both cases, although not significantly (max. $\Delta\text{MPJPE} = 0.4\text{mm}$). Thus, additional pre-processing is not necessary and the post-processing layer after concatenation appears sufficient to prepare the features for the subsequent modules.

Pre-processing pose embedding	Pre-processing noise embedding	MPJPE [mm]
✗	✗	45.2
✗	✓	45.5
✓	✓	45.6

Table 5.7: Ablation study on different design choices for the Auxiliary Spatial Graph-Transformer (ASGT). The evaluation is performed on Human3.6M [Ion+14].

5.4.5 Discussion

Overall, the model is not able to outperform the baseline on the original dataset, but achieves similar performance through the efficient use of spatial and temporal dependencies. Furthermore, it could be shown that estimating input noise is limited due to the constraints imposed by the dataset, i.e. imbalanced classes of accurate and inaccurate keypoints as well as a different distribution of noise in train and test set. Nevertheless, the model is able to determine noisy keypoints to some extent under these constraints, suggesting that better results could be obtained with more suitable data. In particular, keypoints that are far from the 2D ground truth (GT) ($\geq 0.5\,\text{px}$) are well identified as noisy. In contrast, the model has difficulty classifying keypoints that are only slightly noisy (distance from 2D GT $< 0.3\,\text{px}$), especially keypoints that are just on the borderline of being classified as non-noisy (distance from 2D $\in [0.1, 0.2)\,\text{px}$). Note, however, that a slight noise is probably due to inaccuracy of the 2D detector. In contrast, the particularly noisy keypoints are those caused by occlusion and lead to gross errors in the 2D-to-3D lifting pipeline, so their detection is of greater importance. Accordingly, there were obvious changes in the learned relationships of the model. In principle, the learned relationships correspond to those of the baseline, but at the spatial level, it can be seen that joints shift their attention from noisier to more accurate keypoints, demonstrating the effects of estimating input noise. As a result, the model established strong connections especially to joints in the center of the body (i.e. head, neck, root), where most keypoints are very accurate (PCKh@0.2 $> 75\,\%$). The effects of noise prediction are also apparent in the attention maps of the TTM (Fig. 5.17) and STM (Fig. 5.18), since the extracted noise features are further processed at the temporal level. They involve increased temporal consistency in the TTM and a larger receptive field in the STM that extends throughout the entire sequence. Thus, by efficiently using the self-attention mechanism, the model captures long-ranging relationships to compensate for inaccurate keypoints. In addition, the model alternately amplifies and attenuates frames in both TTM and STM depending on whether they contain more or less inaccurate keypoints. These improved relationships also have a positive effect on the gener-

alizability of the model to unseen data from different 2D detectors. However, the occlusion robustness analysis reveals that, contrary to expectations, the improved generalizability cannot be attributed to the model being better at localizing noisy keypoints. Rather, estimating input noise causes noisy keypoints to receive less attention, resulting in other joints being less disturbed by noise from their neighbors. However, to achieve an overall better 3D error on the corresponding joint, it is necessary to adjust all relations so that the reduced focus of one noisy joint does not result in giving more weight to another noisy joint. The model-driven approach had its problems in this regard, resulting in poorer 3D estimates for certain joints, although noisy keypoints were emphasized less. Improvements could possibly be achieved by a more appropriate architecture of the auxiliary model (or spatial module), where pose estimation and noise estimation are trained together from scratch.

5.5 Comparative Analysis

This section starts with a detailed quantitative comparison of the baseline, the data-driven approach and the model-driven approach based on their performance on Human3.6M [Ion+14] for individual actions using CPN [Che+18] detections and 2D GT as input. In addition, qualitative results are presented. Lastly, all three methods are compared with the current state-of-the-art.

5.5.1 Comparison of the Proposed Methods

Tab. 5.8 compares the performance of the three methods proposed in detail for each action using CPN detections as input. In general, it is noticeable that there are significant differences between individual actions. The hardest actions, i.e. those with the largest errors, are Photo, Sitting and SittingDown. Here the MPJPE of all models is > 50.0 mm and the P-MPJPE is > 40.0 mm. For action SittingDown the errors are even > 60.0 mm and > 50.0 mm, respectively. Usually the difficulty of estimating an action correctly lies in the complexity of the poses, which is also reflected in particularly high 2D errors in the CPN detections (SittingDown: 0.6015 px), affecting the performance of the models (see Tab. 5.10). Often the difficulty is also amplified by a small amount of training data (relative to the complexity of the poses), see Tab. 5.9 (e.g. Photo, SittingDown, Purchases). This is sometimes also a reason for high 2D errors of CPN without the actions being particularly complex, as is the case for actions Greeting (0.4460 px) and Waiting (0.4166 px), since the detector was fine-tuned on Human3.6M. In this case, it is helpful to exploit temporal relationships to overcome detection errors, as seen in all three approaches, whose MPJPEs are about $10-20$ mm lower for the actions Greeting and Waiting than for Photo, Sitting and SittingDown. Furthermore, note that the 2D error of action Posing suggests that the action is of low complexity, but the 3D errors

Protocol #1	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
Baseline	41.8	46.0	40.8	43.4	46.2	53.3	43.3	42.5	56.4	61.9	45.7	42.9	46.5	32.2	32.7	45.0
Data-driven approach	41.8	46.5	40.9	44.1	46.1	53.5	43.7	43.5	56.3	63.5	46.1	42.9	46.3	31.9	32.6	45.3
Model-driven approach	41.7	45.4	40.5	43.7	46.0	54.0	43.2	43.9	55.3	64.3	45.1	42.6	46.0	32.4	33.4	45.2
Protocol #2	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
Baseline	33.9	36.8	33.9	36.4	36.6	42.2	34.4	33.9	46.3	50.6	37.4	33.7	37.5	26.2	27.8	36.5
Data-driven approach	33.5	36.6	33.5	36.7	36.1	42.0	34.6	34.2	46.2	51.1	37.7	33.4	37.1	25.9	27.3	36.4
Model-driven approach	33.8	36.7	33.5	36.4	36.2	42.3	34.4	35.0	46.6	52.0	37.4	33.6	37.4	26.6	28.1	36.6

Table 5.8: Quantitative comparison of the proposed methods on Human3.6M [Ion+14] under protocol #1 and protocol #2 using CPN [Che+18] detections as input.

Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Total
100.9	158.8	109.4	72.4	115.8	76.0	69.5	63.1	245.7	129.2	133.3	115.3	79.4	132.7	87.3	1559.8

Table 5.9: Number of frames (in thousands) per action in the train set of Human3.6M [Ion+14].

	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
Err. [px]	0.2056	0.2119	0.2072	0.4460	0.2285	0.2271	0.1848	0.2120	0.2682	0.6015	0.2311	0.4166	0.2219	0.1867	0.1839	0.2689
PCKh@0.2 [%]	60.6	61.9	60.1	49.4	57.5	56.1	65.4	60.0	55.6	43.0	57.9	53.5	56.1	64.3	65.1	57.8

Table 5.10: Average head-normalized distance from 2D ground truth (see Eq. 4.1) and PCKh@0.2 per action for CPN [Che+18] on Human3.6M [Ion+14].

of all models are comparatively quite high. One reason could be that the action is among the least frequent in the training data (69.5 k). The fact that the 3D pose estimators have significantly more problems with the action than the 2D detector also suggests that many of the poses suffer from depth ambiguity. Fairly simple actions, i.e. those with the smallest errors, are Walking and WalkTogether with an MPJPE < 35.0 mm and an P-MPJPE < 30 mm for all models. These actions generally have few full body movements and little variation between individual frames. During WalkDog, on the other hand, the actors sometimes kneel down, for example, which makes the action more difficult compared to the other two walking variations. Overall, the maximum error difference under protocol #1 between the actions is nearly 30 mm and above for all models.

Each method also has individually strong and / or weak actions compared to the others. However, there is no action where the data-driven model performs significantly better than the baseline. This was to be expected, as the advanced complementary exploitation of temporal information is particularly beneficial when parts of the sequence are incomplete, allowing the data-driven model to focus on intact frames. In this experiment, however, the original test set is not manipulated, i.e. no artificial occlusion is applied, so there is no advantage to exploiting temporal relations to frames that are far from the target. Specifically, the model performs worse on actions Discussion, Greeting, Purchases and SittingDown compared to the baseline (ΔMPJPE: 0.5 mm, 0.7 mm, 1.0 mm, 1.6 mm, respectively). For actions Discussion, Greeting and Purchases, this could be due to the fact that the corresponding poses are characterized by rapid, short-lasting upper body movements that strongly vary from frame to frame, so that information about temporally distant frames is of little use (unlike, for example, for actions PhoneTalk and Directions, in which upper body movements also dominate, but persist over several frames, so that exploiting temporal relations to distant frames in a complementary manner is sufficient to maintain the baseline's performance). In the case of Purchases, the actors additionally bend over briefly to simulate picking up their goods, which in combination with little training data (63.1 k) makes the action additionally challenging. The same applies to the action SittingDown, only here the poses are characterized by strongly varying, complicated leg / full body movements, since the actors do not sit on a chair for the entire sequence (as in actions Sitting, Eating, PhoneTalk and Smoking), but sit down on the floor and get up again several times. In addition, the comparatively small number of training data could make generalization from occlusion-augmented to original data more difficult. Overall, however, the same average MPJPE can still be achieved.

The model-driven approach outperforms the baseline on actions Discussion, Sitting, Smoking and WalkDog (ΔMPJPE: 0.6 mm, 1.1 mm, 0.6 mm, 0.5 mm, respectively). In contrast, the baseline outperforms the model-driven approach on actions Photo, Purchases, SittingDown and WalkTogether (ΔMPJPE: 0.7 mm, 1.6 mm, 2.4 mm, 0.7 mm, respectively). Sitting and Smoking, along with PhoneTalk and Eating (where the model-driven approach also performs slightly better than the baseline), are actions in which actors spend most of their time sitting on a chair. Since the actors remain in a seated position and do not move far from the spot,

the model could benefit from its greater temporal consistency compared to the baseline, despite reduced focus on the legs. Furthermore, Smoking, PhoneTalk and Eating are actions in which strong relationships to the head and neck might be beneficial for localizing the wrist performing the action. This may also apply to action Sitting, when actors rest their head on their hands. In addition, these actions are mainly characterized by movements in the upper body, which means that there is little movement in the lower body, namely the root (on which the model puts more focus) and the hips (on which the model puts less focus), thus making the first one more reliable as it already is and less affected by noisy neighboring joints. The same could be true for action Photo, but the model's MPJPE deteriorates by 0.7 mm compared to the baseline. This could be due to the fact that the action takes place in a standing position with both hands near the head instead of just one. This in turn could result in the head being more affected by occlusion-related noise than in other actions, making its detected 2D keypoints not as reliable as usual, which negatively influences the estimation of the entire 3D pose. In addition, Photo, Purchases and SittingDown are full body movements that further have in common that the actors sit/squat/bend down and stand up again several times during the sequence. The model has difficulty with these actions because they are characterized by dominant movements of the lower body that result in heavy occlusion (i.e. noisy lower body keypoints), causing the model to focus less on the part of the body that is central to determining the pose. For action SittingDown, this is compounded by the fact that a large portion of keypoints are noisy, resulting in only the small portion of reliable keypoints being considered (in the case of the model-driven approach), which has a negative impact on the overall pose estimation.

Tab. 5.11 compares the performance of the three methods proposed in detail for each action using 2D ground truth as input. The average performance of the baseline and data-driven approach improves by approximately 5 mm under protocol #1 and #2, while the model-driven approach improves its performance by approximately 7 mm and 6 mm under protocol #1 and protocol #2, respectively, demonstrating better generalizability. As expected, the models especially improve on the actions where CPN has a high error. The MPJPE of all models is < 50.0 mm even for the hardest action SittingDown, reducing the error by more than 10 mm. The models show significantly smaller improvements on actions where the average 2D error for CPN is < 0.2 px, i.e. actions Posing, Walking and WalkTogether. Here, baseline, data-driven and model-driven approaches could only reduce their estimates by a maximum of 0.7 mm, 1.6 mm and 3.9 mm, respectively.
As for CPN-detected input poses, each method has its own strong and / or weak actions compared to the others. The baseline still outperforms the data-driven approach on actions Discussion, Purchases and SittingDown (ΔMPJPE: 0.7 mm, 1.3 mm, 1.6 mm, respectively), which is not surprising as this was already the case when using CPN detections as input. These poses vary greatly from frame to frame, so exploiting frames close to the target is beneficial here, which is why the data-driven model cannot keep up with the baseline's performance. This also seemed to be the case for action Greeting, as the data-driven model performs worse

Protocol #1	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
Baseline approach	38.3	41.9	33.0	39.7	39.2	45.2	42.6	37.3	45.5	48.4	40.8	39.9	41.0	31.6	32.0	39.8
Data-driven approach	38.0	42.6	32.8	39.8	39.7	45.5	42.1	38.6	45.2	50.0	40.6	40.0	42.6	31.8	31.0	40.0
Model-driven approach	36.0	39.8	32.2	37.9	37.7	43.9	39.3	35.2	44.6	49.1	39.7	38.4	38.7	30.2	30.9	38.2
Protocol #2	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
Baseline approach	30.2	33.5	27.1	31.3	31.3	37.3	31.1	28.9	36.3	41.5	33.5	30.5	34.0	25.3	24.8	31.8
Data-driven approach	29.6	33.4	27.0	31.3	30.9	37.3	30.2	29.5	36.7	41.9	33.4	30.6	34.4	25.6	24.6	31.8
Model-driven approach	29.0	32.3	27.3	30.5	30.1	36.1	28.7	28.3	36.3	41.8	32.2	29.4	32.2	24.3	23.9	30.8

Table 5.11: Quantitative comparison of the proposed methods on Human3.6M [Ion+14] under protocol #1 and protocol #2 using 2D ground truth as input.

than the baseline when using CPN detections. However, for 2D GT, the data-driven model performs similarly to the baseline on action Greeting, suggesting that temporal information from distant frames was previously not sufficient to maintain the baseline's performance because the CPN detections are very noisy (see Tab. 5.10). In contrast, the data-driven model performs worse than the baseline for action WalkDog (ΔMPJPE $= 1.6$ mm) when using 2D GT, while the models' performance is about the same when using CPN detections. In this case, however, the reasons are not entirely clear. Apart from that, the data-driven model now outperforms the baseline by 1.0 mm on action WalkTogether and by 0.5 mm on action Posing under protocol #1. Both are actions that, along with action Walking, provide the most accurate CPN detections. Unlike Walking, however, Posing and WalkTogether occur comparatively rarely in the train set (Posing: 69.5 k, WalkTogether: 87.3 k, Walking: 132.7 k), making it more difficult for the baseline to generalize to unseen data that is only slightly better, while the data-driven model manages to do so through its efficient exploitation of temporal information.
The model-driven approach outperforms the baseline in all actions (ΔMPJPE ≥ 0.8 mm), except for SittingDown (where the MPJPE is greater by 0.5 mm), clearly demonstrating the model's weakness. Note that the model-driven approach already performs significantly worse than the baseline on this action when using CPN detections. The action SittingDown differs from the others in that it is extremely complex, but in return it rarely occurs in the train set (129.2 k vs. Sitting: 245.7 k), making it difficult for the model to learn how to handle the amount of noisy keypoints. In contrast, while for CPN detections the model-driven approach performs worse than the baseline on actions Photo, Purchases and WalkTogether, for 2D GT, it now outperforms the baseline with an ΔMPJPE ≥ 1.3 mm. Note that these actions are among the least frequent ones in the train set (see Tab 5.9), which may further degrade the already poor generalizability of the baseline. Thus, instead of the model-driven approach significantly improving its performance for 2D GT, the baseline actually suffers from poor generalizability. This is also evident for action SittingDown, where the model-driven approach manages to reduce the performance gap to the baseline under protocol #1 from 2.4 mm to 0.5 mm when using CPN detections vs. 2D GT.

Fig. 5.20 compares the three methods qualitatively by visualizing the results of 5 randomly selected samples from the test set. The estimated poses of the models do not differ much, which was to be expected since they are all based on the same architecture and achieve a similar MPJPE on the test data. It is apparent to all approaches that errors in the detected 2D poses propagate to the 2D-to-3D pose estimation, which is especially visible in rows 1 (left elbow, left wrist) and 5 (left foot, right elbow, right wrist). In individual cases, however, gross detection errors can be overcome, as seen in row 3. Comparing the detected and true 2D pose of the target frame, it becomes clear that the 2D detection error of the left foot is caused by occlusion. The fact that the error is not transferred to the 3D pose in any of the models suggests that exploiting temporal relationships was crucial to overcoming occlusion in this case. In contrast, in the examples of rows 1 and 5, the detection errors are mainly due to the inaccuracy of the 2D detector and the complexity of the poses, respectively. In these cases, it is likely that similar errors occur in adjacent frames. Therefore, exploiting temporal relationships is not sufficient to overcome the detection errors, which is reflected in the errors of the estimated 3D pose. However, the biggest (albeit still subtle) differences between the model estimates can be seen at the joints that have been incorrectly detected. On the one hand, they show that the data-driven model deviates more from the baseline for joints located at the end of a limb, i.e. wrists and feet, as can be seen in rows 3 and 5. These are joints that are more likely to be affected by noise, so the different utilization of temporal information is particularly noticeable here, as 2D detection errors are thus handled differently. On the other

hand, they illustrate that the estimates of the model-driven approach generally deviate more from those of the baseline and the data-driven methods, due to differences in the extraction of spatial features at the beginning of the pose estimation pipeline. For example, the left wrist is placed much closer to the neck in row 1 and the right knee is placed much higher in row 5. As mentioned above, these are very subtle differences, but they still make the estimated pose fundamentally different from the other two.

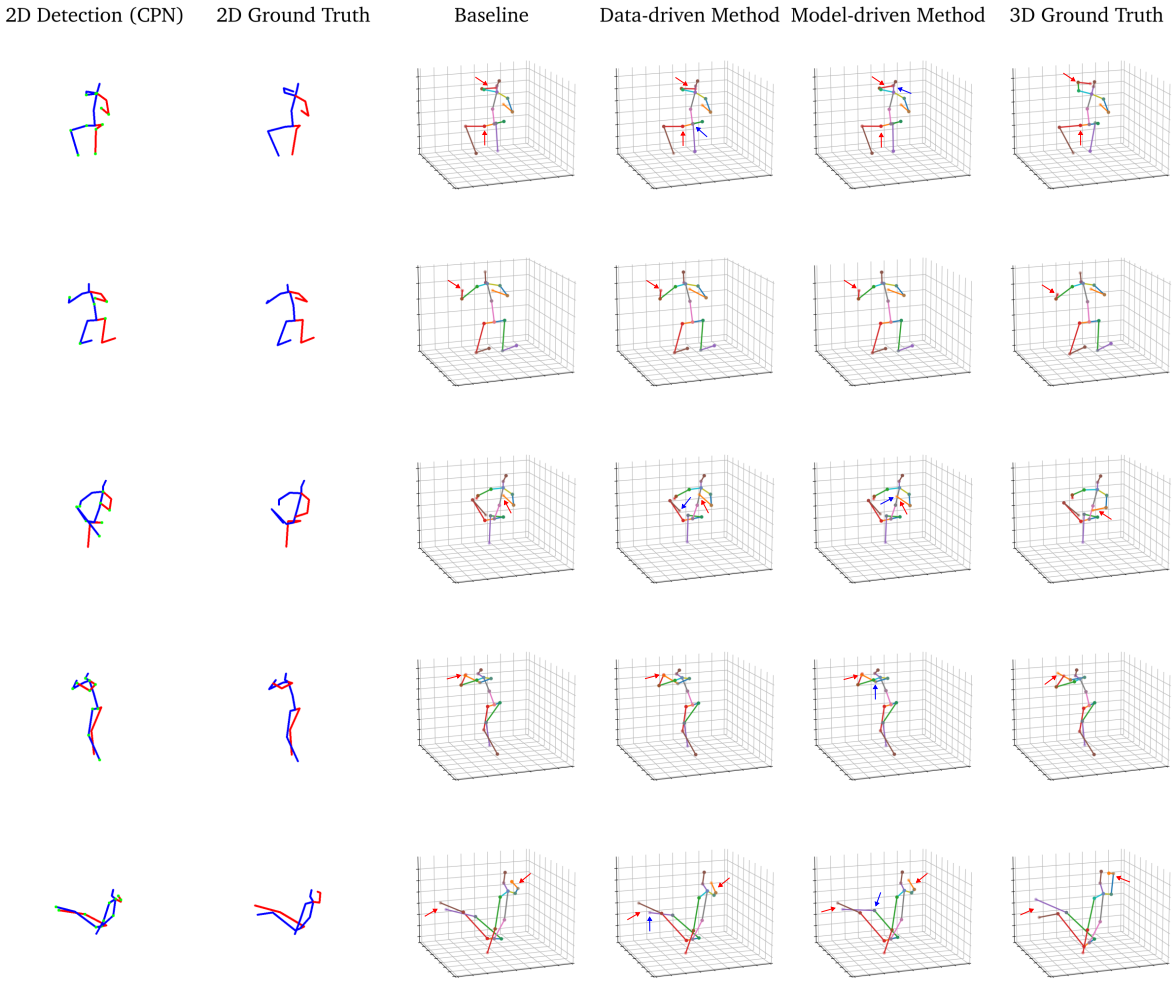


Figure 5.20: Qualitative comparison of the proposed methods on Human3.6M [Ion+14]. Note that the first two columns only show the 2D detection / projection of the target frame, but the models receive a 2D pose sequence of 81 frames as input. In 2D poses, left and right limbs are marked in blue and red, respectively. For CPN detections, inaccurate / noisy keypoints (head-normalized distance to the 2D GT $> 0.2\,\text{px}$) are highlighted in green. In 3D poses, red arrows highlight general differences between estimates and ground truth, while blue arrows highlight differences between individual model predictions.

5.5.2 Comparison with the State-of-the-Art

Tab. 5.12 compares the performance of the three methods presented with the current state-of-the-art (for a detailed summary of each approach see Sec. 3.2). Note that many state-of-the-art methods use a much larger temporal receptive field, which benefits performance, but also increases computational costs. Moreover, most state-of-the-art models are significantly larger, even those with the same number of input frames as the proposed methods, see Tab. 5.13.

Protocol #1	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
MixSTE [Zha+22] ($T = 81$)	39.8	43.0	38.6	40.1	43.4	50.6	40.6	41.4	52.2	56.7	43.8	40.8	43.9	29.4	30.3	42.4
MixSTE [Zha+22] ($T = 243$)	37.6	40.9	37.3	39.7	42.3	49.9	40.1	39.8	51.7	55.0	42.1	39.8	41.0	27.9	27.9	40.9
StridedTransformer [Li+22a] ($T = 81$)	43.3	45.8	42.7	44.3	47.8	53.2	43.4	41.3	56.8	61.1	46.9	44.3	46.7	32.2	33.4	45.5
StridedTransformer [Li+22a] ($T = 351$)	40.3	43.3	40.2	42.3	45.6	52.3	41.8	40.5	55.9	60.6	44.2	43.0	44.2	30.0	30.2	43.7
MHFormer[Li+22b] ($T = 81$)	41.1	45.2	41.2	43.1	45.6	52.7	42.2	42.5	54.4	61.3	45.1	42.8	46.9	31.4	33.1	44.6
MHFormer[Li+22b] ($T = 351$)	39.2	43.1	40.1	40.9	44.9	51.2	40.6	41.3	53.5	60.3	43.7	41.1	43.8	29.8	30.6	43.0
PoseFormer [Zhe+21] ($T = 81$)	41.5	44.8	39.8	42.5	46.5	51.6	42.1	42.0	53.3	60.7	45.5	43.3	46.1	31.8	32.2	44.3
P-STMO [Sha+22] ($T = 81$)	41.7	44.5	41.0	42.9	46.0	51.3	42.8	41.3	54.9	61.8	45.1	42.8	43.8	30.8	30.7	44.1
P-STMO [Sha+22] ($T = 243$)	38.4	42.1	39.8	40.2	45.2	48.9	40.4	38.3	53.8	57.3	43.9	41.6	42.3	29.3	29.3	42.1
CrossFormer [Has+22] ($T = 81$)	40.7	44.1	40.8	41.5	45.8	52.8	41.2	40.8	55.3	61.9	44.9	41.8	44.6	29.2	31.1	43.7
Baseline ($T = 81$)	41.8	46.0	40.8	43.4	46.2	53.3	43.3	42.5	56.4	61.9	45.7	42.9	46.5	32.2	32.7	45.0
Data-driven approach ($T = 81$)	41.8	46.5	40.9	44.1	46.1	53.5	43.7	43.5	56.3	63.5	46.1	42.9	46.3	31.9	32.6	45.3
Model-driven approach ($T = 81$)	41.7	45.4	40.5	43.7	46.0	54.0	43.2	43.9	55.3	64.3	45.1	42.6	46.0	32.4	33.4	45.2
Protocol #2	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
MixSTE [Zha+22] ($T = 81$)	32.0	34.2	31.7	33.7	34.4	39.2	32.0	31.8	42.9	46.9	35.5	32.0	34.4	23.6	25.2	33.9
MixSTE [Zha+22] ($T = 243$)	30.8	33.1	30.3	31.8	33.1	39.1	31.1	30.5	42.4	44.5	34.0	30.8	32.7	22.1	22.9	32.6
StridedTransformer [Li+22a] ($T = 81$)	34.6	36.8	34.9	36.7	37.4	41.9	34.4	32.9	46.6	50.1	38.4	34.2	37.4	26.1	28.3	36.7
StridedTransformer [Li+22a] ($T = 351$)	32.7	35.5	32.5	35.4	35.9	41.6	33.0	31.9	45.1	50.1	36.3	33.5	35.1	23.9	25.0	35.2
MHFormer [Li+22b] ($T = 81$)	31.9	36.0	33.5	35.2	36.1	40.8	32.8	33.0	43.9	49.6	36.9	33.2	36.6	24.8	27.2	35.4
PoseFormer [Zhe+21] ($T = 81$)	32.5	34.8	32.6	34.6	35.3	39.5	32.1	32.0	42.8	48.5	34.8	32.4	35.3	24.5	26.0	34.6
P-STMO [Sha+22] ($T = 243$)	31.3	35.2	32.9	33.9	35.4	39.3	32.5	31.5	44.6	48.2	36.3	32.9	34.4	23.8	23.9	34.4
CrossFormer [Has+22] ($T = 81$)	31.4	34.6	32.6	33.7	34.3	39.7	31.6	31.0	44.3	49.3	35.9	31.3	34.3	23.4	25.5	34.3
Baseline ($T = 81$)	33.9	36.8	33.9	36.4	36.6	42.2	34.4	33.9	46.3	50.6	37.4	33.7	37.5	26.2	27.8	36.5
Data-driven approach	33.5	36.6	33.5	36.7	36.1	42.0	34.6	34.2	46.2	51.1	37.7	33.4	37.1	25.9	27.3	36.4
Model-driven approach	33.8	36.7	33.5	36.4	36.2	42.3	34.4	35.0	46.6	52.0	37.4	33.6	37.4	26.6	28.1	36.6

Table 5.12: Quantitative comparison with state-of-the-art methods on Human3.6M [Ion+14] under protocol #1 and protocol #2. T refers to the number of input frames.

	#Params (M)	MPJPE (mm)
MixSTE [Zha+22] ($T = 243$)	33.70	40.9
StridedTransformer [Li+22a] ($T = 81$)	4.06	45.4
StridedTransformer [Li+22a] ($T = 351$)	4.34	43.7
MHFormer [Li+22b] ($T = 81$)	19.84	44.6
MHFormer [Li+22b] ($T = 351$)	31.52	43.0
PoseFormer [Zhe+21] ($T = 81$)	9.60	44.3
P-STMO [Sha+22] ($T = 81$)	5.40	44.1
P-STMO [Sha+22] ($T = 243$)	6.70	42.8
CrossFormer [Has+22] ($T = 81$)	9.93	43.8
Baseline ($T = 81$)	3.85	45.0
Data-driven approach ($T = 81$)	3.85	45.3
Model-driven approach ($T = 81$)	3.85	45.2

Table 5.13: Comparison of model size and MPJPE on Human3.6M [Ion+14].

Although the proposed models lag behind the best performing state-of-the-art (243-frame MixSTE) [Zha+22] by more than 4.0 mm, they have approximately 1/11 the number of trainable parameters (3.85 M vs. 33.70 M) and only require 1/3 of the input frames (81 vs. 243). The strength of MixSTE lies in the alternating exploitation of spatial and temporal relationships, the latter modeling the trajectory of individual joints instead of considering all joints individually. However, the purely transformer-based architecture has the disadvantage that it requires very high computational costs. Thus, the 81-frame version of MixSTE is with an MPJPE of 42.4 mm still significantly better than the models presented, but probably also requires significantly more parameters. Unfortunately, the size of the 81-frame model variant is not known, although it can be assumed that, similar to other state-of-the-art methods, the size difference between the different variants does not exceed 40 %, which would still leave the 81-frame MixSTE with 20.22 M parameters. For comparability, the remainder of this section continues to focus on models with a temporal receptive field of 81 frames.

PoseFormer pioneered purely transformer-based architectures for single-person 2D-to-3D pose lifting and has long dominated this task. Similar to the proposed methods, PoseFormer differentiates between spatial and temporal relationships through separate modules. With 9.60 M parameters, PoseFormer is more than twice the size of the models presented, but thus also achieves a ≥ 0.7 mm smaller MPJPE. Reasons for this include the more excessive use of spatial relations through a correspondingly larger spatial module. However, due to its purely transformer-based architecture, the module ignores prior geometric information of the human skeleton.

CrossFormer [Has+22] builds upon PoseFormer and attempts to improve the locality of Transformers by integrating depth-wise convolutions into the spatial module. In addition, similar to MixSTE, CrossFormer accounts for the interaction of a joint across subsequent frames (i.e. its trajectory), improving the exploitation of temporal relations. This allows CrossFormer to reduce the MPJPE of PoseFormer by 0.5 mm and outperform the proposed

methods by ≥ 1.2 mm. However, since it is still primarily a transformer-based architecture, it is also significantly larger than the presented methods (9.93 M vs. 3.85 M). Moreover, prior geometric information of the human skeleton is still not taken into account.
The 81-frame P-STMO [Sha+22] is another particularly well performing model, outperforming the proposed methods by around 1.0 mm under protocol #1. P-STMO also captures spatial relationships between joints within a single frame separately from temporal relationships between frames of the entire sequence. Note, however, that instead of a Transformer or GCN, a simple MLP block is used to capture spatial relationships between joints, making P-STMO one of the smallest state-of-the-art models with 5.40 M parameters. However, this still leaves the model with 40 % more parameters than the models presented. P-STMO is also trained using transfer learning and fine-tuning, where parts of the model are first pre-trained on a related task before the entire pipeline is trained end-to-end on the actual task. Similar to the data-driven approach, the related task involves masking parts of the input, however, the goal is to recover the masked 2D pose sequence instead of regressing it directly to 3D. Another difference is that the joints are masked by replacing their coordinates with learnable parameters instead of setting them to 0. It is unclear which of the two aspects is the key factor for the success of P-STMO.
Furthermore, the proposed models achieve similar performance to the 81-frame Strided-Transformer [Li+22a] (MPJPE: 45.5 mm, P-MPJPE: 36.7 mm), while requiring approximately 0.21 M fewer trainable parameters. Recall that both Temporal Transformer Module (TTM) and Strided Transformer Module (STM) of the proposed methods are taken from Strided-Transformer. In contrast, StridedTransformer does not differentiate between spatial relations within a single frame and temporal relations across the entire sequence. The fact that the models presented outperform the 81-frame StridedTransformer, as well as the smaller number of parameters, once again demonstrate the effectiveness of exploiting spatial-temporal relationships over purely temporal ones.
The 81-frame MHFormer [Li+22b] outperforms the proposed models by ≥ 0.4 mm under protocol #1 and by ≥ 1.0 mm under protocol #2. The strength of this method lies in the generation of multiple diverse 2D pose hypotheses, which directly addresses the problem of depth ambiguity. However, as MixSTE, MHFormer has the drawback that it requires very high computational costs due to its extensive, purely transformer-based architecture, which consists of more than 5 times as many trainable parameters as the presented methods (19.84 M vs. 3.85 M). Additionally, as PoseFormer and CrossFormer, substantial structural priors of the human skeleton are ignored, which might be another reason why the performance difference between the 81-frame MHFormer and the presented models is comparatively marginal despite a much larger number of parameters.

5.5.3 Discussion

Overall, baseline, data-driven and model-driven approaches were found to produce similar results on average on the test set, with all models showing significant performance differences of up to 30 mm under protocol #1 between more difficult and easier actions. Particularly hard actions are characterized by complex, self-occluded poses due to extensive full body movements and / or low amount of training data, e.g. SittingDown, Sitting and Photo. Comparatively simple actions involve only a few full body movements and little variation between individual frames, e.g. Walking and WalkTogether. Furthermore, it was shown that while the models tend to consider the same actions as difficult or easy, upon closer inspection there are decisive strengths and weaknesses with respect to certain actions. Specifically, the data-driven approach struggles to maintain the baseline's performance when the poses are characterized by rapid, short-lasting upper body movements that strongly vary from frame

to frame (Discussion, Greeting, Purchases, SittingDown), as information about temporally distant frames is of little use under these conditions. Exploiting temporal relationships in a complementary manner is therefore insufficient, illustrating that the strengths of the data-driven approach lie in its occlusion robustness to missing keypoints. Note, however, that for some actions (e.g. Greeting) this was only the case when using CPN-detected keypoints as input. Additionally, using 2D ground truth (GT), there are also actions where the data-driven model outperforms the baseline (Posing and WalkTogether), suggesting that with some adjustments occlusion-augmented training could also have a positive impact on performance on the original data.
In contrast, the model-driven approach performs particularly well for actions that actors spend mainly sitting on a chair (Eating, PhoneTalk, Sitting, Smoking), leveraging its exceptionally strong spatial relationships to root, head and neck, the latter two of which are also beneficial for localizing the wrist in these actions. However, the strong connections to head and neck are a disadvantage when both joints are more affected by occlusion-related noise (e.g. Photo), making their detected 2D keypoints not as reliable as usual, which negatively influences the estimation of the entire 3D pose. Moreover, the reduced focus on typically noisy keypoints (especially knees and feet) negatively affects the performance on actions characterized by dominant lower body movements, causing the model to focus less on the part of the body that is central to determining the pose (e.g. Purchases, SittingDown). This is amplified when a large portion of keypoints are noisy, resulting in only the small portion of reliable keypoints being considered (e.g. SittingDown). Apart from this, however, the model's significantly better generalizability to unseen data (i.e. 2D GT) compared to the baseline (and now also the data-driven model) was again demonstrated.
Furthermore, it was found that the performance of the proposed models is close to the state-of-the-art when considering the same number of input frames, while requiring fewer trainable parameters. The separate distinction between spatial and temporal dependencies is crucial for the models to get by with a small number of trainable parameters while achieving near state-of-the-art performance, as shown by the comparison with StridedTransformer [Li+22a] and MHFormer [Li+22b]. This once again illustrates the advantages of exploiting spatial-temporal relationships over purely temporal ones, which is also reinforced by P-STMO [Sha+22], one of the smallest best performing state-of-the-art models. In addition, P-STMO not only distinguishes between spatial and temporal relationships, but also is trained with a similar strategy as the proposed methods, demonstrating the advantages of transfer learning and fine-tuning. Similar to the data-driven approach, P-STMO also masks parts of the input. However, the model's strong performance indicates that the full potential of the proposed occlusion augmentation strategy is not yet fully exploited.

5.6 Final Discussion

All methods have shown to be able to effectively capture spatial and temporal information, thereby producing similarly satisfactory results on the test dataset. The ability of the self-attention mechanism to capture long-range dependencies was demonstrated by establishing both spatial relationships beyond neighboring second- and third-order nodes and global temporal relationships across distant frames. Additionally, close relationships were also established between naturally connected joints to exploit the structural geometric information of the human skeleton.
At the same time, it has been shown that the baseline method does not fully exploit the temporal receptive field, because little attention is paid to the frames at the beginning and end of a sequence. This manifests itself in a lack of occlusion robustness against incom-

plete input data, i.e. the baseline's performance deteriorates tremendously under additional artificial occlusion. Furthermore, the baseline model was not able to establish a profound ability to handle noisy keypoints particularly well, as it relies heavily on the spatial relationships between joints while not making optimal use of temporal relationships. Indeed, it has been shown that the 3D error per joint is extremely influenced not only by the underlying 2D error of a joint itself, but also by that of spatially closely related joints. While this once again demonstrates the model's ability to comprehensively capture spatial relationships, it also leads to a lack of generalizability to unseen data from different 2D detectors.
The data-driven approach achieves similar good results on the test set as the baseline, showing that training under synthetic occlusion does not affect performance on the original data. Furthermore, it demonstrated extreme improvements over the baseline in terms of robustness to missing keypoints and is able to maintain its performance even under heavy occlusion by compensating for the missing keypoints through the more efficient use of temporal relations. In particular, the temporal receptive field is systematically exploited by capturing long-range temporal relations at different levels in a complementary manner, demonstrating the effectiveness of occlusion-augmented training. Moreover, the established spatial relationships differ only little from those of the baseline (not least due to the use of the same pre-trained model during transfer learning), demonstrating once again the importance of exploiting temporal relationships to overcome occlusion.
The model-driven approach also performs similarly well on the test set as the baseline, however, it was not able to exceed its results, as learning to estimate input noise is limited due to the constraints imposed by the dataset, i.e. imbalanced classes of accurate and inaccurate keypoints as well as a different distribution of noise in train and test set. However, the model was shown to be able to identify inaccurate keypoints, albeit only up to a certain extent. In particular, keypoints that are far from the 2D ground truth are well identified as noisy, whereas the model has difficulty classifying keypoints that are only slightly noisy, especially keypoints that are just on the borderline of being classified as non-noisy. As a result, the model's focus shifted from clearly inaccurate to more accurate keypoints in the input. This was particularly evident at the spatial level, where especially strong relationships were established to joints in the center of the body (head, neck, root), as this is where most keypoints are very accurate. The effects of estimating input noise were also apparent at the temporal level, involving increased temporal consistency and a larger receptive field that extends throughout the entire sequence. In addition, the model alternately amplifies and attenuates frames depending on whether they contain more or less inaccurate keypoints. While this did not improve robustness to noisy input data, it did improve generalizability to unseen data from different 2D detectors.
The comparison between the models revealed that while the models tend to consider the same actions as difficult or easy, each has its own actions on which it performs particularly well or poorly. In particular, the data-driven approach struggles to maintain the baseline's performance when the poses are characterized by rapid, short-lasting upper body movements that strongly vary from frame to frame, as information about temporally distant frames is of little use and thus exploiting temporal relationships in a complementary manner is insufficient. In contrast, the model-driven approach performs particularly well for actions that actors spend mainly sitting on a chair (Eating, PhoneTalk, Sitting, Smoking), leveraging its particularly strong spatial relationships to root, head and neck. However, the model struggles particularly with the action SittingDown, since here the keypoints are strongly affected by noise. This, in turn, is the action where the baseline far outperforms the other two methods.
Furthermore, it was found that the performance of the presented models is close to the current state-of-the-art when considering the same number of input frames, while requir-

ing fewer trainable parameters. It also became clear that not only the distinction between spatial and temporal context contribute to the success of the presented methods, but also pre-training, transfer learning and fine-tuning.
Overall, it is not clear from the analyses conducted which of the methods presented is the best, as each model has its strengths and weaknesses (which go beyond individual actions). However, the major limitation of the model-driven approach is the constraints imposed by the dataset. Thus, without another, more suitable 3D HPE dataset that provides real visibility flags, the approach is least worth exploring, although the model generalizes best to unseen data from different 2D detectors. In contrast, the baseline and the data-driven approaches offer feasible opportunities for improvement that can be easily implemented with existing datasets, allowing for further research.

Chapter 6

Conclusion and Outlook

Finally, this chapter summarizes the main findings of this thesis and identifies opportunities for future research.

6.1 Summary

The focus of this work was to develop an occlusion-robust method for monocular 2D-to-3D single-person human pose estimation (HPE) by exploiting spatial-temporal relationships. To this end, three different methods were implemented that exploit spatial and temporal relationships through the combined use of Graph Convolutional Networks (GCNs) and Transformers. More specifically, GCNs were employed to leverage prior knowledge of the human skeleton through fixed adjacency matrices, while Transformers capture implicit spatial and long-range temporal relationships through the self-attention mechanism. The mathematical concepts of GCNs and Transformers as well as the fundamental challenges of 3D HPE, especially occlusion, were covered in detail beforehand. Furthermore, existing GCN- and Transformer-based state-of-the-art methods as well as methods explicitly addressing the problem of occlusion in 3D HPE were presented.
Of the three methods presented, the first represents a baseline approach on which the other two were built. The second method is a data-driven solution to the occlusion problem that extends the existing baseline with occlusion-based data augmentation at training-time for increased test-time occlusion robustness. In particular, joints are marked as (artificially) occluded by setting their 2D coordinates to 0, forcing the network to rely on other relevant features to predict a correct 3D pose from an incomplete sequence of 2D poses. The third method is a model-driven solution, i.e. it extends the baseline architecturally by employing an auxiliary method aimed at identifying noisy and therefore unreliable keypoints in the input. This is similar to predicting which joints are occluded, as occlusion leads to noisy 2D detections. For this purpose, artificial target labels indicating noise were created based on the distance of the 2D keypoint detections from the 2D ground truth. All three methods were trained using the same 3-phase training procedure consisting of transfer learning, fine-tuning and pose refinement. Subsequently, the training results of each method were analyzed and evaluated in detail, including an individual occlusion robustness analysis to test the effectiveness of each approach. This was followed by a comparative analysis of all methods, both with each other and with the state-of-the-art, and a final discussion highlighting the strengths and weaknesses of each approach.
All methods have shown to be able to effectively capture spatial and temporal information, thereby producing similarly satisfactory results on the test dataset. In particular, both structural geometric information of the human skeleton and implicit spatial relationships beyond

first- and second-order neighbors are captured, as well as long-range temporal relationships across distant frames. However, it has been shown that the baseline method does not make optimal use of temporal relationships, resulting in a lack of occlusion robustness against missing keypoints. It also has not developed a profound ability to handle noisy keypoints particularly well. The data-driven approach, on the other hand, maintains the baseline's performance, while demonstrating extreme improvements over the baseline in terms of robustness to missing keypoints through more efficient use of temporal relations. In particular, the temporal receptive field is systematically exploited by capturing long-range temporal relations at different levels in a complementary manner. At the same time, the model-driven approach was also able to match, but not exceed, the baseline's performance. The model demonstrated that learning to estimate input noise is possible, but only up to a certain extent due to the constraints imposed by the dataset. In particular, keypoints that are far from the 2D ground truth are well identified as noisy, while it is difficult to classify keypoints that are only slightly noisy. As a result, the model's focus shifted from clearly inaccurate to more accurate keypoints in the input, which subsequently affected the temporal relationships as well. While this did not improve robustness to noisy input data, it did improve generalizability to unseen data from different 2D detectors.
The comparison between the methods revealed that each has its strengths and / or weaknesses in estimating poses from certain actions. Moreover, it was found that the performance of the presented models is close to the state-of-the-art when considering the same number of input frames, while requiring fewer trainable parameters.
In summary, based on the analyses performed, it is not clear which of the presented methods is the best. However, the model-driven approach is the least worth exploring without another, more suitable dataset with real visibility flags, while the baseline and the data-driven approaches offer feasible opportunities for improvement, as discussed in more detail in the following section.

6.2 Future Work

Possible future research includes improvements to the methods presented as well as other approaches that address the problem of occlusion.

Improving the baseline architecture could primarily involve adapting the spatial module to make even better use of spatial dependencies so that the subsequent temporal modules can benefit from more meaningful extracted spatial features. Inspired by Zheng et al. [Zhe+22], these adaptations could include a parallel arrangement of graph convolutional blocks with learnable adjacency matrices to simultaneously capture profound implicit spatial relationships at different levels. This would also allow the model to capture spatial relationships beyond first- and second-order neighbors from the beginning (i.e. in the first layer of the module).

There are several options for adapting the occlusion augmentation strategy in the data-driven approach. First, research on a more advanced probability distribution for selecting joints to occlude could continue. Instead of occluding joints with uniform probability, the probability distribution could be based on the MPJPE per joint. Alternatively, joints could be occluded based on the PCKh or the confidence score of their 2D detections. Second, inspired by Shan et al. [Sha+22], the coordinates of occluded joints could be replaced by trainable parameters instead of setting them to 0 to realize synthetic occlusion. While the model is able to maintain meaningful spatial relationships despite the same coordinates for root and occluded

joints (i.e. there is no confusion between these joints), finer differentiation could open up new possibilities in dealing with occlusion.

One of the weaknesses of the model-driven approach is that even if a joint is fully visible, input noise may be present due to the inaccuracy of 2D detectors. Therefore, the artificially created targets for supervising noise predictions do not necessarily coincide with the occlusion of a joint. Ideally, the model will be trained in the future on an alternative dataset to Human3.6M [Ion+14] that contains real visibility labels. Another option would be to train a separate model for occlusion detection on this alternative dataset, which is then used to create visibility flags for Human3.6M. However, since no existing real-world 3D HPE dataset provides visibility labels, further research could alternatively involve improving the strategy for classifying joints as accurate (not noisy) or inaccurate (noisy) to address the problem of imbalanced classes and different distribution of noise in train and test set.

One of the reasons why occlusion is a challenging problem for 3D HPE is that the possible 3D positions of occluded joints are less constrained because their exact 2D positions cannot be captured, amplifying depth ambiguity. Accordingly, another possible approach to solving the occlusion problem would be to address the depth ambiguity problem directly by generating multiple 3D hypothesis representing different possible solutions. Li et al. [Li+22b] already follow a similar approach, namely generating multiple 2D hypotheses. This is essentially equivalent to simulating a multi-view scenario where the target pose is captured from several different views, resulting in multiple different 2D projections. In contrast, generating multiple 3D hypotheses is equivalent to generating multiple possible solutions, all of which are projected onto the same 2D pose sequence.

Appendix A

Appendix

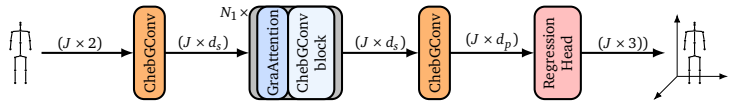


Figure A.1: The architecture of the modified GraFormer used for transfer learning in the baseline and data-driven approaches.

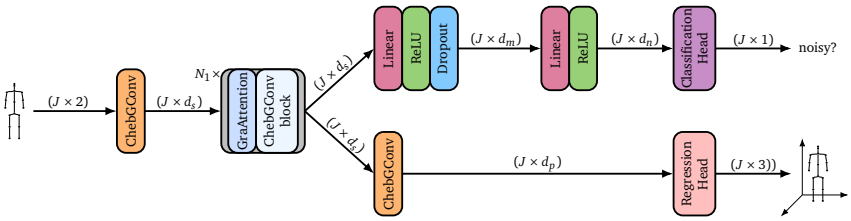


Figure A.2: The architecture of the auxiliary model used for transfer learning in the model-driven approach.

	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	R shoulder	R elbow	R wrist	L shoulder	L elbow	L wrist	Avg.
MPJPE [mm]	47.7	45.5	46.8	46.0	45.6	46.4	46.1	45.7	45.3	45.8	45.9	45.6	45.8	46.2	45.6	45.8	46.4	46.0
ΔMPJPE [mm]	1.3	0.1	1.4	0.6	0.2	1.0	0.7	0.3	-0.1	0.4	0.5	0.2	0.4	0.8	0.2	0.4	1.0	0.6

(a) The subset size q_t is set to 1. The number of occluded frames is set to $q_f = 30$.

	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	R shoulder	R elbow	R wrist	L shoulder	L elbow	L wrist	Avg.
MPJPE [mm]	48.0	45.8	47.1	46.6	45.8	46.8	46.6	45.8	45.5	46.1	46.0	45.8	46.1	46.6	45.8	45.9	46.8	46.3
ΔMPJPE [mm]	2.6	0.4	1.7	1.2	0.4	1.4	1.2	0.4	0.1	0.7	0.6	0.4	0.7	1.2	0.4	0.5	1.4	0.9

(b) The subset size q_t is set to 6. The number of occluded frames is set to $q_f = 30$.

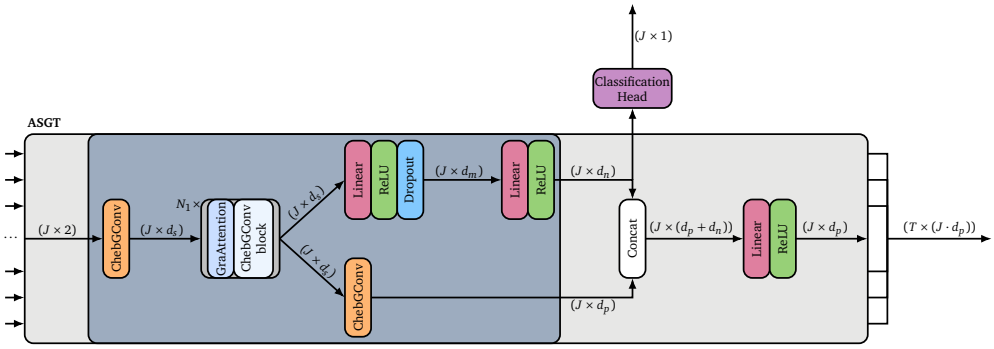
	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	R shoulder	R elbow	R wrist	L shoulder	L elbow	L wrist	Avg.
MPJPE [mm]	48.7	46.1	47.8	47.6	46.2	47.5	47.6	46.2	45.9	46.5	46.4	46.2	46.6	47.3	46.2	46.7	47.6	46.9
ΔMPJPE [mm]	3.3	0.7	2.4	2.2	0.8	2.1	2.2	0.8	0.5	1.1	1.0	0.8	1.2	1.9	0.8	1.3	2.2	1.5

(c) The subset size q_t is set to 10. The number of occluded frames is set to $q_f = 30$.

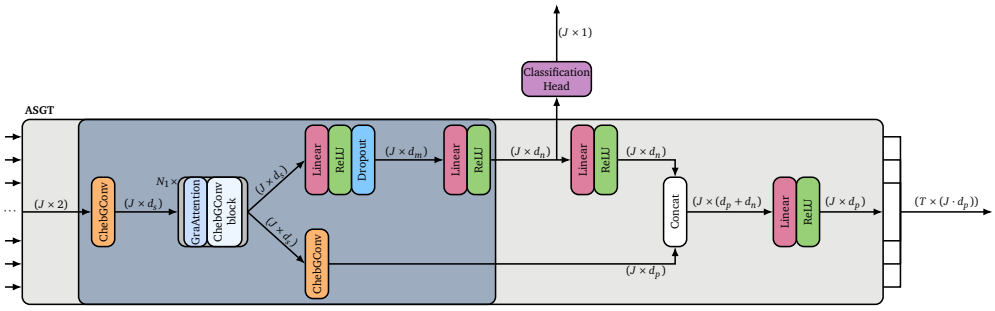
	Root	R hip	R knee	R foot	L hip	L knee	L foot	Spine	Thorax	Neck	Head	R shoulder	R elbow	R wrist	L shoulder	L elbow	L wrist	Avg.
MPJPE [mm]	54.5	49.4	53.3	55.3	49.5	53.0	55.0	48.9	48.8	50.9	49.6	49.7	51.2	52.8	50.1	51.5	53.5	51.6
ΔMPJPE [mm]	9.1	4.0	7.9	9.9	4.1	7.6	9.6	3.5	3.4	5.5	4.2	4.3	5.8	7.4	4.7	6.1	8.1	6.2

(d) The subset size q_t is set to 30. The number of occluded frames is set to $q_f = 30$.

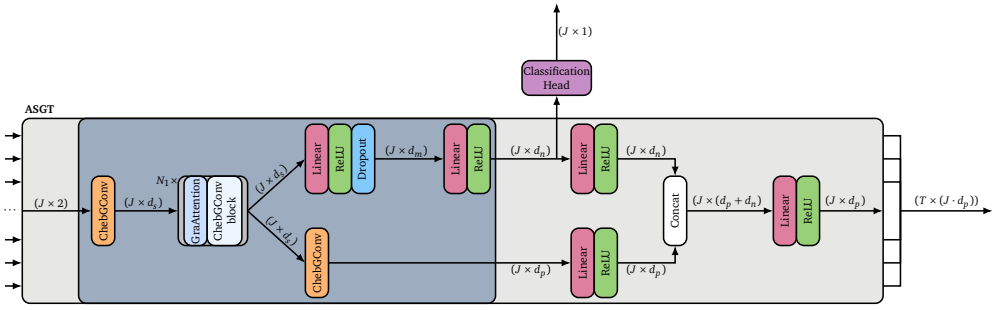
Table A.1: Average MPJPE and error increase of the alternative data-driven approach at different degrees of occlusion of a given joint. During training, occlusion-augmentation was applied with $q_t = 1$ and $p_j = \frac{1}{J}$, i.e. frame were occluded individually rather than in subsets. The model achieves an average MPJPE of $45.4\,\mathrm{mm}$ on the original test set.



(a) The pre-trained pose and noise embeddings are concatenated.



(b) The pre-trained noise embedding is transformed before concatenation.



(c) Both pre-trained pose and noise embeddings are transformed before concatenation.

Figure A.3: The three different design choices for the Auxiliary Spatial Graph-Transformer (ASGT) of the proposed model-driven approach. The pre-trained layer parameters of the auxiliary model are highlighted in darkgray.

Bibliography

[And+14] Andriluka, M., Pishchulin, L., Gehler, P., and Schiele, B. "2D Human Pose Estimation: New Benchmark and State of the Art Analysis". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. June 2014, pp. 3686–3693. DOI: 10.1109/CVPR.2014.471.

[BKH16] Ba, J., Kiros, J., and Hinton, G. "Layer Normalization". In: (July 2016).

[BGK21] Banik, S., Garcia, A., and Knoll, A. "3D Human Pose Regression Using Graph Convolutional Network". In: *2021 IEEE International Conference on Image Processing (ICIP)*. Sept. 2021, pp. 924–928. DOI: 10.1109/ICIP42928.2021.9506736.

[Ben09] Bengio, Y. *Learning Deep Architectures for AI*. Jan. 2009. ISBN: 9781601982957. DOI: 10.1561/9781601982957.

[Bru+13] Bruna, J., Zaremba, W., Szlam, A., and Lecun, Y. "Spectral Networks and Locally Connected Networks on Graphs". In: (Dec. 2013).

[Cai+19] Cai, Y., Ge, L., Liu, J., Cai, J., Cham, T.-J., Yuan, J., and Thalmann, N. "Exploiting Spatial-Temporal Relationships for 3D Pose Estimation via Graph Convolutional Networks". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019, pp. 2272–2281. DOI: 10.1109/ICCV.2019.00236.

[Cao+22] Cao, P., Zhu, Z., Wang, Z., Zhu, Y., and Niu, Q. "Applications of graph convolutional networks in computer vision". In: *Neural Computing and Applications* 34 (Aug. 2022). DOI: 10.1007/s00521-022-07368-1.

[Che+21a] Chen, T., Fang, C., Shen, X., Zhu, Y., Chen, Z., and Luo, J. "Anatomy-Aware 3D Human Pose Estimation With Bone-Based Pose Decomposition". In: *IEEE Transactions on Circuits and Systems for Video Technology* PP (Feb. 2021), pp. 1–1. DOI: 10.1109/TCSVT.2021.3057267.

[Che+18] Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., and Sun, J. "Cascaded Pyramid Network for Multi-person Pose Estimation". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7103–7112. DOI: 10.1109/CVPR.2018.00742.

[Che+21b] Cheng, Y., Wang, B., Yang, B., and Tan, R. T. "Monocular 3D Multi-Person Pose Estimation by Integrating Top-Down and Bottom-Up Networks". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2021, pp. 7645–7655. DOI: 10.1109/CVPR46437.2021.00756.

[Che+20] Cheng, Y., Yang, B., Wang, B., and Tan, R. "3D Human Pose Estimation Using Spatio-Temporal Networks with Explicit Occlusion Training". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34 (Apr. 2020), pp. 10631–10638. DOI: 10.1609/aaai.v34i07.6689.

[Che+19] Cheng, Y., Yang, B., Wang, B., Wending, Y., and Tan, R. "Occlusion-Aware Networks for 3D Human Pose Estimation in Video". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019, pp. 723–732. DOI: 10.1109/ICCV.2019.00081.

[Ci+19] Ci, H., Wang, C., Ma, X., and Wang, Y. "Optimizing Network Structure for 3D Human Pose Estimation". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019, pp. 2262–2271. DOI: 10.1109/ICCV.2019.00235.

[DBV16] Defferrard, M., Bresson, X., and Vandergheynst, P. "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering". In: (June 2016).

[Doo+20] Doosti, B., Naha, S., Mirbagheri, M., and Crandall, D. "HOPE-Net: A Graph-Based Model for Hand-Object Pose Estimation". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020, pp. 6607–6616. DOI: 10.1109/CVPR42600.2020.00664.

[Dos+21] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *International Conference on Learning Representations*. 2021.

[Fuk75] Fukushima, K. "Cognitron: A Self-Organizing Multilayered Neural Network". In: *Biological Cybernetics* 20 (Dec. 1975), pp. 121–36. DOI: 10.1007/BF00342633.

[GM22] Ghafoor, M. and Mahmood, A. "Quantification of Occlusion Handling Capability of 3D Human Pose Estimation Framework". In: *IEEE Transactions on Multimedia* (2022), pp. 1–1. DOI: 10.1109/TMM.2022.3158068.

[Goo56] Good, I. "Some terminology and notation in information theory". In: *Proceedings of the IEE-Part C: Monographs* 103.3 (1956), pp. 200–204.

[GBC16] Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[Gow75] Gower, J. C. "Generalized procrustes analysis". In: *Psychometrika* 40.1 (Mar. 1975), pp. 33–51. ISSN: 0033-3123. DOI: 10.1007/BF02291478. URL: http://link.springer.com/10.1007/BF02291478.

[Han+23] Han, K., Wang, Y., Chen, H., Chen, X., Guo, J., Liu, Z., Tang, Y., Xiao, A., Xu, C., Xu, Y., Yang, Z., Zhang, Y., and Tao, D. "A Survey on Vision Transformer". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.1 (2023), pp. 87–110. DOI: 10.1109/TPAMI.2022.3152247.

[HZ04] Hartley, R. and Zisserman, A. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Mar. 2004. ISBN: 9780511811685. DOI: 10.1017/CBO9780511811685. URL: https://www.cambridge.org/core/product/identifier/9780511811685/type/book.

[Has+22] Hassanin, M., Khamiss, A., Bennamoun, M., Boussaid, F., and Radwan, I. "CrossFormer: Cross Spatio-Temporal Transformer for 3D Human Pose Estimation". In: (Mar. 2022). arXiv: 2203.13387. URL: http://arxiv.org/abs/2203.13387.

[He+17] He, K., Gkioxari, G., Dollár, P., and Girshick, R. "Mask R-CNN". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2980–2988. DOI: 10.1109/ICCV.2017.322.

[He+16] He, K., Zhang, X., Ren, S., and Sun, J. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[Hu+20] Hu, J., Shen, L., Albanie, S., Sun, G., and Wu, E. "Squeeze-and-Excitation Networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42.8 (2020), pp. 2011–2023. DOI: 10.1109/TPAMI.2019.2913372.

[Hu+21] Hu, W., Zhang, C., Zhan, F., Zhang, L., and Wong, T.-T. "Conditional Directed Graph Convolution for 3D Human Pose Estimation". In: *MM 2021 - Proceedings of the 29th ACM International Conference on Multimedia* (2021), pp. 602–611. DOI: 10.1145/3474085.3475219. arXiv: 2107.07797. URL: http://arxiv.org/abs/2107.07797.

[IS15] Ioffe, S. and Szegedy, C. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *32nd International Conference on Machine Learning, ICML 2015*. 2015, pp. 448–456. ISBN: 9781510810587. arXiv: 1502.03167.

[Ion+14] Ionescu, C., Papava, D., Olaru, V., and Sminchisescu, C. "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7 (2014), pp. 1325–1339. DOI: 10.1109/TPAMI.2013.248.

[Isk+19] Iskakov, K., Burkov, E., Lempitsky, V., and Malkov, Y. "Learnable Triangulation of Human Pose". In: (2019), pp. 7717–7726. DOI: 10.1109/ICCV.2019.00781.

[JKK18] J. Reddi, S., Kale, S., and Kumar, S. "On the convergence of Adam & Beyond". In: *6th International Conference on Learning Representations ICLR*. May 2018.

[Ji+20] Ji, X., Fang, Q., Dong, J., Shuai, Q., Jiang, W., and Zhou, X. "A survey on monocular 3D human pose estimation". In: *Virtual Reality & Intelligent Hardware* 2.6 (2020), pp. 471–500. ISSN: 2096-5796. DOI: https://doi.org/10.1016/j.vrih.2020.04.005. URL: https://www.sciencedirect.com/science/article/pii/S2096579620300887.

[Kha+22] Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F., and Shah, M. "Transformers in Vision: A Survey". In: *ACM Computing Surveys* 54 (Jan. 2022). DOI: 10.1145/3505244.

[KB15] Kingma, D. P. and Ba, J. "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. 2015.

[KW17] Kipf, T. N. and Welling, M. "Semi-Supervised Classification with Graph Convolutional Networks". In: *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*. 2017.

[LB98] LeCun, Y. and Bengio, Y. "Convolutional Networks for Images, Speech, and Time Series". In: *The Handbook of Brain Theory and Neural Networks*. Cambridge, MA, USA: MIT Press, 1998, pp. 255–258. ISBN: 0262511029.

[LBH15] LeCun, Y., Bengio, Y., and Hinton, G. "Deep Learning". In: *Nature* 521 (May 2015), pp. 436–444. DOI: 10.1038/nature14539. URL: http://www.nature.com/articles/nature14539.

[LP08] Lek, S. and Park, Y. "Artificial Neural Networks". In: *Encyclopedia of Ecology*. Dec. 2008, pp. 237–245. DOI: 10.1016/B978-008045405-4.00173-7.

[LL19] Li, C. and Lee, G. H. "Generating Multiple Hypotheses for 3D Human Pose Estimation With Mixture Density Network". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 9879–9887. DOI: 10.1109/CVPR.2019.01012.

[Li+22a] Li, W., Liu, H., Ding, R., Liu, M., Wang, P., and Yang, W. “Exploiting Temporal Contexts with Strided Transformer for 3D Human Pose Estimation”. In: *IEEE Transactions on Multimedia* (2022), pp. 1–1. DOI: 10.1109/TMM.2022.3141231.

[Li+22b] Li, W., Liu, H., Tang, H., Wang, P., and Van Gool, L. “MHFormer: Multi-Hypothesis Transformer for 3D Human Pose Estimation”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 13137–13146. DOI: 10.1109/CVPR52688.2022.01280.

[Li+19] Li, X., Fan, Z., Liu, Y., Li, Y., and Dai, Q. “3D Pose Detection of Closely Interactive Humans Using Multi-View Cameras”. In: *Sensors* 19 (June 2019), p. 2831. DOI: 10.3390/s19122831.

[Liu+20a] Liu, K., Ding, R., Zou, Z., Wang, L., and Tang, W. “A Comprehensive Study of Weight Sharing in Graph Networks for 3D Human Pose Estimation”. In: Nov. 2020, pp. 318–334. ISBN: 978-3-030-58606-5. DOI: 10.1007/978-3-030-58607-2_19.

[Liu+20b] Liu, R., Shen, J., Wang, H., Chen, C., Cheung, S.-c., and Asari, V. “Attention Mechanism Exploits Temporal Contexts: Real-Time 3D Human Pose Reconstruction”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 5063–5072. DOI: 10.1109/CVPR42600.2020.00511.

[Liu+22] Liu, W., Bao, Q., Sun, Y., and Mei, T. “Recent Advances in Monocular 2D and 3D Human Pose Estimation: A Deep Learning Perspective”. In: *ACM Computing Surveys* 55.4 (Nov. 2022). ISSN: 0360-0300. DOI: 10.1145/3524497. URL: https://doi.org/10.1145/3524497.

[Liu+21] Liu, Y., Zhang, Y., Wang, Y., Hou, F., Yuan, J., Tian, J., Zhang, Y., Shi, Z., Fan, J., and He, Z. *A Survey of Visual Transformers*. 2021. DOI: 10.48550/ARXIV.2111.06091. URL: https://arxiv.org/abs/2111.06091.

[Lut+22] Lutz, S., Blythman, R., Ghosal, K., Moynihan, M., Simms, C., and Smolic, A. “Jointformer: Single-Frame Lifting Transformer with Error Prediction and Refinement for 3D Human Pose Estimation”. In: *2022 26th International Conference on Pattern Recognition (ICPR)*. 2022, pp. 1156–1163. DOI: 10.1109/ICPR56361.2022.9956366.

[Mal09] Mallat, S. *A Wavelet Tour of Signal Processing*. Third Edition. Academic Press, Jan. 2009. ISBN: 978-0-12-374370-1. DOI: 10.1016/B978-0-12-374370-1.X0001-8.

[NYD16] Newell, A., Yang, K., and Deng, J. “Stacked Hourglass Networks for Human Pose Estimation”. In: *European Conference on Computer Vision (ECCV)*. Vol. 9912. Oct. 2016, pp. 483–499. ISBN: 978-3-319-46483-1. DOI: 10.1007/978-3-319-46484-8_29.

[PNS21] Panigrahi, S., Nanda, A., and Swarnkar, T. “A Survey on Transfer Learning”. In: *Intelligent and Cloud Computing*. Jan. 2021, pp. 781–789. ISBN: 978-981-15-5970-9. DOI: 10.1007/978-981-15-5971-6_83.

[Pas+19] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[PG17] Patterson, J. and Gibson, A. *Deep Learning: A Practitioner's Approach*. 1st. O'Reilly Media, Inc., 2017. ISBN: 1491914254.

[Pav+19] Pavllo, D., Feichtenhofer, C., Grangier, D., and Auli, M. "3D Human Pose Estimation in Video With Temporal Convolutions and Semi-Supervised Training". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 7745–7754. DOI: 10.1109/CVPR.2019.00794.

[Qia99] Qian, N. "On the momentum term in gradient descent learning algorithms". In: *Neural Networks* 12.1 (Feb. 1999), pp. 145–151. ISSN: 08936080. DOI: 10.1016/S0893-6080(98)00116-6.

[RGL15] Rafi, U., Gall, J., and Leibe, B. "A semantic occlusion model for human pose estimation from a single depth image". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2015, pp. 67–74. DOI: 10.1109/CVPRW.2015.7301338.

[Rud16] Ruder, S. "An overview of gradient descent optimization algorithms". In: *ArXiv* (Sept. 2016).

[RN09] Russell, S. and Norvig, P. *Artificial Intelligence: A Modern Approach*. 3rd. USA: Prentice Hall Press, 2009. ISBN: 0136042597.

[Sár+18] Sárándi, I., Linder, T., Arras, K. O., and Leibe, B. "How Robust is 3D Human Pose Estimation to Occlusion?" In: *IROS Workshop - Robotic Co-workers 4.0*. 2018.

[SB12] Schulz, H. and Behnke, S. "Deep Learning". In: *KI - Künstliche Intelligenz* 26.4 (Nov. 2012), pp. 357–363. DOI: 10.1007/s13218-012-0198-z.

[Sha+22] Shan, W., Liu, Z., Zhang, X., Wang, S., Ma, S., and Gao, W. "P-STMO: Pre-Trained Spatial Temporal Many-to-One Model for 3D Human Pose Estimation". In: *17th European Conference of Computer Vision (ECCV)*. 2022, pp. 461–478. ISBN: 978-3-031-20064-9. DOI: 10.1007/978-3-031-20065-6_27. URL: https://doi.org/10.1007/978-3-031-20065-6_27.

[SBB10] Sigal, L., Balan, A. O., and Black, M. J. "HumanEva: Synchronized Video and Motion Capture Dataset and Baseline Algorithm for Evaluation of Articulated Human Motion". In: *International Journal of Computer Vision* 87 (Mar. 2010), pp. 4–27. ISSN: 0920-5691. DOI: 10.1007/s11263-009-0273-6. URL: http://link.springer.com/10.1007/s11263-009-0273-6.

[Sri+14] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

[TK21] Thakur, A. and Konde, A. "Fundamentals of Neural Networks". In: *International Journal for Research in Applied Science and Engineering Technology* 9 (Aug. 2021), pp. 407–426. ISSN: 23219653. DOI: 10.22214/ijraset.2021.37362.

[TH12] Tieleman, T. and Hinton, G. "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude". In: *COURSERA: Neural networks for machine learning* 4.2 (2012), pp. 26–31.

[TV98] Trucco, E. and Verri, A. *Introductiory Techniques for 3-D Computer Vision*. 1998.

[Tsa87] Tsai, R. Y. "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses". In: *IEEE Journal of Robotics and Automation* RA-3.4 (1987), pp. 323–344.

[Vas+17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. "Attention is All You Need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Curran Associates Inc., 2017, pp. 6000–6010. ISBN: 9781510860964.

[Wan+21a] Wang, J., Tan, S., Zhen, X., Xu, S., Zheng, F., He, Z., and Shao, L. "Deep 3D Human Pose Estimation: A Review". In: *Computer Vision and Image Understanding* 210 (Sept. 2021). ISSN: 1077-3142. DOI: 10.1016/j.cviu.2021.103225. URL: https://doi.org/10.1016/j.cviu.2021.103225.

[Wan+21b] Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., Liu, W., and Xiao, B. "Deep High-Resolution Representation Learning for Visual Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.10 (2021), pp. 3349–3364. DOI: 10.1109/TPAMI.2020.2983686.

[Wan+18] Wang, X., Girshick, R., Gupta, A., and He, K. "Non-local Neural Networks". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7794–7803. DOI: 10.1109/CVPR.2018.00813.

[Wu+21] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. "A Comprehensive Survey on Graph Neural Networks". In: *IEEE Transactions on Neural Networks and Learning Systems* 32.1 (2021), pp. 4–24. DOI: 10.1109/TNNLS.2020.2978386.

[XT21] Xu, T. and Takano, W. "Graph Stacked Hourglass Networks for 3D Human Pose Estimation". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 16100–16109. DOI: 10.1109/CVPR46437.2021.01584.

[YTH20] Yucheng, C., Tian, Y., and He, M. "Monocular human pose estimation: A survey of deep learning-based methods". In: *Computer Vision and Image Understanding* 192 (Mar. 2020), p. 102897. DOI: 10.1016/j.cviu.2019.102897.

[Zen+21] Zeng, A., Sun, X., Yang, L., Zhao, N., Liu, M., and Xu, Q. "Learning Skeletal Graph Neural Networks for Hard 3D Pose Estimation". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 11416–11425. DOI: 10.1109/ICCV48922.2021.01124.

[Zha+22] Zhang, J., Tu, Z., Yang, J., Chen, Y., and Yuan, J. "MixSTE : Seq2seq Mixed Spatio-Temporal Encoder for 3D Human Pose Estimation in Video". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 13222–13232. DOI: 10.1109/CVPR52688.2022.01288.

[Zha+19a] Zhang, S., Tong, H., Xu, J., and Maciejewski, R. "Graph convolutional networks: a comprehensive review". In: *Computational Social Networks* 6 (Nov. 2019). DOI: 10.1186/s40649-019-0069-y.

[Zha+19b] Zhao, L., Peng, X., Tian, Y., Kapadia, M., and Metaxas, D. N. "Semantic Graph Convolutional Networks for 3D Human Pose Regression". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019, pp. 3425–3435. DOI: 10.1109/CVPR.2019.00354.

[Zha+21] Zhao, W., Tian, Y., Ye, Q., Jiao, J., and Wang, W. *GraFormer: Graph Convolution Transformer for 3D Pose Estimation*. 2021. DOI: 10.48550/ARXIV.2109.08364. URL: https://arxiv.org/abs/2109.08364.

[Zhe+22] Zheng, C., Mendieta, M., Wang, P., Lu, A., and Chen, C. "A Lightweight Graph Transformer Network for Human Mesh Reconstruction from 2D Human Pose". In: *Proceedings of the 30th ACM International Conference on Multimedia*. MM'22. Association for Computing Machinery, 2022, pp. 5496–5507. DOI: 10.1145/3503161.3547844. URL: https://doi.org/10.1145/3503161.3547844.

[Zhe+20] Zheng, C., Wu, W., Chen, C., Yang, T., Zhu, S., Shen, J., Kehtarnavaz, N., and Shah, M. *Deep Learning-Based Human Pose Estimation: A Survey*. 2020. DOI: 10.48550/ARXIV.2012.13392. URL: https://arxiv.org/abs/2012.13392.

[Zhe+21] Zheng, C., Zhu, S., Mendieta, M., Yang, T., Chen, C., and Ding, Z. "3D Human Pose Estimation with Spatial and Temporal Transformers". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 11636–11645. DOI: 10.1109/ICCV48922.2021.01145.

[Zho+20] Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. "Graph neural networks: A review of methods and applications". In: *AI Open* 1 (2020), pp. 57–81. ISSN: 2666-6510. DOI: https://doi.org/10.1016/j.aiopen.2021.01.001. URL: https://www.sciencedirect.com/science/article/pii/S2666651021000012.

[Zhu+21] Zhu, Y., Xu, X., Shen, F., Ji, Y., Gao, L., and Shen, H. T. "PoseGTAC: Graph Transformer Encoder-Decoder with Atrous Convolution for 3D Human Pose Estimation". In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Aug. 2021, pp. 1359–1365. DOI: 10.24963/ijcai.2021/188.