electronics

MDPI

Article

Graph U-Shaped Network with Mapping-Aware Local Enhancement for Single-Frame 3D Human Pose Estimation

Bing Yu *, Yan Huang, Guang Cheng, Dongjin Huang and Youdong Ding

Shanghai Film Academy, Shanghai University, Shanghai 200072, China; faaally1950@shu.edu.cn (Y.H.); guang_cheng@shu.edu.cn (G.C.); djhuang@shu.edu.cn (D.H.); ydding@shu.edu.cn (Y.D.)
* Correspondence: yubing@shu.edu.cn

Abstract: The development of 2D-to-3D approaches for 3D monocular single-frame human pose estimation faces challenges related to noisy input and failure to capture long-range joint correlations, leading to unreasonable predictions. To this end, we propose a straightforward, but effective U-shaped network called the mapping-aware U-shaped graph convolutional network (M-UGCN) for single-frame applications. This network applies skeletal pooling/unpooling operations to expand the limited convolutional receptive field. For noisy inputs, as local nodes have direct access to the subtle discrepancies between poses, we define an additional mapping-aware local-enhancement mechanism to focus on local node interactions across multiple scales. We evaluated our proposed method on the benchmark datasets Human3.6M and MPI-INF-3DHP, and the experimental results demonstrated the robustness of the M-UGCN against noisy inputs. Notably, the average error in the proposed method was found to be 4.1% lower when compared to state-of-the-art methods adopting similar multi-scale learning approaches.

Keywords: 3D human pose estimation; graph convolutional network; multi-scale feature fusion

Citation: Yu, B.; Huang, Y.; Cheng, G.; Huang, D.; Ding, Y. Graph U-Shaped Network with Mapping-Aware Local Enhancement for Single-Frame 3D Human Pose Estimation. *Electronics* **2023**, *12*, 4120. https://doi.org/10.3390/electronics12194120

Academic Editor: Javid Taheri

Received: 19 August 2023
Revised: 6 September 2023
Accepted: 7 September 2023
Published: 2 October 2023

1. Introduction

Three-dimensional monocular human pose estimation [1–21] involves the extraction of human poses from RGB images and plays a vital role in diverse domains. It assists in behavior recognition for surveillance and healthcare, captures motion in virtual or augmented environments, and finds utility in sports analysis, robotics, and biomechanics. The method proposed in this paper falls into the category of typical 2D-to-3D lifting problems [1–17], decomposing the 2D and 3D pose estimation tasks into a two-stage process. We focused on the second stage: lifting the 2D keypoints obtained by detectors to 3D positions in the view space. In this way, compared with the use of image-based approaches [19–21] to regress a parametric model's shape and pose parameters directly from RGB images, the actual overhead on the 3D pose estimation task is reduced and the feature domain gap is alleviated. In other words, two-stage estimation introduces a compromise between performance and consumption.

However, recovering the 3D joint positions from a 2D keypoints is an ill-posed problem. The fact that different poses may have the same 2D projection results has motivated researchers to exclude most error cases by establishing a priori constraints for humans. Graph convolutional networks (GCNs) [1–15] are competitive approaches to exploit joint correlations. They extend the capabilities of traditional CNNs [22] by conducting convolution operations on non-Euclidean graph-structured data.

In 2D-to-3D GCN-based pose estimation tasks, human joints are represented by nodes, and bones are regarded as edges. Bones connect joints to their associated parents, shaping a sparse topology that can be expressed as an adjacency matrix. Nodes interact depending on the adjacency matrix. The adjacency matrix can only depict the actual physical node correlation. As we all know, when a person is moving, apparent relationships exist between

non-local parts, such as legs and arms. Therefore, to make nodes link to others that are non-physically connected (shown in Figure 1a), some works [5,12,13] utilize high-order graph representation, while others [3,9] use learnable edges to reconstruct the adjacency matrix to an affinity matrix. However, the GCN-based methods [1–15] successfully obtain joint correlation, but are highly dependent on 2D keypoint detectors and will still suffer instability from 2D inputs, worsening their adaptiveness to depth ambiguity. To break through this bottleneck, most GCN-based works turn to exploiting temporal information. Works [6,8] that use short-sequence inputs still suffer from unstable 2D data because of the great pose similarity with short consecutive frames, while works [14,15] that use lengthy sequences restrict practical application. Therefore, our work focused on how to enhance the robustness against noisy inputs in single-frame cases.

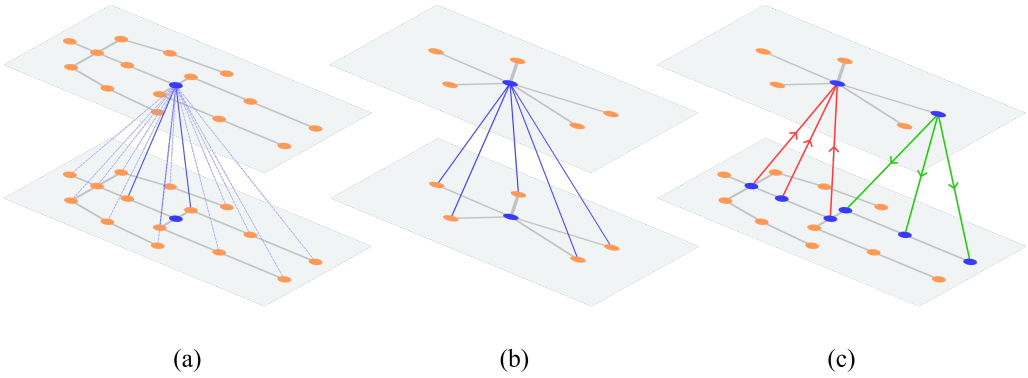


Figure 1. Graph-structured node interaction: (**a**) Node links to the other non-direct neighbor in a feature map; (**b**) down-sampled node interacts with its direct neighbor in a feature map; (**c**) node interacts across two different scaled feature maps through directed mapping edges. These edges represent the node pooling (red arrows) or unpooling (green arrows) direction, guiding cross-scale information exchange.

However, the graph representation such as high-order or affinity matrices is static, which means that the model can only learn in the current step by one fixed modality of the node relationship. We attempted to apply the temporal-based dynamic affinity matrix when noisy inputs are given. As shown in Figure 2a, the robustness against noise improves when the initial affinity is influenced by pre-embedded temporal information. Previous work [23] further confirmed our ideas and showed the same decreasing tendency for prediction errors as more-adjacent frames are taken into the network. As illustrated in Figure 2b, the affinity visualization presents the weak relationships between non-local parts, such as legs and arms (marked as yellow rectangles). In contrast, using a single frame provides little information on which non-local nodes should be given more attention, thus limiting the receptive field for convolution.

Therefore, we designed a new strategy to cope with the invariance that frustrates non-local interaction. Although a node is allowed to collect features from multi-hop neighbors as the network becomes deeper, it has limited learning ability for non-local nodes, reducing the non-local interaction efficiency. The strategy first adopts pooling/unpooling operations to force the model to capture information between non-local nodes directly. Following the idea of multi-scale learning based on graph representation, we adopted a framework based on Graph U-Nets [24], but used skeletal pooling/unpooling [1] layers instead of gPool/gUnpool layers in order to maintain the basic body topology. The framework first samples gradual sub-structures of graph representations to enable high-level feature encoding and receptive field enlargement in the down-scaling stage, then progressively recovers the original scale in the up-scaling stage and decodes the features. The skeletal pooling/unpooling layers introduce multi-scale pose features. At lower scales, feature transformations enable interaction between non-local nodes. At higher scales, feature transformations help communications between nodes and their direct neighbors. In this

process, the non-local relationship is considered, but the local node is ignored as the sub-optimal features are lost in the down-scaling stage to pool and the up-scaling stage to unpool.

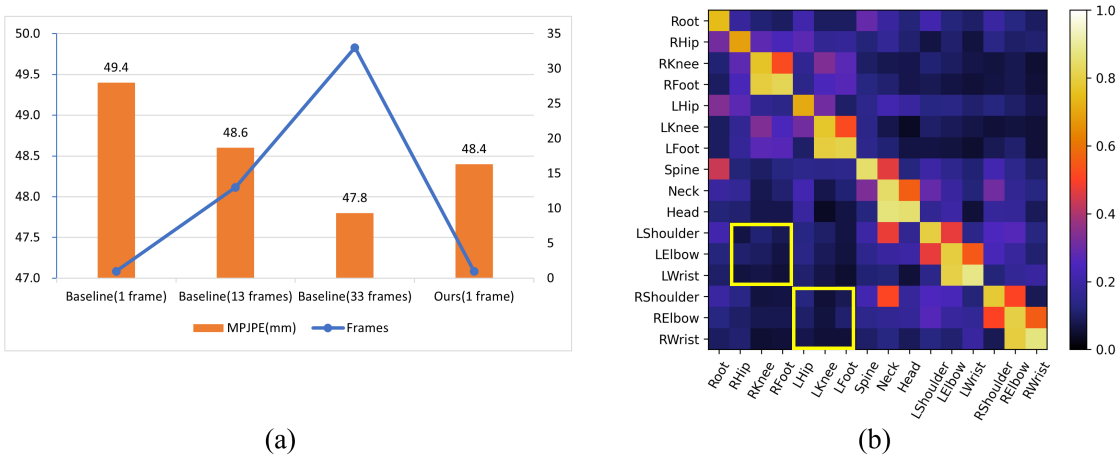


Figure 2. A sequential residual architecture composed of graph convolutional blocks [3] is considered as a baseline for the comparison of static and dynamic affinity: (**a**) The diagram indicates that the mean per-joint position error (MPJPE) decreases significantly with the dynamic affinity matrix inferred through the use of a longer temporal sequence. Our method, while not using temporal information, alleviates the limitations associated with static affinity. All methods were tested on the Human3.6M dataset [25] with noisy inputs. (**b**) Static affinity visualization showing the weak relationships between non-local parts such as legs and arms (marked as yellow rectangles).

In terms of the above considerations, we designed a cross-scale interaction stage between the down-scaling and up-scaling stages. This stage contains several repeated mapping-aware fusion modules to achieve mapping-aware node interactions. In each module, nodes across different scales are connected through a particular type of edge, which we call directed mapping edges. As shown in Figure 1c, the multi-scale feature maps are linked by the mapping edges. These mapping edges simulate the directed mapping relationships and help to develop a novel learning-based sparse node update block, that is the mapping-aware node assignment block (NAB). The NABs pass features along the mapping edges in a node-to-node manner when updating graph nodes. In this way, the NABs allow for communication between the pooled nodes that carry non-local information and the unpooled nodes that retain detailed information from their direct neighbors, thus amplifying subtle differences based on poses. The resulting features are mixed with the initial features in a channelwise manner. Subsequent experiments demonstrated that amplifying the discrepancy among nodes allows the network to learn particular node correlations under different poses, thus enabling more-stable prediction results under noisy inputs.

In summary, our work makes three main contributions:

- We propose a mapping-aware U-shaped graph convolution network (M-UGCN) for 2D-to-3D human pose estimation based on Graph U-Nets [24], and apply skeletal pooling/unpooling operations [1] to perform multi-scale learning. This network, in cooperation with an existing convolutional receptive field expansion strategy (e.g., the affinity matrix), can further enhance the non-local information.
- We propose a cross-scale interaction stage in M-UGCN that bridges the up-scaling and down-scaling stages. In this stage, we establish directed mapping edges to simulate the node pooling and unpooling. Along the mapping edge, information is exchanged across multiple scales, helping the model learn detailed differences among poses. To the best of our knowledge, this is the first attempt to introduce directed graph representation to describe the mapping relationships among nodes across scales.

- The method proposed in this work outperformed most single-frame estimation approaches on two large-scale 3D pose datasets: Human3.6M [25] and MPI-INF-3DHP [26].

In the subsequent sections, we delve deeper into specific aspects of our research. In Section 2, we provide an overview of related works about GCN-based 3D human pose estimation. Section 3 presents the method and details the mapping-aware local enhancement. In Section 4, we implement an ablation study to illustrate the effectiveness of our method. We also constructed a performance comparison with the previous works and provide some visualizations. Finally, the paper ends with the conclusions and a discussion of promising directions.

2. Related Work

2.1. Three-Dimensional Human Pose Estimation

Monocular 3D human-pose-estimation methods can be divided into two categories: model-based [19–21] and skeleton-based [1–17] approaches. The former incorporate a parametric body model such as SMPL [27] to generate a skinned human mesh that is learned using cues from RGB images. The latter approaches are quite different, as they only output sparse 3D joint keypoints that are sufficient to describe human poses. In this type of approach, some works [16,17] have directly inferred 3D poses from images without the use of intermediate representation (e.g., heatmaps). All of the methods mentioned above can be categorized as image-based tasks that suffer from the domain gap, as discrepancies always exist between in-the-wild and indoor images. Motivated by the development of 2D human-pose-estimation methods [28–31], 2D-to-3D lifting methods [1–17] employ a two-stage strategy. In the first stage, a 2D pose detector provides 2D poses, which are used in the second stage to obtain 3D poses. These 2D-to-3D lifting methods bridge the gap between images obtained in various scenario settings, as the scale of 2D image paired datasets is much larger than that of 3D image datasets. Our work falls into the field of 2D-to-3D lifting problems. Given that a human pose can be described topologically, graph convolutional networks (GCNs) play an important role in our work, where the bones are represented by edges and the joints are represented by nodes.

2.2. GCN-Based Works

GCN-based approaches have a strong capacity for feature extraction in the non-Euclidean case, which can be divided into spectral-based [32,33] and spatial-based approaches [34–36]. Spectral-based approaches utilize Chebyshev polynomials or rational polynomials to parameterize the convolutional kernel applied to the frequency domain graph signal. Our work pertains to spatial-based approaches, which instead define convolution operations on the spatial relationships between nodes and their neighbors. In addition, the graph representation itself can be divided into two types, depending on the connection mode between graph nodes.

In an undirected graph , as the primary choice in most works, edges represent bidirectional connections between joints. GCN-based works using undirected graph representations face two main challenges: the weight-sharing problem and the limitation of the receptive field. Considering the first issue, Liu et al. [10] employed a weight matrix for each joint at the cost of a large number of parameters, Zhao et al. [7] extended the adjacency matrix to be channel-specific. Zou and Tang [3] designed a modulation vector for each node used in the feature transformation matrix. As for the second issue, the convolution filter should be re-defined to take the non-direct neighbor nodes into account, which can be divided into two solutions. The first solution involves generating a high-order graph representation, either explicitly or implicitly. Cai et al. [8] proposed various ways to define one-hop neighbor classifications based on human semantic information. Zou et al. [5] applied higher-order polynomials of the graph adjacency matrix to the original convolution operation to obtain multi-hop convolution results and added these results for a larger receptive field. Quan and Hamza [12] also introduced a higher-order graph convolutional framework that concatenates feature representations from multi-hop neighborhoods to

capture long-range dependencies between body joints. Zhou et al. [2] and Zhao and Wang [4] substituted Chebyshev polynomials for the traditional convolutional kernel to obtain an implicit higher-order structure representation. The second solution is to learn a free matrix to define node relationships in the vertex domain [3,9,13], which can reflect the relationships between long-range nodes.

Directed graph representations model the hierarchy of joints and bones. Cheng et al. [14] employed directed-graph-structured data weighted by 2D confidence scores to mitigate noise propagation, and Hu et al. [15] used conditional directed graph convolution to enhance the dependence on non-local nodes. These methods enable cooperation between directed graph convolution and temporal information. The time axis is treated as a distinct dimension, and the associated need for lengthy temporal sequences limits the practical application of such methods. However, directed graph representations allow for an effective interpretation of the directional relationships between nodes. Inspired by this, we do not use such a representation to model the human skeleton, but, instead, employ it as a solution to encode the mapping relationships between nodes across multiple scales.

2.3. Multi-Scale Learning

Many image-based tasks, such as object detection, image classification, and image enhancement, take advantage of multi-scale feature learning. As image down-sampling can expand the receptive field while reducing the model scale, multi-scale learning approaches [37–40] have become prevalent.

Multi-scale learning applied to graph-structured data is a novel idea for 3D human estimation. Graph-structured data can be treated as analogous to image data, but as the graph structure has stronger irregularity, traditional pooling/unpooling operations were considered unsuitable in our case. Gao and Ji [24] developed an encoder–decoder model called Graph U-Nets. The graph pooling operation that they proposed can adaptively select some nodes to form lower-scale graph-structured data, while the graph unpooling operation restores the lower-scale data to higher-scale data. For monocular 3D human pose estimation, given that the topological graphs expressing the skeleton are very sparse, Li et al. [11] proposed a new model that contains three parallel sparse-to-fine graph representations of poses to achieve multi-scale learning. Cai et al. [8] and Xu and Takano [1] re-defined the solution for graph down-sampling to maintain the basic shape of the human body. We extended the ideas of the above-mentioned researchers to design pre-defined sub-graphs of the human body and propose a local-enhancement strategy to raise the upper bound in single-frame monocular human pose estimation.

3. Method

Given the 2D keypoints $\mathrm{J}_{2\mathrm{D}} \in \mathbb{R}^{N\times 2}$ in the pixel space, the aim here was to obtain 3D joint positions $\mathrm{J}_{3\mathrm{D}} \in \mathbb{R}^{N\times 3}$ in the view space. To construct a model for this purpose, we propose a novel GCN-based framework, which is a U-shaped structure. As shown in Figure 3, the proposed mapping-aware U-shaped graph convolutional network (M-UGCN) consists of four stages: the down-scaling, cross-scale interaction, up-scaling, and output stages.

3.1. Preliminaries for GCN

The human skeletal topology can be defined as a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ represents the joint nodes and $\mathcal{E}$ represents the edges (bones) linking joints. The bones can be interpreted as an adjacency matrix $\mathrm{A} \in \{0,1\}^{N\times N}$, where N denotes the number of joints. The index (i, j) of an element in A denotes the two sides of a bone linking nodes i and j. If node j is not directly connected to node i, the element of A corresponding to the index (i, j) is set to zero; otherwise, if node j is directly connected to node i, the associated element is set to one.

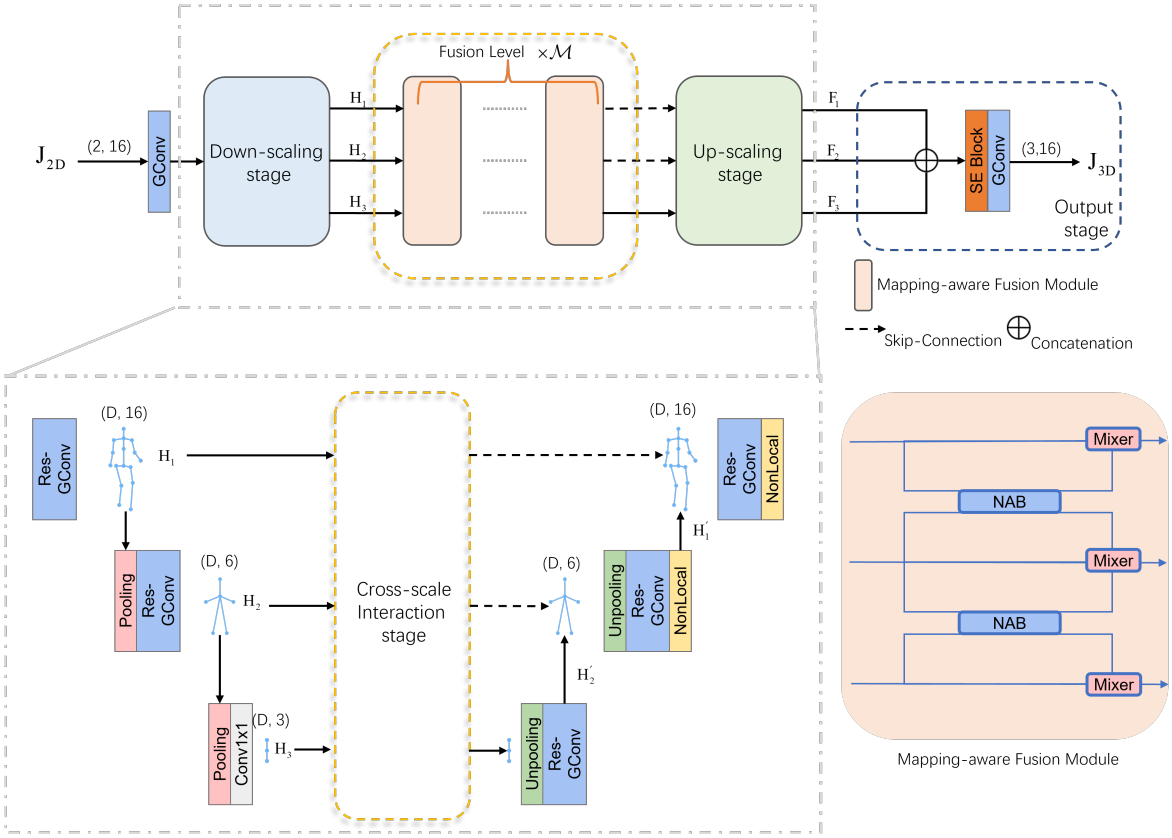


Figure 3. Overview of the proposed M-UGCN framework. The structure is roughly divided into four stages: the down-scaling, cross-scale interaction, up-scaling, and output stages. The cross-scale interaction stage, with $\mathcal{M}$-repeated mapping-aware fusion modules, acts as a bridge between the down-scaling and up-scaling stages. The node-assignment blocks (NABs) and mixer blocks (Mixers) are critical components of this stage. The output stage performs the final multi-level feature fusion and outputs the predictions.

After adding the identity matrix, A becomes $\tilde{A}$ containing self-connection edges. Each node can be described as a D-dimensional feature vector $x_i \in \mathbb{R}^D, \forall i \in [0, N-1]$. The features of all nodes can be written as $X \in \mathbb{R}^{D \times N}$. A graph convolutional (GConv) layer [35] updates the node features according to the following equation:

$$X' = \sigma(W X \hat{A}), \tag{1}$$

where $\sigma(.)$ is the activation function, $W \in \mathbb{R}^{D' \times D}$ is the learnable weight matrix for feature transformation, and $\hat{A}$ is a symmetrically normalized matrix of $\tilde{A}$. This vanilla GCN shares one weight matrix W, which limits its ability to capture diverse spatial relationships.

We applied modulated graph convolution [3], which creates a free modulation vector for each node, to transform the sharing weight matrix W. In this way, Equation (1) can be changed to:

$$H' = \sigma((M \odot (WH))\hat{A}), \tag{2}$$

where $M \in \mathbb{R}^{D' \times D}$ is the collection of modulation vectors m_i and the symbol $\odot$ denotes elementwise multiplication. In this work, ReLU [41] was used as the activation function for training.

In addition, the initial adjacency matrix A can be extended to an affinity matrix. Equation (3) details the generation of the affinity matrix:

$$A \leftarrow A + O. \tag{3}$$

Specifically, a learnable mask $O \in \mathbb{R}^{N \times N}$ is added to the adjacency matrix A, which changes the weights of edges to link non-local nodes. Although this operation helps to expand the receptive field for convolution, O in each GConv layer is fixed once training has been completed, giving non-local nodes less chance to interact. Therefore, we propose a novel model based on multi-scale learning. We shortened the hop distance between non-local nodes by sampling down-scaled graph-structured nodes. Furthermore, we applied mapping-aware local enhancement in this network to remedy the negative effects of information loss and the static node correlation modality.

3.2. Network Architecture

As shown in Figure 3, we first applied a graph-embedding layer to convert 2D keypoints into a high-dimensional graph-based representation. The down-scaling stage gradually down-samples these graph data, allowing for the expansion of the receptive field, while the up-scaling stage gradually restores the low-scale graph data. The additional part—that is, the cross-scale interaction stage—between these two stages implements cross-scale feature fusion through several stacked mapping-aware fusion modules. We focus on the module at this stage in Section 3.3. Finally, high-level features from groups obtained in the up-scaling stage are fed into the output stage for the final decoding.

3.2.1. Pooling and Unpooling

In this part, we introduce the pooling and unpooling layers. The pooling layer performs a skeletal pooling operation [1], the same as the process shown in Figure 4b, allowing for a more-compact physical structure. Unlike general graph pooling in Figure 4a, this operation maintains the physical connections between nodes. The original graph nodes will be cut into several parts, where each is reconnected after a maximum pooling operation.

The unpooling layer performs a skeletal unpooling operation [1] to restore the pooled nodes, the same as the process shown in Figure 5b. The general graph unpooling operation in Figure 5a sets nodes discarded previously to null, while the skeletal unpooling operation duplicates the pooled nodes to the corresponding vacant location.

The skeletal pooling/unpooling operation depends on the body joints' pre-defined structure and sub-structures. The design is shown in Figure 6, which has three configurations. We divided the original S_1-structured nodes into six parts (Head, Body, Left/Right Arm, and Left/Right Leg). S_1-structure nodes are then aggregated as new nodes and rewired to obtain the structure S_2 (Head, Body, Left/Right Arm, Left/Right Leg). S_3-structured nodes are pooled from S_2. Not that the number of structure configurations decides the number of module group stacks in the up-scaling and down-scaling stages, which means we used three modules in each stage.

3.2.2. Down-Scaling Stage

The down-scaling stage is comprises stacking three encoding module groups, with a skeletal pooling layer inserted between every two groups. It starts with the S_1-structured node features. The features pass through the residual graph convolutional (Res-GConv) layer and, then, are down-scaled into S_2-structured node features by a pooling layer. We repeat this process to generate S_3-structured node features and feed them into a normal 1×1 convolutional layer.

The scales of the S_1-, S_2-, and S_3-structured node features differ. Let these three differently scaled node features be collected as an atlas $\mathbb{H} = \{H_s | s \in \mathbb{S}\}$, where $\mathbb{S} = \{1, 2, 3\}$. When the subscript of the scale is $s = 1$, $H_s \in \mathbb{R}^{D \times 16}$ is expressed as S_1-structured graph features. $H_2 \in \mathbb{R}^{D \times 6}$ and $H_3 \in \mathbb{R}^{D \times 3}$ are the progressive pooling results of different levels corresponding to H_1. The compression of graph nodes shortens the long-range distances, especially the distances between arms and legs. This provides an efficient way to aggregate information from higher-order neighbors.

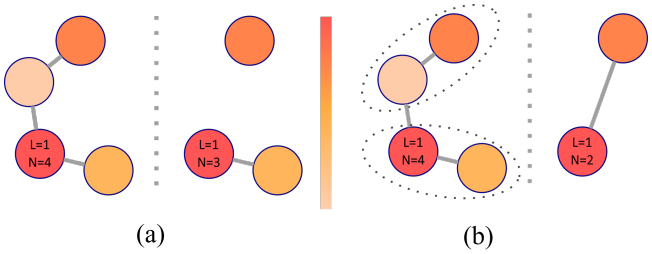


Figure 4. Illustrations of the pooling operation, where L denotes the level and N is the number of nodes in the current graph structure: (**a**) graph pooling [24]; (**b**) skeletal pooling [1].

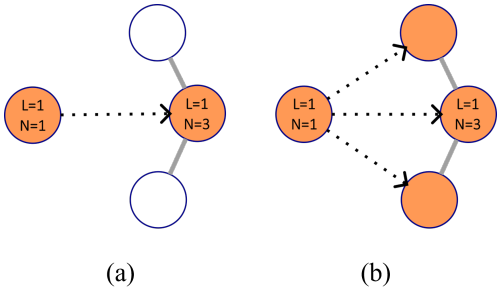


Figure 5. Illustrations of the unpooling operation, where L denotes the level and N is the number of nodes in the current graph structure: (**a**) graph unpooling [24]; (**b**) skeletal unpooling [1].

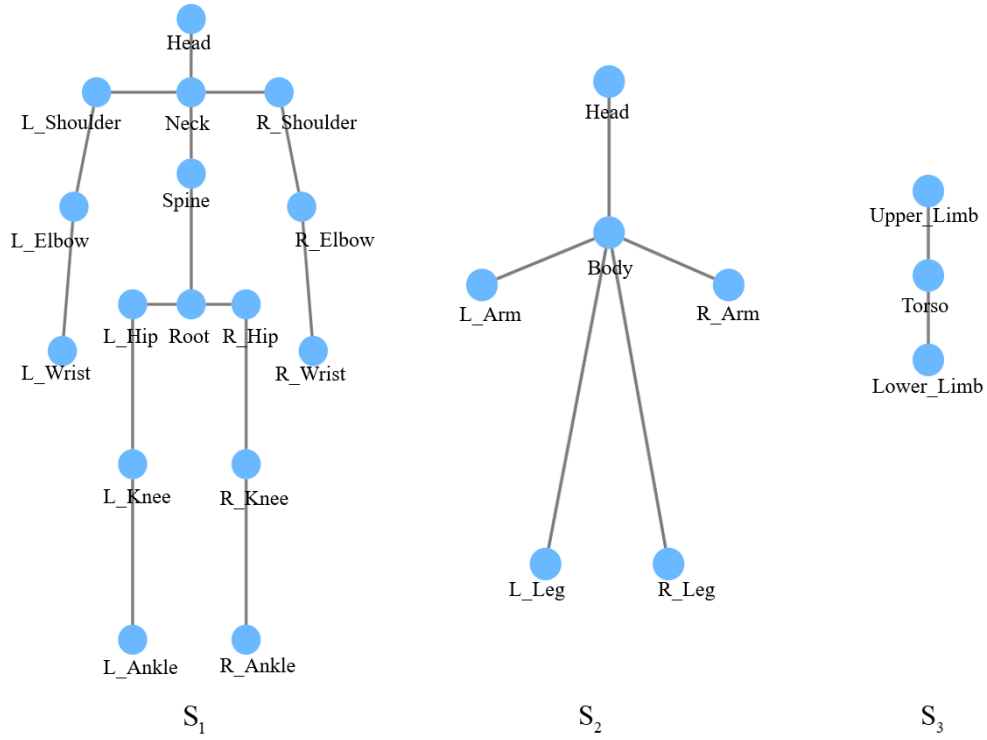


Figure 6. Description of the structure of the body joints and its sub-structures. We designed these pre-defined structures to retain the physical human topology.

3.2.3. Up-Scaling Stage

The up-scaling stage uses three stacked module groups and starts from the lowest-scale feature H_3. The unpooling layer changes the features from the scale $s+1$ to the new scale s. The unpooled features are fed into the Res-GConv layer to obtain H'_s. Then, H_s in the down-scaling stage uses skip-connections to fuse with H'_s, if the cross-scale interaction stage is not utilized. Once the features have returned to the highest scale, an additional non-local

layer is added at the end of module group. The non-local layer is utilized to recompute the attention weights for every pair of nodes when the potential node correlations have been explicitly reassigned and the current scale is the highest.

For each group in the up-scaling stage, the respective outputs are F_3, F_2, and F_1. The features at different levels in the decoding process can provide valuable information for the final prediction, and so, F_3, F_2, and F_1 are all fed into the output stage.

3.2.4. Output Stage

The output stage is applied to process the multi-level features obtained in the up-scaling stage. We used the skeletal unpooling operation mentioned in Section 3.2.3 to expand the scale of intermediate features to the highest scale, concatenate all three features, and feed them into a squeeze-and-excitation block (SE block) [42]. The weights of each feature are re-calculated in this block. The final layer is a graph convolutional block, which is used for prediction.

3.2.5. Cross-Scale Interaction Stage

This stage lies between the down-scaling and up-scaling stages. It contains repeated mapping-aware fusion modules to target the local node interactions across adjacent scales. The number of fusion modules controls the node feature update and fusion level. In our experimental setup, the fusion level $\mathcal{M}$ of 2 was found to be optimal. In the next section, we detail the mechanism of the mapping-aware fusion module.

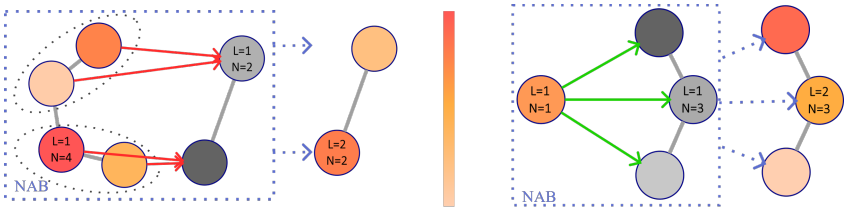
3.3. Mapping-Aware Fusion Module

The naïve multi-scale learning method has been proven useful for receptive field expansion, but comes at the cost of sub-optimal feature loss. In other words, the original version improves the global level, but ignores the local individuality, which might pose difficulties with regard to the ability to capture the subtle differences among poses. In particular, in noisy cases, unstable distant neighbors transfer noise as the learning process goes deeper. We believe that these abandoned features contain the most-direct valuable semantics from their nearest neighbors, thus helping to resist noise. Therefore, in the cross-scale interaction stage, we designed mapping-aware fusion modules and channelwise cross-scale feature fusion to allow interaction between the pooled nodes that carry non-local information and the unpooled nodes that retain detailed information from their direct neighbors. We took one mapping-aware fusion module as an example, in order to simplify the explanation, which means that the fusion level $\mathcal{M}$ is 1. The structure is depicted in Figure 3.

Each of the node-assignment blocks (NABs) in the mapping-aware fusion module has two input streams. In each stream, node features are organized into two types of node structures: target graph and source graph. For example, in a particular stream, the node features to be updated (which we treated as the target) are denoted by H_2, and the source can be one of the adjacent-scaled features (H_1 or H_3). The purpose of the NAB is to transmit information from the source to the target graph nodes where mapping edges participate. We refer to this node-to-node information interaction with explicit direction as mapping-aware interaction to achieve local enhancement.

The NAB focuses on the feature transformation across scales and updates the target graph nodes, which depends significantly on the pre-calculated mapping edges and the corresponding source graph nodes. The NAB can process two situations through the mapping edges: (i) as shown in Figure 7a, when the target graph is at a lower scale than the source, the NAB can pass the information along a mapping edge from the source feature map and re-assign the attention to the corresponding pooled node in the target feature map; (ii) as shown in Figure 7b, when the target graph is at a higher scale than the source, the NAB extracts information from the pooled node in the source feature map along a cluster of mapping edges to update the target feature map. The updated feature maps are then

fed into the channelwise mixer block and fused with the initial target graph features in a channelwise manner.



(a) (b)

Figure 7. Illustration of mapping-aware interaction, where L denotes the level and N is the number of nodes in the current graph structure: the directed (**a**) or inverted directed (**b**) mapping edges (green arrows) from the source feature map (colored circles) to the target feature map (gray circles) guide cross-scale interaction in a node-to-node manner through the NAB. Then, the target graph nodes are updated.

3.3.1. Preparation for Mapping Edges

As the unpooling and pooling between lower- and higher-scale feature maps is directional, we interpreted these relationships as directed mapping edges that can be described by a 2D matrix. We first initialized a 2D zero matrix. Then, when node i in a higher-scale feature map has a mapping relationship with node j in a lower-scale feature map, the matrix element corresponding to the 2D index (i, j) is 1. The mapping matrix A^{state} is constructed following the above rule, where the subscript $state \in \{+, -\}$ represents the mapping direction between nodes across scales. Its elements are called mapping edges. The start and end of a mapping edge represent the node candidates that are about to interact.

Notably, A^{+} is the transpose of A^{-}, and both are adjacency matrix representations for directed graphs. To avoid numerical instabilities, we followed the idea of [14] and normalized the directed adjacency matrix using the following equation:

$$\tilde{\mathrm{A}}^{state} = D^{-1}\mathrm{A}^{state}, \tag{4}$$

where $D \in \mathbb{R}^{N \times N}$ is the degree matrix of A^{state} and N represents the total number of nodes in both the target- and source-scale feature maps.

Next, the NAB concatenates the target $\mathrm{H}_{tar} \in \mathbb{R}^{D \times N_{tar}}$ with the source graph features $\mathrm{H}_{src} \in \mathbb{R}^{D \times N_{src}}$ along the spatial axis to obtain H, as follows:

$$\begin{aligned} &\mathrm{H} = \mathrm{Concat}(\mathrm{H}_{tar}, \mathrm{H}_{src}) \in \mathbb{R}^{D \times (N_{tar} + N_{src})}, \\ &\forall tar, src \in \mathbb{S},\ src = tar \pm 1, \end{aligned} \tag{5}$$

where N_{tar} and N_{src} are the number of nodes in the target and source graphs, respectively, and src denotes the adjacent scale index relative to tar.

3.3.2. Mapping-Aware Node-Assignment Block

The NAB utilizes $\tilde{\mathrm{A}}^{state}$ as a filter to activate a source node, whose features will be passed from the start to the end of a directed mapping edge. We first introduced the nodewise learnable parameters $\mathrm{W}_{ij} \in \mathbb{R}^{D \times D}$ between nodes i and j in a modulated manner [3] to enhance the node discrepancy. Then, we can obtain the updated node features using the mapping function $\mathcal{F}$:

$$\mathcal{F}_i(\mathrm{H}, \tilde{\mathrm{A}}^{state}) = \sum_{j \in \mathcal{N}_i} \mathrm{W}_{ij}\mathrm{h}_j(\tilde{a}_{ij}^{state} + e_{ij}), \tag{6}$$

where $\mathcal{N}_i$ is a collection of all direct neighbors (including the self-node), $\tilde{a}_{ij}^{state}$ denotes a mapping edge between nodes i and j, and e is an element of the identity matrix. Note

that the original mapping matrix A^{state} is transposed depending on the target and source graph scales. If $tar < src$, $\mathcal{F}$ uses $\tilde{\mathrm{A}}^{+}$ to represent nodes from the source to the target graph with the forward mapping relationships; otherwise, $\mathcal{F}$ uses $\tilde{\mathrm{A}}^{-}$. To make this easier to understand, we show these two cases for the mapping function $\mathcal{F}$ in Figure 8. Finally, we processed the updated features $\mathcal{F}(\mathrm{H}, \tilde{\mathrm{A}}^{state})$ through a pointwise convolutional layer. The result is sliced into $\mathrm{H}_{src\_tar} \in \mathbb{R}^{D \times N_{tar}}$, consistent with the scale of the initial target graph. The final process is detailed below:

$$\mathrm{H}_{src\_tar} = \mathrm{Conv}(\mathcal{F}(\mathrm{H}, \tilde{\mathrm{A}}^{state}))(:, : tar), \tag{7}$$

where $(:, : tar)$ slices the generated feature maps into the size of the target.

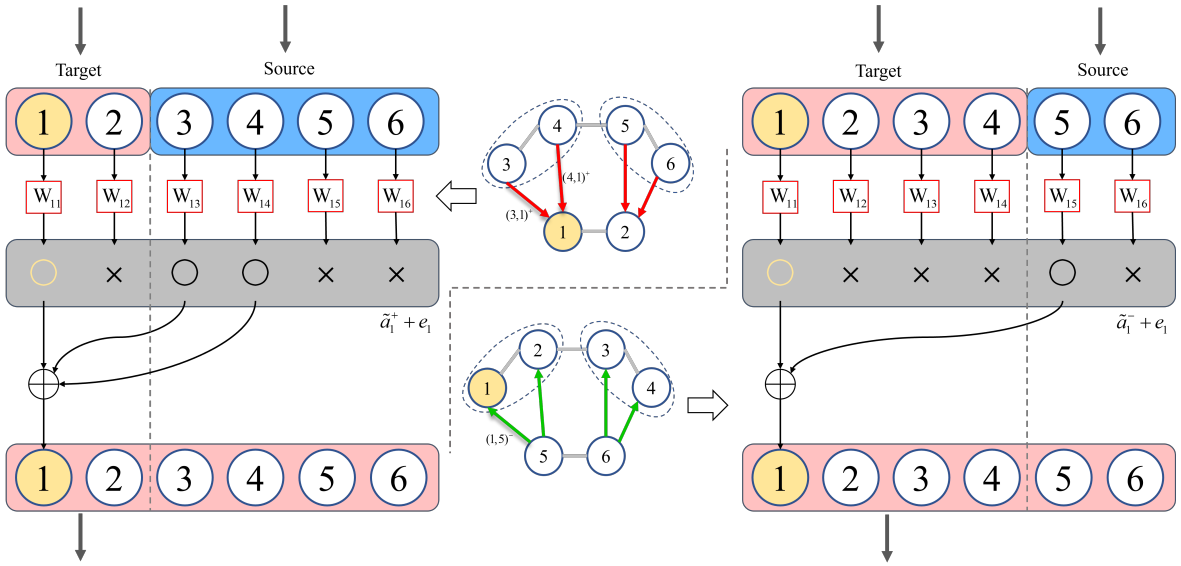


Figure 8. Illustrations of the mapping function $\mathcal{F}$ in two cases, that is non-reversed (**top**) and reversed (**bottom**) mapping relationships between multi-scaled feature maps. The update of Node 1 is taken as an example.

As the NAB only pays attention to both ends of the mapping edge, such a mapping-aware transformation enhances the local information across scales.

3.3.3. Channelwise Mixer Block

After the NABs update node features, we grouped the updated target and initial target node features into a set $\mathcal{H}_{tar} = \{\mathrm{H}_{tar}, \mathrm{H}_{src\_tar}\}$. Each element in the set $\mathcal{H}_{tar}$ is written as $\mathrm{E} \in \mathbb{R}^{D \times N_{tar}}$. In the Mixer block, channelwise summation for fusion is applied to obtain the new fused features, after which the H_{tar} are overwritten as inputs for the next mapping-aware fusion module. Specifically, we first concatenated the features in $\mathcal{H}_{tar}$ and applied global average pooling to obtain a channelwise item $\mathrm{b} \in \mathbb{R}^{1 \times \mathcal{N}D}$, where $\mathcal{N}$ is the number of elements in $\mathcal{H}_{tar}$, equal to either 2 or 3. Then, we used b to calculate the learnable blending vectors β, as follows:

$$\beta = \mathrm{Sigmoid}(\mathrm{Conv1d}_k(\mathrm{b})), \tag{8}$$

where k is an adaptable kernel size, as mentioned in [43]. Next, we reshaped β from $1 \times \mathcal{N}D$ to $\mathcal{N} \times D$ and fused the features to overwrite $\mathcal{H}_{tar}$ using:

$$\mathrm{H}_{tar} \leftarrow \sum_{n=0}^{n<\mathcal{N}} \beta_n \odot \mathrm{E}_n, \tag{9}$$

where $\beta_n \in \mathbb{R}^D$ is the n^{th} learnable blending vector of β and E_n is the n^{th} element in $\mathcal{H}_{tar}$. The symbol $\odot$ denotes the matrix elementwise multiplication that should broadcast.

Compared with direct element addition, this solution fuses elements in a channelwise style, which was found to enhance the performance under our experimental settings.

3.4. Training

Most skeleton-based models aim to minimize the errors of the predicted $J_{3D} = \{j_i | i = 1, \ldots, N\}$ and ground truth $\tilde{J}_{3D} = \{\tilde{j}_i | i = 1, \ldots, N\}$ poses, which can be formulated as:

$$\mathcal{L}(\tilde{J}_{3D}, J_{3D}) = \sum_{i=1}^{N} \| \tilde{j}_i - j_i \|^2. \quad (10)$$

We used a weighted combination of the Mean-Squared Errors (MSEs) and the Mean Absolute Errors (MAEs) as a loss function, following previous works [1,3,8].

Before training, for $J_{2D} \in \mathbb{R}^{N \times 2}$, we normalized the keypoints in the pixel space within the range $[-1, 1]$ and retained the scales of images in every skeleton. As some prior visual knowledge exists in the global translation, we did not remove it. For $\tilde{J}_{3D}$, as the paired 3D ground truth, we transformed the original position from the world to view space and made the local coordinate system of every joint relative to the root. The training was then supervised with respect to $\tilde{J}_{3D} \in \mathbb{R}^{N \times 3}$ for the root-relative predictions $J_{3D} \in \mathbb{R}^{N \times 3}$ in the view space.

4. Experiments

In this section, we first introduce the experimental settings, including the datasets used, evaluation metrics, and implementation details, in Sections 4.1 and 4.2. The ablation study and performance comparison are detailed in Sections 4.3 and 4.4, respectively. Finally, some visualization results are provided in Section 4.5.

4.1. Datasets and Evaluation

We conducted evaluation on two datasets: (a) Human3.6M (H36M) [25] is one of the most-widely used paired single-human pose datasets captured under laboratory conditions with 3.6-million frames. Four cameras were simultaneously used to capture daily human postures from different perspectives. The motion capture system allowed for precise 3D joint positions to be obtained simultaneously. Following previous work [1–5], the data for Subjects S1, S2, S3, S5, and S7 were defined as the training dataset, while the data of Subjects S9 and S11 were used for further testing. Before training, we flipped the paired data to expand the coverage of the dataset. (b) MPI-INF-3DHP (MPII) [26] is a paired dataset with 1.3-million frames obtained from a multi-view markerless motion capture in various scenario settings, including indoor scenes with green screens, outdoor scenes without green screens, and outdoor scenes. We only used its validation set to test the generalizability of the proposed method. We followed the method of Posaug [44] to complete data pre-processing. The redirection of the skeleton format was the same as the suggestion in [26].

Following previous works [1–5], we utilized two evaluation metrics: (a) MPJPE/PA-MPJPE, where MPJPE denotes the Euclidean distance between the ground truth and estimated 3D joint after aligning their roots, and PA-MPJPE is an additional metric that employs a rigid Procrustes [45] alignment to remove the influence of rotation and scale first and, then, calculates the error. We only used MPJPE, as the former is stricter and more representative. (b) PCK/AUC, of which the percentage of correct keypoints (PCK) is a metric typically used in 2D human pose estimation that can be extended to a 3D version, interpreted as the proportion of joints within a certain error threshold (typically set at 150 mm) relative to the total number of joints. The AUC score is interpreted as the area under the curve of the PCK. We used both of these metrics to validate the generalizability of the proposed model.

4.2. Implementation Details

Following previous works [1,3–5,11,12], the 2D detection results were obtained using the cascaded pyramid network (CPN) [31] pre-trained on the COCO [46] dataset and

fine-tuned on Human3.6M [25]. All experiments were implemented in PyTorch with an AMD Ryzen 7 5800X 8-Core Processor and a single NVIDIA RTX 3090 GPU in Ubuntu 20.04 and optimized using Adam [47]. We applied a modulated graph convolutional layer [3] as our GConv block, and the weight initialization was the same as described in [3]. The GConv layer also employed the self-connection decoupling strategy to retain the self-loop information of nodes.

We trained our model in an end-to-end manner. The batch size was set to 256, and the feature dimension was set to 128. The learning rate was initially 0.002, and the learning rate decay value was 0.96, with the rate decayed every 4 epochs. Dropout [48] with a probability of 0.2 was applied to each GConv layer, in order to avoid over-fitting. The cross-scale fusion level $\mathcal{M}$ was set to 2.

4.3. Ablation Study

We constructed an ablation study on the Human3.6M [25] validation sets, employing MPJPE as the evaluation metric. Two different basic configurations were considered. One was the Simple-UGCN (S-UGCN), which removes the cross-scale interaction stage, but retains the U-shaped structure, and the other was the proposed method M-UGCN. We chose the ground truth keypoints as input, in order to exclude disturbances derived from noise.

4.3.1. Cross-Scale Fusion Level

Multi-scale node features can pass through the cross-scale interaction stage to enhance the local information through the use of stacked mapping-aware fusion modules. We denote the utilization frequency as the fusion level. To discuss the impact of the number of mapping-aware fusion modules, we tuned the fusion level from 0 to 3. The obtained mean errors are shown in Figure 9. It is evident that, when we adopted a fusion level of 2, the errors of the joint positions were sharply decreased, compared to Levels 1 and 3. Level 0—the same configuration as the Simple-UGCN—presented the closest results to Level 2. However, Fusion Level 0 (i.e., without local enhancement) is more likely to encounter bottlenecks under noisy inputs, as discussed in Section 4.3.

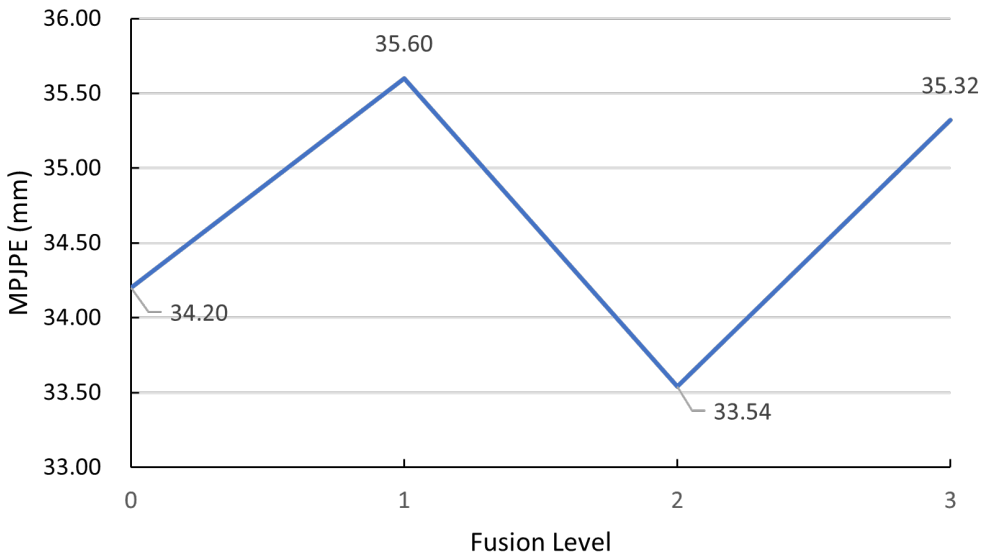


Figure 9. Fusion level ablation study at the cross-scale interaction stage.

4.3.2. Components of the Mapping-Aware Fusion Module

This module achieves local information exchange from one side of a mapping edge to the other and contains two key components. The NAB passes edge information of edges from the source to target graph nodes along the mapping edges, while the channelwise Mixer block mixes the updated target graph nodes with the initial nodes in a learnable manner. To study the importance of both components in the mapping-aware fusion module, we designed four variants: (i) Strategy 1 is a Simple-UGCN, which removes the whole cross-scale interaction stage; (ii) Strategy 2 performs feature fusion with given blending factors under the weighted rate 1:3, taking the place of the channelwise mixer block; (iii) Strategy 3 adopts two spatial linear layers [49], whose hidden size is 512, to update the target graph node features, rather than the feature transformation and mapping function in Equation (6); (iv) Strategy 4 is the M-UGCN containing both of the components. All variants were constructed under Cross-Fusion Level 2, and the results are given in Table 1.

Table 1. Ablation study considering the components of the cross-scale interaction stage. In addition to the S-UGCN and M-UGCN, two other variants are added for comparison. The best are in bold.

Method	NAB	Mixer (Channelwise)	Params (M)	MPJPE↓ (mm)
Strategy 1 (S-UGCN)	✗	✗	**0.87**	34.20
Strategy 2	✓	✗	1.25	35.35
Strategy 3	✗	✓	1.25	36.48
Strategy 4 (M-UGCN)	✓	✓	1.25	**33.53**

↓ Lower is better.

We can see that the NAB and Mixer block cooperated reasonably. When either of them was absent, the errors obviously increased. The results obtained under Strategy 2 demonstrated the importance of the learnable fusion operations from the opposite perspective. Strategy 3, with spatial linear layers, caused every output node feature to depend on all other nodes, going against our mapping-aware local enhancement for explicit node-to-node exchange. This led to the poorest result.

4.3.3. Mixer Type

The Mixer block fuses features in a learnable way. We compared two types for the Mixer. The first was discussed in Section 3.3, which learns the weights of each node feature map, and the final error was 34.51 mm. The second involved the common weighted summation of multiple features, with which the error was 1.02 mm higher. The results indicated that the channelwise Mixer block has more flexibility to sort the importance of features, thus boosting the final performance.

4.3.4. Mapping Edge Setting

The mapping edges, represented by A^{state}, establish node-to-node interaction paths across different scales, playing an essential role in mapping-aware local enhancement. The participation of the mapping edges helps the NABs focus on nodes across scales that have explicit mapping relationships, generated by the pooling and unpooling processes. We compared four different settings to prove the effectiveness of mapping edges: (i) CNN-like methods—standard convolutional blocks used to replace the whole NAB for node updating; (ii) free edges—where the weights of mapping edges were learned from training data freely to generate A^{state}; (iii) undirected edges—mapping relationships considered to be bidirectional and A^{state} symmetrically normalized before being fed into the NAB; (iv) directed edges—where the proposed mapping edges were interpreted as unidirectional links to simulate the pooling and unpooling processes. Figure 10 validates that the CNN-like method cannot adapt to deal with non-Euclidean data, for which GCN-based methods are suitable. The generation of free mapping edges was also not suited to our objective of capturing the node-to-node mapping relationships. In contrast, both undirected and

directed edges succeeded in encouraging information exchange between local nodes across adjacency scales. However, the mapping direction offered further cues to inform the NAB about the state of the input feature (i.e., whether it was up-scaled or down-scaled from the source graph data).

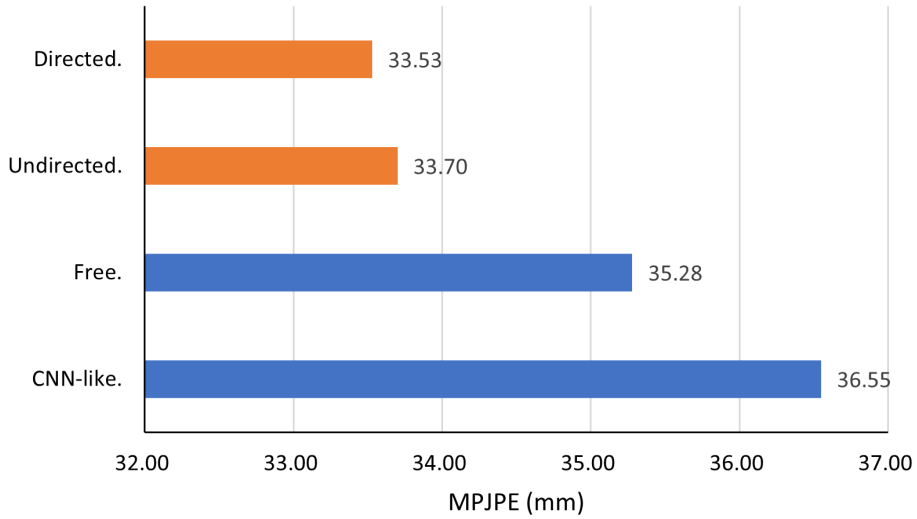


Figure 10. Ablation study of the mapping edges. The directed mapping edges are the most-suitable representatives for mapping relationships across adjacency scales.

4.4. Performance Comparison

4.4.1. Comparison with the State-of-the-Art

Using the MPJPE metric, we compared the Simple-UGCN and M-UGCN with previous works on Human3.6M. We used two types of input: the 2D keypoints detected by the CPN [31] and the ground truth. Note that some works have applied a post-refinement module [8] to process the final outputs when considering noisy detections. The x and y components of the outputs can be derived in two ways. One involves obtaining predictions directly in a regressive manner, while the other involves using the projection model to transform the x and y components from the pixel space to the view space. The post-refinement module mixes the results of these two solutions for more-stable predictions.

For the sake of fairness, we made respective comparisons depending on whether the post-refinement module was used or not, as the camera parameters corresponding to the images are not always available in practical application scenarios.

The results in Table 2 suggest that M-UGCN surpassed all other methods when considering noisy detections, even when the post-refinement module was excluded. Moreover, the use of mapping-aware fusion modules allowed for Simple-UGCN to overcome its bottleneck, especially for challenging, but relatively static poses such as sitting down. This means that the optimized M-UGCN had a stronger capability to balance short- and long-range joint relationships.

Table 2. Quantitative evaluation results on Human3.6M under MPJPE. 2D keypoints detected by the the CPN [31] were used as the input. The table is categorized into two groups. The bottom group methods employ the post-refinement module [8], while the top group methods do not. The best are in bold.

Methods	Dire.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
Martincz et al. [50]	51.8	56.2	58.1	59.0	69.5	78.4	55.2	58.1	74.0	94.6	62.3	59.1	65.1	49.5	52.4	62.9
Wang et al. [51]	49.2	55.5	53.6	53.4	63.8	67.7	50.2	51.9	70.3	81.5	57.7	51.5	58.6	44.6	47.2	57.8
Pavlakos et al. [17]	48.5	54.4	54.4	52.0	59.4	65.3	49.9	52.9	65.8	71.1	56.6	52.9	60.9	44.7	47.8	56.2
Ci et al. [9]	46.8	52.3	**44.7**	50.4	52.9	68.9	49.6	46.4	60.2	78.9	51.2	50.0	54.8	40.4	43.3	52.7
Pavllo et al. [52]	47.1	50.6	49.0	51.8	53.6	61.4	49.4	47.4	59.3	67.4	52.4	49.5	55.3	39.5	42.7	51.8
Liu et al. [10]	46.3	52.2	47.3	50.7	55.5	67.1	49.2	46.0	60.4	71.1	51.5	50.1	54.5	40.3	43.7	52.4
Zou et al. [5]	49.0	54.5	52.3	53.6	59.2	71.6	49.6	49.8	66.0	75.5	55.1	53.8	58.5	40.9	45.4	55.6
Li et al. [11]	47.8	52.5	47.7	50.5	53.9	60.7	49.5	49.4	60.0	66.3	51.8	48.8	55.2	40.5	42.6	51.8
Xu et al. [1]	45.2	49.9	47.5	50.9	54.9	66.1	48.5	46.3	59.7	71.5	51.4	48.6	53.9	39.9	44.1	51.9
Zhao et al. [2]	45.2	50.8	48.0	50.0	54.9	65.0	48.2	47.1	60.2	70.0	51.6	48.7	54.1	39.7	43.1	51.8
Wu et al. [13]	**44.4**	50.1	46.0	**47.1**	52.6	66.3	**47.0**	**44.4**	59.9	77.8	51.0	48.1	**45.2**	38.3	**40.5**	50.9
Zhao and Wang [4]	45.0	50.0	45.8	50.4	53.8	62.7	47.8	45.6	58.7	66.2	51.4	48.0	53.0	38.4	41.3	50.5
Ours (Simple-UGCN)	46.8	51.5	44.8	50.7	52.2	**59.3**	49.5	46.7	58.6	69.1	51.2	48.2	55.3	39.7	42.1	51.0
Ours (M-UGCN)	44.5	**49.3**	46.6	49.4	**51.2**	**59.3**	47.7	45.1	**57.1**	**65.9**	**50.1**	**47.5**	53.3	**38.2**	40.9	**49.7**
Cai et al. [8]	46.5	48.8	47.6	50.9	52.9	61.3	48.3	45.8	59.2	64.4	51.2	48.4	53.5	39.2	41.2	50.6
Zou et al. [3]	45.4	49.2	45.7	49.4	50.4	58.2	47.9	46.0	57.5	**63.0**	49.7	46.6	52.2	38.9	40.8	49.4
Lin et al. [53]	**42.8**	48.6	45.1	**48.0**	51.0	**56.5**	**46.2**	44.9	56.5	63.9	49.6	**46.2**	**50.5**	37.9	**39.5**	48.5
Ours (Simple-UGCN)	44.8	50.2	**43.9**	49.0	50.7	56.9	48.1	45.4	57.1	65.4	49.9	46.8	52.4	38.5	40.8	49.3
Ours (M-UGCN)	43.3	**48.4**	45.4	48.3	**49.8**	57.6	46.6	**44.3**	**55.6**	**63.0**	**48.9**	46.3	51.5	**37.4**	39.9	**48.4**

Table 3 shows the significant performance enhancement when all the methods reached their upper bounds for precise inputs. Compared to the method in [1] (with 3.70 M parameters), which also takes advantage of multi-scale feature learning, the Simple-UGCN reduced the average error by 1.6 mm with only 0.87 M parameters, while the M-UGCN reduced the error by 2.3 mm with 1.25 M parameters. Our two models achieved a 90.8% and 86.8% reduction in the number of parameters, respectively, and had far fewer parameters than the second-place method [54] (9.49 M).

Table 3. Quantitative evaluation results on Human3.6M under MPJPE. Ground-truth keypoints were used as the input. The best results are indicated in bold.

Methods	Dire.	Disc.	Eat	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
Martinez et al. [50]	45.2	46.7	43.3	45.6	48.1	55.1	44.6	44.3	57.3	65.8	47.1	44.0	49.0	32.8	33.9	46.8
Zhao et al. [7]	37.8	49.4	37.6	40.9	45.1	41.4	40.1	48.3	50.1	42.2	53.5	44.3	40.5	47.3	39.0	43.8
Liu et al. [10]	36.8	40.3	33.0	36.3	37.5	45.0	39.7	34.9	40.3	47.7	37.4	38.5	38.6	29.6	32.0	37.8
Xu et al. [1]	35.8	38.1	31.0	35.3	35.8	43.2	37.3	31.7	38.4	45.5	35.4	36.7	36.8	27.9	30.7	35.8
Zhao et al. [2]	32.0	38.0	30.4	34.4	34.7	43.3	35.2	31.4	38.0	46.2	34.2	35.7	36.1	**27.4**	30.6	35.2
Li et al. [54]	33.1	38.7	29.3	34.4	34.1	**38.0**	39.0	32.1	38.4	**39.4**	34.2	37.5	**34.5**	28.0	**28.8**	34.6
Ours (Simple-UGCN)	32.0	36.2	**29.2**	33.6	34.3	39.9	36.0	30.7	**36.7**	42.9	33.9	35.2	35.7	27.6	29.1	34.2
Ours (M-UGCN)	**27.9**	**34.8**	30.6	**32.0**	**33.7**	39.8	**34.5**	**30.0**	37.7	44.1	**33.6**	**32.7**	34.6	27.5	29.3	**33.5**

4.4.2. Robustness against Noise

In practical applications, data inputs are typically noisy and unstable. To highlight the capacity for noise resistance, we trained our models and the other state-of-the-art models using the ground truth data with various levels of Gaussian noise added. Figure 11 clearly shows that the accuracy of the 2D detection directly affected the performance of the 2D-to-3D pose estimation tasks. Note that the works [3,5,9,11–13] adopted the traditional receptive field expansion idea for GCN-based methods, while our method utilizes multi-scale learning. Moreover, the M-UGCN learned the subtle differences between poses through the mapping-aware local enhancement, thus keeping the prediction errors at a much lower level.

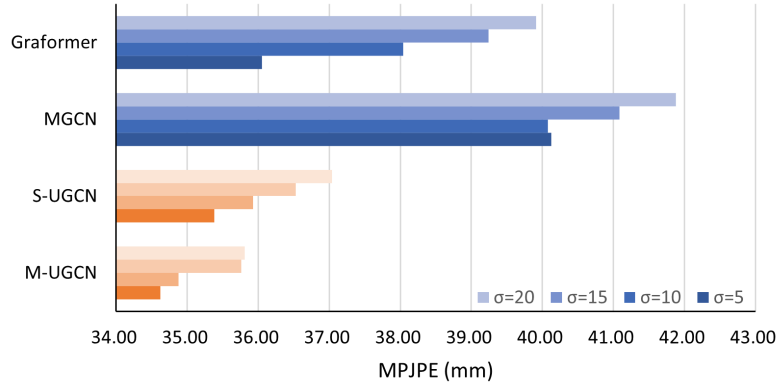


Figure 11. Noise tolerance comparisons with the models Graformer [2] and MGCN [3] models. Here, we applied Gaussian noise with mean zero and various values for the standard deviation σ to the 2D ground truth data.

4.4.3. Model Scale

In Table 4, we report the scales of our two model configurations, in order to show that our solutions achieved competitive results while controlling the number of parameters to be as small as possible. Note that all methods considered in the comparison were under their full parameter configurations and used the ground truth as the input. The most-robust and -effective method was that of Zhao and Wang [4], due to it having lowest prediction errors and relatively small model scale. However, the M-UGCN presented a more-powerful ability to deal with noisy inputs, as indicated in Table 2 and Figure 11.

We also compared the computational efficiency and inference time with the state-of-the-art in Table 4. With the cross-scale interaction stage, the MPJPE of the M-UGCN was 33.5, but its FLOPs and inference time rose because several parallel branches run in the model. This is acceptable because the eyes can struggle to determine tiny temporal discrepancy. When turning to a more-lightweight model configuration, the Simple-UGCN was more-computationally and -memory efficient.

Table 4. Model scale comparisons among methods under their full configurations. Inf_time is the abbreviation of inference time. We computed the inference time as the average time required to infer a sample with a batch size of 256 after the model is warmed up.

Methods	Params (M)	FLOPs↓ (G)	Inf_time↓ (ms)	MPJPE↓ (mm)
Martinez et al. [50]	4.29	-	-	45.5
Liu et al. [10]	4.22	-	-	37.8
Xu et al. [1]	3.70	-	-	35.8
Zhao et al. [2]	0.65	0.0119	0.0560	35.2
Zou et al. [3]	2.74	0.0447	0.0503	34.1
Zhao and Wang [4]	1.99	-	-	33.2
Simple-UGCN	0.87	0.0121	0.0558	34.2
M-UGCN	1.25	0.0187	0.0695	33.5

↓ Lower is better.

4.4.4. Generalization Validation

We applied the proposed model trained on the Human3.6M [25] dataset to the MPI-INF-3DHP [26] validation set, in order to validate the generalizability of our method. Compared to the Human3.6M dataset, the validation set of 3DHP contains more-diverse motions and unseen outdoor environments. Although we used only the Human3.6M [25] dataset comprising laboratory scenarios for training, the results provided in Table 5 suggest that our method had strong compatibility with unseen in-the-wild data.

Table 5. Results on the MPI-INF-3DHP [26] test set, which includes three scenario settings: studio with green screen (GS), studio without green screen (noGS), and outdoors (Outdoor).

Methods	Training Data	PCK↑				AUC↑
		GS	noGS	Outdoor	All	
Yang et al. [55]	H36M+MPII	-	-	-	69.0	32.0
Zhou et al. [56]	H36M+MPII	75.6	71.3	80.3	75.3	38.0
Xu et al. [1]	H36M	81.5	81.7	75.2	80.1	45.8
Zhao et al. [2]	H36M	80.1	77.9	74.1	79.0	43.8
Zhao and Wang [4]	H36M	81.8	78.2	77.5	80.1	44.0
Wu et al. [13]	H36M	81.3	75.8	75.2	77.8	46.9
Ours (M-UGCN)	H36M	85.2	85.8	77.8	82.4	50.4

↑ Higher is better.

4.5. Qualitative Results

Figure 12 shows some visualization results obtained using the M-UGCN on the Human3.6M dataset. The discrepancy between the predictions and ground truth can be seen to be negligible. Our method can reason properly for some challenging poses. We believe that the effectiveness of the skeleton-based method will benefit further applications, such as pose-based digital-character-driven solutions and action recognition.

Figure 13 shows qualitative comparisons between our method and a baseline method on challenging poses in the Human3.6M [25] dataset. The 2D keypoints detected by the CPN [31] were used as the input. In order to demonstrate robustness in noisy cases, we selected images with actors not facing the camera and severe self-occlusion. Our method obtained more-stable predictions when met with challenging poses.

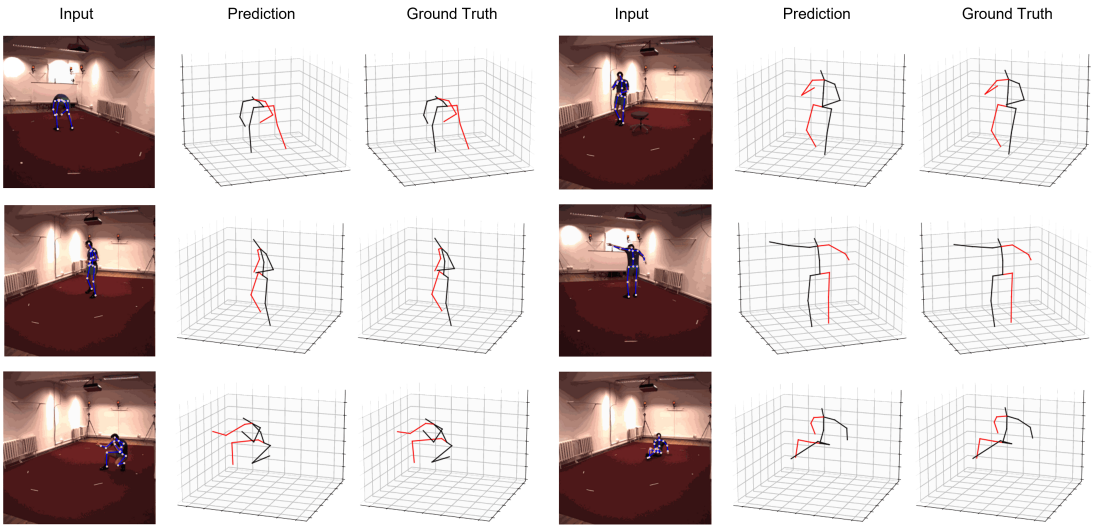


Figure 12. Qualitative results of our method on the Human3.6M [25] dataset. The discrepancy between the predictions and ground truth is negligible.

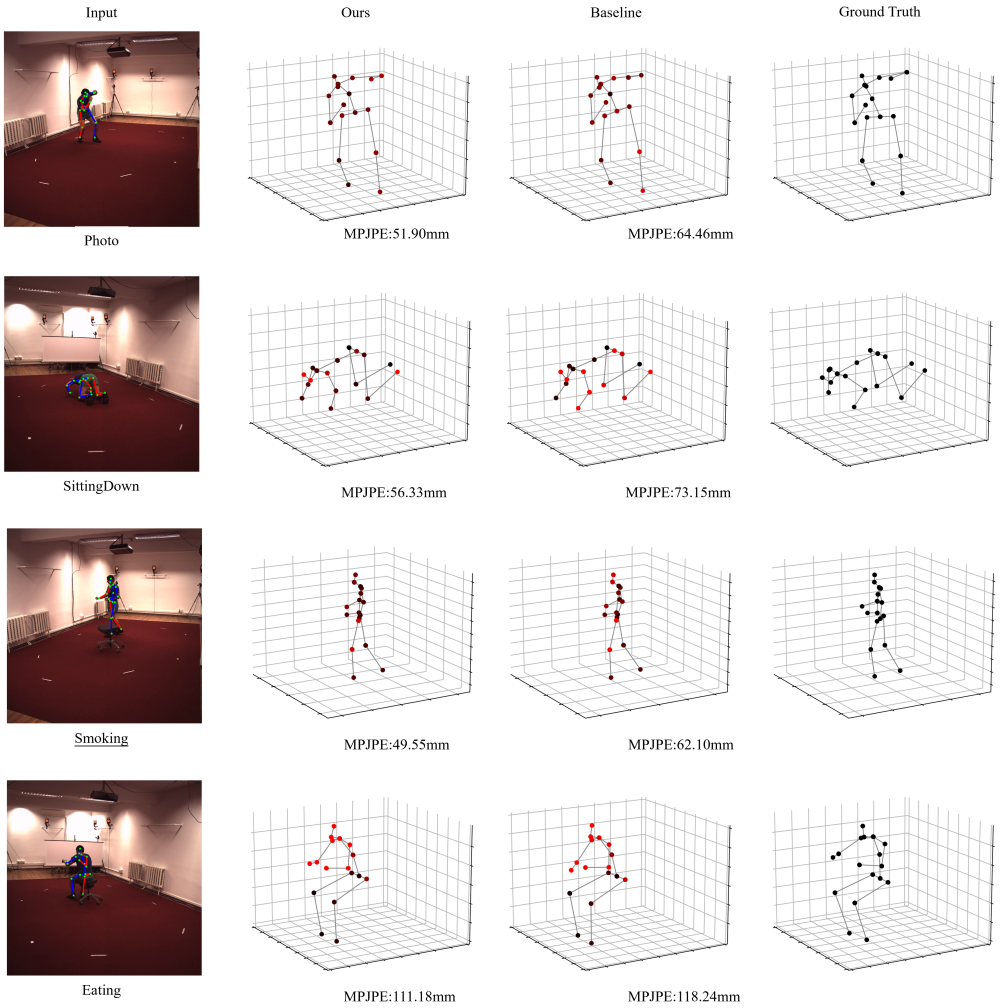


Figure 13. Qualitative comparisons between our proposed method and a baseline method [3] on challenging poses in the Human3.6M [25] dataset. The 2D keypoints detected by the CPN [31] were used as the input samples, and the brightness of points represents the scale of the error, amplifying the visual discrepancy from the ground truth.

We further compared our method with the baseline [3] on in-the-wild images with unseen poses, as illustrated in Figure 14. The same 2D pose detector, the CPN [31], was applied for fairness. Thanks to the local enhancement strategy in the cross-scale interaction stage, our method presented improved partial stability and generated more-plausible limbs for the poses.

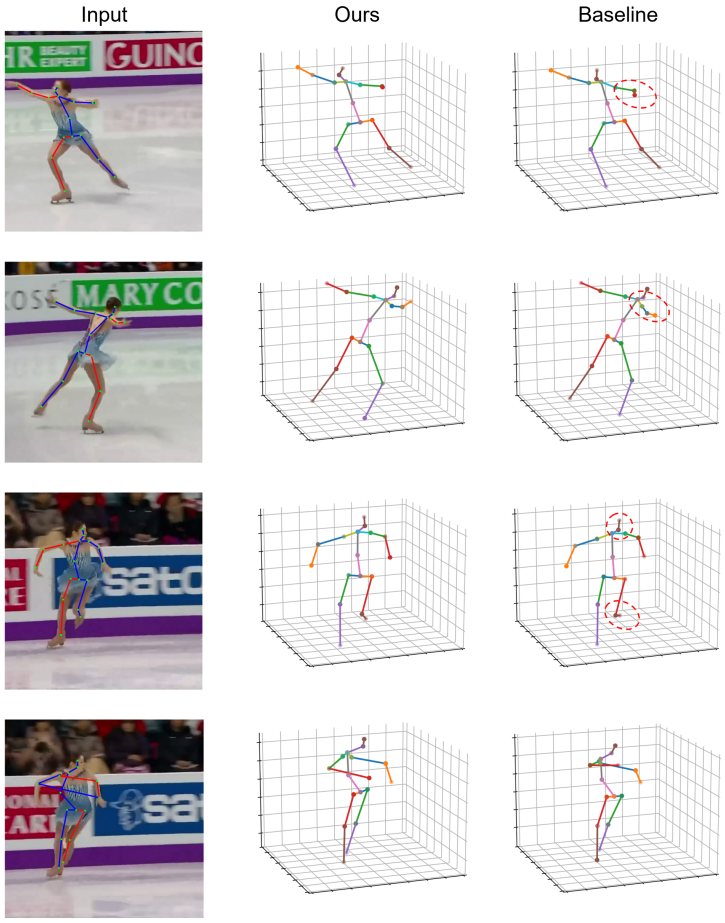


Figure 14. Qualitative comparisons on challenging in-the-wild images between our method and a baseline method [3]. The last row shows that our method can reason more-believable poses in the keypoint-missing case.

5. Conclusions

We presented a lightweight 2D-to-3D method, the M-UGCN, for monocular 3D human pose estimation that reduced the number of parameters and reasoned more-believable poses with only one frame. A skeletal pooling and unpooling operation was introduced to U-shaped nets to exploit global features. The mapping-aware interaction was able to capture subtle discrepancies in local joint correlation. As far as we know, our method is the first attempt to apply directional mapping relationships described as directed graphs to multi-scale feature fusion in the sparse graph case. We built mapping edges across feature scales to simulate the graph-structured nodes' pooling and unpooling process, thus contributing to precise information complementarityand exchange. We implemented ablation experiments to prove the validation of the mapping-aware local enhancement.

Compared to the temporal-based method, the M-UGCN may show weaknesses in its ability to cope with challenging poses, but still surpassed some methods with short sequences of poses (shown in Table 6) and most of the SOTA single-frame methods (shown in Table 2). Despite the M-UGCN achieving promising performance, as a 2D-to-3D approach, it is highly dependent on 2D keypoint detectors. The M-UGCN may not be able to predict reasonable poses with poor 2D keypoint inputs. There are multiple solutions to enhance the M-UGCN on poor single-frame inputs. We believe that, with the incorporation of pre-refinement [57] on 2D inputs or post-refinement [8] on 3D outputs, the M-UGCN on noisy inputs can be enhanced even more. Besides, multiple 3D pose candidates' generation and sampling [58] comprise another solution to avoid unreasonable predictions.

Table 6. Comparisons between the proposed method and previous temporal-based methods in terms of the number of parameters, the length of the pose sequence, and MPJPE.

Methods	Params (M)	Frames	MPJPE↓ (mm)
ST-GCN * [8]	5.18	7	48.8
PoseFormer [18]	9.58	9	49.9
STFormer [6]	7.21	3	50.6
Simple-UGCN	0.87	1	51.0
Simple-UGCN *	1.01	1	49.3
M-UGCN	1.25	1	49.7
M-UGCN *	1.48	1	48.4

Methods marked by * employed the post-refinement module [8]. ↓ means lower is better.

In the future, considering the restriction of lengthy sequences in practical applications, we aim to improve the accuracy and decrease the computation to extend our methods for pose-based human driving or monocular motion capture. As an intermediate component, our model has a reliable backend of accuracy improvements and would be the key to transferring 3D poses into the 3D human mesh.

Author Contributions: Methodology, B.Y., Y.H., G.C., D.H. and Y.D.; software, B.Y. and Y.H.; writing—original draft preparation, B.Y., Y.H. and G.C.; writing—review and editing, B.Y., Y.H., D.H. and Y.D. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Shanghai Natural Science Foundation (19ZR1419100) and the Shanghai Talent Development Funding (2021016).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study.

References

1. Xu, T.; Takano, W. Graph stacked hourglass networks for 3d human pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 16105–16114.
2. Zhao, W.; Wang, W.; Tian, Y. GraFormer: Graph-oriented transformer for 3D pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 20438–20447.
3. Zou, Z.; Tang, W. Modulated graph convolutional network for 3D human pose estimation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 11477–11487.
4. Zhao, W.; Wang, W. K-order graph-oriented transformer with GraAttention for 3D pose and shape estimation. *arXiv* **2022**, arXiv:2208.11328.
5. Zou, Z.; Liu, K.; Wang, L.; Tang, W. High-order graph convolutional networks for 3D human pose estimation. In Proceedings of the British Machine Vision Conference (BMVC), Virtual, 22–25 November 2020.
6. Wang, H.; Shi, Q.; Shan, B. Three-dimensional human pose estimation with spatial–temporal interaction enhancement transformer. *Appl. Sci.* **2023**, *13*, 5093. [CrossRef]
7. Zhao, L.; Peng, X.; Tian, Y.; Kapadia, M.; Metaxas, D.N. Semantic graph convolutional networks for 3d human pose regression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 3425–3435.
8. Cai, Y.; Ge, L.; Liu, J.; Cai, J.; Cham, T.J.; Yuan, J.; Thalmann, N.M. Exploiting spatial-temporal relationships for 3d pose estimation via graph convolutional networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 2272–2281.
9. Ci, H.; Wang, C.; Ma, X.; Wang, Y. Optimizing network structure for 3d human pose estimation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 2262–2271.
10. Liu, K.; Ding, R.; Zou, Z.; Wang, L.; Tang, W. A Comprehensive Study of Weight Sharing in Graph Networks for 3D Human Pose Estimation. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; pp. 318–334.
11. Li, H.; Shi, B.; Dai, W.; Chen, Y.; Wang, B.; Sun, Y.; Guo, M.; Li, C.; Zou, J.; Xiong, H. Hierarchical graph networks for 3D human pose estimation. *arXiv* **2021**, arXiv:2111.11927.
12. Quan, J.; Hamza, A.B. Higher-order implicit fairing networks for 3D human pose estimation. *arXiv* **2021**, arXiv:2111.00950.
13. Wu, W.; Zhou, D.; Zhang, Q.; Dong, J.; Wei, X. High-order local connection network for 3D human pose estimation based on GCN. *Appl. Intell.* **2022**, *52*, 15690–15702. [CrossRef]

14. Cheng, Y.; Wang, B.; Yang, B.; Tan, R.T. Graph and temporal convolutional networks for 3d multi-person pose estimation in monocular videos. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 2–9 February 2021; Volume 35, pp. 1157–1165.
15. Hu, W.; Zhang, C.; Zhan, F.; Zhang, L.; Wong, T.T. Conditional directed graph convolution for 3d human pose estimation. In Proceedings of the 29th ACM International Conference on Multimedia, Chengdu, China, 20–24 October 2021; pp. 602–611.
16. Sun, X.; Shang, J.; Liang, S.; Wei, Y. Compositional human pose regression. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2602–2611.
17. Pavlakos, G.; Zhou, X.; Daniilidis, K. Ordinal depth supervision for 3d human pose estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7307–7316.
18. Zheng, C.; Zhu, S.; Mendieta, M.; Yang, T.; Chen, C.; Ding, Z 3d human pose estimation with spatial and temporal transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 11656–11665.
19. Bogo, F.; Kanazawa, A.; Lassner, C.; Gehler, P.; Romero, J.; Black, M.J. Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; pp. 561–578.
20. Lin, K.; Wang, L.; Liu, Z. End-to-end human pose and mesh reconstruction with transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2021; pp. 1954–1963.
21. Joo, H.; Neverova, N.; Vedaldi, A. Exemplar fine-tuning for 3d human model fitting towards in-the-wild 3d human pose estimation. In Proceedings of the 2021 International Conference on 3D Vision (3DV), Virtual, 1–3 December 2021; pp. 42–52.
22. Krichen. Convolutional Neural Networks: A survey. *Computers* **2023**, *12*, 151. [CrossRef]
23. Zeng, A.; Sun, X.; Yang, L.; Zhao, N.; Liu, M.; Xu, Q. Learning skeletal graph neural networks for hard 3d pose estimation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 11436–11445.
24. Gao, H.; Ji, S. Graph U-Nets. In Proceedings of the Machine Learning Research (PMLR), Long Beach, CA, USA, 9–15 June 2019; pp. 2083–2092.
25. Ionescu, C.; Papava, D.; Olaru, V.; Sminchisescu, C. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *36*, 1325–1339. [CrossRef] [PubMed]
26. Mehta, D.; Rhodin, H.; Casas, D.; Fua, P.; Sotnychenko, O.; Xu, W.; Theobalt, C. Monocular 3d human pose estimation in the wild using improved cnn supervision. In Proceedings of the 2017 International Conference on 3D Vision (3DV), Qingdao, China, 10–12 October 2017; pp. 506–516.
27. Loper, M.; Mahmood, N.; Romero, J.; Pons-Moll, G.; Black, M.J. SMPL: A skinned multi-person linear model. *ACM Trans. Graph. (TOG)* **2015**, *34*, 1–16. [CrossRef]
28. Sun, K.; Xiao, B.; Liu, D.; Wang, J. Deep high-resolution representation learning for human pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 5693–5703.
29. Newell, A.; Yang, K.; Deng, J. Stacked hourglass networks for human pose estimation. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; pp. 483–499.
30. Cao, Z.; Simon, T.; Wei, S.E.; Sheikh, Y. Realtime multi-person 2d pose estimation using part affinity fields. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7291–7299.
31. Chen, Y.; Wang, Z.; Peng, Y.; Zhang, Z.; Yu, G.; Sun, J. Cascaded pyramid network for multi-person pose estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7103–7112.
32. Defferrard, M.; Bresson, X.; Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 3844–3852.
33. Levie, R.; Monti, F.; Bresson, X.; Bronstein, M.M. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Trans. Signal Process* **2018**, *67*, 97–109. [CrossRef]
34. Tekin, B.; Katircioglu, I.; Salzmann, M.; Lepetit, V.; Fua, P. Structured prediction of 3d human pose with deep neural networks. *arXiv* **2016**, arXiv:1605.05180.
35. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
36. Battaglia, P.W.; Hamrick, J.B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; Faulkner, R.; et al. Relational inductive biases, deep learning, and graph networks. *arXiv* **2018**, arXiv:1806.01261.
37. Nah, S.; Hyun Kim, T.; Mu Lee, K. Deep multi-scale convolutional neural network for dynamic scene deblurring. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3883–3891.
38. Cai, Z.; Fan, Q.; Feris, R.S.; Vasconcelos, N. A unified multi-scale deep convolutional neural network for fast object detection. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; pp. 354–370.
39. Zhang, Y.; Hu, N.; Li, Z.; Ji, X.; Liu, S.; Sha, Y.; Song, X.; Zhang, J.; Hu, L.; Li, W. Lumbar spine localisation method based on feature fusion. *CAAI Trans. Intell. Technol.* **2023**, *8(3)*, 931–945. [CrossRef]
40. Wu, C.; Wei, X.; Li, S.; Zhan, A. MSTPose: Learning-Enriched Visual Information with Multi-Scale Transformers for Human Pose Estimation. *Electronics* **2023**, *12*, 3244. [CrossRef]
41. Nair, V.; Hinton, G.E. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010; pp. 807–814.

42. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 7132–7141.
43. Wang, Q.; Wu, B.; Zhu, P.; Li, P.; Zuo, W.; Hu, Q. ECA-Net: Efficient channel attention for deep convolutional neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–18 June 2020; pp. 11534–11542.
44. Gong, K.; Zhang, J.; Feng, J. Poseaug: A differentiable pose augmentation framework for 3d human pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Montreal, BC, Canada, 11–17 October 2021; pp. 8575–8584.
45. Gower, J.C. Generalized procrustes analysis. *Psychometrika* **1975**, *40*, 33–51. [CrossRef]
46. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; pp. 740–755.
47. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
48. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
49. Tolstikhin, I.O.; Houlsby, N.; Kolesnikov, A.; Beyer, L.; Zhai, X.; Unterthiner, T.; Yung, J.; Steiner, A.; Keysers, D.; Uszkoreit, J.; et al. Mlp-mixer: An all-mlp architecture for vision. *Adv. Neural Inf. Process Syst.* **2021**, *34*, 24261–24272.
50. Martinez, J.; Hossain, R.; Romero, J.; Little, J.J. A simple yet effective baseline for 3d human pose estimation. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2640–2649.
51. Wang, M.; Chen, X.; Liu, W.; Qian, C.; Lin, L.; Ma, L. DRPose3D: Depth ranking in 3D human pose estimation. In Proceedings of the 27th International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018; pp. 978–984.
52. Pavllo, D.; Feichtenhofer, C.; Grangier, D.; Auli, M. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7753–7762.
53. Lin, H.; Chiu, Y.; Wu, P. AMPose: Alternatively mixed global-local attention model for 3D human pose estimation. *arXiv* **2022**, arXiv:2210.04216.
54. Li, W.; Liu, H.; Guo, T.; Tang, H.; Ding, R. GraphMLP: A graph MLP-like architecture for 3D human pose estimation. *arXiv* **2022**, arXiv:2206.06420.
55. Zhou, X.; Huang, Q.; Sun, X.; Xue, X.; Wei, Y. Towards 3d human pose estimation in the wild: A weakly-supervised approach. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 398–407.
56. Zhou, K.; Han, X.; Jiang, N.; Jia, K.; Lu, J. Hemlets pose: Learning part-centric heatmap triplets for accurate 3d human pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 2344–2353.
57. Xu, J.; Yu, Z.; Ni, B.; Yang, J.; Yang, X.; Zhang, W. Deep kinematics analysis for monocular 3d human pose estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–18 June 2020; pp. 899–908.
58. Choi, J.; Shim, D.; Kim, H. Diffupose: Monocular 3d human pose estimation via denoising diffusion probabilistic model. *arXiv* **2022**, arXiv:2212.02796.