GraFormer: Graph-oriented Transformer for 3D Pose Estimation

Weixi Zhao, Weiqiang Wang, Yunjie Tian
University of Chinese Academy of Sciences, Beijing, China
{zhaoweixi19,tianyunjie19}@mails.ucas.ac.cn
wqwang@ucas.ac.cn

Abstract

In 2D-to-3D pose estimation, it is important to exploit the spatial constraints of 2D joints, but it is not yet well modeled. To better model the relation of joints for 3D pose estimation, we propose an effective but simple network, called GraFormer[1]*, where a novel transformer architecture is designed via embedding graph convolution layers after multi-head attention block. The proposed GraFormer is built by repeatedly stacking the GraAttention block and the ChebGConv block. The proposed GraAttention block is a new transformer block designed for processing graph-structured data, which is able to learn better features through capturing global information from all the nodes as well as the explicit adjacency structure of nodes. To model the implicit high-order connection relations among non-neighboring nodes, the ChebGConv block is introduced to exchange information between non-neighboring nodes and attain a larger receptive field. We have empirically shown the superiority of GraFormer through extensive experiments on popular public datasets. Specifically, GraFormer outperforms the state-of-the-art GraghSH [38] on the Human3.6M dataset yet only contains 18% parameters of it.*

1. Introduction

3D pose estimation has attracted much attention in recent years from computer vision community due to its numerous applications such as action recognition [16, 19, 37, 39], virtual reality [9, 23], etc. The 2D-to-3D human pose estimation task aims to convert 2D joint coordinates in images into corresponding 3D coordinates in the physical world. It is challenging since less information is contained in 2D coordinates. Previous works [25, 26] have shown that the 2D coordinates plus connection structure information are vital to learning feature representations for 3D pose estimation. However, the CNN-based method [26] is weak to directly model the connection structure information.

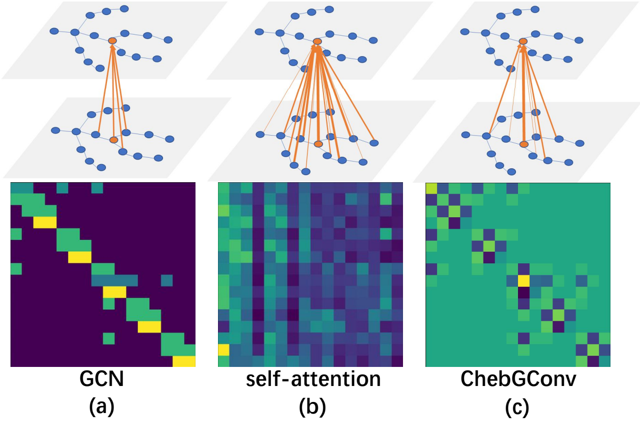

Figure 1. Node information transfer diagrams. (a): Normalized adjacency matrix. The adjacency matrix is hard to find all implicit relations between a node and other nodes, except the relation between it and its neighbors. (b): Attention weight matrix. The weight matrix can model the implicit relationship of all nodes based on the feature values. (c): Graph Laplacian matrix. The ChebGConv block can not only find many implicit relations but also remain the graph characteristics of 2D joints.

To better model the explicit structure information, recent works [7, 22, 38, 42] employ graph convolution networks (GCNs) to learn the representation of these graph-structured data. These techniques achieve good performances but suffer from the limited receptive field when learning better representations, where graph convolution filters [18] operate only on the first-order neighboring nodes, as illustrated in Figure 1 (a). Although this issue can be alleviated by stacking multiple GCN layers, the performance can still degrade due to the over-smoothing problem. Other methods are also proposed to enlarge the receptive field. For example, Zhao *et al.* [42] utilize non-local modules to increase the network receptive field. Lin *et al.* [21] utilize the recent popular transformer model to capture the global visual information over entire RGB images. In [21, 42], the self-attention modules of transformers facilitate the interaction among all

[1] Codes:https://github.com/zhaoweixi/GraFormer

nodes, and thus the relations of them are modeled, as illustrated in Figure 1 (b). However, the self-attention mechanism builds upon calculating the similarities of nodes and ignores the graph structure information among nodes (the adjacency relation of nodes).

To better utilize self-attention to model the structural representations of nodes in a graph, we propose a graph-based attention block (**GraAttention**) where the transformer and a graph convolution layer with a learnable adjacency matrix (referred to as **LAM-GConv**) [7, 18] are combined. Concretely, we replace the multiple layer perceptron (MLP) of conventional transformers with LAM-GConv. Although the conventional self-attention module facilitates the interaction among nodes in a graph, it ignores the graph structure information among nodes. In our method, the introduction of LAM-GConv can further boost the interaction among nodes by modeling graph structure. By introducing the LAM-GConv into the conventional transformer, the GraAttention block is able to learn better features through capturing global information as well as the adjacency structure of nodes.

Although the direct adjacent relations can be modeled by LAM-GConv, two non-neighboring nodes also have implicit relations. For example, some implicit relations for the two knee joints of humans exist, though they are not physically connected. We note that the conventional self-attention can only model the global relationship of nodes, and the LAM-GConv can only model direct adjacency relations, but the implicit relations among nodes are not well modeled. Thus, we propose to use ChebGConv block [6] to model the aforementioned implicit connection, as shown in Figure 1 (c). The proposed ChebGConv block can exchange information according to the high-order structure relations of nodes to attain a larger receptive field than the vanilla graph convolutions [18].

We demonstrate the effectiveness of our method by conducting comprehensive evaluation experiments and ablation studies on standard 3D benchmarks. The experimental results show that the proposed network, Graph-oriented Transformer (**GraFormer**) outperforms the state of the arts on Human3.6M [15] with only 0.65M parameters. In particular, for the 2D ground truth inputs, we achieve 13 first-place results out of 15 categories on Human3.6M. In addition, the superiority of GraFormer is also verified on ObMan [12], FHAD [10], and GHD [27]. The proposed GraFormer is task-independent and thus can be easily applied to other graph regression tasks.

The contribution of this paper can be summarized into two aspects. First, we propose a new transformer architecture, called GraAttention block, for processing graph-structured data. In the proposed GraAttentioin block, the conventional multi-head attention computation is followed by a graph convolution layer instead of MLP, which can capture not only the global information from all the nodes but also model the adjacency structure among nodes with a learnable adjacency matrix. Second, we propose to use the ChebGConv block to further model the implicit high-order connection relations among nodes. By stacking the GraAttentioin block and ChebGConv block, we construct a novel network called GraFormer, which can simultaneously model the explicit and implicit relationships between nodes, enlarge the receptive field of node information transmission, and effectively improve the performance of 2D-to-3D pose estimation tasks.

2. Related works

2.1. 3D Pose Estimation

One-stage 3D pose estimation methods usually directly estimate 3D pose using image features. Pavlakos *et al.* [28] first predict the 3D heat map and then yield the 3D pose. Mehta *et al.* [26] utilize transfer learning to produce multi-modal data, which are fused to predict 3D pose. Tekin *et al.* [34] utilize 3D YOLO model [29] combined with the temporal information to predict the 3D pose of hands and objects simultaneously. Li *et al.* [20] group and predict the joint points in a multi-tasks manner.

Differently, multi-stage methods first adopt CNN networks to detect 2D joint coordinates, and then they are used as inputs for 3D pose estimation. To yield 3D pose, Chen *et al.* [3] propose to match 2D coordinates with a 3D pose database. Based on 2D coordinates, Martinez *et al.* [25] propose a simple and effective network architecture consisting of linear layers, batch normalization, dropout, and ReLU activation function to regress the 3D pose. Simon *et al.* [31] propose a multi-view method that estimates 3D hand pose by triangulating multiple 2D joint coordinates. Hossain *et al.* [13] consider 2D coordinate information as a sequence and utilize temporal information to predict the 3D coordinates in a sequence manner.

2.2. Transformer-based Methods

Different from the conventional graph convolution [11, 18] which aggregates the information of neighbors equally, the GAT [36] method uses the self-attention to learn the weight of each node to aggregate the information of neighbors. The aGCN [40] method also learns the weights of neighboring nodes through the self-attention mechanism but it uses a different activation function and transformation matrix from GAT. Although the GAT and aGCN methods are effective, their receptive fields are still limited. To attain the global receptive field, Zhao *et al.* [42] utilize the non-local layer to learn the relationships between 2D joints. Lin *et al.* [21] modify the standard transformer encoder and adjust the dimension of encoder layers. However, such interaction ignores the adjacency structure of nodes. Com-

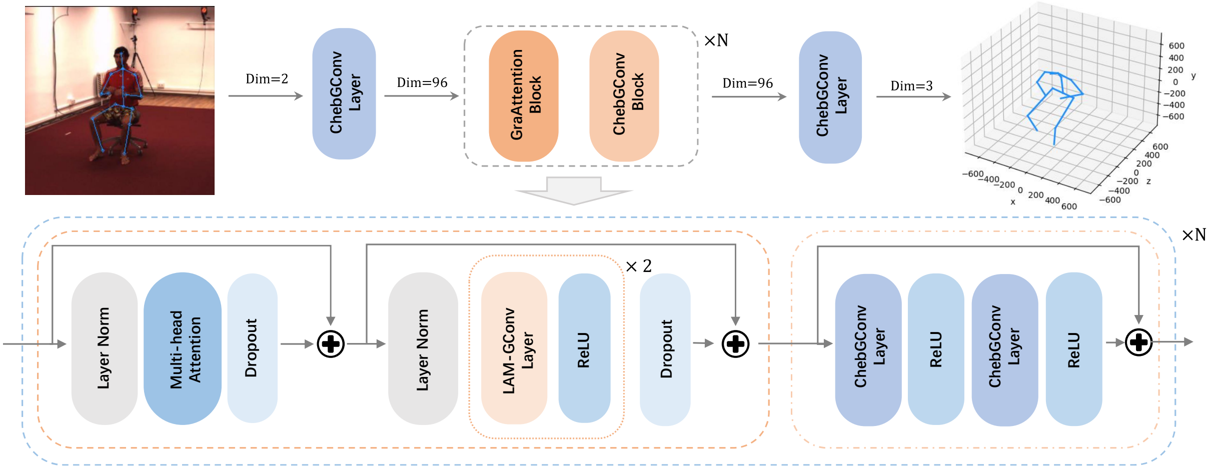

Figure 2. Framework of GraFormer. The core part is the stack of GraAttention block and ChebGConv block, which boosts performance for 2D-to-3D pose estimation tasks by exploiting relations among 2D joints.

paratively, our method enlarges the receptive field through self-attention and models graph structure by GCNs to effectively improve the performance on 3D pose estimation.

2.3. GCN-based methods

Recently, some works in the 3D pose estimation task [7, 11, 22, 38, 42] have achieved state-of-the-art results by using graph convolutional networks. Zhao *et al.* [42] propose semantic graph convolution, which learns the weights among neighbor joints. Non-local modules are used to enhance interaction among 2D joints. Doosti *et al.* [7] propose the modified graph pooling and unpooling operations to make the up-sampling and down-sampling procedures trainable for graph-structured data. Xu *et al.* [38] propose the graph hourglass network and adopt the SE Block [14] to fuse features extracted from different layers of the network. In this paper, we use a new transformer architecture by embedding graph convolution operations to improve the 3D pose estimation.

3. Method

As shown in Figure 2, the proposed GraFormer takes 2D joint coordinates as inputs and predicts the 3D pose as a target. It is built by repeatedly stacking GraAttention blocks and ChebGConv blocks. GraAttention blocks mainly consist of two parts, i.e., the conventional multi-head attention and two graph convolution layers with learnable adjacency matrix (LAM-GConv), and the residual shortcut is added for each of them. ChebGConv blocks mainly consist of two Chebyshev graph convolutional layers, and also a residual shortcut is added. More details are given in subsections 3.2, 3.3.

3.1. Preliminaries

Multi-head self-attention and graph convolution layers are the fundamental building blocks of the proposed GraFormer. Graph Convolution Networks (GCNs) have the ability to handle graph-structured data. Formally, let $X^l \in R^{j\times d_l}$ denote the input of the l-th layer of a GCN, which contains j nodes and each node corresponds to a d_l-dimensional representation vector. For example, the input of GraFormer, $X^0 \in R^{j\times 2}$, is the 2D joint coordinates of human bodies or hands. The output X^{l+1} of the l GCN layer can be computed by

$$X^{l+1} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}X^l\Theta\right), \tag{1}$$

where σ is the ReLU activation function, $\Theta \in R^{d_l\times d_{l+1}}$ denotes a learnable weight matrix, $A \in R^{j\times j}$ is the adjacency matrix, $\widetilde{A} = A + I_j$ and $\widetilde{D}$ is the diagonal node degree matrix, and I_j is the identity matrix of order j. We make $\widetilde{A}$ learnable for LAM-GConv in GraAttention block.

The gains achieved by transformer-based methods [21] mainly come from integrating the global information from a sequence. Concretely, let $X^l \in R^{j\times d_l}$ denote the input of the l-th multi-head attention layer, and it is first fed into three fully connected layers parallelly to produce three outputs, i.e., query Q^l, key K^l and value V^l with the same dimension of $j \times d$. Then, the self-attention output Y^l is computed by

$$Y^l = \tau(\frac{Q^l \cdot {K^l}^T}{\sqrt{d}})V^l, \tag{2}$$

where τ is the softmax function on the row.

3.2. GraAttention block

As illustrated in Figure 2, in the application of 3D pose estimation of the human body, the pose of human body in images is represented by sixteen 2D joint coordinates. After they are first pre-processed by a ChebGConv layer, the corresponding output is fed into the architecture which is built by repeatedly stacking GraAttention blocks and ChebGConv blocks. Before it is processed by the multi-head self-attention in GraAttention block, the input feature vectors are first normalized by layer normalization (LN) [1], which is commonly used in transformer models [35].

The multi-head self-attention block is inherited from the transformer encoder layers [35]. Different from other applications of transformer in 3D pose estimation like [21], we remove the MLP layer from the standard transformer since we observe that the existence of MLP makes the system performance degrade greatly. Then, the self-attention output is processed by a dropout [32]. It should be noted that positional encoding is not used in our system. Although each dimension of the output of the multi-head attention block contains all 2D joint information, which is decided by the characteristic of multi-head attention computation, spatial topological relations among nodes are not encoded. Since the standard transformer can only encode the linear topological relation, it is not consistent with the general graph topology structure, so the removal of it aims to avoid the introduction of noises for the following modeling of spatial topological relation.

Finally, the output is normalized by the LN layer and followed by a combination of two GCN layers and two ReLU activation layers. In the GraAttention blocks, the GCN layers are added in order to encode the information of graph topology structure. It should be noted that different from the standard vanilla graph convolution layer, we make the adjacency matrix to be learnable and it is shared by multiple LAM-GConv layers so that the GCN layer becomes more flexible to learn graph-structured data. We name the GCN layer with a learnable adjacency matrix as LAM-GConv. At the end, a dropout is added to make the system more robust. It can be seen that we propose a new transformer block specific for processing graph-structured data, called GraAttention block, which is a combination of multi-head self-attention without position information encoding and GCN layers. Both parts include a shortcut connection so as to make the training easier, as shown in Figure 2.

3.3. ChebGConv block

Although we have utilized the vanilla graph convolution layer in the GraAttention block to model the non-linear topological relation among nodes, it should be noted that the topological relation is obtained by learning and is virtual, since the adjacency matrix consists of learnable parameters. Different from graph convolution layers in the GraAttention block, those in ChebGConv blocks have the fixed adjacency matrix, which truly encodes the objective topological relation among nodes. We aim to use ChebGConv blocks to model the implicit high-order graph structure information. For Chebyshev graph convolution layers [6], the normalized graph Laplacian is computed by

$$L = \ I - \tilde{D}^{-\frac{1}{2}} A \tilde{D}^{-\frac{1}{2}}, \tag{3}$$

and Chebyshev graph convolution is defined as

$$X^{l+1} = \sum_{k=0}^{K-1} T_k \left(\tilde{L}\right) X^l \theta_k, \tag{4}$$

where $T_k\left(x\right) = 2xT_{k-1}\left(x\right) - T_{k-2}\left(x\right)$ denotes the Chebyshev polynomial of degree k, $T_0 \ = \ 1, T_1 \ = \ x$, and $\tilde{L} \in R^{j\ \times j}$ denotes the rescaled Laplacian, $\tilde{L} = 2L/\lambda_{\max} - I$, $\lambda_{\max}$ is the maximum eigenvalue of L. $\theta_k \in R^{d_l \times d_{l+1}}$ denotes the trainable parameters in the graph convolutional layer. Since the convolution kernel is a K-order polynomial graph Laplacian, ChebGConv block is able to fuse information among the K-hop neighbors of a joint, which brings a larger receptive field. Our experimental results also show that ChebGConv blocks indeed boost the system performance. Though ChebGConv involves more expensive computation than a vanilla GCN layer, its total computation cost increases slightly since the size of the graph is small with only 16 joints and the topological structure is very simple.

3.4. Training

To train the GraFormer, we apply a loss function to the final output to minimize the error between the 3D predictions and ground truth. Given dataset $S = \left\{J_i^{2d}, J_i^{3d}\right\}_{i=1}^{N}$, where $J_i^{2d} \ \in \ R^{j\times 2}$ is 2D coordinates of joints of human bodies or hands, $j = 16$ for the human datasets and 21 for the hand datasets, and $J_i^{3d} \in R^{j\times 3}$ is the 3D ground truth coordinates. N denotes the total number of training examples.

We use the mean squared errors (MSE) as the loss between 3D predictions and ground truth coordinates, i.e.,

$$L_{mse} = \frac{1}{N} \sum_{i=1}^{N} \left(\left\| \tilde{J}_i^{3d} - J_i^{3d} \right\|_2^2 \right), \tag{5}$$

where $\tilde{J}_i^{3d} \in R^{j\times 3}$ denotes the predicted 3D coordinates, $\left\|\cdot\right\|_2$ is the 2-norm of the vector.

4. Experiments

In this section, we first introduce the experimental details and training settings. Next, we compare GraFormer with other state-of-the-art methods and analyze the results. Finally, we conduct ablation studies to verify the effectiveness of GraFormer.

Methods	Direct.	Discuss	Eating	Greet	Phone	Photo	Pose	Purch.	Sitting	SittingD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
Pavlakos [28] CVPR17	67.4	71.9	66.7	69.1	72.0	77.0	65.0	68.3	83.7	96.5	71.7	65.8	74.9	59.1	63.2	71.9
Metha [26] 3DV17	52.6	64.1	55.2	62.2	71.6	79.5	52.8	68.6	91.8	118.4	65.7	63.5	49.4	76.4	53.5	68.6
Zhou [44] ICCV17	54.8	60.7	58.2	71.4	62.0	65.5	53.8	55.6	75.2	111.6	64.1	66.0	51.4	63.2	55.3	64.9
Martinez [25] ICCV17	51.8	56.2	58.1	59.0	69.5	78.4	55.2	58.1	74.0	94.6	62.3	59.1	65.1	49.5	52.4	62.9
Sun [33] ICCV17	52.8	54.8	54.2	54.3	61.8	**53.1**	53.6	71.7	86.7	61.5	67.2	53.4	47.1	61.6	53.4	59.1
Fang [8] AAAI18	50.1	54.3	57.0	57.1	66.6	73.3	53.4	55.7	72.8	88.6	60.3	57.7	62.7	47.5	50.6	60.4
Yang [41] CVPR18	51.5	58.9	50.4	57.0	62.1	65.4	49.8	52.7	69.2	85.2	57.4	58.4	**43.6**	60.1	47.7	58.6
Hossain [13] ECCV18	48.4	50.7	57.2	55.2	63.1	72.6	53.0	51.7	66.1	80.9	59.0	57.3	62.4	46.6	49.6	58.3
Zhao [42] CVPR19	48.2	60.8	51.8	64.0	64.6	53.6	51.1	67.4	88.7	**57.7**	73.2	65.6	48.9	64.8	51.9	60.8
Ci [5] ICCV19	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
Liu [22] ECCV20	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
Xu [38] CVPR21	**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
Ours	**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**
Martinez [25] (GT)	37.7	44.4	40.3	42.1	48.2	54.9	44.4	42.1	54.6	58.0	45.1	46.4	47.6	36.4	40.4	45.5
Hossain [13] (GT)	35.2	40.8	37.2	37.4	43.2	44.0	38.9	35.6	42.3	44.6	39.7	39.7	40.2	32.8	35.5	39.2
Zhao [42] (GT)	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 [22] (GT)	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 [38] (GT)	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
Ours (GT)	**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**

Table 1. Quantitative evaluation results using MPJPE in millimeter on Human3.6M [15]. Best in bold.

4.1. Experimental Details

Dataset We use 3 popular hand datasets, including ObMan [12], FHAD [10], GHD [27], and 1 human pose dataset Human3.6M [15] to evaluate the GraFormer.

ObMan is a large synthetic dataset of hand-object interaction scenarios. The hands are generated from MANO [30] and the objects are selected from the Shapenet [2] dataset. The ObMan dataset contains 141,550 training samples and 6,285 evaluation samples. Each sample contains an RGB image, a depth image, a 3D mesh of the hand and object, and 3D coordinates for the hand.

FHAD [10] contains videos of manipulating different objects from the first-person perspective. There are a total of 21,501 frames of images, where 11,019 frames are used for training and 10,482 frames for testing.

GHD [27] contains 143,449 images without objects, and 188,050 images with objects. The no object part of the dataset consists of 141 sets of images, each containing 1024 images, except the last set. We use the first 130 sets for training and the last 11 sets for testing.

Human3.6M [15] is the most widely used dataset in the 3D human pose estimation task. It provides 3.6M accurate 3D poses captured by the MoCap system in the indoor environment. It contains 15 actions performed by seven actors taken by four cameras. There are two common evaluation protocols for splitting training and testing set in previous methods [22,25,38,42]. The first protocol uses subjects S1, S5, S6, S7, and S8 for training, and S9 and S11 for testing. Errors are calculated after the ground truth and predictions are aligned with the root joints. The second protocol uses subjects S1, S5, S6, S7, S8, and S9 for training, and S11 for testing. We conduct experiments using the first protocol. In this way, there are 1,559,752 frames for training, and 543,344 frames for testing.

Evaluation Metric We follow the same evaluation metric as [42]. The evaluation metric is the Mean Per Joint Position Error (MPJPE) in millimeters, which is calculated between the ground truth and the predicted 3D coordinates across all cameras and joints after aligning the predefined root joints (the pelvis joint).

Training Settings For the three hand datasets ObMan, FHAD and GHD, we directly take 2D image coordinates and 3D camera coordinates as the inputs and ground truth. The 2D coordinates and 3D ground truth provided by ObMan can be used by simply converting the ground truth from meter to millimeter. The 3D ground truth provided by FHAD are world coordinates, and we use the extrinsic matrix to calculate the corresponding camera coordinates. The 3D ground truths of GHD are already the camera coordinate system, and the 2D coordinates need to be cropped, scaled and restored. We use the hand coordinates as input but object coordinates for all hand datasets. For Human3.6M, because of the multiple camera views, it needs to be normalized according to [42] before training and evaluation.

In our experiment, we set the number of N in Figure 2 to 5 and adopt 4 heads for self-attention. Different from the feature dimension value of 64 or 128 in previous works [38, 42], we set the middle feature dimension of the model to 96 for GraFormer with a dropout rate of 0.25. We adopt Adam [17] optimizer for optimization with an initial learning rate of 0.001 and mini-batches of 64. For Human3.6M, we multiply the learning rate by 0.9 every 75000 steps. For hand datasets, the learning rate decays by 0.9 every 30 epochs. We train GraFormer for 100 epochs on Human3.6M, 900 epochs on Obman and GHD and 3000 epochs on FHAD.

4.2. Performance and Comparison

In this section, we evaluate the GraFormer on several popular datasets and analyze the performances compared with other state-of-the-art methods.

Performance on Human Pose Dataset The results of previous works on Human3.6M can be categorized into two groups as shown in Table 1. The top group methods take images as inputs and then yield 3D poses using the learned features. In our method, we adopt 2D coordinates detected by Cascaded Pyramid Network (CPN) [4]. In the bottom group, we compare GraFormer with the most advanced methods using the same 2D inputs. The results show that GraFormer surpasses all the methods, which indicates the superiority of our method. The bottom group methods take 2D ground truth as inputs to predict 3D pose coordinates directly. Compared with the previous methods, GraFormer achieves the best performance which indicates the effectiveness of our method. In particular, GraFormer obviously improves scores in direction, eating, greet, phone, pose, smoke, and wait with only 0.65M parameters (18% of [38]). Interestingly, we find that these actions have a large range of motions (the average distance between adjacent joints is 55.25 pixels), which implies longer distances among 2D joints than the actions of Photo and SittingD (the average distance between adjacent joints is 49.89 pixels). This means GraFormer has a more powerful capability to capture information of 2D joints with a larger range of motions.

Performance on Hand Datasets We verify GraFormer on hand datasets via comparing with Linear model [25], Graph U-Net [7] and SemGCN [42] since all of these methods regress 3D pose results by taking 2D ground truth coordinates as inputs. The difference between human pose data and hand data is the number of joints and skeleton structure.

The results on three hand datasets are shown in Table 2. Results show that GraFormer achieves the best performance on ObMan and GHD. In particular, GraFormer surpasses other methods by a large margin on GHD. We note that small datasets are not friendly to self-attention. Even so, GraFormer still beats the Linear model and Graph U-Net on the extremely small dataset FHAD. Note that for the hand data set, we use SemGConv [42] to replace ChebGConv [6] in GraFormer, since SemGConv is easier to train on small datasets

Generalization Ability To evaluate the generalization capabilities of our model, we use our model trained on Human3.6M and evaluate on the test set of MPI-INF-3DHP [26]. The test set of MPI-INF-3DHP includes 3 settings, studio with green screen(GS), studio without green screen(noGS) and outdoors. We use the same metric as [26], including 3D Percentage of Correct Keypoints (3D PCK) and the Area Under the Curve (AUC). Although the GraAttention is sensitive to values, our method still outperforms most state-of-the-art methods on MPI-INF-3DHP while only using Human3.6M for training. The related results are shown in Table 3.

Methods	ObMan	FHAD	GHD	Methods	ObMan	FHAD	GHD
Martinez [25]	23.64	26.15	39.25	Zhao [42]	2.34	8.49	13.1
Doosti [7]	7.63	13.82	8.45	Ours	1.71	10.42	2.20

Table 2. MPJPE (mm) results compared with Linear model, SemGCN and Graph U-Net on three hand datasets, ObMan, FHAD and GHD.

Methods	Training Date	PCK				AUC
		GS	noGS	Outdoor	Avg	All
Martinez [25]	H36M	49.8	42.5	31.2	42.5	17.0
Mehta [26]	H36M	70.8	62.3	58.8	64.7	31.7
Yang [41]	H36M+MPII	-	-	-	69.0	32.0
Zhou [44]	H36M+MPII	71.1	64.7	72.7	69.2	32.5
Luo [24]	H36M	71.3	59.4	65.7	65.6	33.2
Ci [5]	H36M	74.8	70.8	77.3	74.0	36.7
Zhou [43]	H36M+MPII	75.6	71.3	80.3	75.3	38.0
Xu [38]	H36M	81.5	81.7	75.2	80.1	45.8
ours	H36M	80.1	77.9	74.1	79.0	43.8

Table 3. Results on MPI-INF-3DHP test set.

Runtime On an Nvidia RTX2080Ti GPU, the inference of a single batch with size 64 requires just 0.015 seconds.

Methods	Params	MPJPE(mm)	Methods	Params	MPJPE(mm)
GAT [36]	0.16M	82.9	FC [25]	4.29M	45.5
ST-GCN [39]	0.27M	57.4	Pre-agg [22]	4.22M	37.8
SemGCN [42]	0.43M	43.8	GraphSH [38]	3.70M	35.8
GraFormer-small	0.12M	38.9	GraFormer	0.65M	35.2

Table 4. Results on Human3.6M dataset under different parameter configurations.

4.3. Ablation Study

Discussion on model parameters We start the ablation experiments by comparing GraFormers with different parameter configurations to other methods on the Human3.6M dataset, and the results are released in Table 4. We report the results of our models of two configurations to show that our method can achieve better results with fewer parameters than other methods. GraFormer-small has only 2 layers, and the feature dimension is 64 with a dropout rate of 0.1. GraFormer has 5 layers, and the feature dimension is set to 96 with a dropout rate of 0.25. The GraFormer achieves better results with even much fewer parameters than GraphSH, etc. The lightweight version, GraFormer-small, with 72% fewer parameters than SemGCN, beats SemGCN by 4.9.

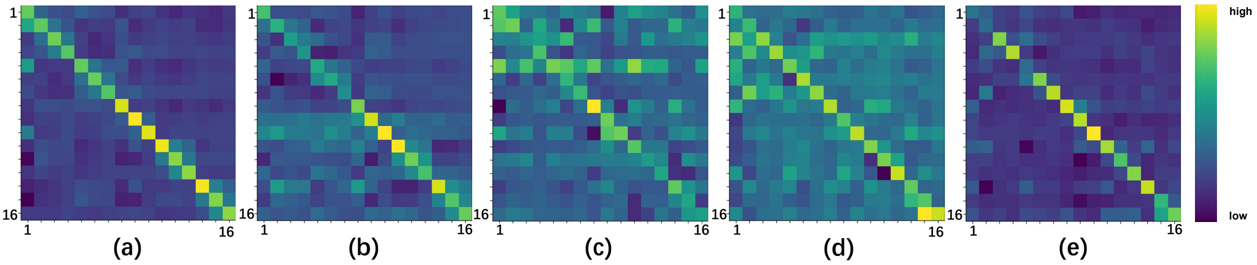

Figure 3. Visualization of the learned adjacency matrices of different LAM-GConv layers in the GraAttention.

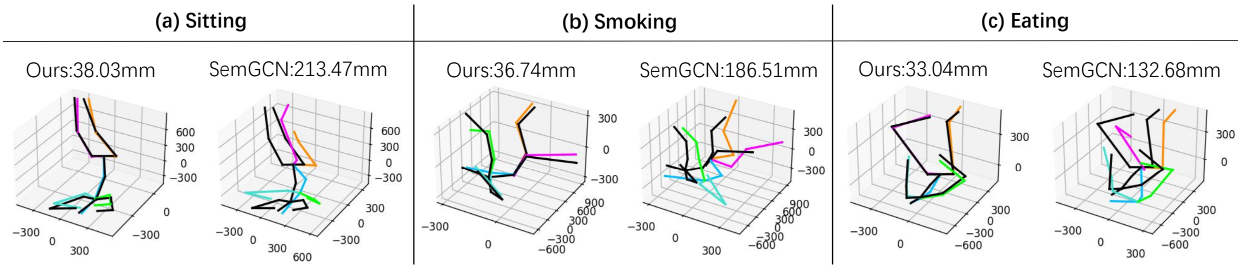

Figure 4. Quantitatively compare our method with SemGCN on different actions of Human3.6M [15]. The 3D ground truth and 3D predictions are shown in black and color, respectively.

It should be noted that in the top group of Table 1, we set N to 6, model dimension to 128, and dropout rate to 0.45. However, even so, GraFormer has only 37% as many parameters as GraphSH [38].

Effects of GraFormer Modules Next, we test GraFormer modules on Human3.6M [15], ObMan [12], and FHAD [10]. We design 5 models by removing or replacing GraFormer's modules to test the effects of our method. All parameters are the same for the 5 models if not particularly indicated. Specifically, Model-T is formed by replacing the stack of GraAttention and ChebGConv block with transformer encoders. Model-C removes GraAttention from GraFormer. Model-M replaces GraAttention with self-attention. Model-AT removes the ChebGConv from GraFormer. And model-AM reserves MLP compared to GraFormer. The results are shown in Table 5.

Models	Human3.6M	ObMan	FHAD
model-T	51.76	15.54	20.14
model-C	47.81	8.51	16.30
model-M	42.19	5.02	13.52
model-AT	37.78	7.29	14.39
model-AM	42.44	3.46	13.49
GraFormer	35.17	3.29	11.68

Table 5. MPJPE (mm) results on Human3.6M, ObMan and FHAD by removing or replacing GraFormer's modules.

From the results of model-T, we can find that the transformer is poor for 2D-to-3D pose estimation. This is because the transformer ignores graph structure information. The results of model-C and model-M are worse than GraFromer, which shows that GraAttention is necessary and more effective than self-attention. The loss of ChebGConv block in model-AT brings worse performance, which indicates the effectiveness of ChebGConv block. The performance also degrades when the MLP layer is plugged after self-attention in GraAttention, which verifies that the MLP layer impedes the learning of 3D poses actually. Figure 6 shows the test errors of model-AM and GraFormer. We find that the test error of model-AM is almost unchanged at 250 epochs while the test error of GraFomer still decreases until below 12mm. This reconfirms that the MLP layer degrades transformer in the 2D-to-3D pose estimation.

Comparison of different graph convolutional layers In Table 6, we show the results of Graformer on Human3.6M using Chebyshev graph convolution [6] or semantic graph convolution [42]. The ChebGConv contains a vital parameter, the order of the graph Laplacian polynomial, we show the results for orders 1, 2, and 3.

Visualization Figure 3 shows the visualization results of learned adjacency matrices of LAM-GConv layers of GraAttention block. In Figure 3(a), the color of some 3×3 regions is obviously brighter than other regions, which indicates that interaction among these joints takes greater

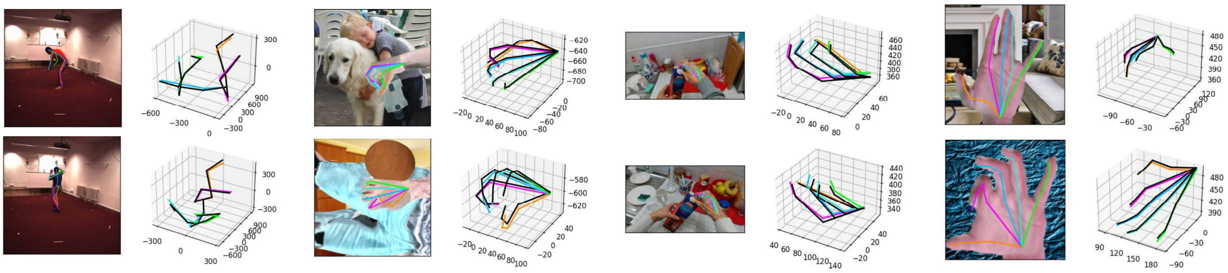

Figure 5. Skeleton results predicted by GraFormer on Human3.6M [15], ObMan [12], FHAD [10], and GHD [27].

	cheb k=1	cheb k=2	cheb k=3	SemGConv
dim 64	39.04	36.19	39.31	38.53
dim 96	37.77	35.17	38.13	38.56

Table 6. MPJPE (mm) results on Human3.6M using different graph convolutional layers.

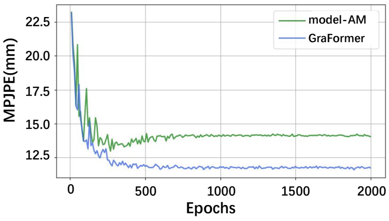

Figure 6. Test errors of model-AM and GraFormer on FHAD.

weights and these joints are closely connected. Interestingly, we note that joints 2-4, 5-7, 11-13 and 14-16 are four limbs of the human, which are activated in 3 × 3 regions. This implies that the relations of joints on a limb are strongly connected, and the GraFormer is able to find these relations effectively. The regions of Figure 3 (b) to (d) become larger, which shows that GraFormer finds long-range relationships in these layers. The Figure 3(e) illustrates that the interaction regions become much smaller, which implies that joints mainly retain their own information, and a little information interacts among joints.

Figure 4 shows the results of a quantitative comparison of our method with SemGCN [42] over three actions, sitting, smoking, and eating, which have a large range of motions. The results show that our method can significantly improve the performance on some types of actions.

In Figure 5, we show the predicted 3D body results on Human3.6M [15](columns 1-2) and hand results on ObMan [12](columns 3-4), FHAD [10] (columns 5-6) and GHD [27] (columns 7-8). The images in columns 1, 3, 5, and 7 are skeleton figures drawn using 2D ground truth. In columns 2, 4, 6, and 8, the colored skeletons are drawn using the 3D predictions, and the black skeletons are drawn using the 3D ground truth. We can find that our method is able to estimate the 3D poses accurately using 2D coordinates. It shows that our method could effectively learn 3D poses by exploiting the relationship among 2D joints.

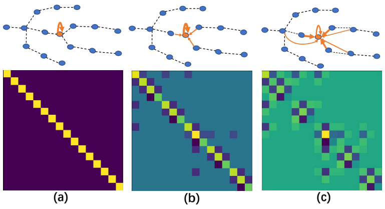

Figure 7. Visualization of graph Laplacian of ChebGConv.

Figure 7 is a visualization of graph Laplacian of different orders of Chebyshev graph convolution. The top row shows the schematic diagrams of joint information aggregation according to the bottom row. The width of the line implies the weights between 2D joints. The bottom row shows the visualization of the corresponding Laplacian matrix with 0-order (a), 1-order (b) and 2-order (c) respectively. It is easy to find that the bigger orders matrix activates more 2D joints, which implies that bigger orders of the Laplacian matrix have the capability to find more implicit relations.

5. Conclusions

In this paper, we present a new graph-oriented transformer network GraFormer and apply it to 2D-to-3D pose estimation task. In the proposed GraFormer, two function blocks (GraAttention block and ChebGConv blocks) are presented and the stacking of them makes the GraFormer can not only fuse the information of all the nodes but also model the implicit and explicit topological structure. Extensive experiments have been conducted on the popular benchmarks and the results show that the proposed GraFormer achieves the state-of-the-art performances but contains much fewer model parameters.

Acknowledgement This work is supported by NSFC projects under Grant 61976201, and NSFC Key Projects of International (Regional) Cooperation and Exchanges under Grant 61860206004.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 4

[2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 5

[3] Ching-Hang Chen and Deva Ramanan. 3d human pose estimation= 2d pose estimation+ matching. In *CVPR*, pages 7035–7043, 2017. 2

[4] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *CVPR*, pages 7103–7112, 2018. 6

[5] Hai Ci, Chunyu Wang, Xiaoxuan Ma, and Yizhou Wang. Optimizing network structure for 3d human pose estimation. In *ICCV*, pages 2262–2271, 2019. 5, 6

[6] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *arXiv preprint arXiv:1606.09375*, 2016. 2, 4, 6, 7

[7] Bardia Doosti, Shujon Naha, Majid Mirbagheri, and David J Crandall. Hope-net: A graph-based model for hand-object pose estimation. In *CVPR*, pages 6608–6617, 2020. 1, 2, 3, 6

[8] Hao-Shu Fang, Yuanlu Xu, Wenguan Wang, Xiaobai Liu, and Song-Chun Zhu. Learning pose grammar to encode human body configuration for 3d pose estimation. In *AAAI*, volume 32, 2018. 5

[9] Ji Gan and Weiqiang Wang. In-air handwritten english word recognition using attention recurrent translator. *Neural Computing and Applications*, 31(7):3155–3172, 2019. 1

[10] Guillermo Garcia-Hernando, Shanxin Yuan, Seungryul Baek, and Tae-Kyun Kim. First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In *CVPR*, pages 409–419, 2018. 2, 5, 7, 8

[11] Liuhao Ge, Zhou Ren, Yuncheng Li, Zehao Xue, Yingying Wang, Jianfei Cai, and Junsong Yuan. 3d hand shape and pose estimation from a single rgb image. In *CVPR*, pages 10833–10842, 2019. 2, 3

[12] Yana Hasson, Gul Varol, Dimitrios Tzionas, Igor Kalevatykh, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning joint reconstruction of hands and manipulated objects. In *CVPR*, pages 11807–11816, 2019. 2, 5, 7, 8

[13] Mir Rayat Imtiaz Hossain and James J Little. Exploiting temporal information for 3d human pose estimation. In *ECCV*, pages 68–84, 2018. 2, 5

[14] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 3

[15] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *TPAMI*, 36(7):1325–1339, 2013. 2, 5, 7, 8

[16] Songyao Jiang, Bin Sun, Lichen Wang, Yue Bai, Kunpeng Li, and Yun Fu. Skeleton aware multi-modal sign language recognition. In *CVPR*, pages 3413–3423, 2021. 1

[17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[18] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 1, 2

[19] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In *CVPR*, pages 3595–3603, 2019. 1

[20] Moran Li, Yuan Gao, and Nong Sang. Exploiting learnable joint groups for hand pose estimation. In *AAAI*, volume 35, pages 1921–1929, 2021. 2

[21] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. In *CVPR*, pages 1954–1963, 2021. 1, 2, 3, 4

[22] Kenkun Liu, Rongqi Ding, Zhiming Zou, Le Wang, and Wei Tang. A comprehensive study of weight sharing in graph networks for 3d human pose estimation. In *ECCV*, pages 318–334. Springer, 2020. 1, 3, 5, 6

[23] Duo Lu and Linzhen Luo. Fmkit: An in-air-handwriting analysis library and data repository. In *CVPR Workshop on Computer Vision for Augmented and Virtual Reality, 2020*, 2020. 1

[24] Chenxu Luo, Xiao Chu, and Alan Yuille. Orinet: A fully convolutional network for 3d human pose estimation. *BMVC*, 2018. 6

[25] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *ICCV*, pages 2640–2649, 2017. 1, 2, 5, 6

[26] Dushyant Mehta, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt. Monocular 3d human pose estimation in the wild using improved cnn supervision. In *3DV*, pages 506–516. IEEE, 2017. 1, 2, 5, 6

[27] Franziska Mueller, Florian Bernard, Oleksandr Sotnychenko, Dushyant Mehta, Srinath Sridhar, Dan Casas, and Christian Theobalt. Ganerated hands for real-time 3d hand tracking from monocular rgb. In *CVPR*, pages 49–59, 2018. 2, 5, 8

[28] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G Derpanis, and Kostas Daniilidis. Coarse-to-fine volumetric prediction for single-image 3d human pose. In *CVPR*, pages 7025–7034, 2017. 2, 5

[29] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *CVPR*, pages 7263–7271, 2017. 2

[30] Javier Romero, Dimitrios Tzionas, and Michael J Black. Embodied hands: Modeling and capturing hands and bodies together. *TOG*, 36(6):1–17, 2017. 5

[31] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *CVPR*, pages 1145–1153, 2017. 2

[32] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way

to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014. 4

[33] Xiao Sun, Jiaxiang Shang, Shuang Liang, and Yichen Wei. Compositional human pose regression. In *ICCV*, pages 2602–2611, 2017. 5

[34] Bugra Tekin, Federica Bogo, and Marc Pollefeys. H+ o: Unified egocentric recognition of 3d hand-object poses and interactions. In *CVPR*, pages 4511–4520, 2019. 2

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 4

[36] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017. 2, 6

[37] Junwu Weng, Chaoqun Weng, and Junsong Yuan. Spatio-temporal naive-bayes nearest-neighbor (st-nbnn) for skeleton-based action recognition. In *CVPR*, pages 4171–4180, 2017. 1

[38] Tianhan Xu and Wataru Takano. Graph stacked hourglass networks for 3d human pose estimation. In *CVPR*, pages 16105–16114, 2021. 1, 3, 5, 6, 7

[39] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI*, 2018. 1, 6

[40] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph r-cnn for scene graph generation. In *ECCV*, pages 670–685, 2018. 2

[41] Wei Yang, Wanli Ouyang, Xiaolong Wang, Jimmy Ren, Hongsheng Li, and Xiaogang Wang. 3d human pose estimation in the wild by adversarial learning. In *CVPR*, pages 5255–5264, 2018. 5, 6

[42] Long Zhao, Xi Peng, Yu Tian, Mubbasir Kapadia, and Dimitris N Metaxas. Semantic graph convolutional networks for 3d human pose regression. In *CVPR*, pages 3425–3435, 2019. 1, 2, 3, 5, 6, 7, 8

[43] Kun Zhou, Xiaoguang Han, Nianjuan Jiang, Kui Jia, and Jiangbo Lu. Hemlets pose: Learning part-centric heatmap triplets for accurate 3d human pose estimation. In *ICCV*, pages 2344–2353, 2019. 6

[44] Xingyi Zhou, Qixing Huang, Xiao Sun, Xiangyang Xue, and Yichen Wei. Towards 3d human pose estimation in the wild: a weakly-supervised approach. In *ICCV*, pages 398–407, 2017. 5, 6