Dynamic Multiscale Graph Neural Networks for 3D Skeleton-Based Human Motion Prediction

Maosen Li[1], Siheng Chen[2]⊠, Yangheng Zhao[1], Ya Zhang[1]⊠, Yanfeng Wang[1], and Qi Tian[1]

[1] Cooperative Medianet Innovation Center, Shanghai Jiao Tong University
[2] Mitsubishi Electric Research Laboratories

{maosen_li, zhaoyangheng-sjtu, ya_zhang, wangyanfeng} @sjtu.edu.cn, schen@merl.com, wywqtian@gmail.com

Abstract

We propose novel dynamic multiscale graph neural networks (DMGNN) to predict 3D skeleton-based human motions. The core idea of DMGNN is to use a multiscale graph to comprehensively model the internal relations of a human body for motion feature learning. This multiscale graph is adaptive during training and dynamic across network layers. Based on this graph, we propose a multiscale graph computational unit (MGCU) to extract features at individual scales and fuse features across scales. The entire model is action-category-agnostic and follows an encoder-decoder framework. The encoder consists of a sequence of MGCUs to learn motion features. The decoder uses a proposed graph-based gate recurrent unit to generate future poses. Extensive experiments show that the proposed DMGNN outperforms state-of-the-art methods in both short and long-term predictions on the datasets of Human 3.6M and CMU Mocap. We further investigate the learned multiscale graphs for the interpretability. The codes could be downloaded from https://github.com/limaosen0/DMGNN.

1. Introduction

3D skeleton-based human motion prediction forecasts future poses given the past motions based on the human-body-skeleton. The motion prediction helps machines understand human behaviors, attracting considerable attention [9, 20, 32, 5, 12, 2]. The related techniques can be widely applied to many computer vision and robotics scenarios, such as human-computer interaction [24, 23, 17, 13], autonomous driving [6], and pedestrian tracking [1, 15, 3].

Many methods, including the conventional state-based methods [25, 44, 38, 37, 36] and deep-network-based methods [9, 32, 10, 7, 12, 14, 11, 33, 43], have been proposed to

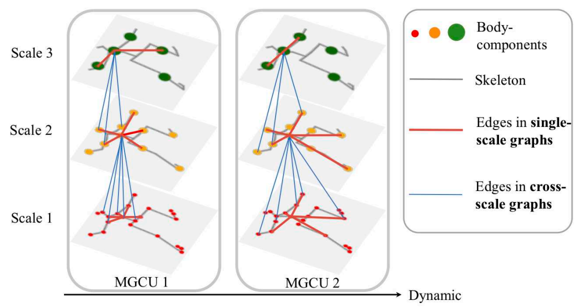

Figure 1. Two learned multiscale graphs on 'Posing'. We show strong relations associated with torsos in single scales and across scales. Two multiscale graphs are dynamic from one MGCUs to another, capturing local and distant relations, respectively.

achieve promising motion prediction. However, most methods did not explicitly exploit the relations or constraints between different body-components, which carry crucial information for motion prediction. A recent work [31] built graphs across body-joints for pairwise relation modeling; however, such a graph was still insufficient to reflect a functional group of body-joints. Another work [43] builds predefined sturctures to aggregate body-joint features to represent fixed body-parts, while the model only considers the body physical constraints without exploiting the movement coordination and relations. For example, the action of 'Walking' tends to be understood based on the collaborative movements of abstract arms and legs, rather than the detailed locations of fingers and toes.

To model more comprehensive relations, we propose a new representation for a human body: a *multiscale graph*, whose nodes are body-components at various scales and edges are pairwise relations between components. To model a body at multiple scales, a multiscale graph consists of two types of sub-graphs: *single-scale graphs*, connecting body-components at the same scales, and *cross-scale graphs*, con-

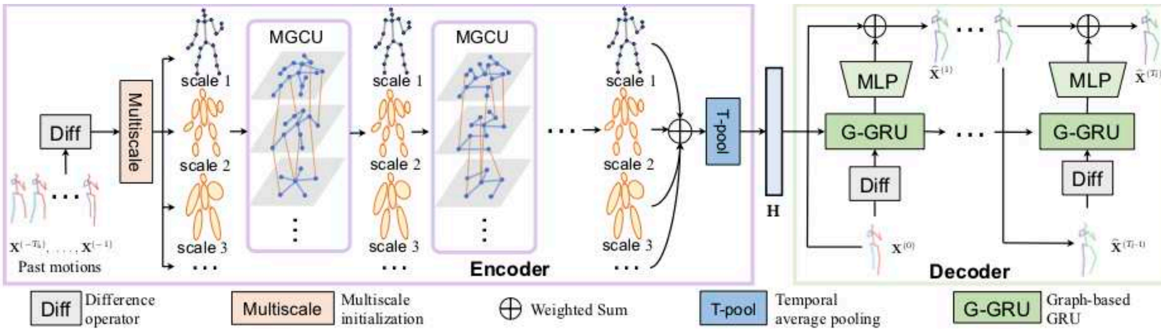

Figure 2. The architecture of DMGNN, which uses an encoder-decoder framework for motion prediction. In the encoder, cascaded multiscale graph computational blocks (MGCU) leverage dynamic muliscale graphs to extract spatio-temporal features. In the decoder, we propose a graph-based GRU (G-GRU) to predict poses.

necting body-components across two scales; see Figure 1. The single-scale graphs together provide a pyramid representation of a body skeleton. Each cross-scale graph is a bipartite graph, bridging one single-scale graph to another. For example, an "arm" node in a coarse-scale graph could connect to "hand" and "elbow" nodes in a fine-scale graph. This multiscale graph is initialized by predefined physical connections and adaptively adjusted in training to be motion-sensitive. Overall, this multiscale representation provides a new potentiality to model body relations.

Based on the multiscale graph, we propose a novel model, called *dynamic multiscale graph neural networks* (DMGNN), which is action-category-agnostic and follows from an encoder-decoder framework to learn motion representations for prediction. The encoder contains a cascade of *multiscale graph computational units* (MGCU), where each is associated with a multiscale graph. One MGCU includes two key components: *single-scale graph convolution block* (SS-GCB), leveraging single-scale graphs to exact features at individual scales, and *cross-scale fusion block* (CS-FB), inferring cross-scale graphs to convert features from one scale to another and enable fusion across scales. The multiscale graph has adaptive and trainable inbuilt topology; it is also dynamic because the topology is changing from one MGCU to another; see the learned dynamic multiscale graphs in Figure 1. Notably, cross-scale graphs in CS-FBs are constructed adaptively to input motions, and reflect discriminative motion patterns for category-agnostic prediction.

As for the decoder, we adopt a *graph-based gated recurrent unit* (G-GRU) to sequentially produce predictions given the last estimated poses. The G-GRU utilizes trainable graphs to further enhance state propagation. We also use residual connections to stabilize the prediction. To learn richer motion dynamics, we introduce difference operators to extract multiple orders of motion differences as the proxies of positions, velocities, and accelerations. The architecture of DMGNN is illustrated in Figure 2.

To verify the superiority of our DMGNN, extensive experiments are conducted on two large-scale datasets: Human 3.6M [19] and CMU Mocap[1]. The experimental results show that our model outperforms most state-of-the-art works for both short-term and long-term prediction in terms of both effectiveness and efficiency. The main contributions of this paper are as follow:

- We propose dynamic multiscale graph neural networks (DMGNN) to extract deep features at multiple scales and achieve effective motion prediction;
- We propose two key components: a multiscale graph computational unit, which leverages a multiscale graph to extract and fuse features across multiple scales, as well as a graph-based GRU to enhance state propagation for pose generation; and
- We conduct extensive experiments to show that the proposed DMGNN outperforms most state-of-the-art methods for short and long-term motion prediction on two large datasets. We further visualize the learned graphs for interpretability and reasoning.

2. Related Work

Human motion prediction: To forecast motions, some traditional methods, e.g., hidden Markov models [25], Gaussian-process [44] and random forests [25], were developed. Recently, deep networks are playing increasingly crucial roles: some recurrent-network-based models generated future poses step-by-step [9, 20, 32, 41, 45, 11, 30, 12, 28]; some feed-forward networks [26, 31] tried to reduce error accumulation for stable prediction; imitation-learning algorithm was also proposed [42]. However, these methods rarely considered enough relations from various scales, which carry comprehensive information for human behaviors understanding. In this work, we build dynamic multiscale graphs to capture rich multiscale relations and extract flexible semantics for motion prediction.

Graph deep learning: Graphs, expressing data associated with non-grid structures, preserve the dependencies among internal nodes [46, 40, 39]. Many studies focused

[1]http://mocap.cs.cmu.edu/

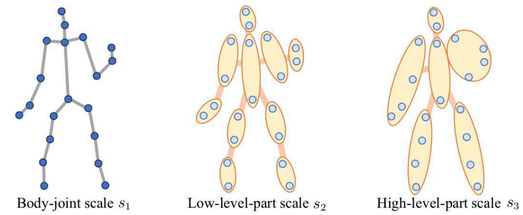

Figure 3. Three body scales on Human 3.6M. In s_1, we consider 20 joints with non-zero exponential maps [18]; In s_2 and s_3, we consider 10 and 5 parts, respectively.

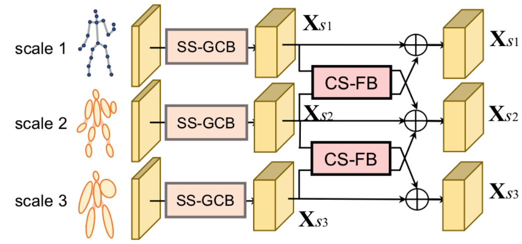

Figure 4. An MGCU uses single-scale graph convolution blocks (SS-CB) cross-scale fusion blocks (CS-FB).

on graph representation learning and the relative applications [29, 8, 22, 16, 46, 35]. Based on fixed graph structures, previous works explored propagating node features according to either the graph spectral domain [8, 22] or the graph vertex domain [16]. Several graph-based models have been employed for skeleton-based action recognition [46, 27, 34], motion prediction [31] and 3D pose estimation [47]; Different from any previous works, our model considers multiscale graphs and corresponding operations.

3. Problem Formulation

Suppose that the historical 3D skeleton-based poses are $\mathbb{X}_{-T_\mathrm{h}:0} = [\mathbf{X}^{(-T_\mathrm{h})}, \ldots, \mathbf{X}^{(0)}] \in \mathbb{R}^{M \times (T_\mathrm{h}+1) \times D_\mathbf{x}}$ and the future poses are $\mathbb{X}_{1:T_\mathrm{f}} = [\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(T_\mathrm{f})}] \in \mathbb{R}^{M \times T_\mathrm{f} \times D_\mathbf{x}}$, where $\mathbf{X}^{(t)} \in \mathbb{R}^{M \times D_\mathbf{x}}$ with M joints and $D_\mathbf{x} = 3$ feature-dimensions depicts the 3D pose at time t. The goal of motion prediction is to generate future poses given the past observed ones; mathematically, we need to propose a model $\mathcal{F}_{pred}(\cdot)$ to predict $\widehat{\mathbb{X}}_{1:T_\mathrm{f}} = \mathcal{F}_{pred}(\mathbb{X}_{-T_\mathrm{h}:0})$, where $\widehat{\mathbb{X}}_{1:T_\mathrm{f}}$ is the predicted motion close to the target $\mathbb{X}_{1:T_\mathrm{f}}$.

To exploit rich body relations, we represent a body as a multiscale graph across multiscale body-components. Theoretically, we could use arbitrary number of scales. Based on human nature, we specifically adopt 3 scales: the body-joint scale, the low-level-part scale, and the high-level-part scale. To initialize multiscale body graphs, we merge spatially nearby joints to coarser scales based on human prior; see Figure 3. With the multiscale graphs, we propose *dynamic multiscale graph neural networks* (DMGNN) to predict future poses in an end-to-end fashion.

4. Key Components

To construct our dynamic multiscale graph neural networks (DMGNN), we consider three basic components: a multiscale graph computational unit (MGCU), a graph-based GRU (G-GRU), and a difference operator.

4.1. Multiscale graph computational unit (MGCU)

The functionality of a MGCU is to extract and fuse features at multiple scales based on a multiscale graph, which is trained adaptively and individually. One MGCU includes two types of building blocks: single-scale graph convolution blocks, which leverage single-scale graphs to extract features at each scale, and cross-scale fusion blocks, which leverage cross-scale graphs to convert features from one scale to another and enable effective fusion across scales; see Figure 4. We now introduce each block in detail.

Single-scale graph convolution block (SS-GCB). To extract spatio-temporal features at each scale, we propose a *single-scale graph convolution block* (SS-GCB). Let the trainable adjacency matrix of the single-scale graph at scale s be $\mathbf{A}_s \in \mathbb{R}^{M_s \times M_s}$, where M_s is the number of body-components. $\mathbf{A}_s$ is first initialized by a skeleton graph whose nodes are body-components and edges are physical connections, modeling a prior of the physical constraints; see Figure 3. During training, each element in $\mathbf{A}_s$ is adaptively tuned to capture flexible body relations.

Based on the single-scale graph, SS-GCB effectively extracts deep features through two steps: 1) a graph convolution extracts spatial features of body-components; and 2) a temporal convolution extracts temporal features from motion sequences. Let the input feature at scale s be $\mathbf{X}_s \in \mathbb{R}^{M_s \times D_\mathbf{x}}$, the spatial graph convolution is formulated as

$$\mathbf{X}_{s,\mathrm{sp}} = \mathrm{ReLU}(\mathbf{A}_s \mathbf{X}_s \mathbf{W}_s + \mathbf{X}_s \mathbf{U}_s) \in \mathbb{R}^{M_s \times D'_\mathbf{x}}, \quad (1)$$

where $\mathbf{W}_s, \mathbf{U}_s \in \mathbb{R}^{D_\mathbf{x} \times D'_\mathbf{x}}$ are trainable parameters. Through (1), we extract the spatial features from correlated body-components. $\mathbf{A}_s$ in each SS-GCB is trained individually and stays fixed during test. To capture motions along time, we then develop a temporal convolution on the feature sequences. The single-scale graphs in different SS-GCBs are dynamic, showing flexible relations. Note that features extracted at various scales have different dimensionalities and reflect information with different receptive fields.

Cross-scale fusion block (CS-FB). To enable information diffusion across scales, we propose a *cross-scale fusion block* (CS-FB) which uses a cross-scale graph to convert features from one scale to another. A cross-scale graph is a bipartite graph that corresponds the nodes in one single-scale graph to the nodes in another single-scale graph. For example, the features of an "arm" node in the low-level-part scale s_2 can potentially guide the feature learning of a "hand" node in the body-joint scale s_1. We aim to infer this cross-scale graph adaptively from data. Here we present CS-FB from s_1 to s_2 as an example.

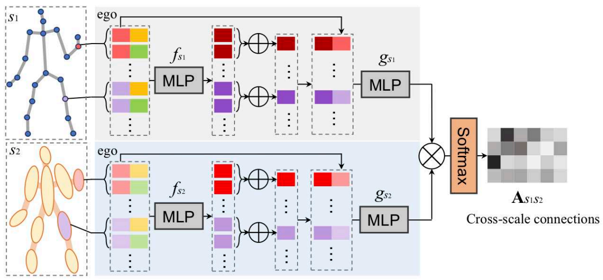

Figure 5. The inference of a cross-scale graph.

We first infer the cross-scale graph with adjacent matrix $\mathbf{A}_{s_1s_2} \in [0,1]^{M_{s_2} \times M_{s_1}}$ to model the cross-scale relations. Let the feature of the ith joint and the kth part along time be $(\mathbb{X}_{s_1})_{:,i,:} \in \mathbb{R}^{T_{s_1} \times D'_{\mathbf{x}}}$ and $(\mathbb{X}_{s_2})_{:,k,:} \in \mathbb{R}^{T_{s_2} \times D'_{\mathbf{x}}}$, we vectorize them as $\mathbf{p}_{s_1,i} = \mathrm{vec}(\mathrm{conv}_{s_1,\tau}((\mathbb{X}_{s_1})_{:,i,:};\mu))$ and $\mathbf{p}_{s_2,k} = \mathrm{vec}(\mathrm{conv}_{s_2,\tau}((\mathbb{X}_{s_2})_{:,k,:};\mu))$ to leverage temporal information, where τ and μ denote the temporal convolution kernel size and stride. We infer the edge weight between the ith joint and kth part $(\mathbf{A}_{s_1s_2})_{k,i}$ through

$$
\begin{aligned}
\mathbf{r}_{s_1,i} &= \sum_{j=1}^{M_{s_1}} f_{s_1}\left([\mathbf{p}_{s_1,i}, \mathbf{p}_{s_1,j} - \mathbf{p}_{s_1,i}]\right) && \text{(2a)}\\
\mathbf{h}_{s_1,i} &= g_{s_1}\left([\mathbf{p}_{s_1,i}, \mathbf{r}_{s_1,i}]\right) && \text{(2b)}\\
\mathbf{r}_{s_2,k} &= \sum_{j=1}^{M_{s_2}} f_{s_2}\left([\mathbf{p}_{s_2,k}, \mathbf{p}_{s_2,j} - \mathbf{p}_{s_2,k}]\right) && \text{(2c)}\\
\mathbf{h}_{s_2,k} &= g_{s_2}\left([\mathbf{p}_{s_2,k}, \mathbf{r}_{s_2,k}]\right) && \text{(2d)}\\
(\mathbf{A}_{s_1s_2})_{k,i} &= \mathrm{softmax}(\mathbf{h}_{s_2,k}^{\top}\mathbf{h}_{s_1,i}) \in [0,1], && \text{(2e)}
\end{aligned}
$$

where $f_{s_1}(\cdot)$, $g_{s_1}(\cdot)$, $f_{s_2}(\cdot)$ and $g_{s_2}(\cdot)$ denotes MLPs; $\mathrm{softmax}(\cdot)$ is a softmax operator along the raw of inner product matrix and $[\cdot,\cdot]$ is concatenation. (2a) and (2c) aggregate the relative features of all the components to the ith and the kth components in two scales, which are then updated by (2b) and (2d); and (2e) obtains adjacent matrix through inner product and softmax, thus we model the normalized effects from a body in s_1 to each component in s_2. The intuition behind this design is to leverage the global relative information to augment body-component features, and we use the inner product of two augmented features to obtain the edge weight. Figure 5 illustrates the inference of $\mathbf{A}_{s_1s_2}$. Notably, different from the fixed single-scale graphs during inference, the cross-scale graphs are efficiently inferred online and adaptive to motion features, which are flexible to capture distinct patterns for individual inputs.

We next fuse the joint features to the part-scale with $\mathbf{A}_{s_1s_2}$. Given the joint features at a certain time stamp $\mathbf{X}_{s_1} \in \mathbb{R}^{M_{s_1} \times D'_{\mathbf{x}}}$, the part-scale feature is updated as

$$\mathbf{X}_{s_2} \leftarrow \mathbf{A}_{s_1s_2}\mathbf{X}_{s_1}\mathbf{W}_{\mathrm{F},s_1} + \mathbf{X}_{s_2} \in \mathbb{R}^{M_{s_2} \times D'_{\mathbf{x}}},$$

where $\mathbf{W}_{\mathrm{F},s_1} \in \mathbb{R}^{D'_{\mathbf{x}} \times D'_{\mathbf{x}}}$ is trainable. Thus, each body-part in s_2 adaptively absorbs detailed information from the corresponding joints in s_1. The fused $\mathbf{X}_{s_2}$ is fed into the SS-CB of the next MGCU in s_2. In the other way around, we can define the fusion from s_2 to s_1 with similar operations.

4.2. Graph-based GRU

The functionality of a graph-based GRU (G-GRU) is to learn and update hidden states with the guide of a graph. The key is to use a trainable graph to regularize the states, which are used to generate future poses. Let $\mathbf{A}_{\mathrm{H}} \in \mathbb{R}^{M \times M}$ be the adjacent matrix of the inbuilt graph, which is initialized with the skeleton-graph and trained to build adaptive edges, and $\mathbf{H}^{(0)} \in \mathbb{R}^{M \times D_{\mathbf{h}}}$ be the initial state of G-GRU. At time $t > 0$, G-GRU takes two inputs: the initial state, $\mathbf{H}^{(t)}$, and the online 3D skeleton-based information, $\mathbf{I}^{(t)} \in \mathbb{R}^{M \times d}$. Then, G-GRU$(\mathbf{I}^{(t)}, \mathbf{H}^{(t)})$ works as

$$
\begin{aligned}
\mathbf{r}^{(t)} &= \sigma(r_{\mathrm{in}}(\mathbf{I}^{(t)}) + r_{\mathrm{hid}}(\mathbf{A}_{\mathrm{H}}\mathbf{H}^{(t)}\mathbf{W}_{\mathrm{H}})),\\
\mathbf{u}^{(t)} &= \sigma(u_{\mathrm{in}}(\mathbf{I}^{(t)}) + u_{\mathrm{hid}}(\mathbf{A}_{\mathrm{H}}\mathbf{H}^{(t)}\mathbf{W}_{\mathrm{H}})),\\
\mathbf{c}^{(t)} &= \tanh(c_{\mathrm{in}}(\mathbf{I}^{(t)}) + \mathbf{r}^{(t)} \odot c_{\mathrm{hid}}(\mathbf{A}_{\mathrm{H}}\mathbf{H}^{(t)}\mathbf{W}_{\mathrm{H}})),\\
\mathbf{H}^{(t+1)} &= \mathbf{u}^{(t)} \odot \mathbf{H}^{(t)} + (1 - \mathbf{u}^{(t)}) \odot \mathbf{c}^{(t)},
\end{aligned}
$$

where $r_{\mathrm{in}}(\cdot)$, $r_{\mathrm{hid}}(\cdot)$, $u_{\mathrm{in}}(\cdot)$, $u_{\mathrm{hid}}(\cdot)$, $c_{\mathrm{in}}(\cdot)$ and $c_{\mathrm{hid}}(\cdot)$ are trainable linear mappings; $\mathbf{W}_{\mathrm{H}}$ denotes the trainable weights. For each G-GRU cell, it applies a graph convolution on the hidden states for information propagation and produces the state for next frame.

4.3. Difference operator

The motion states like velocity and acceleration carry important dynamics. To use them, we propose a difference operator to compute high-order differences of input sequences, guiding the model to learn richer dynamics. At time t, the 0-order difference is $\Delta^0\mathbf{X}^{(t)} = \mathbf{X}^{(t)} \in \mathbb{R}^{M \times D_{\mathbf{x}}}$, and the β-order difference ($\beta > 0$) of the pose, $\Delta^{\beta}\mathbf{X}^{(t)}$, is $\Delta^{\beta}\mathbf{X}^{(t)} = \Delta^{\beta-1}\mathbf{X}^{(t)} - \Delta^{\beta-1}\mathbf{X}^{(t-1)}$. We use zero paddings after computing the differences to handle boundary conditions. Overall, the difference operator works as

$$\mathrm{diff}_{\beta}(\mathbf{X}^{(t)}) = \left[\Delta^0\mathbf{X}^{(t)} \quad \cdots \quad \Delta^{\beta}\mathbf{X}^{(t)}\right].$$

Here we consider $\beta = 2$. The three elements reflects positions, velocities, and accelerations.

5. DMGNN Framework

Here we present the architecture of our DMGNN, which contains a multiscale graph-based encoder and a recurrent graph-based decoder for motion prediction.

5.1. Encoder

Capturing semantics from observed motions, the encoder aims to provide the decoder with motion states for prediction. In the encoder, for each motion sample, we first concatenate its $0, 1, 2$-order of differences as input. And we initialize 3 body scales by averaging joint clusters in s_1 to spatially corresponding components in coarser scales. For example, we average two "right hand" joints in s_1 to the "right arm" part in s_2. We then use a cascade of MGCUs to extract spatio-temporal features. Note that the multi-scale graph associated with each MGCU is trained individually, thus the graph topology can be dynamically changing from one MGCU to another. To finally combine the three scales for comprehensive semantics, the output features are weighted summed. Since the numbers of body-components are different across scales, we broadcast the coarser components to match their spatially corresponding joints. Let the broadcast output features of the three scale be $\mathbb{H}_{s_1}, \mathbb{H}_{s_2}, \mathbb{H}_{s_3} \in \mathbb{R}^{T' \times M \times D_{\mathbf{h}}}$, the summed feature is

$$\mathbb{H} = \mathbb{H}_{s_1} + \lambda(\mathbb{H}_{s_2} + \mathbb{H}_{s_3}), \tag{3}$$

where λ is a hyper-parameter to balance different scales. We next use a temporal average pooling to remove the time dimension of $\mathbb{H}$ and obtain $\mathbf{H} \in \mathbb{R}^{M \times D_{\mathbf{h}}}$, which aggregates historical information as the initial state of the decoder.

5.2. Decoder

The decoder aims to predict future poses sequentially. The core of the decoder is the proposed graph-based GRU (G-GRU), which further propagates motion states for sequence regression. We first use the difference operator to extract three orders of differences as motion priors, and then feed them into G-GRU to update the hidden state. We next generate future pose displacement with an output function. Finally, we add the displacements to the input pose to predict the next frame. At frame t, the decoder works as

$$\widehat{\mathbf{X}}^{(t+1)} = \widehat{\mathbf{X}}^{(t)} + f_{\mathrm{pred}}\left(\text{G-GRU}\left(\mathrm{diff}_2(\widehat{\mathbf{X}}^{(t)}), \mathbf{H}^{(t)}\right)\right),$$

where $f_{\mathrm{pred}}(\cdot)$ represents an output function, implemented by MLPs. The initial state $\mathbf{H}^{(0)} = \mathbf{H}$, which is the final output of encoder.

5.3. Loss function

To train our DMGNN, we consider the ℓ_1 loss. Let the nth sample of predictions be $(\widehat{\mathbb{X}}_{1:T_{\mathrm{f}}})_n \in \mathbb{R}^{T_{\mathrm{f}} \times M \times D_{\mathbf{x}}}$ and the corresponding ground truth be $(\mathbb{X}_{1:T_{\mathrm{f}}})_n$. For N training samples, the loss function is

$$\mathcal{L}_{\mathrm{pred}} = \frac{1}{N} \sum_{n=1}^{N} \left\| (\mathbb{X}_{1:T_{\mathrm{f}}})_n - (\widehat{\mathbb{X}}_{1:T_{\mathrm{f}}})_n \right\|_1,$$

where $||\cdot||_1$ denotes the ℓ_1 norm. ℓ_1 loss gives sufficient gradients to joints with small losses to promote even more precise prediction; ℓ_1 loss also gives stable gradients to joints with large losses, alleviating gradient explosion. In our experiments, ℓ_1 loss leads to more precise predictions than ℓ_2 loss. All the weights in the proposed DMGNN are trained end-to-end with the stochastic gradient descent [4].

6. Experiments

6.1. Datasets and experimental setup

Human 3.6m (H3.6M). H3.6M dataset [19] has 7 subjects performing 15 different classes of actions. There are 32 joints in each subject, and we transform the joint positions into the exponential maps and only use the joints with non-zero values (20 joints remain). Along the time axis, we downsample all sequences by two. Following previous paradigms [32], the models are trained on 6 subjects and tested on the specific clips of the 5th subject.

CMU motion capture (CMU Mocap). CMU Mocap consists of 5 general classes of actions: 'human interaction', 'interaction with environment', 'locomotion', 'physical activities & sports', and 'situations & scenarios', where each subject has 38 joints and we preserve 26 joints with non-zero exponential maps. Be consistent with [26], we select 8 detailed actions: 'basketball', 'basketball signal', 'directing traffic', 'jumping', 'running', 'soccer', 'walking' and 'washing window'. We evaluate our model with the same approach as we do for H3.6M.

Model configuration. We implement DMGNN with PyTorch 1.0 on one RTX-2080Ti GPU. We set 3 scales, which contains body-joints, 10 and 5 body-components for both datasets. We use 4 cascaded MGCUs, whose feature dimensions are 32, 64, 128 and 256, respectively. In the first two MGCUs, we use both SS-GCBs and CS-FBs to extract spatio-temporal features and fuse cross-scale features; In the last two MGCUs, we only use SS-GCBs. In the decoder, the dimension of the G-GRU is 256, and we use a two-layer MLP for pose output. In training, we set the batch size 32 and clip the gradients to a maximum ℓ_2-norm of 0.5; we use Adam optimizer [21] with learning rate 0.0001. All the hyper-parameters are selected with validation sets.

Baseline methods. We compare the proposed DMGNN with many recent works, which learned motion patterns from pose vectors, e.g. Res-sup. [32], CSM [26], TP-RNN [7], AGED [12], and Imit-L [42], or separated bodies e.g. Skel-TNet [14], and Traj-GCN [31]. We reproduce, Res-sup., CSM and Traj-GCN based on their released codes. We also employ a naive baseline, ZeroV [32], which sets all predictions to be the last observed pose at $t = 0$.

6.2. Comparison to state-of-the-art methods

To validate the proposed DMGNN, we show the prediction performance for both short-term and long-term motion prediction on Human 3.6M (H3.6M) and CMU Mocap. We quantitatively evaluate various methods by the mean angle error (MAE) between the generated motions and ground-truths in angle space. We also illustrate the predicted samples for qualitative evaluation.

Short-term motion prediction. Short-term motion prediction aims to predict the future poses within 500 millisec-

Table 1. Mean angle errors (MAE) of different methods for short-term prediction on 4 representative actions of H3.6M. We also present different DMGNN variants, including using fixed graphs in SS-GCB (fixed $\mathbf{A}_s$), no graph in GRU (no G-GRU), and only one scale (single). The complete DMGNN outperform others methods at most time stamp.

Motion	Walking				Eating				Smoking				Discussion			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
ZeroV [32]	0.39	0.68	0.99	1.15	0.27	0.48	0.73	0.86	0.26	0.48	0.97	0.95	0.31	0.67	0.94	1.04
Res-sup. [32]	0.27	0.46	0.67	0.75	0.23	0.37	0.59	0.73	0.32	0.59	1.01	1.10	0.30	0.67	0.98	1.06
CSM [26]	0.33	0.54	0.68	0.73	0.22	0.36	0.58	0.71	0.26	0.49	0.96	0.92	0.32	0.67	0.94	1.01
TP-RNN [7]	0.25	0.41	0.58	0.65	0.20	0.33	0.53	0.67	0.26	0.47	0.88	0.90	0.30	0.66	0.96	1.04
AGED [12]	0.21	0.35	0.55	0.64	0.18	0.28	0.50	0.63	0.27	0.43	0.81	0.83	0.26	0.56	0.77	0.84
Skel-TNet [14]	0.31	0.50	0.69	0.76	0.20	0.31	0.53	0.69	0.25	0.50	0.93	0.89	0.30	0.64	0.89	0.98
Imit-L [42]	0.21	0.34	0.53	0.59	0.17	0.30	0.52	0.65	0.23	0.44	0.87	0.85	0.23	0.56	0.82	0.91
Traj-GCN [31]	**0.18**	0.32	**0.49**	**0.56**	**0.17**	0.31	0.52	0.62	0.22	0.41	0.84	0.79	**0.20**	**0.51**	**0.79**	**0.86**
DMGNN (fixed $\mathbf{A}_s$)	0.20	0.35	0.54	0.63	0.20	0.34	0.53	0.66	0.23	0.41	0.86	0.83	0.26	0.65	0.92	1.02
DMGNN (no G-GRU)	0.22	0.33	0.53	0.61	0.19	0.32	0.53	0.66	0.23	0.42	0.87	0.82	0.27	0.65	0.90	0.98
DMGNN ($S = 1$)	0.20	0.33	0.54	0.60	0.18	0.31	0.52	0.62	0.22	0.41	0.83	0.80	0.25	0.64	0.95	1.00
DMGNN	**0.18**	**0.31**	**0.49**	0.58	**0.17**	**0.30**	**0.49**	**0.59**	**0.21**	**0.39**	**0.81**	**0.77**	0.26	0.65	0.92	0.99

Table 2. MAEs of different methods for short-term motion prediction on other 11 actions of H3.6M.

Motion	Directions				Greeting				Phoning				Posing				Purchases				Sitting			
millisecond	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Res-sup [32]	0.41	0.64	0.80	0.92	0.57	0.83	1.45	1.60	0.59	1.06	1.45	1.60	0.45	0.85	1.34	1.56	0.58	0.79	1.08	1.15	0.41	0.68	1.12	1.33
CSM [26]	0.39	0.60	0.80	0.91	0.51	0.82	1.21	1.38	0.59	1.13	1.51	1.65	0.29	0.60	1.12	1.37	0.63	0.91	1.19	1.29	0.39	0.61	1.02	1.18
Traj-GCN [31]	0.26	0.45	0.70	0.79	**0.35**	**0.61**	0.96	1.13	**0.53**	1.02	1.32	1.45	0.23	0.54	1.26	1.38	0.42	0.66	1.04	1.12	0.29	0.45	0.82	**0.97**
DMGNN	**0.25**	**0.44**	**0.65**	**0.71**	0.36	**0.61**	**0.94**	**1.12**	**0.52**	**0.97**	**1.29**	**1.43**	**0.20**	**0.46**	**1.06**	**1.34**	**0.41**	**0.61**	**1.05**	1.14	**0.26**	**0.42**	**0.76**	**0.97**

Motion	Sitting Down				Taking Photo				Waiting				Walking Dog				Walking Together				Average			
millisecond	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Res-sup. [32]	0.47	0.88	1.37	1.54	0.28	0.57	0.90	1.02	0.32	0.63	1.07	1.26	0.52	0.89	1.25	1.40	0.27	0.53	0.74	0.79	0.40	0.69	1.04	1.18
CSM [26]	0.41	0.78	1.16	1.31	0.23	0.49	0.88	1.06	0.30	0.62	1.09	1.30	0.59	1.00	1.32	1.44	0.27	0.52	0.71	0.74	0.38	0.68	1.01	1.13
Traj-GCN [31]	**0.30**	**0.63**	**0.89**	**1.01**	**0.15**	0.36	0.59	0.72	0.23	0.50	0.92	1.15	0.46	0.80	**1.12**	**1.30**	**0.15**	0.35	0.52	**0.57**	**0.27**	0.53	0.85	0.96
DMGNN	0.32	0.65	0.93	1.05	**0.15**	**0.34**	**0.58**	**0.71**	**0.22**	**0.49**	**0.88**	**1.10**	**0.42**	**0.72**	1.16	1.34	**0.15**	**0.33**	**0.50**	**0.57**	**0.27**	**0.52**	**0.83**	**0.95**

Table 3. MAEs of different methods for long-term prediction on the 4 representative actions of H3.6M dataset.

Motion	Walking		Eating		Smoking		Discussion		Average	
milliseconds	560	1k	560	1k	560	1k	560	1k	560	1k
ZeroV [32]	1.35	1.32	1.04	1.38	1.02	1.69	1.41	1.96	1.21	1.59
Res-sup. [32]	0.93	1.03	0.95	1.08	1.25	1.50	1.43	1.69	1.14	1.33
CSM [26]	0.98	0.92	1.01	1.24	0.97	1.62	1.56	1.86	1.13	1.41
AGED [12]	0.78	0.91	0.86	**0.93**	1.06	**1.21**	**1.25**	**1.30**	0.99	**1.09**
Skel-TNet [14]	0.94	0.92	0.97	1.23	0.99	1.59	1.51	1.82	1.10	1.39
Imit-L [42]	0.67	0.69	0.79	1.13	0.95	1.63	1.34	1.81	0.94	1.32
Traj-GCN [31]	**0.65**	**0.67**	0.76	1.12	0.87	1.57	1.33	1.70	0.90	1.27
DMGNN	0.66	0.75	**0.74**	1.14	**0.83**	1.52	1.33	1.45	**0.89**	1.21

onds. We compare DMGNN to state-of-the-art methods for predicting poses in 400 milliseconds on H3.6M dataset. We first test 4 representative actions: 'Walking', 'Eating', 'Smoking' and 'Discussion'. Table 1 shows MAEs of DMGNN and some baselines. We also present the performance of several variants of DMGNN: we use fixed body-graphs in SS-GCBs (fixed $\mathbf{A}_s$); the common GRU without a graph (no G-GRU); or only the joint-scale ($S = 1$) bodies. We see that, i) the complete DMGNN obtain the most precise prediction among all the variants; ii) compared to baselines, DMGNN has the lowest prediction MAEs on 'Eating' and 'Smoking', and obtains competitive results on 'Walking' and 'Discussion'. Table 2 compares the proposed DMGNN with some recent baselines on the remaining 11 actions in H3.6M. We see that DMGNN achieves the best performance in most actions (also for average MAEs).

Long-term motion prediction. Long-term motion prediction aims to predict the poses over 500 milliseconds, which is challenging due to the action variation and non-linearity movements. Table 3 presents the MAEs of various models for predicting 4 actions and average MAEs across the 4 actions in the future 560 ms and 1000 ms on H3.6M dataset. We see that DMGNN outperforms the competitors on actions 'Eating', and 'Discussion' at 560 ms, and obtains competitive performances on other cases.

We also train our DMGNN for short-term and long-term prediction on 8 classes of actions in CMU Mocap dataset. Table 4 shows the MAEs across the future 1000 ms. We see that DMGNN significantly outperforms the state-of-the-art methods on actions 'Basketball', 'Basketball Signal', 'Running' and 'Walking' and obtains competitive performance on the other actions.

Predicted sample visualization. We compare the synthesized samples of DMGNN to those of Res-sup., CSM and Traj-GCN on H3.6M. Figure 6 illustrates the future poses of 'Taking Photo' in 1000 ms with the frame interval of 80 ms. Comparing to baselines, we see that DMGNN completes the action accurately and reasonably, providing significantly better predictions. Res-sup. has large discontinuity between the last observed pose the first predicted one (red box); CSM and Traj-GCN have large errors after the 280th ms (blue box); three baselines give large posture errors in long-term (yellow box). We show more prediction

Table 4. Comparisons of MAEs between our model and the state-of-the-art methods on the 8 actions of CMU Mocap dataset. We evaluate the model and present the MAEs at both short and long-term prediction time stamps.

Motion	Basketball					Basketball Signal					Directing Traffic					Jumping				
milliseconds	80	160	320	400	1000	80	160	320	400	1000	80	160	320	400	1000	80	160	320	400	1000
Res-sup. [32]	0.49	0.77	1.26	1.45	1.77	0.42	0.76	1.33	1.54	2.17	0.31	0.58	0.94	1.10	2.06	0.57	0.86	1.76	2.03	2.42
CSM [26]	0.36	0.62	1.07	1.17	1.95	0.33	0.62	1.05	1.23	1.98	0.26	0.58	0.91	1.04	2.08	0.38	0.60	1.36	1.58	2.05
Traj-GCN [31]	0.33	0.52	**0.89**	**1.06**	1.71	0.11	0.20	0.41	0.53	**1.00**	**0.15**	0.32	**0.52**	**0.60**	2.00	**0.31**	**0.49**	**1.23**	**1.39**	1.80
DMGNN	**0.30**	**0.46**	**0.89**	1.11	**1.66**	**0.10**	**0.17**	**0.31**	**0.41**	1.26	**0.15**	**0.30**	0.57	0.72	**1.98**	0.37	0.65	1.49	1.71	**1.79**

Motion	Running					Soccer					Walking					Washing Window				
milliseconds	80	160	320	400	1000	80	160	320	400	1000	80	160	320	400	1000	80	160	320	400	1000
Res-sup. [32]	0.32	0.48	0.65	0.74	1.00	0.29	0.50	0.87	0.98	1.73	0.35	0.45	0.59	0.64	0.88	0.31	0.47	0.74	0.93	1.37
CSM [26]	0.28	0.43	0.54	0.57	0.69	0.28	0.48	0.79	0.90	1.58	0.35	0.44	0.46	0.51	0.77	0.30	0.47	0.79	1.00	1.39
Traj-GCN [31]	0.33	0.55	0.73	0.74	0.95	**0.18**	**0.29**	**0.61**	**0.71**	**1.40**	0.33	0.45	0.49	0.53	0.61	0.22	0.33	**0.57**	**0.75**	1.20
DMGNN	**0.19**	**0.31**	**0.47**	**0.49**	**0.64**	0.22	0.32	0.79	0.91	1.54	**0.30**	**0.34**	**0.38**	**0.43**	**0.60**	**0.20**	**0.27**	0.62	0.81	**1.09**

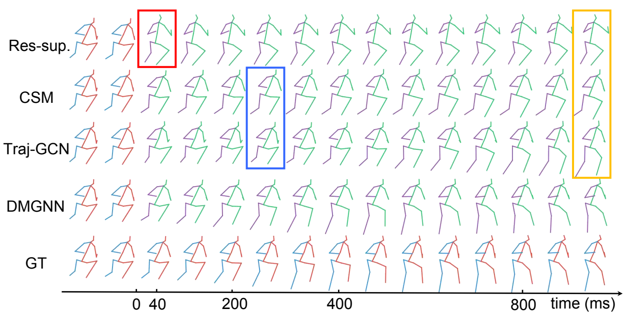

Figure 6. Qualitative comparison on the action 'Taking Photo' of H3.6M for both short and long-term prediction.

Table 5. Average time cost comparison between DMGCNN with the latest models on H3.6M dataset.

Model	Time cost (ms)	
milisecond	400	1000
TP-RNN [7]	48.96	127.41
Skel-TNet [14]	33.29	98.17
Traj-GCN [31]	71.43	144.93
DMGNN	**29.18**	**86.04**

images and videos in Appendix.

Effectiveness and efficiency test. We compare the running time costs of DMGNN to several latest models. Table 5 presents the running time of different methods for short and long-term motion prediction on H3.6M dataset. We see that DMGNN achieves the shortest running time while generating future poses over both 400 or 1000 ms, compared with the other competitors [32, 26, 31]. DMGNN takes only 29.18 ms to generate motions in 400 ms, indicating that DMGNN with multiscale graphs has efficient operations.

6.3. Ablation study

We now investigate some crucial elements of DMGNN.

Effects of multiple scales. To verify the proposed multiscale representation, we employ various scales in DMGNN for 3D skeleton-based motion prediction. Besides the three scales in our model, we introduce additional two scales: s_4, which represents a body as $M_{s_4} = 3$ parts: left limbs, right limbs and torso, and s_5, which contains $M_{s_5} = 2$ parts: upper body and lower body; see illustrations of s_4 and s_5 in Appendix. Table 6 presents the MAEs with various scales. We see that, when we combine s_1, s_2 and s_3, lowest prediction error is achieved. Notably, using two scales (s_1, s_2 or s_1, s_3) is significant better than using only s_1; but involving too abstract scales (s_4 or s_5) tends to hurt prediction.

Table 6. Average MAEs of DMGNN with different scales for short-term prediction at different time stamps.

	Node numbers M_s					MAEs			
Scales	20	10	5	3	2	80	160	320	400
1	✓					0.29	0.55	0.87	1.00
1, 2	✓	✓				**0.27**	0.53	0.85	0.97
1, 2, 3	✓	✓	✓			**0.27**	**0.52**	**0.83**	**0.95**
1, 3	✓		✓			0.28	0.53	0.84	**0.92**
1, 2, 3, 4	✓	✓	✓	✓		0.28	0.54	0.87	0.98
1, 4	✓			✓		0.28	0.54	0.86	0.97
1, 2, 3, 5	✓	✓	✓		✓	0.28	0.55	0.86	0.99
1, 5	✓				✓	0.29	0.55	0.87	1.00

Table 7. MAEs and running times of DMGNN with different numbers of MGCUs for short and long-term prediction on H3.6M.

	MAE at different time stamps (ms)						running time (ms)	
MGCUs	80	160	320	400	560	1000	400	1000
1	0.30	0.56	0.87	1.02	1.25	1.52	27.42	83.01
2	0.29	0.53	0.85	0.99	1.20	1.52	27.89	83.95
3	**0.27**	0.54	**0.83**	**0.95**	1.18	1.49	28.34	84.89
4	**0.27**	**0.52**	**0.83**	**0.95**	**1.16**	**1.48**	29.18	86.04
5	0.28	0.55	**0.83**	0.96	1.17	1.51	30.37	88.39
6	0.29	0.54	0.84	0.98	1.19	1.54	31.55	91.15

Table 8. Average MAEs of DMGNN with different numbers of CS-FBs and feature aggregators over 400 ms on H3.6M.

	Average MAE across 400 ms			
CS-FB numbers	1	2	3	0
without relative	0.623	0.622	0.618	0.630
with relative	0.618	**0.613**	0.616	

Effects of the number of MGCUs. To validate the effects of multiple MGCUs in the encoder, we tune the numbers of MGCUs from 1 to 6 and show the prediction errors and running time costs for short and long-term prediction on H3.6M, which are presented in Table 7. We see that, when we adopt 1 to 4 MGCUs, the prediction MAEs fall and time costs rise continuously; when we use 5 or 6 MGCUs, the prediction errors are stably low, but the time costs rise higher. Therefore, we select to use 4 MGCUs, resulting in precise prediction and high running efficiency.

Effects of CS-FBs. Here, we evaluate 1) the effectiveness of using relative features during cross-scale graph inference in CS-FBs; 2) different numbers of CS-FBs in a

Table 9. Average MAEs for different orders of motion differences.

	MAE at different time stamps (ms)			
Difference Order	80	160	320	400
$\beta = 0$	0.34	0.60	0.86	1.01
$\beta = 0, 1$	0.28	0.54	**0.83**	0.97
$\beta = 0, 1, 2$	**0.27**	**0.52**	**0.83**	**0.95**

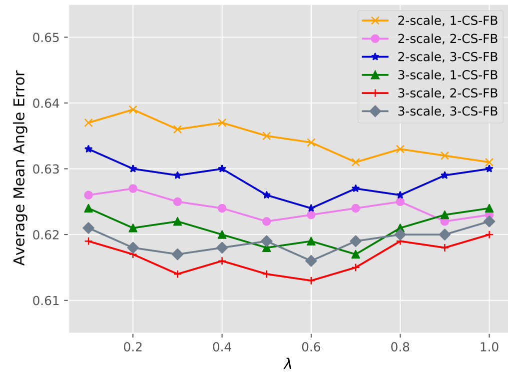

Figure 7. Average MAEs of DMGNN variants with different final fusion coefficient λ for short-term motion prediction.

sequence of 4 MGCUs. For 0 CS-FB, the model only fuses all scales at the end of the encoder. Table 8 presents the average MAEs with different CS-FBs and relative-feature mechanisms across 400 ms on H3.6M. We see that 1) using relative features leads to lower MAEs, validating the effectiveness of such augmented features; 2) 2 CS-FBs leads to the best prediction performance. The intuition is that 0 or 1 CS-FB fuse insufficiently and 3 CS-FBs tend to fuse redundant information to confuse the model.

Effect of λ in final fusion. The hyper-parameter λ in the final fusion (3) balances the influence between joint-scale and more abstract scales. Figure 7 illustrates the average MAE with different body scales and CS-FBs for short-term prediction on H3.6M. We see that the performance reach its best when we use 3 scales, 2 hierarchical CS-FBs and $\lambda = 0.6$, even though it is robust to the change of λ.

Effect of high-order motion differences. We study the effects of various orders of motion differences fed into the encoder and decoder of our model. We evaluate DMGNN with combinations of $0, 1, 2$-orders of pose differences. Table 9 presents the MAEs of DMGNN with various input differences for short-term motion prediction. We see that the proposed DMGNN obtains the lowest MAEs when it adopts the $0, 1, 2$-orders of motion differences. This indicates that high-order differences improve the prediction performance significantly.

6.4. Analysis of category-agnostic property

Here we validate that DMGNN can learn discriminative motion features for category-agnostic prediction.

We first visualize the learned cross-scale graphs for different actions to test the discriminative power. Figure 8 shows the graphs in two CS-FBs on 'Walking' and 'Directions' in H3.6M. For each action, we show some strong relations from detailed scales to the right arms in coarse scales. We see that i) for each action, the CS-FBs capture diverse ranges of a human body: the graph in the first CS-FB focuses on nearby body-components; the second CS-FB captures more global and action-related effects; i.e. hands and feet affects arms during walking; and ii) the cross-scale graphs are different for various actions, especially in the second CS-FB, capturing distinct patterns.

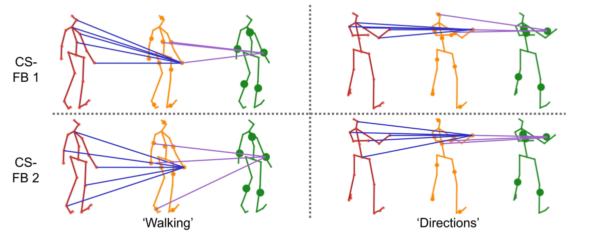

Figure 8. The learned dynamic cross-scale graphs on two CS-FBs for two actions: 'Walking' and 'Directions' in H3.6M.

Table 10. Classification accuracies on cross-scale graphs and motion features of DMGNN and other methods on H3.6M.

Methods	On CS-FB 1	On CS-FB 2	On $\mathbf{H}$	Res-sup. [32]	TP-RNN [7]
Accuracy	28.6%	40.1%	45.7%	22.6%	24.4%

We next conduct action classification on the intermediate representations to test the discriminative power. We isolatedly train a two-layer MLP to classify each dynamic cross-scale graph. We also classify the outputs from the encoders of DMGNN, Res-sup. (class-aware) and TP-RNN (class-agnostic). Table 10 presents the average classification accuracies on 15 categories of actions. We see that the cross-scale graph in the second CS-FB is more informative than the one in the first CS-FB for action recognition. Comparing to baselines, DMGNN obtains the highest the classification accuracies on encoder representation, indicating that DMGNN captures discriminative information for class-agnostic prediction.

7. Conclusion

We build dynamic mutiscale graphs to represent a human body and propose dynamic multiscale graph neural networks (DMGNN) with an encoder-decoder framework for 3D skeleton-based human motion prediction. In the encoder, We develop multiscale graph computational units (MGCU) to extract features; in the decoder, we develop a graph-based GRU (G-GRU) for pose generation. The results show that the proposed model outperforms most state-of-the-art methods for both short and long-trem prediction in terms of both effectiveness and efficiency.

Acknowledgement: This work is supported by the National Key Research and Development Program of China (No. 2019YFB1804304), SHEITC (No. 2018-RGZN-02046), NSFC (No. 61521062), 111 plan (No. B07022), and STCSM (No. 18DZ2270700).

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Feifei Li, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–971, June 2016.

[2] Emad Barsoum, John Kender, and Zicheng Liu. Hp-gan: Probabilistic 3d human motion prediction via gan. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1531–1540, June 2018.

[3] Apratim Bhattacharyya, Mario Fritz, and Bernt Schiele. Long-term on-board prediction of people in traffic scenes under uncertainty. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4194–4202, June 2018.

[4] Lon Bottou. Large-scale machine learning with stochastic gradient descent. In *International Conference on Computational Statistics (COMPSTAT)*, pages 177–187, August 2010.

[5] Judith Butepage, Michael Black, Danica Kragic, and Hedvig Kjellstrom. Deep representation learning for human motion prediction and classification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1591–1599, July 2017.

[6] Siheng Chen, Baoan Liu, Chen Feng, Carlos Vallespi-Gonzalez, and Carl Wellington. 3d point cloud processing and learning for autonomous driving. *IEEE Signal Processing Magazine Special Issue on Autonomous Driving*, 2020.

[7] Hsukuang Chiu, Ehsan Adeli, Borui Wang, DeAn Huang, and Juan Niebles. Action-agnostic human pose forecasting. *CoRR*, abs/1810.09676, 2018.

[8] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3844–3852, December 2016.

[9] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 4346–4354, December 2015.

[10] Partha Ghosh, Jie Song, Emre Aksan, and Otmar Hilliges. Learning human motion models for long-term predictions. *CoRR*, abs/1704.02827, 2017.

[11] Anand Gopalakrishnan, Ankur Mali, Dan Kifer, Lee Giles, and Alexander Ororbia. A neural temporal model for human motion prediction. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12116–12125, June 2019.

[12] Liangyan Gui, Yuxiong Wang, Xiaodan Liang, and Jose Moura. Adversarial geometry-aware human motion prediction. In *The European Conference on Computer Vision (ECCV)*, pages 786–803, September 2018.

[13] Liangyan Gui, Kevin Zhang, Yuxiong Wang, Xiaodan Liang, Jose Moura, and Manuela Veloso. Teaching robots to predict human motion. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2018.

[14] Xiao Guo and Jongmoo Choi. Human motion prediction via learning local structure representations and temporal dependencies. In *AAAI Conference on Artificial Intelligence*, February 2019.

[15] Ankur Gupta, Julieta Martinez, James Little, and Robert Woodham. 3d pose from motion for cross-view action recognition via non-linear circulant temporal encoding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2061–2068, June 2014.

[16] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1024–1034, December 2017.

[17] Dean Huang and Kris Kitani. Action-reaction: Forecasting the dynamics of human interaction. In *The European Conference on Computer Vision (ECCV)*, pages 489–504, July 2014.

[18] Du Huynh. Metrics for 3d rotations: Comparison and analysis. *Journal of Mathematical Imaging and Vision*, 35(2):155–164, October 2009.

[19] 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, July 2014.

[20] Ashesh Jain, Amir Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5308–5317, June 2016.

[21] Diederik Kingma and Jimmylei Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, pages 1–15, May 2015.

[22] Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, pages 1–14, April 2017.

[23] Hema Koppula and Ashutosh Saxena. Learning spatio-temporal structure from rgb-d videos for human activity detection and anticipation. In *International Conference on Machine Learning (ICML)*, pages 792–800, June 2013.

[24] Hema Koppula and Ashutosh Saxena. Anticipating human activities using object affordances for reactive robotic response. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 38(1):14–29, January 2016.

[25] Andreas Lehrmann, Peter Gehler, and Sebastian Nowozin. Efficient nonlinear markov models for human motion. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1314–1321, June 2014.

[26] Chen Li, Zhen Zhang, Wee Sun Lee, and Gim Hee Lee. Convolutional sequence to sequence model for human dynamics. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5226–5234, June 2018.

[27] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3595–3603, June 2019.

[28] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Symbiotic graph neural networks for 3d skeleton-based human action recognition and motion prediction. *CoRR*, abs/1910.02212, 2019.

[29] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. In *International Conference on Learning Representations (ICLR)*, pages 1–20, May 2016.

[30] Zhenguang Liu, Shuang Wu, Shuyuan Jin, Qi Liu, Shijian Lu, Roger Zimmermann, and Li Cheng. Towards natural and accurate future motion prediction of humans and animals. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10004–10012, June 2019.

[31] Wei Mao, Miaomiao Liu, Mathieu Salzmann, and Hongdong Li. Learning trajectory dependencies for human motion prediction. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[32] Julieta Martinez, Michael Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4674–4683, July 2017.

[33] Dario Pavllo, David Grangier, and Michael Auli. Quaternet: A quaternion-based recurrent model for human motion. In *British Machine Vision Converence (BMVC)*, pages 1–14, September 2018.

[34] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with directed graph neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7912–7921, June 2019.

[35] Chenyang Si, Ya Jing, Wei Wang, Liang Wang, and Tieniu Tan. Skeleton-based action recognition with spatial reasoning and temporal stack learning. In *The European Conference on Computer Vision (ECCV)*, pages 103–118, September 2018.

[36] Ilya Sutskever, Geoffrey Hinton, and Graham Taylor. The recurrent temporal restricted boltzmann machine. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1601–1608, December 2009.

[37] Graham Taylor and Geoffrey Hinton. Factored conditional restricted Boltzmann machines for modeling motion style. In *International Conference on Machine Learning (ICML)*, pages 1025–1032, June 2009.

[38] Graham Taylor, Geoffrey Hinton, and Sam Roweis. Modeling human motion using binary latent variables. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1345–1352, December 2007.

[39] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. Learning localized generative models for 3d point clouds via graph convolution. In *International Conference on Learning Representations (ICLR)*, pages 1–15, May 2019.

[40] Nitika Verma, Edmond Boyer, and Jakob Verbeek. Feastnet: Feature-steered graph convolutions for 3d shape analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2598–2606, June 2018.

[41] Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The pose knows: Video forecasting by generating pose futures. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 3332–3341, October 2017.

[42] Borui Wang, Ehsan Adeli, Hsukuang Chiu, Dean Huang, and JuanCarlos Niebles. Imitation learning for human pose prediction. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[43] He Wang, Edmond Ho, Hubert Shum, and Zhanxing Zhu. Spatio-temporal manifold learning for human motions via long-horizon modeling. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, PP(99), August 2019.

[44] Jack Wang, Aaron Hertzmann, and David Fleet. Gaussian process dynamical models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1441–1448, December 2006.

[45] Tianfan Xue, Jiajun Wu, Katherine Bouman, and Bill Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 91–99. December 2016.

[46] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 7444–7452, February 2018.

[47] Long Zhao, Xi Peng, Yu Tian, Mubbasir Kapadia, and Dimitris N. Metaxas. Semantic graph convolutional networks for 3d human pose regression. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3425–3435, June 2019.