V-DMGNN-GAN: Spatial Inpainting for Human Motion Prediction

Anushree Bannadabhavi[*] **Guanxiong Chen**

Yunpeng (Larry) Liu

Kaitai Tong

Abstract

For the task of modelling and predicting human motion trajectories, existing generative models suffer from occlusions and mislabeling that are commonplace in human motion datasets. We tackle this problem by building a network that can accurately infer trajectories of skeletal joints missing from input, by building upon previous work on deterministic multi-scale motion prediction. We build a GAN-based architecture consisting of a variational auto-encoder (VAE) as our generator network to learn the distribution of human motion and a recurrent neural network-based discriminator for explicitly penalizing the unrealistic motion that is generated from our generator. The source code of our paper can be found at `https://github.com/ericchen321/V-DMGNN-GAN`.

1 Introduction

Human motion modeling plays a crucial role in many applications, ranging from video game [8], autonomous driving [6], and human robot interactions [21]. This problem requires generating future skeletons given initial observations of the ground truth sequences. Previously, researchers propose many deep learning algorithms rely on deep generative models such as variation auto-encoder [6], generative adversarial network [26] and normalizing flow based [29] models which demonstrate convincing results given fully observed joints in the past sequences. However, in many real-world applications mocap systems need to deal with partially occluded, mislabelled or inaccurate joint poses [28, 29]. For example, in workplaces such as construction sites, lower-body joint poses are often unavailable because optical marker readings can be highly inaccurate due to foot-floor impacts and occlusions in cluttered capture spaces [28]. In autonomous driving, majority of the time a vehicle is only able to observe pedestrians partially by the RGB camera or Lidar, given the complex road environment. An accurate modeling of pedestrians could improve the safety of the autonomous agent.

Formally, we consider the task in hand as a spatial sequence-to-sequence (seq-to-seq) inpainting: predicting a sequence of future full skeletal poses from a sequence of past, partially occluded skeletal poses. We tackle it by building upon DMGNN [15], a recent work of deterministic multi-scale graph based motion prediction. Although DMGNN can achieve fairly good performance on predicting future poses from unoccluded past poses, it is not the ideal solution to our problem, because a deterministic model often fails to learn the multi-modality of human motion [27], especially when predicting motion in the far future [5]. We further believe that multi-modality can manifest itself to a greater extent when past poses are partially occluded. Meanwhile, State-of-the-art GAN-based models [11, 5, 14] have achieved competitive results on motion prediction, with MotionGAN [11] specifically built for spatial inpainting.

[*]Authors listed alphabetically by last name. Equal contribution from all authors.

Therefore we aim to replace DMGNN's deterministic, encoder-decoder motion generation network with a stochastic, variational auto-encoder (VAE)-based generation network, so it better learns the multi-modality human motion [27]. Also, we aim to augment DMGNN's motion generation network with a discriminator to leverage GAN-based models' ability to synthesize human motions with higher level of realism [5, 11].

The main contributions of this project are:

1) We replace DMGNN [15]'s existing deterministic motion generation network with a variational generation network, while retaining DMGNN's GCN-based multi-scale pose feature learning and synthesis.
2) On top of the variational generator network, we implement a discriminator network to explicitly discriminate unrealistic generated motion. Putting the discriminator and the generator together, we have the Variational DMGNN-GAN (V-DMGNN-GAN). We show that it can achieve performance on-par with the "vanilla" DMGNN when predicting future poses from occluded past poses.
3) Drawing inspiration from MotionGAN [11], we develop two masking techniques, lower-body masking and random masking to represent two common cases of joint occlusion.
4) In addition to evaluating our Variational DMGNN-GAN and the "vanilla" DMGNN on the CMU dataset that has already been preprocessed in the DMGNN paper [15] and is ready-to-use, we evaluate their performance on the ACCAD dataset [1] to show that our proposed network as well as "vanilla" DMGNN generalizes well to new datasets.

2 Related Work

In this section we discuss previous work on learning human motion with deep neural networks. We focus on networks that inpaint motion trajectories either spatially or temporally. Three main works that we drew inspiration from are (i) a GAN based model - MotionGAN[11] that provides long term pose predictions even when the input has missing joints or occlusions, (ii) a VAE based model - MT-VAE[27] that addresses the multi-modality of human motion and produces diverse motion predictions and (iii) Dynamic Multiscale Graph Neural Network (DMGNN)[15] that uses a multiscale graph to represent human pose and captures the internal relations of the human body for motion feature learning. While these models have several advantages, they have some drawbacks:

1) MotionGAN [11] does not use multiple scales for modelling the human body.
2) MT-VAE [27] only takes full poses as input.
3) DMGNN is a deterministic model and does not address the issue of missing joints or occlusions.

Therefore in our work **Variational DMGNN-GAN** (V-DMGNN-GAN), we are proposing a GAN-based model to infer the trajectories of skeletal joints missing from input. The generator of GAN is stochastic (VAE-based model) in order to obtain diverse outputs and the encoder of the generator has multi-scale graph units for better learning of human poses.

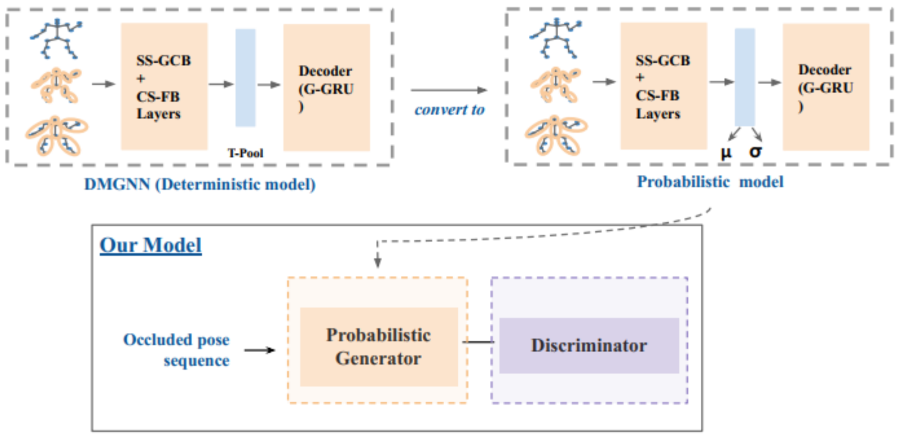

Figure 1: Variational DMGNN-GAN: architectural overview.

In the following subsections, we discuss common cases of occlusion and look into how prior work formulated the occlusion problem. Then we discuss generative models built for motion prediction. Next, we look into works that exploit the spatio-temporal relationships between joints and other scene structures with graph-based neural networks. Finally we discuss human motion datasets available for us to use.

2.1 Joint Occlusion and Masking

Based on existing studies in motion capture of occluded, mislabelled and inaccurate poses [28, 29, 19], we broadly categorize joint occlusion into two cases: a) lower-body occlusion, which is common when motion capture takes place in cluttered environments [28] such as construction cites, with users having to perform heavy duties such that foot-floor impacts inject considerable noise to marker readings on lower-body [28]. b) random occlusion, when joint pose readings are missing or inaccurate due to simple optical occlusion [29, 19].

MotionGAN [11] employs binary masking to emulate structural occlusion (specific joints occluded) and random occlusion (random joints occluded). Following their approach, we develop lower-body masking and random joint masking, considering lower-body occlusion as a case of structural occlusion. More details on our masking techniques are in Section 6.2.

2.2 Human Motion Prediction with Generative Models

Recent work typically employ generative models such as GANs [11, 5, 14], VAEs [27, 7, 24] or recurrent models [9, 25] to infer future human skeletal motion from motions in the past. Hernandez et. al. [11] proposed a GAN model with a linear combination of reconstruction loss, limb distances loss, bone length loss and regularized adversarial loss, trained under a self-supervised learning manner by skeleton inpainting. MT-VAE [27] used a VAE-based architecture to make stochastic motion sequence predictions and addressed the multi-modality of human motions. [9] on the other hand combined RNN with VAE for long-term motion prediction.

2.3 Learning Skeletal Motion with Graph Neural Networks

For action recognition and skeletal motion prediction, recent deep learning based approaches have leveraged graph-based neural networks to exploit the spatio-temporal relations among joints, as well as the relationships between the human and surrounding objects (contextual information).

Recent work on action recognition such as [26, 22, 20] built human skeletons as graphs and applied graph convolution operations to extract features that allow their networks to distinguish actions. [22] proposes using directed acyclic graph (DAG) to represent the relationship between joints and bones in skeleton data for improving action recognition task. [15] proposed a dynamic multiscale graph neural network (DMGNN) with an encoder-decoder framework for 3D skeleton based human motion prediction. This work exploits the relations between different joints and different body parts by using a multiscale graph representation for the human body. [29] used spatial-temporal graph convolutional network (ST-GCN) to capture the spatio-temporal relations of skeletal joints and extended MoGlow [10] to generate motion patterns from incomplete past pose inputs.

2.4 Human Motion Datsets

Large, diverse human motion datasets are essential for deep neural networks to learn to predict or classify human actions. Such datasets come in a variety of modalities — e.g. marker based ([2, 18]), video-based ([13, 23]) or multimodal ([12]); they can also be broadly categorized into fully-body motion datasets [2, 17, 13, 16] and facial datasets [30, 3] based on the body part for which motions are captured. For our training and evaluation, we use the CMU Mocap dataset [2] already preprocessed in DMGNN [15]. AMASS [16] is a collection of a multitude of human motion datasets, including the aformentioned CMU Mocap dataset and the KIT dataset. We pick the ACCAD dataset [1] from AMASS to evaluate our proposed V-DMGNN-GAN and our baselines.

3 Method

We denote a sequence of human skeletal poses by $x_{n=1:N}^{t=1:T}$, where N represents the number of joints within a pose, and T represents the number of frames captured. The pose of a single joint in a single time frame, $x_n^t \in \mathbb{R}^3$ is represented by three Euler angles that define the rotation of a joint with respect to its parent joint on the human skeleton kinematic chain. We define the seq-to-seq motion prediction problem as follows: given an observed sequence of poses, $x^{-(T_{in}-1):0}$, we want to generate a sequence of future poses, $x^{1:T_{out}}$.

In the following sections, we first discuss the original DMGNN's encoder module and the changes we made to make the deterministic DMGNN probabilistic. Then we briefly demonstrate the DMGNN's decoder as we do not modify the original decoder. As we are proposing a GAN model, we also talk about our discriminator design in section 3.2 with our GAN loss.

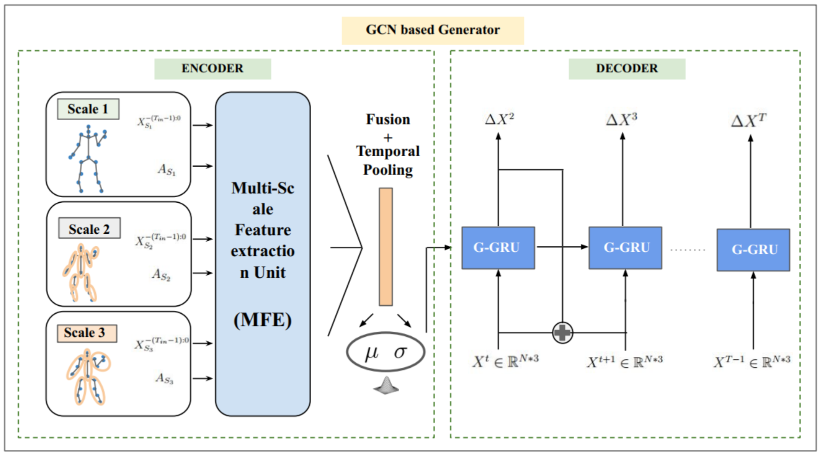
GCN based Generator
ENCODER
DECODER
Scale 1
Scale 2
Scale 3
$X_{S_1}^{-(T_{in}-1):0}$
A_{S_1}
$X_{S_2}^{-(T_{in}-1):0}$
A_{S_2}
$X_{S_3}^{-(T_{in}-1):0}$
A_{S_3}
Multi-Scale Feature extraction Unit
(MFE)
Fusion + Temporal Pooling
μ σ
ΔX^2
ΔX^3
ΔX^T
G-GRU
G-GRU
G-GRU
$X^t \in \mathbb{R}^{N*3}$
$X^{t+1} \in \mathbb{R}^{N*3}$
$X^{T-1} \in \mathbb{R}^{N*3}$

Figure 2: Probabilistic Multiscale GCN-based generator. The input of the generator consists of three scales. The node of the graph, A_{s1} in the first scale is the original human joints. For the second and third scales, the node of the graph is artificially chosen to capture higher-level parts of the body. The encoder takes in these three scale features and outputs a latent distribution, $q_\phi(z|x^{-(T_{in}-1):0})$. A sample from this distribution is used as the initial hidden state for the decoder. The decoder is a recurrent network that takes in the initial frame $t = 1$of pose and outputs the difference for the next frame. At the next timestep, $t = 2$, the difference is added to the previous frame as the input.

3.1 Probabilistic Generator

The generator of our GAN-based model is a VAE model that built on top of a existing deterministic human motion prediction model, DMGNN. To make our model probabilistic, we replace the original DMGNN's hidden state H with a latent variable z for producing a probabilistic future prediction. The overall structure of our generator is shown in Figure 2.

Multi-scale feature extraction unit (MFE). We first briefly explain the encoding step of the original DMGNN for learning the hidden state, a detailed formulation can be found in the Appendix section 6.3. A single scale input consists of the joints $x_{1:N}^{-(T_{in}-1):0} \in \mathbb{R}^{M_s \times T_{in} \times D}$. The corresponding graph represented by a adjacency matrix $A_s \in \mathbb{R}^{M_s \times M_s}$ at scale s, where M_s represents the number of body parts at scale s and the D is the number of features of a single component. DMGNN initializes the A_s by the skeleton graph. At scale $s = 1$, each node represents a single joint and the edges represents the physical connection of the body joints.

The MFE units takes in the multi-scale inputs and outputs enlarged channel features for each scale that linear combined into a single hidden state, $H \in \mathbb{R}^{T_h \times D_h \times M_{s1}}$. Two essential modules of the MFE are the single-scale graph convilution block, (SS-GCB) and the cross-scale fusion block (CS-FB). The SS-GCB consists of spatial graph convolution and temporal convolution networs that learns spatial temporal features for a single scale. The CS-FB then fusing this spatial temporal features across scales using an attention graph. Figure 3 includes our detailed MFE block diagram.

Learning Latent variable z. So far we have been following the DMGNN design for the encoder, we need to learn the conditional distribution of z for our probabilistic generator's encoder, $q_\phi(z|x^{-(T_{in}-1):0})$ where ϕ represents the learnable parameters. To do so, we first add an additional temporal convolutional network layer for reducing the temporal dimension from $T_h$to T_z. Then

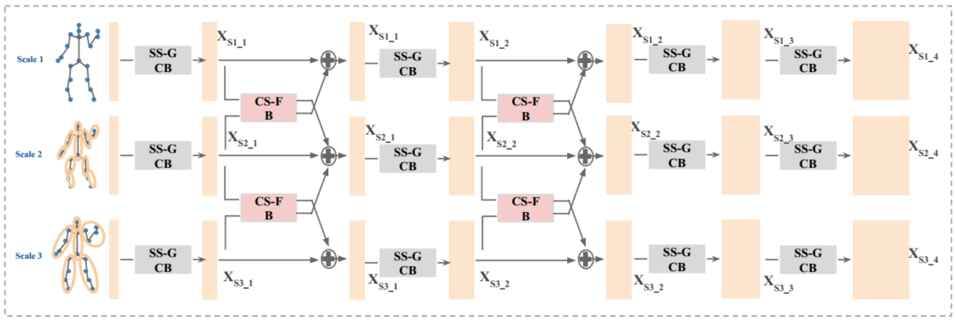

Figure 3: Multi-scale feature extraction unit (MFE), consists of 4 layers of SS-GCB blocks and the two cross-scale fusion blocks are applied after the first and second layer of the MFE.

same as DMGNN we apply temporal pooling to aggregate the temporal information, results in $H_z \in \mathbb{R}^{M_{s1} \times D_h}$. Two fully connected layers takes in H_z and outputs the mean, μ and the covariance σ, i.e.:

$$\mu = W_\mu H_z \in \mathbb{R}^{M_{s1} \times D_z} \tag{1}$$

$$\sigma = W_\sigma H_z \in \mathbb{R}^{M_{s1} \times D_z} \tag{2}$$

$$\kappa \sim N(0,1) \tag{3}$$

$$z = \mu + \sigma * \kappa \in \mathbb{R}^{M_{s1} \times D_z}. \tag{4}$$

The W_μ and W_σ are the weights for the fully connected layer. Different to common VAE that the latent space is in 1-dimensional, our latent variable z represents hidden information at each joint.

DMGNN decoder. The DMGNN is a graph-based GRU that originally takes the temporal pooled hidden state, $H' \in \mathbb{R}^{M_{s1} \times D_h}$ as the initial hidden state to the GRU. We do not modify this decoder as it is capable of maintaining the joints structure while predicting the future poses. Compare to the original GRU, the G-GRU also includes a trainable graph, represents by the adjacency matrix, $A_H \in \mathbb{R}^{M_{s1} \times M_{s1}}$, which uses the skeleton-graph as initialization. We put a detailed formulation for the G-GRU in the Appendix section 6.1 for reference.

Probabilistic generator loss. Unlike the common VAE model that tries to reconstruct the input, our VAE model tries to generate a sequence of future poses, $x^{1:T_{out}}$. Our objective function of the generator is as follows:

$$L_{generator} = \frac{1}{T} * \sum_{t=1}^{T} ||(\hat{x}^t - x^t)||_1 + \lambda KL(q_\phi(z|x^{-(T_{in}-1):0})||N(0,I)) \tag{5}$$

The first term of the objective function is the L1 loss between the predicted future pose $\hat{x}^t$ and the ground truth pose x^t at time t. The second term acts as a regularizer. The overall objective is to output realistic future poses.

3.2 GAN Discriminator

Our probabilistic model is able to generate future poses given a past sequence, but it does not explicitly penalize that the generated sequence is realistic motion. Therefore, we design GAN includes a discriminator that tries to distinguish between real motion and generated fake motion as shown in Figure 4.

To capture the conditional probability, $p(x^{1:T}|x^{-(T_{in}-1):0})$ which models the variations of motion that generator produces, inspired by [26][5], we use a recurrent neural network to learn the hidden representation of the motion as shown in Figure 5. Conveniently, we adapt the architecture of the G-GRU unit from the decoder to use as our recurrent neural network. We discard the output of the G-GRU unit and feed in the sequence of poses at each timestep for learning the hidden motion representation, h_T^{Dis}. Multiple fully connected layers then take in h_T^{Dis} and output the label of this sequence, label$\in [0,1]$. The adversarial loss for our discriminator is as follows

$$L_D = \mathbb{E}[log(1 - D(\{x_{1:N}^{-(T_{in}-1):0}, G(x_{1:m}^{-(T_{in}-1):0})\}))] + \mathbb{E}[log(D(x_{1:N}^{-(T_{in}-1):T}))], \tag{6}$$

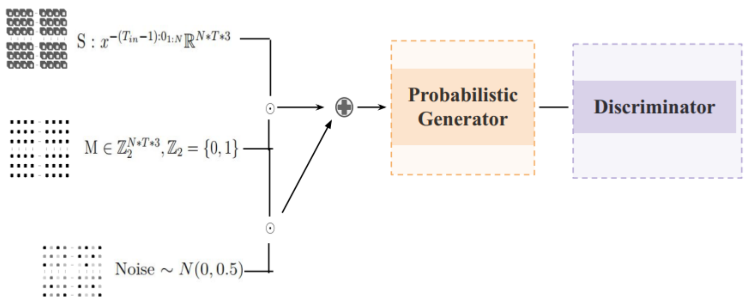
S : $x^{-(T_{in}-1):0_{1:N}}\mathbb{R}^{N*T*3}$
M $\in \mathbb{Z}_2^{N*T*3}, \mathbb{Z}_2 = \{0,1\}$
Noise $\sim N(0, 0.5)$
Probabilistic Generator
Discriminator

Figure 4: Proposed GAN architecture. The generator takes in the masked past sequence of poses and generates the future sequence of poses. The discriminator takes in the full ground truth sequence of poses from$-(T_{in}-1)$ to T and concatenated generated sequence with the ground truth past sequence, $\{x^{-(T_{in}-1):0}, \hat{x}^{1:T}\}$ to distinguish realistic and fake motion.

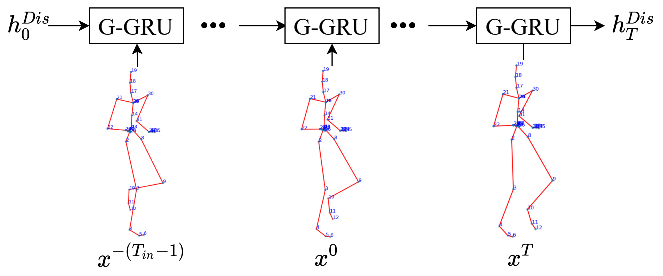

Figure 5: Discriminator recurrent network head, the initial hidden, h^{Dis} is initialized with zeros. Compare to a auto-regressive model, we do not feed in the output of GRU at time t to the next

where $\{,\}$ represents concatenation of two sequences and $x_{1:m}^{-(T_{in}-1):0}$ represents the masked input. Additional adversarial loss is added to Equation 5.

$$L_{generator} = \frac{1}{T} * \sum_{t=1}^{T} ||(\hat{x}^t - x^t)||_1 + \lambda_{KL} KL(q_\phi(z|x^{-(T_{in}-1):0})||N(0,I)) + \boldsymbol{\lambda_{gan}} \mathbb{E}[\boldsymbol{log(D(\{x_{1:N}^{-(T_{in}-1):0}, G(x_{1:m}^{-(T_{in}-1):0})\}))}] \quad (7)$$

Training GAN to achieve equilibrium between the generator loss and discriminator loss is challenging. The ideal scenario is the generator is able to generate realistic future motion that the discriminator can not distinguish. We clip our discriminator parameters after each parameters update as suggested in [4] to help stabilize the training.

4 Experiments

In this section we discuss the experiments we set up to compare our network with several baselines on lower-body inpainting and random inpainting.

4.1 Datasets and Preprocessing

We use two datasets for our prediction comparision experiment:

1) the CMU Mocap dataset [2]. This dataset contains 71,124 diverse motion sequences from eight classes of actions: basketball, basketball signalling, directing traffic, jumping, running, playing soccer, walking and window washing. It has already been preprocessed by DMGNN [15]'s authors and we can use them out-of-the-box;
2) the ACCAD dataset [1] as a part of the AMASS human motion dataset collection[16]. We pick this dataset because for two reasons: (a) It contains an even-larger number of motion sequences (85,826) from 15 action classes; (b) It has been used to evaluate HG-VAE [7] which is a baseline

we picked, so using it would lead to meaningful benchmarking. Because ACCAD was not evaluated on the original DMGNN, we created our own preprocessing pipeline to convert ACCAD data into files compatible with DMGNN as well as our GAN network.

Both the CMU Mocap dataset and the ACCAD dataset come with sequences of fully-visible joint pose sequences. To simulate occlusions we use the masking techniques detailed in 6.2, and we feed the masked sequence to our model.

4.2 Training

We see from Figure 6 that the generator loss gradually decreases over the iterations, while the discriminator loss plateaus, suggesting our generator is able to learn from the training process. For more details on our training process, including hyperparameter choices and training platform specifications, please refer to Section 6.4.

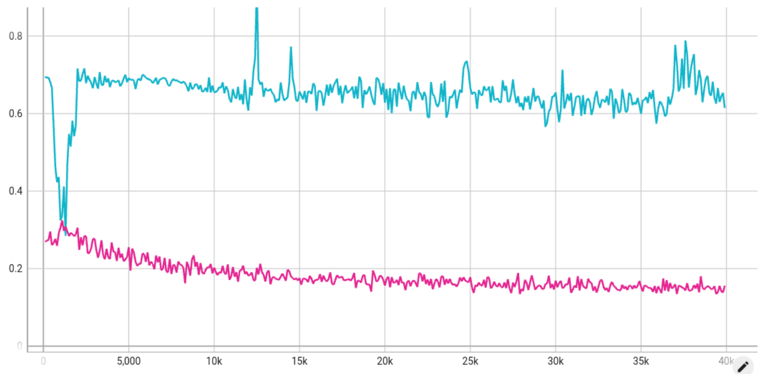

Figure 6: Loss curve from training our V-DMGNN-GAN. The red curves shows the progression of our generator loss; the blue curve shows the discriminator loss.

4.3 Metrics

We quantitatively evaluate our model's performance using the Mean Angle Error (MAE) metric. For each predicted frame, we compute the Euclidean norm of the errors between the predicted and ground-truth Euler angles, and average over the entire batch (size m).

$$MAE = \frac{1}{m} \sum_{i=1}^{m} ||\overrightarrow{\theta_{i,gt}^{t}} - \overrightarrow{\theta_{i,pred}^{t}}||_2, \tag{8}$$

4.4 Baselines

We compared the MAE metric of our V-DMGNN-GAN with the following baselines:

1) **The "Vanilla" DMGNN [15]**: To train and evaluate DMGNN on the ACCAD dataset, we created our own preprocessing pipeline to convert ACCAD data into files compatible with DMGNN.
2) **HG-VAE [7]:** another VAE-based network that was specifically designed to predict future poses from past occluded poses. Note that HG-VAE measures the quality of its prediction with the Mean per-joint Position Error (MPJPE). Since we are using the MAE metric as used in DMGNN [15], we built a pipeline to convert from MPJPE to MAE.

4.5 Result Analysis

In this section we anaylze our results from the prediction quality experiments, and we comment on the impact of several design choices based on the results from our ablation study.

Quantitative analysis. Our GAN model performs competitively with the original DMGNN for the structured occlusion at 40ms as shown in Table 1. However, our model performs worst than the original DMGNN as the prediction horizon increases. We think one reason that contributes to this result is our model is a probabilistic model rather than deterministic, and the MAE metric only measures the difference to the ground truth prediction but does not favour generating diverse future poses. We slightly trade-off our ground truth reconstruction performance for generating probabilistic future poses. In addition, we are not using velocity and acceleration information in our encoder module which we observed would help improve the MAP performance in our ablation study. In general, our model and DMGNN perform worst to unstructured random joints, we believe random joint masking is hard for the sequence to sequence model to learn as the inconsistent memory produced through random masking is hard to recover. This observation is useful to consider for our future work.

Frame	40 ms (↓)		240 ms (↓)		400 ms (↓)	
Masking / Action Class	Lower-body	Random	Lower-body	Random	Lower-body	Random
Basketball	0.477 / 0.427	0.620 / 0.415	1.074 / 0.789	**1.028** / 1.120	1.500 / 1.205	1.341 / 1.476
Basketball signalling	**0.360** / 0.422	0.609 / 0.407	0.686 / 0.494	0.736 / 0.716	0.905 / 0.669	0.971 / 0.937
Directing traffic	0.854 / 0.802	0.960 / 0.636	0.845 / 0.749	0.965 / 0.770	1.040 / 0.899	1.189 / 0.984
Jumping	0.558 / 0.507	**0.968** / 0.981	**1.039** / 1.068	**1.112** / 1.153	**1.796** / 1.798	**1.739** / 1.885
Running	0.387 / 0.238	0.615 / 0.278	0.674 / 0.422	0.584 / 0.517	0.906 / 0.451	0.726 / 0.554
Soccer	**0.401** / 0.437	0.519 / 0.345	0.747 / 0.670	0.783 / 0.774	0.992 / 0.838	**0.978** / 1.002
Walking	0.276 / 0.272	0.506 / 0.310	0.425 / 0.302	0.393 / 0.389	0.553 / 0.407	**0.486** / 0.519
Washing window	**0.279** / 0.289	0.578 / 0.346	0.677 / 0.473	0.728 / 0.568	1.060 / 0.836	1.089 / 0.841

Table 1: MAE evaluated with 'vanilla' DMGNN and our GAN implementation on the eight action classes from the CMU dataset [2]; Lower-body masking and random masking; **Ours** / DMGNN. We highlight test cases for which our model performs better.

Qualitative analysis. We see from Figure 7 that for this particular sequence, while our GAN achieves

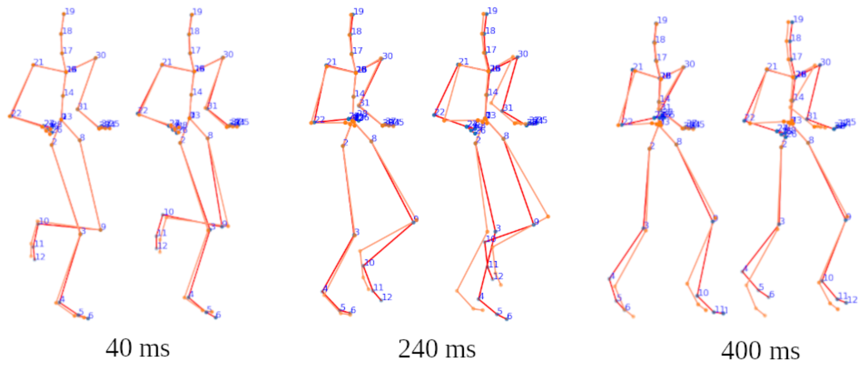

Figure 7: Target (orange) vs. predicted (red) poses on a running sequence from the CMU dataset [2]. Joints on the right leg are masked. Here we visualize skeletal poses at three frames: 40 ms, 240 ms and 400 ms. For each frame we have prediction from the "vanilla" DMGNN model shown on the left, and prediction from our V-DMGNN-GAN shown on the right.

roughly comparable performance at 40 ms with DMGNN, its deviation from the target pose becomes larger than DMGNN on 240 ms and 400 ms. The discrepancy is most obvious at 240 ms on the lower body on which we are masking. Also, we see our GAN not only makes less accurate prediction on the masked lower body, but also on the unmasked upper body. This observation suggests our model has failed to capture motion dynamics local to body parts not affected by occlusion.

4.6 Ablation Study

For our GAN model, we investigate the impact of several architectural decisions on the quality of motion synthesis, both in terms of short-range prediction (40 ms) and long-range prediction (400 ms). The results are shown in Table 2.

1) L1 vs L2 reconstruction loss. Our base network follows "vanilla" DMGNN [15]'s design of using the L1 loss function to guide our gradient descent. DMGNN's authors argued they used L1 loss since it leads to more stable gradients and better testing performance [15]. Suprisingly, we see that their claim is not true for our GAN model: with L2 loss the model performs better than with L1 loss.

2) Joint (angular) velocity and acceleration. In "vanilla" DMGNN the network takes derivatives of joint angle with respect to time to get joint angular velocity and acceleration. Then joint angle, velocity and acceleration are then fed to the encoder and processed independently, concatenated to a single latent vector and passed to the decoder. We removed velocity and acceleration from the input to our generator network since we thought masked joint angles would make derivatives too noisy to be learnable features. Nevertheless velocity and acceleration contain abundant amount of information of human motion dynamics[15]. So we add velocity and acceleration back as inputs to the encoder network of our generator, and we find that with velocity and acceleration the GAN's performance boosts significantly on all three time scales. This result together with our qualitative analysis that our base model fails to capture the dynamics of non-occluded joints suggests velocity and acceleration are pivotal to motion prediction.

Model Variant	40 ms (↓)	240 ms (↓)	400 ms (↓)
Base (L1 reconstruction loss, no velocity or acceleration input to generator)	0.449	0.771	1.094
L2 reconstruction Loss	0.491	0.685	0.996
Added velocity and acceleration as generator input	0.442	0.656	0.960

Table 2: Comparision between three GAN model variants: 1) our base setup; 2) L2 reconstruction loss; 3) added velocity and acceleration as generator input. Lower-body masking. The MAE metrics are averaged over eight classes on the CMU test set.

5 Conclusion and Future Work

We have proposed a novel GAN with a variational generator network built on top of the GCN-based multi-scale DMGNN [15] for future skeletal motion prediction from partially occluded past poses. To emulate pose occlusion we developed two masking techniques, lower-body masking and random joint masking. By leveraging the stochastic nature of VAE and GAN's ability to synthesize realistic samples, we are able to achieve prediction performance (measured by MAE) comparable to the "vanilla" DMGNN on the CMU dataset [2], as well as the ACCAD dataset [1] which has never been evaluated on DMGNN before. Visualization of our predicted poses and results from our ablation study suggests that we can further improve our network by integrating joint velocity and acceleration as inputs to our network, or using L2 loss for training. Also, due to time constraints we are not able to report MAE from the HG-VAE baseline. This should be left as a part of our future work.

References

[1] Accad mocap system and data. `https://accad.osu.edu/research/motion-lab/system-data`. Accessed: 2022-04-16.

[2] Cmu graphics lab motion capture database. `http://mocap.cs.cmu.edu/`. Accessed: 2022-02-22.

[3] N. Aifanti, C. Papachristou, and A. Delopoulos. The mug facial expression database. In *11th International Workshop on Image Analysis for Multimedia Interactive Services WIAMIS 10*, pages 1–4. IEEE, 2010.

[4] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223. PMLR, 06–11 Aug 2017. URL `https://proceedings.mlr.press/v70/arjovsky17a.html`.

[5] E. Barsoum, J. Kender, and Z. Liu. Hp-gan: Probabilistic 3d human motion prediction via gan. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1418–1427, 2018.

[6] A. Bhattacharyya, M. Fritz, and B. Schiele. Long-term on-board prediction of people in traffic scenes under uncertainty. *CoRR*, abs/1711.09026, 2017. URL `http://arxiv.org/abs/1711.09026`.

[7] A. Bourached, R. Gray, R.-R. Griffiths, A. Jha, and P. Nachev. Hierarchical graph-convolutional variational autoencoding for generative modelling of human motion. *arXiv preprint arXiv:2111.12602*, 2021.

[8] N. M. Gamage, D. Ishtaweera, M. Weigel, and A. Withana. So predictable! continuous 3d hand trajectory prediction in virtual reality. In J. Nichols, R. Kumar, and M. Nebeling, editors, *UIST '21: The 34th Annual ACM Symposium on User Interface Software and Technology, Virtual Event, USA, October 10-14, 2021*, pages 332–343. ACM, 2021. doi: 10.1145/3472749.3474753. URL `https://doi.org/10.1145/3472749.3474753`.

[9] I. Habibie, D. Holden, J. Schwarz, J. Yearsley, and T. Komura. A recurrent variational autoencoder for human motion synthesis. In *BMVC*, 2017.

[10] G. E. Henter, S. Alexanderson, and J. Beskow. Moglow: Probabilistic and controllable motion synthesis using normalising flows. *ACM Trans. Graph.*, 39(6), nov 2020. ISSN 0730-0301. doi: 10.1145/3414685.3417836. URL `https://doi.org/10.1145/3414685.3417836`.

[11] A. Hernandez, J. Gall, and F. Moreno-Noguer. Human motion prediction via spatio-temporal inpainting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7134–7143, 2019.

[12] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013.

[13] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[14] J. N. Kundu, M. Gor, and R. V. Babu. Bihmp-gan: Bidirectional 3d human motion prediction gan. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 8553–8560, 2019.

[15] M. Li, S. Chen, Y. Zhao, Y. Zhang, Y. Wang, and Q. Tian. Dynamic multiscale graph neural networks for 3d skeleton based human motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[16] N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. J. Black. AMASS: Archive of motion capture as surface shapes. In *International Conference on Computer Vision*, pages 5442–5451, Oct. 2019.

[17] C. Mandery, Ö. Terlemez, M. Do, N. Vahrenkamp, and T. Asfour. The kit whole-body human motion database. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 329–336. IEEE, 2015.

[18] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Mocap database hdm05. *Institut für Informatik II, Universität Bonn*, 2(7), 2007.

[19] J. Niu, X. Wang, D. Wang, and L. Ran. A novel method of human joint prediction in an occlusion scene by using low-cost motion capture technique. *Sensors*, 20(4):1119, 2020.

[20] W. Peng, X. Hong, H. Chen, and G. Zhao. Learning graph convolutional network for skeleton-based human action recognition by neural searching, 2019.

[21] P. Sermanet, C. Lynch, J. Hsu, and S. Levine. Time-contrastive networks: Self-supervised learning from multi-view observation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 486–487. IEEE, 2017.

[22] L. Shi, Y. Zhang, J. Cheng, and H. Lu. Skeleton-based action recognition with directed graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7912–7921, 2019.

[23] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[24] J. Walker, C. Doersch, A. Gupta, and M. Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *European Conference on Computer Vision*, pages 835–851. Springer, 2016.

[25] H. Wang, J. Dong, B. Cheng, and J. Feng. Pvred: A position-velocity recurrent encoder-decoder for human motion prediction. *IEEE Transactions on Image Processing*, 30:6096–6106, 2021.

[26] S. Yan, Y. Xiong, and D. Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[27] X. Yan, A. Rastogi, R. Villegas, K. Sunkavalli, E. Shechtman, S. Hadap, E. Yumer, and H. Lee. Mt-vae: Learning motion transformations to generate multimodal human dynamics. In *Proceedings of the European conference on computer vision (ECCV)*, pages 265–281, 2018.

[28] D. Yang, D. Kim, and S.-H. Lee. Lobstr: Real-time lower-body pose prediction from sparse upper-body tracking signals. In *Computer Graphics Forum*, volume 40, pages 265–275. Wiley Online Library, 2021.

[29] W. Yin, H. Yin, D. Kragic, and M. Björkman. Graph-based normalizing flow for human motion generation and reconstruction. *CoRR*, abs/2104.03020, 2021. URL `https://arxiv.org/abs/2104.03020`.

[30] S. Zafeiriou, D. Kollias, M. A. Nicolaou, A. Papaioannou, G. Zhao, and I. Kotsia. Aff-wild: valence and arousal'in-the-wild'challenge. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 34–41, 2017.

6 Appendix

6.1 G-GRU

G-GRU is proposed in DMGNN that built upon the existing recurrent nerual network model, gated recurrent units, GRU. GRU includes update gate operation, reset gate operation, and one operation for finding the current memory. Finally, the output of GRU is the next hidden state which summarizes all the history of the past.

These operations are updated taking into the graph structure for G-GRU,

$$\begin{aligned} r^t &= \text{Sigmoid}(\phi_{r_{in}}(x^t_{1:N}) + \phi_{r_{hid}}(A_H H^t W_H)) \\ u^t &= \text{Sigmoid}(\phi_{u_{in}}(x^t_{1:N}) + \phi_{u_{hid}}(A_H H^t W_H)) \\ c^t &= \tanh\big(\phi_{c_{in}}(x^t_{1:N}) + r^t \circledast \phi_{c_{hid}}(A_H H^t W_H)\big) \\ H^{t+1} &= u^t \circledast H^t + (1 - u^t) \circledast c^t. \end{aligned}$$

The W_H is the trainable parameters and $x^t_{1:N}$ is the input pose at time t and ϕ_* represent linear layers. Graph convolution, $\circledast$ is done for updating the hidden state.

Our discriminator relies on the final output hidden state at T which capture the full history of motion for the full sequence to decide if the motion is real or fake.

6.2 Joint Masking

As introduced in Section 2.1, we develop two masking techniques, lower-body masking and random masking to emulate two broad categories of joint occlusion. We visualize them in Figure 8.

Lower-body masking. Given a sequence of input pose matrix of shape $N \times T \times 3$, the lower-body masking matrix S_L has the same shape, with non-occluded joint angles being 1 and occluded joint angles being 1.

Random masking. The random masking matrix S_R also has the same shape as an input pose matrix, with each joint angle's probability of being visible following the Bernoulli distribution $Bernoulli(p = 0.8)$.

We then element-wise multiply the masking matrix with the full input sequence $x^{t=1:T}_{n=1:N}$ to get a sequence of partially occluded poses. We also recognize that in reality, a joint measurement when measured under occlusion or impact [28] is noisy as opposed to being plain zero. Therefore for each occluded joint measurement we inject a Gaussian noise.

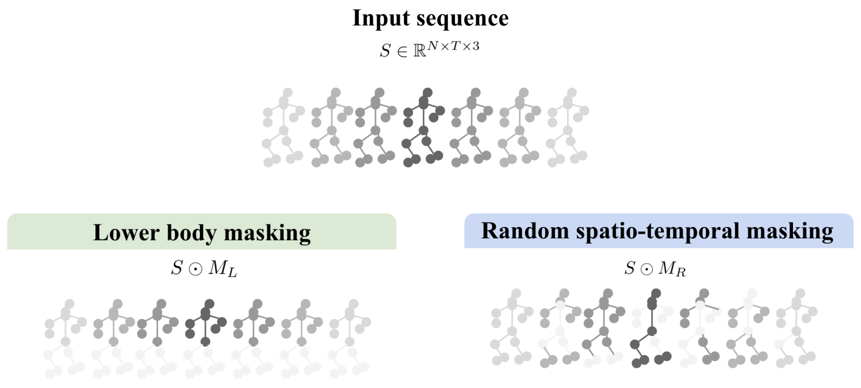
Input sequence
$S \in \mathbb{R}^{N \times T \times 3}$
Lower body masking
$S \odot M_L$
Random spatio-temporal masking
$S \odot M_R$

Figure 8: Proposed masking techniques and their effects.

6.3 DMGNN Encoder

DMGNN's encoder relies on multiple layers of single-scale graph convolution block (SS-GCB) combined with cross-scale fusion blocks(CS-FB) introduced from DMGNN as shown in Figure 3.

The SS-GCB block consists of a spatial graph convolutional neural network(SP-GCN) and a temporal convolutional neural network(TCN). The SP-GCN extract spatial features from neighbouring nodes.

$$X_{out} = ReLU(A_s x_{1:N}^{-(T_{in}-1):0} W_s + x_{1:N}^{-(T_{in}-1):0} B_s) \tag{9}$$

The $X_{out} \in M_s \times T_{in} \times \hat{D}$ and $W_s, B_s \in D \times \hat{D}$ are the training parameters. Then the temporal convolutional network extracts feature from the time domain. The kernel size of the TCN decides the number of sequences that are considered for a single frame temporally.

After extracting spatial-temporal features at each scale separately, DMGNN designed a attention graph, $A_{s,s'} \in M_s' \times M_s$ for fusing features at different scales. This fusion step, CS-FB happens between two SS-GCB blocks for a single level as shown in Figure 3. This fusion step embeds extra information into the learned single scale features from other scales which feed to the next SS-GCB block for feature learning. Intuitively, the arm movement learned at a higher scale is beneficial for learning the hand movement at a lower scale.

Given two features $X_{s,i}$, $X_{s',k} \in \mathbb{R}^{T_l \times D_{l,s}}$ from scale s and s' where $D_{l,s}$ represents the output feature from the SS-GCB block l at scale s. DMGNN first learns the embeddings $h_{s,i}$ and $h_{s',k}$ for each component using multilayer perceptron (MLP) where $i \in [1, M_s]$ and $k \in [1, M_s]'$, at each scale from vectorized $x_{s,i}^{-(T_{in}-1):0}$ and $x_{s',i}^{-(T_{in}-1):0}$. Then the attention graph $A_{s,s'}$ is calculated as follow:

$$A_{s,s'}(k,i) = \text{softmax}(h_{s',k}^{\top} h_{s,k}) \tag{10}$$

This attention graph $A_{s,s'}$ indicates which part at scale s' is more relevant to the part at scale s. The output of the fusion for X_s' is:

$$X_{s',out} = A_{s,s'} X_s W_{fuse,s} + X_{s'}, \tag{11}$$

where $W_{fuse,s}$ is the learnable parameter $\in D_{l,s} \times D_{l,s'}$. For X_s, it can be calculated using the same fashion as Equation, 11 with transposed $A_{s,s'}$.

DMGNN claims that applying the two CS-FB blocks separately for the first SS-GCB blocks works the best as three CS-FB blocks fuse redundant information.

The final outputs of the multi-scale feature extraction unit(MFE), $H_{s_1}, H_{s_2}, H_{s_3}$ are fused linearly with hyper-parameter λ to $H \in \mathbb{R}^{T_h \times D_h \times M_{s1}}$:

$$H = H_{s1} + \lambda(H_{s2} + H_{s3}) \tag{12}$$

6.4 More Training Details

Hyperparameters.

Name	
λ_{KL}	0.1
λ_{GAN}	0.07
Parameter clipping	(-0.25, 0.25)
Discriminator learning rate	0.000004
Generator learning rate	0.00005

Table 3: Hyperparameters used for training V-DMGNN-GAN.

Training GAN to achieve equilibrium between the generator loss and discriminator loss is challenging. The ideal scenario is the generator is able to generate realistic future motion that the discriminator can not distinguish. Even though we did not have enough time and resources for grid searching for the best hyper-parameters. We followed the general advice for prohibiting the discriminator collapse too early, ie. the discriminator updates less iteration then the generator update, and the learning rate for the discriminator is much smaller than the generator. In addition, we clip our discriminator parameters after each parameters update as suggested in [4] to help stabilize the training. Our finalized hypperparameter selection can be found in Table 3

For both our V-DMGNN-GAN and the baselines, we train for 40,000 steps.

Training platform. We conducted the majority of our experiments on a desktop with Intel i7-11700 CPU and NVIDIA RTX3090 GPU.

6.5 MAE Comparision of V-DMGNN-GAN with the Baselines on the ACCAD dataset

Frame	41.65 ms (↓)		241.57 ms (↓)		399.84 ms (↓)	
Masking / Action Class	Lower-body	Random	Lower-body	Random	Lower-body	Random
Female 1: general	0.149 / 0.119 /	0.661 / 0.226 /	0.335 / 0.300 /	0.544 / 0.432 /	0.438 / 0.335 /	0.481 / 0.448 /
Female 1: gestures	0.843 / 0.770 /	1.260 / 0.819 /	1.694 / 1.300 /	1.570 / 1.364 /	1.970 / 1.664 /	1.665 / 1.589 /
Female 1: running	1.452 / 1.125 /	**1.454** / 1.474 /	2.953 / 2.117 /	**2.424** / 2.799 /	**3.272** / 2.397 /	**2.690** / 2.872 /
Female 1: walking	0.567 / 0.313 /	0.629 / 0.387 /	0.707 / 0.443 /	0.660 / 0.472 /	0.874 / 0.514 /	0.778 / 0.516 /
Male 1: general	1.083 / 0.437 /	0.775 / 0.631 /	2.580 / 1.020 /	3.190 / 1.388 /	1.663 / 1.419 /	4.215 / 1.624 /
Male 1: running	1.165 / 0.955 /	**1.051** / 1.151 /	1.760 / 1.338 /	1.625 / 1.460 /	**1.299** / 1.591 /	1.510 / 1.393 /
Male 1: walking	0.338 / 0.299 /	0.514 / 0.358 /	0.519 / 0.486 /	0.562 / 0.531 /	0.460 / 0.437 /	**0.462** / 0.519 /
Male 2: general	0.240 / 0.091 /	0.383 / 0.215 /	0.338 / 0.226 /	0.372 / 0.358 /	0.446 / 0.385 /	**0.437** / 0.505 /
Male 2: martial arts, extended	**0.827** / 0.880 /	1.133 / 1.002 /	2.084 / 1.889 /	**2.001** / 2.179 /	**2.109** / 2.143 /	**2.027** / 2.057 /
Male 2: martial arts, kicks	0.634 / 0.600 /	0.910 / 0.878 /	0.970 / 0.632 /	0.979 / 0.661 /	0.941 / 0.595 /	1.077 / 0.646 /
Male 2: martial arts, punches	0.701 / 0.532 /	1.228 / 0.969 /	1.281 / 1.128 /	1.682 / 1.307 /	1.680 / 1.564 /	2.042 / 1.751 /
Male 2: martial arts, stances	**0.804** / 0.951 /	**0.711** / 0.882 /	1.743 / 1.656 /	1.947 / 1.887 /	2.423 / 2.148 /	2.677 / 2.373 /
Male 2: running	1.357 / 1.067 /	1.473 / 1.187 /	1.986 / 1.119 /	1.768 / 1.245 /	2.140 / 1.261 /	1.546 / 1.305 /
Male 2: walking	0.697 / 0.389 /	0.787 / 0.553 /	0.963 / 0.597 /	0.886 / 0.585 /	1.119 / 0.675 /	0.954 / 0.738 /
Male 2: martial arts, walks and turns	**0.533** / 0.555 /	0.964 / 0.721 /	**1.034** / 1.173 /	**0.815** / 1.581 /	**1.175** / 1.212 /	**0.921** / 1.422 /

Table 4: MAE evaluated with 'vanilla' DMGNN and our GAN implementation on the eight action classes from the ACCAD dataset [1]; Lower-body masking and random masking; **Ours** / DMGNN / HG-VAE.

Originally, we have planned to compare our proposed model against DMGNN and HG-VAE on the ACCAD dataset on MAE metric, unfortunately, HG-VAE converts inputs axis-angle representation forms to joints and further implements their experiments in Mean Per Joint Position Error (MPJPE). Due to shortage of time, we will leave the conversion as part of the future work.