Semantic Graph Convolutional Networks for 3D Human Pose Regression

Long Zhao[1] Xi Peng[2] Yu Tian[1] Mubbasir Kapadia[1] Dimitris N. Metaxas[1]
[1]Rutgers University [2]Binghamton University
{lz311,yt219,mk1353,dnm}@cs.rutgers.edu, xpeng@binghamton.edu

Abstract

In this paper, we study the problem of learning Graph Convolutional Networks (GCNs) for regression. Current architectures of GCNs are limited to the small receptive field of convolution filters and shared transformation matrix for each node. To address these limitations, we propose Semantic Graph Convolutional Networks (SemGCN), a novel neural network architecture that operates on regression tasks with graph-structured data. SemGCN learns to capture semantic information such as local and global node relationships, which is not explicitly represented in the graph. These semantic relationships can be learned through end-to-end training from the ground truth without additional supervision or hand-crafted rules. We further investigate applying SemGCN to 3D human pose regression. Our formulation is intuitive and sufficient since both 2D and 3D human poses can be represented as a structured graph encoding the relationships between joints in the skeleton of a human body. We carry out comprehensive studies to validate our method. The results prove that SemGCN outperforms state of the art while using 90% fewer parameters. The code can be found at `https://github.com/garyzhao/SemGCN`.

1. Introduction

Convolutional Neural Networks (CNNs) have successfully tackled classic computer vision problems such as image classification [12, 29, 31, 52], object detection [19, 46, 55, 63, 74, 79] and generation [43, 58, 71, 73, 80], where the input image has a grid-like structure. However, many real-world tasks, *e.g.*, molecular structures, social networks and 3D meshes, can only be represented in the form of irregular structures, where CNNs have limited applications.

In order to address this limitation, Graph Convolutional Networks (GCNs) [17, 28, 49] have been introduced recently as a generalization of CNNs that can directly deal with a general class of graphs. They have achieved state-of-the-art performance when applied to 3D mesh deformation [45, 64], image captioning [70], scene understanding [68], and video recognition [66, 67]. These works utilize GCNs to model relations of visual objects for classification. In this paper, we investigate using deep GCNs for regression, which is another core problem of computer vision with many real-world applications.

However, GCNs cannot be directly applied to regression problems due to the following limitations in baseline methods [28, 64, 67]. First, to handle the issue that graph nodes may have various numbers of neighborhoods, the convolution filter shares the same weight matrix for all nodes, which is not comparable with CNNs. Second, previous methods are simplified by restricting the filters to operate in a one-step neighborhood around each node according to the guidance of [28]. The receptive field of the convolution kernel is limited to one due to this formulation, which severely impairs the efficiency of information exchanging especially when the network goes deeper.

In this work, we propose a novel graph neural network architecture for regression called *Semantic Graph Convolutional Networks (SemGCN)* to address the above limitations. Specifically, we investigate learning semantic information encoded in a given graph, *i.e.*, the local and global relations of nodes, which is not well-studied in previous works. SemGCN does not rely on hand-crafted constraints [10, 13, 51] to analyze the patterns for a specific application, and thus can be easily generalized to other tasks.

In particular, we study SemGCN for *2D to 3D human pose regression*. Given a 2D human pose (and the optional relevant image) as input, we aim to predict the locations of its corresponding 3D joints in a certain coordinate space. Using SemGCN to formulate this problem is intuitive. Both 2D and 3D poses are able to be naturally represented by a canonical skeleton in the form of 2D or 3D coordinates, and SemGCN can explicitly exploit their spatial relations, which are crucial for understanding human actions [67].

Our work makes the following contributions. First, we propose an improved graph convolution operation called *Semantic Graph Convolution (SemGConv)* which is derived from CNNs. The key idea is to learn channel-wise weights for edges as priors implied in the graph, and then combine them with kernel matrices. This significantly improves the power of graph convolutions. Second, we introduce

SemGCN where *SemGConv* and non-local [65] layers are interleaved. This architecture captures both local and global relationships among nodes. Third, we present an end-to-end learning framework to show that SemGCN can also incorporate external information, such as image content, to further boost the performance for 3D human pose regression.

The effectiveness of our approach is validated by comprehensive evaluation with a rigorous ablation study and comparisons with state of the art on standard 3D benchmarks. Our approach matches the performance of state-of-the-art techniques on Human3.6M [24] using only 2D joint coordinates as inputs and 90% fewer parameters. Meanwhile, our approach outperforms state of the art when incorporating image features. Furthermore, we also show the visual results of SemGCN, which demonstrate the effectiveness of our approach qualitatively. Note that the proposed framework can be easily generalized to other regression tasks, and we leave this for future work.

2. Related Work

Graph convolutional networks. Generalizing CNNs to inputs with graph-like structures is an important topic in the field of deep learning. In the literature, there have been several attempts to use recursive neural networks to process data represented in graph domains as directed acyclic graphs [14]. GNNs were introduced in [17, 28, 49] as a more common solution to handle arbitrary graph data. The principle of constructing GCNs on graph generally follows two streams: the spectral perspective and the spatial perspective. Our work belongs to the second stream [28, 39, 60], where the convolution filters are applied directly on the graph nodes and their neighbors.

Recent studies on computer vision have achieved state-of-the-art performance by leveraging GCNs to model the relations among visual objects [68, 70] or temporal sequences [66, 67]. This paper follows the spirit of them, while we explore applying GCNs for regression tasks, especially, 2D to 3D human pose regression.

3D pose estimation. Lee and Chen [30] first investigated inferring 3D joints from their corresponding 2D projections. Later approaches either exploited nearest neighbors to refine the results of pose inference [18, 25] or extracted hand-crafted features [1, 23, 47] for later regression. Other methods created over-complete bases which are suitable for representing human poses as sparse combinations [2, 4, 44, 62, 77]. More and more studies focus on making use of deep neural networks to find the mapping between 2D and 3D joint locations. A couple of algorithms directly predicted 3D pose from the image [75], while others combined 2D heatmaps with volumetric representation [41], pairwise distance matrix estimation [36] or image cues [56] for 3D human pose regression.

Recently, it has been proven that 2D pose information is crucial for 3D pose estimation. Martinez *et al.* [34] introduced a simple yet effective method which predicted 3D key points purely based on 2D detections. Fang *et al.* [13] further extended this approach through pose grammar networks. These works focus on 2D to 3D pose regression, which are most relevant to the context of this paper.

Other methods use synthetic datasets which are generated from deforming a human template model with the ground truth [8, 42, 48] or introduce loss functions involving high-level knowledge [40, 53, 69] in addition to joints. They are complementary to the others. Remaining works target at exploiting temporal information [11, 18, 21, 57] for 3D pose regression. They are out of the scope of this paper, since we aim at handling the 2D pose from one single image. However, our method can be easily extended to sequence inputs, and we leave it for future work.

3. Semantic Graph Convolutional Networks

We propose a novel graph network architecture to handle general regression tasks involving data that can be represented in the form of graphs. We first provide the background of GCNs and related baseline method. Then we introduce the detailed design of SemGCN.

We assume that graph data share the same topological structure, such as human skeletons [10, 26, 61, 67], 3D morphable models [33, 45, 72] and citation networks [50]. Other problems which own different graph structures in the same domain, *e.g.*, protein-protein interaction [60] and quantum chemistry [15], are out of the scope of this paper. This assumption makes it possible to learn priors implied in the graph structure, which motivates SemGCN.

3.1. ResGCN: A Baseline

We will start by briefly recapping the 'vanilla' GCNs as proposed in [28]. Let $\mathcal{G} = \{\mathbf{V}, \mathbf{E}\}$ denote a graph where $\mathbf{V}$ is the set of K nodes and $\mathbf{E}$ are edges, while $\vec{\boldsymbol{x}}_i^{(l)} \in \mathbb{R}^{D_l}$ and $\vec{\boldsymbol{x}}_i^{(l+1)} \in \mathbb{R}^{D_{l+1}}$ are the representations of node i before and after the l-th convolution respectively. A graph based convolutional propagation can be applied to node i in two steps. First, node representations are transformed by a learnable parameter matrix $\mathbf{W} \in \mathbb{R}^{D_{l+1} \times D_l}$. Second, these transformed node representations are gathered to node i from its neighboring nodes $j \in \mathcal{N}(i)$, followed by a non-linear function (ReLU [37]). If node representations are collected into a matrix $\mathbf{X}^{(l)} \in \mathbb{R}^{D_l \times K}$, the convolution operation can be written as:

$$\mathbf{X}^{(l+1)} = \sigma\Big(\mathbf{W}\mathbf{X}^{(l)}\tilde{\mathbf{A}}\Big), \tag{1}$$

where $\tilde{\mathbf{A}}$ is symmetrically normalized from $\mathbf{A}$ in conventional GCNs. $\mathbf{A} \in [0, 1]^{K \times K}$ is the adjacency matrix of $\mathcal{G}$, and we have $a_{ij} = 1$ for node $j \in \mathcal{N}(i)$ and $a_{ii} = 1$.

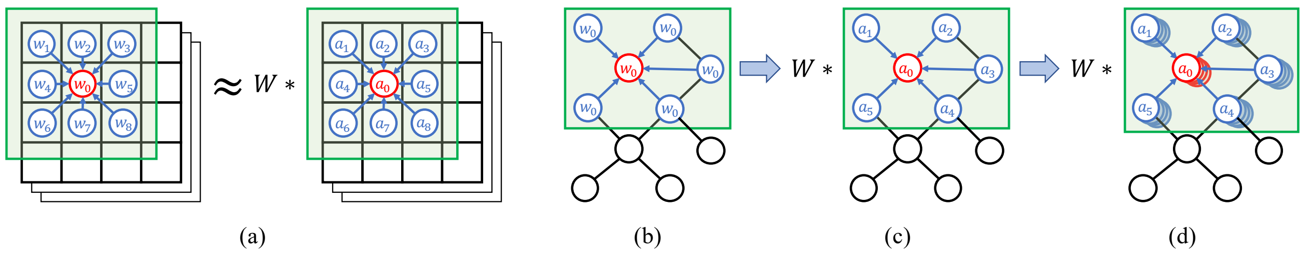
w_1 w_2 w_3 w_4 w_0 w_5 w_6 w_7 w_8
$\approx W *$
a_1 a_2 a_3 a_4 a_0 a_5 a_6 a_7 a_8
$W *$
$W *$
(a)
(b)
(c)
(d)

Figure 1. Illustration of the proposed Semantic Graph Convolutions. (a) The 3×3 convolution kernel of CNNs (highlighted in *green*) learns a different transformation matrix $\boldsymbol{w}_i$ for each position inside the kernel. We approximate it by learning a weighting vector $\boldsymbol{a}_i$ for each position and a shared transformation matrix $\mathbf{W}$. (b) Conventional GCNs only learn a shared transformation matrix $\boldsymbol{w}_0$ for all nodes. (c) The approximated formulation in (a) can be directly extended to (b): we add an additional learnable weight a_i for each node in the graph. (d) We further extend (c) to learn a channel-wise weighting vector $\boldsymbol{a}_i$ for each node. After combining them with the vanilla transformation matrix $\mathbf{W}$ in GCNs, we can obtain a new kernel operation for graphs which owns comparable learning capability with CNNs. The learned weight vectors show the local semantic relationships of neighboring nodes implied in the graph.

Wang *et al.* [64] rephrased a very deep graph network based on Eq. 1 with residual connections [20] to learn the mapping between image features and 3D vertexes. We adopt its network architecture and treat it as our baseline which is denoted as ResGCN.

There are two clear drawbacks in Eq. 1. First, in order to make the graph convolution work on nodes with arbitrary topologies, the learned kernel matrix $\mathbf{W}$ is shared for all edges. As a result, the relationships of neighboring nodes, or the internal structure in the graph, is not well exploited. Second, previous works only collect features from the first-order neighbors of each node. This is also limited because the receptive field is fixed to 1.

3.2. Semantic Graph Convolutions

We show that learning semantic relationships of neighboring nodes implied in edges of the graph is effective to address the limitation of the shared kernel matrix.

The proposed approach builds on concepts from CNNs. Fig. 1(a) shows a CNN with a convolution kernel of size 3×3. It learns nine transformation matrices which are different from each other to encode features inside the kernel in the spatial dimension. This makes the operation own expressive power to model feature patterns contained in images. We find that this formulation can be approximated by learning a weighting vector $\vec{\boldsymbol{a}}_i$ for each position, and then combining them with a shared transformation matrix $\mathbf{W}$. If we represent the image feature map as a square grid graph whose nodes represent pixels, this approximated formulation can be directly extended to GCNs as shown in Fig. 1(c).

To this end, we propose *Semantic Graph Convolution (SemGConv)*, where we add a learnable weighting matrix $\mathbf{M} \in \mathbb{R}^{K \times K}$ to conventional graph convolutions. And then Eq. 1 is transformed to:

$$\mathbf{X}^{(l+1)} = \sigma\Big(\mathbf{W}\mathbf{X}^{(l)}\rho_i(\mathbf{M} \odot \mathbf{A})\Big), \tag{2}$$

where ρ_i is Softmax nonlinearity which normalizes the input matrix across all choices of node i; $\odot$ is an element-wise operation which returns m_{ij} if $a_{ij} = 1$ or negatives with large exponents saturating to zero after ρ_i; $\mathbf{A}$ serves as a mask which forces that for node i in the graph, we only compute the weights of its neighboring nodes $j \in \mathcal{N}(i)$.

As illustrated in Fig. 1(d), we can further extend Eq. 2 by learning a set of $\mathbf{M}_d \in \mathbb{R}^{K \times K}$, so that a different weighting matrix is applied to each channel d of output node features:

$$\mathbf{X}^{(l+1)} = \Big\Vert_{d=1}^{D_{l+1}} \sigma\Big(\vec{\boldsymbol{w}}_d\mathbf{X}^{(l)}\rho_i(\mathbf{M}_d \odot \mathbf{A})\Big), \tag{3}$$

where $\|$ represents channel-wise concatenation, and $\vec{\boldsymbol{w}}_d$ is the d-th row of the transformation matrix $\mathbf{W}$.

Comparison to previous GCNs. Both aGCN [68] and GAT [60] follow a self-attention strategy [59] to compute the hidden representations of each node in the graph by attending over its neighbors. They aim to estimate a weighting function *depending on inputs* for edges to modulate information flow throughout the graph. By contrast, we target at learning *input-independent* weights for edges which represent priors implied in the graph structures, *e.g.*, how one joint influences other body parts in human pose estimation.

The edge importance weighting mask introduced in ST-GCN [67] is the most related work to ours but with following two sharp differences. First, no Softmax nonlinearity is leveraged after weighting by [67], while we find it stabilizes the training and obtains better results, since the contributions of nodes to their neighbors are normalized by Softmax. Second, ST-GCN applies only one single learnable mask to all channels, but our Eq. 3 learns channel-wise different weights for edges. As a result, our model owns better capability to fit the data mapping.

3.3. Network Architecture

Capturing global and long-range relationships among nodes in the graph is able to efficiently address the problem

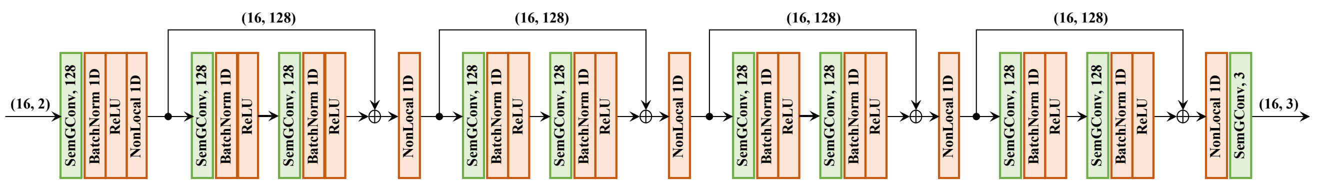

Figure 2. Example of the proposed Semantic Graph Convolutional Networks. The building block of our network is one residual block [20] built by two *SemGConv* layers with 128 channels, followed by one non-local layer [65]. This block is repeated four times. All *SemGConv* layers are followed by batch normalization [22] and a ReLU activation [37] except the last one.

of the limited receptive field. However, in order to maintain the behavior of GCNs, we restrict the feature updating mechanism by computing responses between nodes based on their representations other than learning new convolution filters. Therefore, we follow the non-local mean concept [5, 65] and define the operation as:

$$\vec{\boldsymbol{x}}_i^{(l+1)} = \vec{\boldsymbol{x}}_i^{(l)} + \frac{W_x}{K} \sum_{j=1}^{K} f(\vec{\boldsymbol{x}}_i^{(l)}, \vec{\boldsymbol{x}}_j^{(l)}) \cdot g(\vec{\boldsymbol{x}}_j^{(l)}), \quad (4)$$

where W_x is initialized as zero; f is a pairwise function to compute the affinity between node i and all other j; g computes the representation of the node j. In practice, Eq. 4 can be implemented by the non-local layers proposed in [65].

Based on Eq. 3 and 4, we propose a new network architecture for regression tasks called *Semantic Graph Convolutional Networks*, where *SemGConv* and non-local layers are interleaved to capture local and global semantic relations of nodes. Fig. 2 shows an example. In this work, SemGCN in all blocks has the same structure, which consists of one residual block [20] built by two *SemGConv* layers with 128 channels, and then followed by one non-local layer. This block is repeated several times to make the network deeper. At the beginning of the network, one *SemGConv* is used for mapping the inputs into the latent space; and we have an additional *SemGConv* which projects the encoded features back to the output space. All *SemGConv* layers are followed by batch normalization [22] and a ReLU activation [37] except the last one. Note that if *SemGConv* layers are replaced with vanilla graph convolutions and all non-local layers are removed, SemGCN downgrades to ResGCN in Sect. 3.1.

Intuitively, SemGCN can be regarded as a form of neural message passing system [15] where the forward pass has two phases: messages are updated locally and then refined by the global state of the system. These two phases take turns to process messages so that the efficiency of information exchanging is improved for the whole system.

4. 3D Human Pose Regression

In this section, we present a novel end-to-end trainable framework which incorporates SemGCN in Sect. 3 with image features for 3D human pose regression.

4.1. Framework Overview

Recently, it is proven that accurate 3D pose estimation can be directly obtained by using only 2D human poses as system inputs [34]. Formally, given a series of 2D joints $\mathbf{P} \in \mathbb{R}^{K\times 2}$ and their corresponding 3D joints $\mathbf{J} \in \mathbb{R}^{K\times 3}$ in a predefined camera coordinate system (K is the numbers of joints), the system aims to learn a regression function $\mathcal{F}^*$ which minimizes the following error over a dataset containing N human poses:

$$\mathcal{F}^* = \underset{\mathcal{F}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\mathcal{F}(\mathbf{P}_i), \mathbf{J}_i). \quad (5)$$

We argue that image content is able to offer important cues for solving ambiguous cases, such as the classic turning ballerina optical illusion. Therefore, we extend Eq. 5 by treating image content as an additional constraint. The extended formulation can be denoted as:

$$\mathcal{F}^* = \underset{\mathcal{F}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\mathcal{F}(\mathbf{P}_i|I_i), \mathbf{J}_i), \quad (6)$$

where I_i is the image containing the aligned human pose of the 2D joints $\mathbf{P}_i$. In practice, $\mathbf{P}$ may be obtained as 2D ground truth locations under known camera parameters or from a 2D joint detector. In the latter case, the 2D detector has already encoded the perceptual features of the input image during the training process. This observation motivates the design of our framework.

An overview of our framework is shown in Fig. 3. The whole framework consists of two neural networks. Given an image, one deep convolutional network is leveraged for 2D joints prediction; at the same time, it also serves as a backbone network and image features are pooled from its intermediate layers. Since 2D and 3D joint coordinates can be encoded in a human skeleton, the proposed SemGCN is used for automatically capturing the patterns embedded in the spatial configuration of the human joints. It predicts 3D coordinates according to the 2D pose as well as perceptual features from the backbone network.

Note that our framework effectively reduces to Eq. 5 when image features are not considered. As we demonstrate

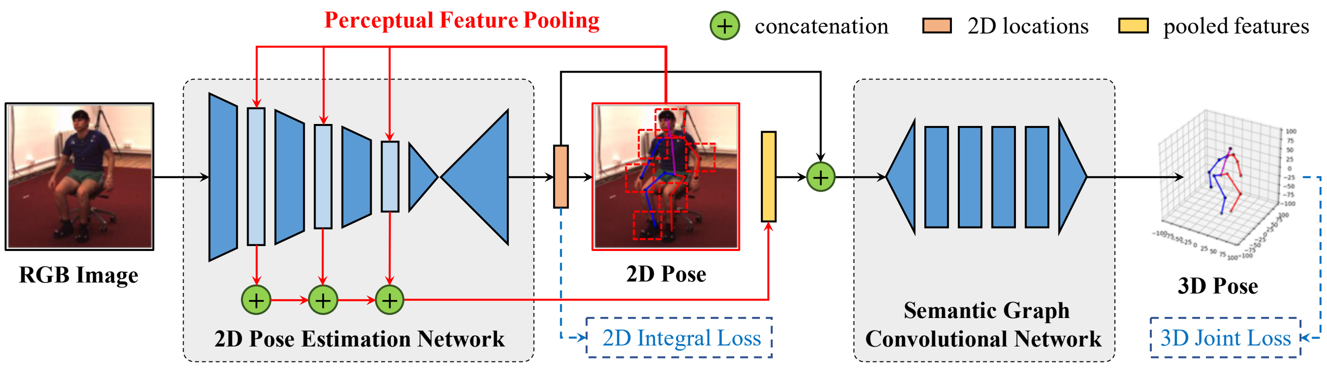

Figure 3. Illustration of our framework incorporating image features for 3D human pose estimation. We pre-train a 2D pose estimation network to predict 2D joint locations. It also serves as a backbone network where we pool image features. The proposed SemGCN predicts 3D pose from 2D joints as well as image features. Note that the whole framework is end-to-end trainable.

in experiments, SemGCN manages to effectively encode the mapping from 2D to 3D poses, and the performance can be further boosted when incorporating image content.

4.2. Perceptual Feature Pooling

ResNet [20] and HourGlass [38] are widely adopted in conventional human pose detection problems. Empirically, we employ ResNet as the backbone network since its intermediate layers provide hierarchical features from images which are useful in computer vision problems such as object detection and segmentation [46, 74].

Given the coordinate of each 2D joint in the input image, we pool features from multiple layers in ResNet. In particular, we concatenate features extracted from layer *conv_1* to *conv_4* using RoIAlign [19]. These perceptual features are then concatenated with the 2D coordinates and fed into SemGCN. Note that since all joints in the input image share the same scale, we pool the features in a squared bounding box centered on each joint with a fixed size, *i.e.*, the mean bone length of the skeleton. This is illustrated in Fig. 3.

4.3. Loss Function

Most previous regression-based methods directly minimize the mean squared errors (MSE) of the predicted and ground truth joint positions [6, 34, 57, 76] or bone vectors [53]. Following the spirit of them, we employ a simple combination of joint and bone constraints in human poses as our loss function, which is defined as:

$$\mathcal{L}(\mathcal{B},\mathcal{J}) = \underbrace{\sum_{i=1}^{M} ||\tilde{\mathbf{B}}_i - \mathbf{B}_i||^2}_{\text{bone vectors}} + \underbrace{\sum_{i=1}^{K} ||\tilde{\mathbf{J}}_i - \mathbf{J}_i||^2}_{\text{joint positions}}, \qquad (7)$$

where $\mathcal{J} = \{\tilde{\mathbf{J}}_i | i = 1, \ldots, K\}$ are predicted 3D joint coordinates and $\mathcal{B} = \{\tilde{\mathbf{B}}_i | i = 1, \ldots, M\}$ are bones computed from $\mathcal{J}$; $\mathbf{J}_i$ and $\mathbf{B}_i$ are corresponding ground truth in the dataset. Each bone is a directed vector pointing from the starting joint to its associated parent as defined in [53].

5. Experiments

In this section, we first introduce settings and implementation details for evaluation, and then conduct an ablation study on components in our method, and finally report our results and comparisons with state-of-the-art methods.

5.1. Implementation Details

As suggested in the previous works [34, 53, 75], it is impossible to train an algorithm to infer the 3D joint locations in an arbitrary coordinate space system. Therefore, we choose to predict 3D pose in the camera coordinate system [11, 32, 41, 57], which makes the 2D to 3D regression problem similar across different cameras.

We make use of the ground truth 2D joint locations provided in the dataset to align the 3D and 2D poses following the setting of [75]. This implies that we implicitly use the camera calibration information. Then, we zero-center both the 2D and 3D poses around the predefined root joint, *i.e.*, the pelvis joint, which is in line with previous works and the standard protocol. Moreover, we do not use data augmentation during the training process for simplicity.

Network training. We use ResNet50 in [54] as our backbone network, which is compatible with the integral loss and pre-trained on ImageNet [9]. During training, we employ Adam [27] for optimization with a initial learning rate of 0.001 and use mini-batches of size 64. The learning rate is dropped with a decay rate of 0.5 when the loss on the validation set saturates. We initialize weights of the graph network using the initialization described in [16].

In our preliminary experiments, we observe that the direct end-to-end training of the whole network from scratch cannot achieve the best performance. We argue that this is likely because of the highly non-linear dependency between the graph network and conventional deep convolutional module for 2D pose estimation. Therefore, we utilize a multi-stage training scheme which is more stable and effective in practice. We first train the backbone network for 2D pose estimation from images using 2D ground truth. As described in [54], the integral loss is used. Then we fix the 2D pose estimation module and train the graph network for 2D to 3D pose regression using the output of 2D estimation module and the 3D ground truth. In this stage, the loss function defined in Eq. 7 is employed. At last, the whole network is fine-tuned with all data. Both integral loss and Eq. 7 are activated. Note that the final stage is end-to-end.

5.2. Datasets and Evaluation Protocols

Our proposed approach is comprehensively evaluated on the most widely used dataset for 3D human pose estimation: Human3.6M [24], following the standard protocol.

Datasets. *Human3.6M* [24] is currently the largest publicly available dataset for 3D human pose estimation. This dataset contains 3.6 million of images captured by a MoCap System in an indoor environment, where 7 professional actors perform 15 everyday activities such as walking, eating, sitting, making a phone call and engaging in a discussion. Both 2D and 3D ground truth are available for supervised learning. Following the setting of [75], the videos are down-sampled from 50fps to 10fps for both the training and testing sets to reduce redundancy. We also use *MPII* dataset [3], the state-of-the-art benchmark for 2D human pose estimation, for pre-training the 2D pose detector and qualitatively evaluation in the experiment.

Evaluation protocols. For Human3.6M [24], there are two common evaluation protocols using different training and testing data split in the literature. One standard protocol uses all 4 camera views in subjects S1, S5, S6, S7 and S8 for training and the same 4 camera views in subjects S9 and S11 for testing. Errors are calculated after the ground truth and predictions are aligned with the root joint. We refer to this as *Protocol #1*. The other protocol makes use of six subjects S1, S5, S6, S7, S8 and S9 for training, and evaluation is performed on every 64th frame of S11. It also utilizes a rigid transformation to further align the predictions with the ground truth. This protocol is referred as *Protocol #2*. In this work, we use Protocol #1 in all the experiments for evaluation, since it is more challenging and matches the settings of our method.

The evaluation metric is the Mean Per Joint Position Error (MPJPE) in millimeter between the ground truth and the predicted 3D coordinates across all cameras and joints after aligning the pre-defined root joints (the pelvis joint). We use this metric for evaluation in the following sections.

Our network predicts the normalized locations of 3D joints. During testing, to calibrate the scale of the outputs,

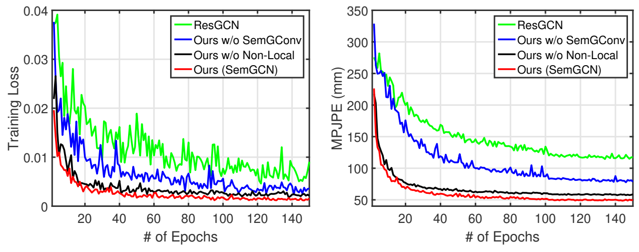

Figure 4. Training curves (left) and testing errors (right) of our networks with different settings. Our full model has lower and smoother learning curves as well as better testing results.

Method	# of params	MPJPE (mm)
ResGCN	0.14M	94.4
Ours w/o SemGConv	0.30M	65.9
Ours w/o Non-Local	0.27M	52.5
Ours (SemGCN)	0.43M	43.8

Table 1. 2D to 3D pose regression errors and the parameter numbers of our networks with different settings on Human3.6M dataset [24]. Our full model achieves the best performance.

we require that the sum of length of all 3D bones is equal to that of a canonical skeleton as shown in [41, 75, 78]. Therefore, we follow the method in [75] for calibration.

Configurations. Our method is evaluated with the following two different configurations for 3D human pose estimation on Human3.6M.

Configuration #1. We only leverage 2D joints of the human pose as inputs. SemGCN in Sect. 3 is trained for regression and the *SemGConv* layer defined in Eq. 2 is utilized. 2D ground truth (GT) or outputs from pre-trained 2D pose detectors are used for training and testing. In order to be in line with the setting of previous works [13, 34], we employ HourGlass [38] (HG) as the 2D detector. It is first pre-trained on MPII and then fine-tuned on Human3.6M. Only the joint loss in Eq. 7 is employed.

Configuration #2. We use 2D images as inputs, and the proposed framework in Sect. 4 is trained for regression. The channel-wise weighted *SemGConv* defined in Eq. 3 is employed. ResNet50 [20] is utilized as the backbone network for 2D pose estimation and feature pooling (RN w/ FP).

5.3. Ablation Study

We conduct the ablation study on the proposed method in Sect. 3. Configuration #1 is employed. Our SemGCN consists of two main components: *SemGConv* and non-local layers. To verify them, we train two variants of SemGCN: one only uses *SemGConv* and the other only uses non-local layers. Then we evaluate them together with the baseline method in Sect. 3.1 (ResGCN) and our full model in Sect. 3.3 on Human3.6M. Note that in order to get rid of the

Protocol #1	Direct.	Discuss	Eating	Greet	Phone	Photo	Pose	Purch.	Sitting	SittingD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
Ionescu *et al*. [24] PAMI'16	132.7	183.6	132.3	164.4	162.1	205.9	150.6	171.3	151.6	243.0	162.1	170.7	177.1	96.6	127.9	162.1
Tekin *et al*. [57] CVPR'16	102.4	147.2	88.8	125.3	118.0	182.7	112.4	129.2	138.9	224.9	118.4	138.8	126.3	55.1	65.8	125.0
Zhou *et al*. [77] CVPR'16	87.4	109.3	87.1	103.2	116.2	143.3	106.9	99.8	124.5	199.2	107.4	118.1	114.2	79.4	97.7	113.0
Du *et al*. [11] ECCV'16	85.1	112.7	104.9	122.1	139.1	135.9	105.9	166.2	117.5	226.9	120.0	117.7	137.4	99.3	106.5	126.5
Chen & Ramanan [7] CVPR'17	89.9	97.6	89.9	107.9	107.3	139.2	93.6	136.0	133.1	240.1	106.6	106.2	87.0	114.0	90.5	114.1
Pavlakos *et al*. [41] CVPR'17	67.4	71.9	66.7	69.1	72.0	77.0	65.0	68.3	83.7	96.5	71.7	65.8	74.9	59.1	63.2	71.9
Mehta *et al*. [35] 3DV'17	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 *et al*. [75] ICCV'17	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 *et al*. [34] ICCV'17	51.8	56.2	58.1	59.0	69.5	78.4	55.2	58.1	74.0	94.6	62.3	59.1	65.1	49.5	52.4	62.9
Sun *et al*. [53] ICCV'17	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 *et al*. [13] AAAI'18	50.1	54.3	57.0	57.1	66.6	73.3	53.4	55.7	72.8	88.6	60.3	57.7	62.7	47.5	50.6	60.4
Yang *et al*. [69] CVPR'18	51.5	58.9	**50.4**	57.0	62.1	65.4	49.8	52.7	69.2	85.2	**57.4**	58.4	43.6	60.1	47.7	58.6
Hossain & Little [21] ECCV'18	48.4	**50.7**	57.2	55.2	63.1	72.6	53.0	**51.7**	**66.1**	80.9	59.0	57.3	62.4	**46.6**	49.6	58.3
Ours (HG)	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
Ours (RN w/ FP)	**47.3**	60.7	51.4	60.5	**61.1**	**49.9**	**47.3**	68.1	86.2	**55.0**	67.8	61.0	**42.1**	60.6	**45.3**	**57.6**
Ours (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

Table 2. Quantitative comparisons of Mean Per Joint Position Error (mm) between the estimated pose and the ground truth on Human3.6M [24] under Protocol #1. We show the results of our model (Sect. 3) trained and tested with the 2D predictions of HourGlass [38] (HG) as inputs using Configuration #1, and the results of our network presented in Sect. 4 which incorporate image features (RN w/ FP) during training and testing under Configuration #2. We also show an upper bound of our method which uses 2D ground truth (GT) as the input for training and testing. The top two best methods of each action are highlighted in bold and underlined respectively.

Method	# of params	MPJPE (mm)
aGCN [68] / GAT [60]	0.16M	82.9
ST-GCN [67]	0.27M	57.4
FC [34]	4.29M	45.5 (62.9)
FC [34] w/ PG [13]	-	43.3 (60.4)
Ours	0.43M	43.8 (61.1)
Ours w/ PG [13]	-	42.5 (59.8)

Table 3. Evaluation of 2D to 3D pose regression on Human3.6M datasets [24]. Errors within the parentheses are computed by using the 2D estimations from HG [38] as inputs during training and testing. Otherwise, 2D ground truth is utilized. Our method advances other GCN-based approaches by 20% and achieves the state-of-the-art performance using 90% fewer parameters than [34].

influence from the 2D pose detector, we report the results using 2D ground truth for training and testing.

All models are trained based on the architecture shown in Fig. 2 after 200 epochs. Results are shown in Table 1. We also show their curves of training losses and testing errors in Fig. 4. We can see that our model with more components performs better than those with fewer components, which indicates the efficacy of each part of our algorithm. Moreover, our networks with *SemGConv* have much smoother training curves which demonstrates that learning local relations among nodes stabilizes the training process as well.

5.4. Evaluation on 3D Human Pose Regression

2D to 3D pose regression. We first evaluate our method for 2D to 3D pose regression and only Configuration #1 is leveraged. We compared ours with three GCN-based methods: aGCN [68], GAT [60] and ST-GCN [67], and two state-of-the-art approaches: FC [34] and PG [13]. As ST-GCN [67] is designed for videos, we set its temporal dimension to one for images. PG proposed a framework to refine the 3D pose, which is complementary to FC and ours. Therefore, we also report our results refined by PG.

The results are reported in Table 3. Our approach outperforms other GCN-based approaches by a large margin (about 20%). More importantly, our method achieves the state-of-the-art performance with around 90% fewer parameters than [34]. Meanwhile, the runtime of SemGCN reduces 10% compared with [34], which is around 1.8ms for a forward pass on a Titan Xp GPU. After we refined our results by PG, our approach obtains the best performance.

Comparison with the state of the art. We show evaluation results under Configuration #1 and #2. Note that many leading methods have sophisticated frameworks or learning strategies. Some of them aim at in-the-wild images [54, 69, 75] or exploit temporal information [11, 18, 21, 57], while some other approaches use complex loss functions [53, 69]. These methods are with different research targets compared to ours. Therefore, we include some of them during evaluation for completeness. Table 2 reports the results.

We find that our method using only 2D joints as inputs is able to match the state-of-the-art performance. After incorporating image features, our network sets the new state of the art. Especially, we improve previous methods by a large margin for the action of directions, taking photo, posing, sitting down, walking dog and walking together. We hypothesize that this is due to the severe self-occlusions in these actions, while they can be effectively encoded by our SemGCN using relations within graphs. The result of our method trained and tested with ground truth 2D joint loca-

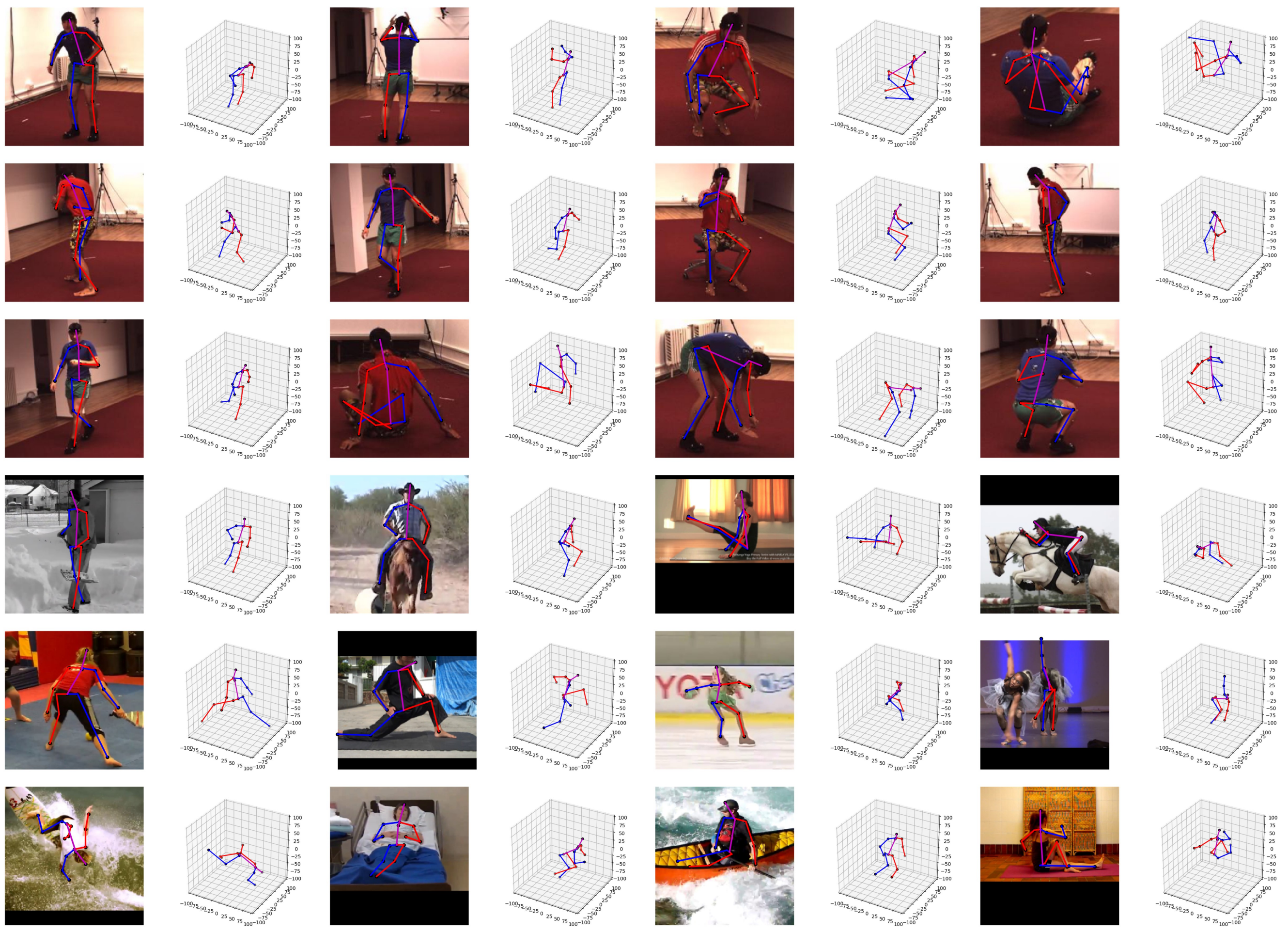

Figure 5. Visual results of our method on Human3.6M [24] and MPII [3]. The first three rows show results on Human3.6M. Results on MPII are drawn in the last three rows. The bottom row shows four typical failure cases. Best viewed in color.

tions shows our upper bound.

Qualitative results. In Fig. 5, we show the visual results of our method on Human3.6M and the test set of MPII. MPII contains in-the-wild images with novel human poses which are not similar to the examples in Human3.6M. As seen, our method is able to accurately predict 3D pose for both indoor and most in-the-wild images. It indicates that SemGCN can effectively encode relationships among joints and further generalize them to some novel cases.

The bottom row of Fig. 5 also shows typical failure cases of our method. These images include extreme poses which are largely different from those in Human3.6M. Our method failed to handle them but still yields reasonable 3D poses.

6. Conclusions

We present a novel model for 3D human pose regression, the Semantic Graph Convolutional Networks (SemGCN). Our method has addressed the key challenges of GCNs by learning local and global semantic relations among nodes in the graph. The combination of SemGCN and features pooled from image content further improves the performance in 3D human pose estimation. Comprehensive evaluation results show that our network obtains state-of-the-art performance with 90% fewer parameters compared with the closest work. The proposed SemGCN also opens up many possible directions for future works. For example, how to incorporate temporal information, such as videos, into SemGCN becomes a natural question.

Acknowledgments. This work was funded partly by grant BAAAFOSR-2013-0001 to Dimitris Metaxas. This work was also partly supported by NSF 1763523, 1747778, 1733843 and 1703883 Awards. Mubbasir Kapadia was funded partly by NSF IIS-1703883, NSF S&AS-1723869, and DARPA SocialSim-W911NF-17-C-0098.

References

[1] Ankur Agarwal and Bill Triggs. Recovering 3D human pose from monocular images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 28(1):44–58, 2006.

[2] Ijaz Akhter and Michael J Black. Pose-conditioned joint angle limits for 3D human pose reconstruction. In *CVPR*, pages 1446–1455, 2015.

[3] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. In *CVPR*, pages 3686–3693, 2014.

[4] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J Black. Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In *ECCV*, pages 561–578, 2016.

[5] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *CVPR*, 2005.

[6] Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *CVPR*, pages 4733–4742, 2016.

[7] Ching-Hang Chen and Deva Ramanan. 3D Human Pose Estimation= 2D Pose Estimation+ Matching. In *CVPR*, 2017.

[8] Wenzheng Chen, Huan Wang, Yangyan Li, Hao Su, Zhenhua Wang, Changhe Tu, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Synthesizing training images for boosting human 3D pose estimation. In *International Conference on 3D Vision (3DV)*, pages 479–488, 2016.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009.

[10] Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *CVPR*, pages 1110–1118, 2015.

[11] Yu Du, Yongkang Wong, Yonghao Liu, Feilin Han, Yilin Gui, Zhen Wang, Mohan Kankanhalli, and Weidong Geng. Marker-less 3D Human Motion Capture with Monocular Image Sequence and Height-Maps. In *ECCV*, pages 20–36, 2016.

[12] Mohamed Elhoseiny, Yizhe Zhu, Han Zhang, and Ahmed Elgammal. Link the head to the "beak": Zero shot learning from noisy text description at part precision. In *CVPR*, 2017.

[13] 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*, 2018.

[14] Paolo Frasconi, Marco Gori, and Alessandro Sperduti. A general framework for adaptive processing of data structures. *IEEE transactions on Neural Networks*, 9(5):768–786, 1998.

[15] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *ICML*, 2017.

[16] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, volume 9, pages 249–256, 2010.

[17] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *IJCNN*, pages 729–734, 2005.

[18] Ankur Gupta, Julieta Martinez, James J Little, and Robert J Woodham. 3d pose from motion for cross-view action recognition via non-linear circulant temporal encoding. In *CVPR*, pages 2601–2608, 2014.

[19] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, pages 2980–2988, 2017.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[21] Mir Rayat Imtiaz Hossain and James J Little. Exploiting temporal information for 3D human pose estimation. In *ECCV*, 2018.

[22] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[23] Catalin Ionescu, Fuxin Li, and Cristian Sminchisescu. Latent structured models for human pose estimation. In *ICCV*, pages 2220–2227, 2011.

[24] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 36(7):1325–1339, 2014.

[25] Hao Jiang. 3D Human Pose Reconstruction Using Millions of Exemplars. In *ICPR*, pages 1674–1677, 2010.

[26] Qiuhong Ke, Mohammed Bennamoun, Senjian An, Ferdous Sohel, and Farid Boussaid. A New Representation of Skeleton Sequences for 3D Action Recognition. In *CVPR*, pages 4570–4579, 2017.

[27] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014.

[28] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017.

[29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012.

[30] Hsi-Jian Lee and Zen Chen. Determination of 3D Human-Body Postures From a Single View. *Computer Vision Graphics and Image Processing*, 30(2):148–168, 1985.

[31] Changsheng Li, Xiangfeng Wang, Weishan Dong, Junchi Yan, Qingshan Liu, and Hongyuan Zha. Joint active learning with feature selection via cur matrix decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018.

[32] Sijin Li, Weichen Zhang, and Antoni B Chan. Maximum-margin structured learning with deep networks for 3d human pose estimation. In *ICCV*, pages 2848–2856, 2015.

[33] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A Skinned Multi-Person Linear Model. *ACM Transactions on Graphics (TOG)*, 34(6):248:1–248:16, Oct. 2015.

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

[35] 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 *International Conference on 3D Vision (3DV)*, pages 506–516, 2017.

[36] Francesc Moreno-Noguer. 3D human pose estimation from a single image via distance matrix regression. In *CVPR*, pages 1561–1570, 2017.

[37] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814, 2010.

[38] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, pages 483–499, 2016.

[39] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *ICML*, pages 2014–2023, 2016.

[40] Sungheon Park and Nojun Kwak. 3D Human Pose Estimation with Relational Networks. In *BMVC*, 2018.

[41] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G Derpanis, and Kostas Daniilidis. Coarse-to-Fine Volumetric Prediction for Single-Image 3D Human Pose. In *CVPR*, pages 1263–1272, 2017.

[42] Xi Peng, Zhiqiang Tang, Fei Yang, Rogerio S Feris, and Dimitris Metaxas. Jointly optimize data augmentation and network training: Adversarial data augmentation in human pose estimation. In *CVPR*, pages 2226–2234, 2018.

[43] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.

[44] Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Reconstructing 3D human pose from 2D image landmarks. In *ECCV*, pages 573–586, 2012.

[45] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J. Black. Generating 3D faces using convolutional mesh autoencoders. In *ECCV*, 2018.

[46] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015.

[47] Grégory Rogez, Jonathan Rihan, Srikumar Ramalingam, Carlos Orrite, and Philip HS Torr. Randomized trees for human pose detection. In *CVPR*, 2008.

[48] Grégory Rogez and Cordelia Schmid. MoCap-guided Data Augmentation for 3D Pose Estimation in the Wild. In *NIPS*, pages 3108–3116, 2016.

[49] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[50] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–106, 2008.

[51] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis. In *CVPR*, 2016.

[52] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[53] Xiao Sun, Jiaxiang Shang, Shuang Liang, and Yichen Wei. Compositional human pose regression. In *ICCV*, 2017.

[54] Xiao Sun, Bin Xiao, Shuang Liang, and Yichen Wei. Integral human pose regression. In *ECCV*, 2018.

[55] Zhiqiang Tang, Xi Peng, Shijie Geng, Lingfei Wu, Shaoting Zhang, and Dimitris Metaxas. Quantized Densely Connected U-Nets for Efficient Landmark Localization. In *ECCV*, pages 339–354, 2018.

[56] Bugra Tekin, Pablo Marquez Neila, Mathieu Salzmann, and Pascal Fua. Learning to fuse 2D and 3D image cues for monocular body pose estimation. In *ICCV*, 2017.

[57] Bugra Tekin, Artem Rozantsev, Vincent Lepetit, and Pascal Fua. Direct Prediction of 3D Body Poses from Motion Compensated Sequences. In *CVPR*, pages 991–1000, 2016.

[58] Yu Tian, Xi Peng, Long Zhao, Shaoting Zhang, and Dimitris N Metaxas. CR-GAN: learning complete representations for multi-view generation. In *IJCAI*, pages 942–948, 2018.

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[60] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. In *ICLR*, 2018.

[61] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. Human Action Recognition by Representing 3D Skeletons as Points in a Lie Group. In *CVPR*, pages 588–595, 2014.

[62] Chunyu Wang, Yizhou Wang, Zhouchen Lin, Alan L Yuille, and Wen Gao. Robust estimation of 3D human poses from a single image. In *CVPR*, pages 2361–2368, 2014.

[63] Chaoyang Wang, Long Zhao, Shuang Liang, Liqing Zhang, Jinyuan Jia, and Yichen Wei. Object proposal by multi-branch hierarchical segmentation. In *CVPR*, pages 3873–3881, 2015.

[64] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *ECCV*, 2018.

[65] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.

[66] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *ECCV*, 2018.

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

[68] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph R-CNN for Scene Graph Generation. In *ECCV*, 2018.

[69] 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*, 2018.

[70] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Exploring visual relationship for image captioning. In *ECCV*, 2018.

[71] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks. In *ICCV*, 2017.

[72] Long Zhao, Fangda Han, Xi Peng, Xun Zhang, Mubbasir Kapadia, Vladimir Pavlovic, and Dimitris N. Metaxas. Cartoonish sketch-based face editing in videos using identity deformation transfer. *Computers & Graphics*, 79:58 – 68, 2019.

[73] Long Zhao, Xi Peng, Yu Tian, Mubbasir Kapadia, and Dimitris Metaxas. Learning to forecast and refine residual motion for image-to-video generation. In *ECCV*, pages 387–403, 2018.

[74] Xiangyun Zhao, Shuang Liang, and Yichen Wei. Pseudo mask augmented object detection. In *CVPR*, 2018.

[75] 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*, 2017.

[76] Xingyi Zhou, Xiao Sun, Wei Zhang, Shuang Liang, and Yichen Wei. Deep kinematic pose regression. *ECCV Workshop on Geometry Meets Deep Learning*, 2016.

[77] Xiaowei Zhou, Menglong Zhu, Spyridon Leonardos, Konstantinos G Derpanis, and Kostas Daniilidis. Sparseness meets deepness: 3D human pose estimation from monocular video. In *CVPR*, pages 4966–4975, 2016.

[78] Xiaowei Zhou, Menglong Zhu, Georgios Pavlakos, Spyridon Leonardos, Konstantinos G Derpanis, and Kostas Daniilidis. Monocap: Monocular human motion capture using a CNN coupled with a geometric prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018.

[79] Yizhe Zhu and Ahmed Elgammal. A multilayer-based framework for online background subtraction with freely moving cameras. In *ICCV*, 2017.

[80] Yizhe Zhu, Mohamed Elhoseiny, Bingchen Liu, Xi Peng, and Ahmed Elgammal. A generative adversarial approach for zero-shot learning from noisy texts. In *CVPR*, 2018.

Appendix A. Supplementary Material

This supplementary material provides additional results supporting the claims of the main paper. First, we provide more details about Semantic Graph Convolutional Networks (SemGCN), including the skeleton representation for building the graph (Sect. A.1) and the implementation of graph convolutions (Sect. A.2) and non-local layers (Sect. A.3). Additionally, to better understand the proposed Semantic Graph Convolutions, we provide the visualization results of the learned weights implied in the graph after training (Sect. A.4).

A.1. Skeleton Representation

Following the setting of previous works [13, 34, 53, 54, 75], we utilize a common human skeleton representation for Human3.6M [24] and MPII [3] to build the graph of SemGCN. This skeleton is visualized in Fig. 6(left). It consists of 16 joints and we define the *pelvis* joint as the root joint. Note that the skeleton is initialized as an *undirected* graph in SemGCN before training. After we finish training the network, it will transform to a *weighted directed* graph represented by $\rho_i\big(\mathbf{M} \odot \mathbf{A}\big)$ in Eq. 2 and 3.

In Fig. 6(left), we also show the bone vectors we employed in Eq. 7 to compute the bone loss. Let the bone $\mathbf{B}_k$ be directed from the joint $\mathbf{J}_{parent(k)}$ to the target joint $\mathbf{J}_k$, and we define the bone vector as:

$$\mathbf{B}_k = \mathbf{J}_{parent(k)} - \mathbf{J}_k. \tag{8}$$

This formulation is consistency with [53]. However, in order to be in line with the setting of previous works [13, 34] for fair comparison, the bone loss is not employed in *Configuration #1* of the experiments.

A.2. Implementation of Graph Convolutions

Some previous approaches [64, 67] proposed to leverage two different transformation matrixes other than one in the graph convolutions. To be specific, when the graph convolutional filter is applied to node i in the graph, one matrix $\mathbf{W}_0$ is employed to transform the representation of node i while the other matrix $\mathbf{W}_1$ is learned for all its neighbors. According to this formulation, we rewrite Eq. 1 to:

$$\mathbf{X}^{(l+1)} = \sigma\Big(\mathbf{I} \otimes \mathbf{W}_0 \mathbf{X}^{(l)} \tilde{\mathbf{A}} + \big(\mathbf{1} - \mathbf{I}\big) \otimes \mathbf{W}_1 \mathbf{X}^{(l)} \tilde{\mathbf{A}}\Big), \tag{9}$$

where $\otimes$ denotes element-wise multiplication and $\mathbf{I}$ is the identity matrix. We also implement the proposed *SemGConv* defined by Eq. 2 and 3 in the similar manner.

A.3. Non-local Layers

We follow the guidance of Wang *et al.* [65] to implement the non-local layers in SemGCN. For computational efficiency, we down-sample both the feature dimension and number of nodes in the graph when calculating the embedding of $f(\vec{\boldsymbol{x}}_i^{(l)}, \vec{\boldsymbol{x}}_j^{(l)})$ in Eq. 4.

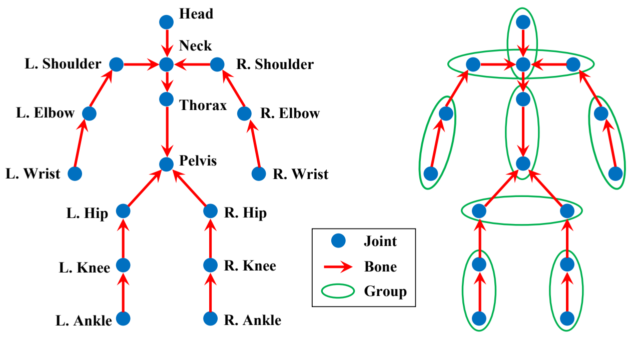

Figure 6. Left: the template human skeleton we utilized to build the graph. Each red arrow represents the bone vector for computing the loss function. Right: the grouping (down-sampling) strategy of the human skeleton we employed in the non-local layers.

Feature embedding. We use "concatenation" for the implementation of f. Two mapping functions $\theta(\cdot)$ and $\phi(\cdot)$ are employed to down-sample the feature of each node from 128 to 64 channels. They are implemented as convolutions with the kernel size of 1. Then we define f as:

$$f(\vec{\boldsymbol{x}}_i^{(l)}, \vec{\boldsymbol{x}}_j^{(l)}) = \mathrm{ReLU}(\mathbf{w}_f[\theta(\vec{\boldsymbol{x}}_i^{(l)}) \| \phi(\vec{\boldsymbol{x}}_j^{(l)})]), \tag{10}$$

where $[\cdot\|\cdot]$ denotes concatenation, and $\mathbf{w}_f$ is the parameters to be learned to project the concatenated vector to a scalar.

Node grouping. We also use max pooling to down-sample the number of nodes in the graph. Fig 6(right) illustrates the grouping strategy we employed. The number of nodes contained in the graph reduces from 16 to 8 after the max pooling operation. This strategy is used for all non-local layers in SemGCN. In the experiments, we find that this pooling operation can speed up the runtime, while it does not influence the final accuracy of the regression.

A.4. Visualization of Weights in SemGCN

To better understand the proposed SemGCN, we visualize the learned weighting matrix $\mathbf{M}$ of each *SemGConv* layer in the network. For simplicity, we utilize a simplified version of SemGCN, where Eq. 2 is employed so that all feature channels share the same $\mathbf{M}$. We use the network architecture as illustrated in Fig. 2 and train it according to *Configuration #1* of the experiments.

The trained network consists of 4 residual blocks where each block contains 2 *SemGConv* layers. Therefore, we visualize the weighting matrixes of these 8 *SemGConv* layers respectively. The matrixes are shown in Fig. 7. We have made two important observations. First, although all *SemGConv* layers share the same graph structure in the network, each of them has learned a different weighting matrix. Second, we can find that *SemGConv* layers have learned higher

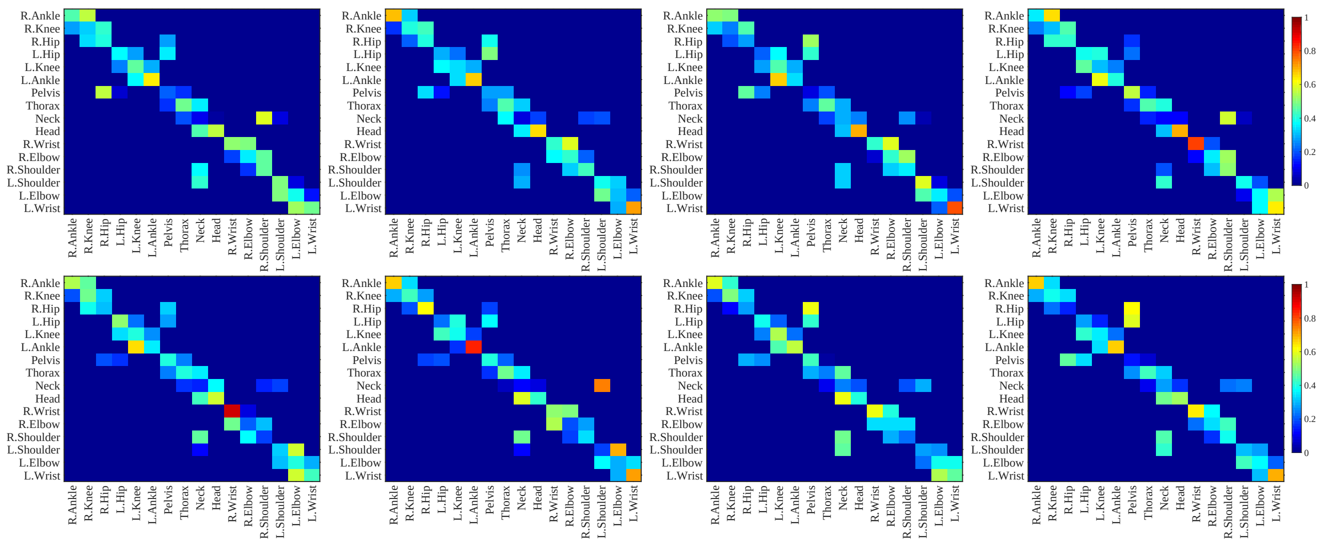

Figure 7. Visualization of learned weighting matrixes in SemGCN. Each *SemGConv* has learned a different weighting matrix for the graph in the network, while all of them have higher weights for nodes farer from the gravity center of the skeleton. Best viewed in color.

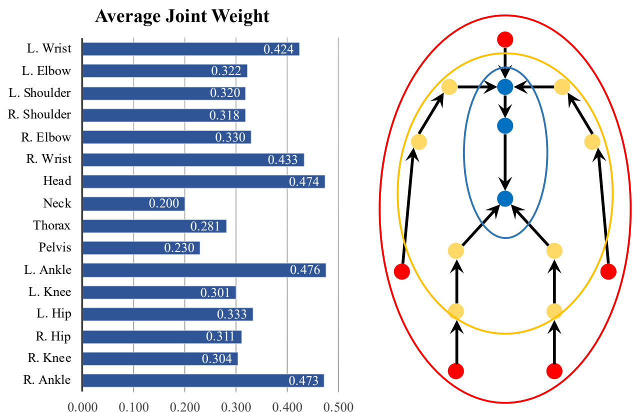

Figure 8. Left: the average learned weight of each joint among all *SemGConv* layers in the network. Right: a regional map of the skeleton where joints are grouped into three regions according to their weights.

weights for nodes which are farer from the gravity center of the human skeleton on average.

To further illustrate the second observation, we compute the average learned weight of each joint among all 8 *SemGConv* layers. The quantitative results are shown in Fig. 8(left). We can see that the left wrist, right wrist, left ankle, right ankle and head own the top highest weights which are greater than 0.4; while the neck, thorax and pelvis have the lowest weights less than 0.3. Other joints have quite similar weights around 0.3. This result can be better visualized by representing the human skeleton with a regional map where joints are grouped into three regions according to their weights. Fig. 8(right) shows the result.

This result is intuitive since joints farer from the center always encode more information of the pose while central joints determine the position of the skeleton. This observation is also consistency with [53, 67]. This demonstrates that the proposed SemGCN is able to effectively encode spatial relationships of nodes in the graph. However, we only rely on the ground truth for supervision, and no additional hand-crafted constraints or rules are employed.