Multitask Non-Autoregressive Model for Human Motion Prediction

Bin Li, Jian Tian, Zhongfei Zhang, Hailin Feng, and Xi Li*

***Abstract*—Human motion prediction, which aims at predicting future human skeletons given the past ones, is a typical sequence-to-sequence problem. Therefore, extensive efforts have been devoted to exploring different RNN-based encoder-decoder architectures. However, by generating target poses conditioned on the previously generated ones, these models are prone to bringing issues such as error accumulation problem. In this paper, we argue that such issue is mainly caused by adopting autoregressive manner. Hence, a novel Non-AuToregressive model (NAT) is proposed with a complete non-autoregressive decoding scheme, as well as a context encoder and a positional encoding module. More specifically, the context encoder embeds the given poses from temporal and spatial perspectives. The frame decoder is responsible for predicting each future pose independently. The positional encoding module injects positional signal into the model to indicate the temporal order. Besides, a multitask training paradigm is presented for both low-level human skeleton prediction and high-level human action recognition, resulting in the considerable improvement for the prediction task. Our approach is evaluated on Human3.6M and CMU-Mocap benchmarks and outperforms state-of-the-art autoregressive methods.**

***Index Terms*—Human motion prediction, non-autoregressive model, multitask learning.**

I. Introduction

As an important and challenging problem in computer vision, human motion prediction is typically formulated as a sequence modeling problem, which aims to predict a set of future human skeletons based on some existing real skeleton sequence data. Therefore, a natural solution to such a problem is to establish effective inertial motion models for capturing the temporal dependency among consecutive human skeleton frames [1]–[14].

Manuscript received February 15, 2020; revised September 14, 2020; accepted October 5, 2020. This work is supported in part by Science and Technology Innovation 2030 – "New Generation Artificial Intelligence" Major Project (No. 2018AAA0100904), Zhejiang Provincial Natural Science Foundation of China under Grant LR19F020004, key scientific technological innovation research project by Ministry of Education, Zhejiang University K. P. Chao's High Technology Development Foundation, NSFC (No. 61672456, 61702448, U19B2043), Artificial Intelligence Research Foundation of Baidu Inc., the funding from HIKVision and Horizon Robotics, and ZJU Converging Media Computing Lab. *(Corresponding author: Xi Li)*

B. Li is with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: bin_li@zju.edu.cn).

J. Tian and X. Li are with the College of Computer science and Technology, Zhejiang University, Hangzhou 310027, China (e-mail: {tianjian29, xilizju}@zju.edu.cn).

Z. Zhang is with the Computer Science Department, Binghamton University, State University of New York, Binghamton, NY 13902 (e-mail: zzhang@binghamton.edu).

H. Feng is with School of Information Engineering, Zhejiang A&F University, Hangzhou, 310027, China (email: hlfeng@zafu.edu.cn).

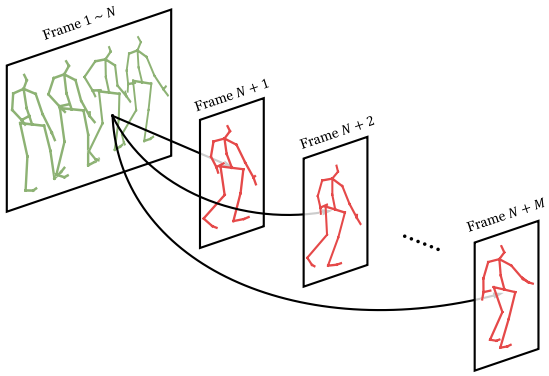


Fig. 1. Given a real human motion sequence (in green color), we independently predict each frame of future data (in red color) in a complete non-autoregressive scheme, in which the subsequent predicted frame would not be affected by the accuracy of the preceding one.

In general, these inertial motion models are based on either sequential autoregression or sequence-to-sequence encoder-decoder learning, which generates a sequence of future human skeletons in a recurrent frame-by-frame way, *i.e.*, predicting the next generated frame depending on the existing real frames as well as the current generated frame. Usually, such inertial motion models are likely to face the following two challenges: 1) Error accumulation: the prediction accuracy of the current frame relies heavily on that of previous frames, resulting in the recurrent prediction error propagation over time [15]. 2) Mean pose problem: these models often converge to an undesired mean pose in the long-term predictions, *i.e.*, the predictor gives rise to static predictions similar to the mean of the ground truth of future sequences [5]. In this paper, we mainly focus on the error accumulation problem.

Previous works on error accumulation problem could be categorized as two groups. 1) Architecture. The conventional chain-structured RNN models rely on recursive dependency, which makes the length of the information propagation path very long, typically resulting in an error accumulation problem. In contrast, the hierarchical structure of either CNNs [5] or GCNs [9], [14] contains skip connections, and encourages dense connections among data units, which ensures the diversity of the information flow patterns with different lengths. 2) Function Approximation. Through periodic functions with different frequencies (*e.g.*, the DCT consine function basis [9]), the conventional methods are able to construct a holistic representation of skeleton sequence from the top-down view within a single run.

In this paper, to solve the error accumulation problem, we argue that the autoregressive decoding pipeline for human motion prediction over the subsequent frames often suffers

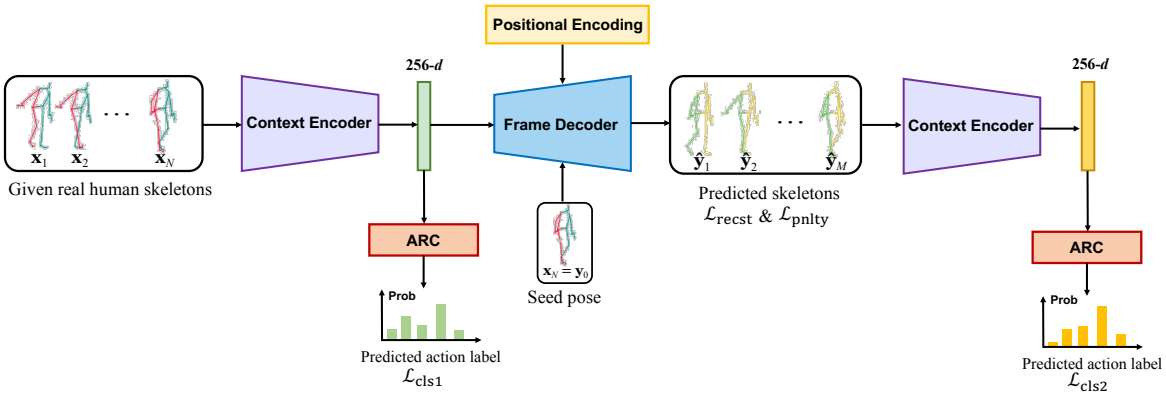


Fig. 2. **Overview of Multitask Non-AuToregressive Model (mNAT).** A real human skeleton sequence is first sent to the context encoder, which is composed of multiple GCN-TCN blocks with residual connection, to obtain the 256-*d* context feature (in green color). This feature, together with the seed pose and positional encoding vectors, is further sent to the frame decoder to generate each future frame independently. The frame decoder owns the same structure with the context encoder except that the kernel size is 1 in frame decoder while 9 in context encoder. Both the features of the given real skeletons and the features of the predicted skeletons are sent to the same action recognition classifier (ARC) to predict the action category. Note that the two context encoders (in purple color) are the same one and so are the two classifiers (in red color).

from the misguidance from the preceding prediction results and the evaluation criteria. Besides, the continuity and diversity of generated frames, are not guaranteed as well. A natural question is whether we have a more feasible decoding pipeline, *e.g.*, breaking the temporal dependency, that allows the subsequent frames to bypass the preceding ones directly during decoding.

To this end, we propose a novel Non-AuToregressive framework (NAT), which largely eases the aforementioned issue. Specifically, NAT is composed of a context encoder (embedding the given poses), a frame decoder (predicting each future skeleton independently), and a positional encoding module (indicating temporal order). *Context encoder* is modeled from both temporal and spatial perspectives by a TCN-based (Temporal Convolutional Network) [16] temporal encoder and a GCN-based (Graph Convolutional Network) [17] spatial encoder through a skeleton kinematic tree, respectively. In principle, it encodes the existing real skeleton sequence data into a context feature space. In addition, *frame decoder* sets up a prediction model to forecast each generated frame based on its corresponding direct connections to the existing real frames, as shown in Fig. 1. Since temporal dependency is broken under our non-autoregressive setting, inspired by recent success from natural language processing [18], we also propose a *positional encoding module* which outputs a combination of sinusoidal waves with different frequencies as the representation of position. This representation can be viewed as a trajectory with physical constraints in the code space, and the frame decoder generates frames with the fusion of the representation and the context feature. The physical constraints, *i.e.*, sinusoidal waves with different frequencies, guarantee the continuity and diversity. Therefore, the quality of the generated frames improves according to the evaluation criteria. Meanwhile, such a non-autoregressive setting enables parallel processing of multiple frames during decoding instead of sequential frame decoding used by conventional recurrent approaches.

In addition, despite the above success, previous works rarely investigated the relation between the low-level human skeletons and the high-level human action category. The motivation behind this is that unlike deep models, human beings always generate intention first and then implement it. If the model knows which action is to be generated, the forecasting would be much easier. For example, "smoking" and "phoning" may have very similar beginning. It is unlikely to forecast the following frames without knowing the exact action category. To this end, inspired by recent success on skeleton-based action recognition [19]–[23], we propose a simple yet effective multitask training paradigm, namely Multitask Non-AuToregressive model (mNAT), which is further empowered by the merit of action recognition. Specifically, as shown in Fig. 2, we build a shared action recognition classifier (ARC) for both given real human skeletons and predicted ones, ensuring that our model is capable of predicting both the low-level and high-level future information. The experimental results show that the human motion prediction task achieves an obvious promotion for this multitasking scheme.

In summary, the main contributions of this work are summarized as follows

1) We propose to solve the human motion prediction task with a novel Non-AuToregressive model (NAT), which largely alleviates the error accumulation problem.
2) We present a multitask training paradigm which is empowered by the merit of action recognition to predict both the low-level human skeletons and high-level human action category.
3) Extensive experiments on both Human3.6M [24] and CMU-Mocap[1] benchmarks yield state-of-the-art results.

II. Related Work

A. Human Motion Prediction

Recently, human-centric computer vision understanding has attracted an enormous amount of attention. Aiming at human-centric tasks, great efforts have been paid from the perspective of learning mechanism. 1) Transfer learning. To capture multi-level information of source and target domain data, Zhou *et*

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

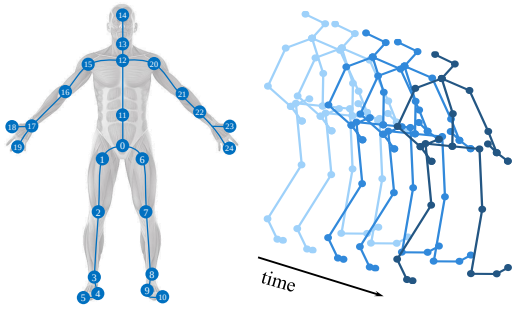
al. [25] presented a multi-mutual consistency learning strategy to reduce the distribution difference between two domains. 2) Cascaded learning. Zhou *et al.* [26] proposed to mine the intrinsic complexity of HOI by building up a cascade architecture for a multi-stage, coarse-to-fine HOI understanding. 3) Graph reasoning. GPNN [27] recognized HOI sceen graph by inferring and reasoning a parse graph iteratively. GRN [28] proposed to explore the topology structure of human body to help generate more convincing pseudo labels for human parsing. Although employing various learning approaches, these works have a common goal of exploring better representation of human structure.

Among these, Human motion prediction is a classical and challenging problem which has long been studied over years. Holden *et al*. [1] first showed that human motion can be formulated as manifold via auto-encoders. Due to the temporal nature of human motion prediction problem, much of current state-of-the-art work is based on RNN-based encoder-decoder structure. Fragkiadaki *et al.* [2] proposed Encoder-Recurrent-Decoder (ERD), which, for the first time, maps the pose into hidden state and propagates through an LSTM layer. Also, Structural-RNN [3] was presented to formulate human motion sequence as spatio-temporal graphs. Martinez *et al.* [4] designed a residual-based GRU (Res-GRU) model by predicting the relative residual between two consecutive frames instead of absolute skeleton. This residual modeling works so well that it becomes the *de facto* standard for the subsequent work.

Since then, human motion prediction has been roughly categorized into two groups. one group of approaches tend to seek better representations of human skeleton. This group of models explore either the spatial dependency or the different representation of each joint, *e.g.*, exponential map, Euler angle. Li *et al.* [5] introduced a convolutional neural networks to model spatial and temporal dependency via a rectangle receptive field. Guo *et al.* [6] designed SkelNet which divides a human skeleton into five non-overlapping parts. To evaluate the effectiveness of different joint representation, QuaterNet [10], [11] was introduced to replace the commonly used exponential map representation by quaternion, which avoids common rotational problems such as non-uniqueness, discontinuity, and gimbal locks [29]. Recently, Discrete Cosine Transform (DCT) [9] was proposed to encode temporal information via a series of DCT coefficients and achieved the state-of-the-art result so far. The second group of methods explore better measurements between ground truth and predicted skeletons. Gui *et al.* [12] presented Adversarial Geometry-Aware encoder-decoder (AGED), which replaced commonly used $L1$ distance by a geodesic loss to better model the motion. Also, Hernandez *et al.* [13] presented Spatio-Temporal Motion Inpainting (STMI-GAN) which formulates human motion forecasting as an image inpainting problem and further solved it with an improved GAN structure.

B. Non-autoregressive Models

RNN-based models, such as LSTM [30] or GRU [31], achieved great success in sequence modeling, especially in Neural Machine Translation (NMT) [32]–[34]. In general, these methods generated tokens in a sequential manner, *i.e.*, the new output word was dependent on the previously generated output. Such left-to-right decoding manner suffered from problems like low efficiency and error accumulation [35], [36]. To remedy these issues, several efforts were devoted to avoiding recurrence in sequence modeling. Gehring *et al.* [37] proposed a sequence model based entirely on convolutional neural networks. Vaswani *et al.* [18] proposed the Transformer network, which stacks multiple self-attention layers to model the dependency on each token pairs. Recently, Gu *et al.* [38] proposed non-autoregressive transformer that makes use of fertilities which represents how many times each source tokens are copied.



(a) (b)

Fig. 3. (a) Illustration of human skeleton of Human3.6M dataset. The blue circles indicate the detailed joint indexes. (b) Illustration of human motion sequence.

As a sequence modeling task, human motion prediction shares similar nature with NMT and meanwhile possesses unique characteristics. 1) Human motion prediction is usually formulated as a continuous prediction task rather than a discrete one. Hence, several useful tricks like Beam Search [34], [39] in NLP could not be directly adopted. 2) Human motion sequence contains rich skeletal structure information, which natural language token rarely owns. Therefore, in this paper, we propose a GCN equipped context encoder to learn the rich skeletal structure as well as a frame decoder to implement non-autoregressive decoding.

C. Graph Neural Networks

Graph Neural Networks (GNNs) is proposed for data whose structure is defined by a graph with either spectral [40], [41] method or spatial method [17]. The previous success of GNNs could be categorized from the perspective of granularity of graph structure. 1) Intra-frame. Li *et al.* [28] utilized the topology structure of human body, *i.e.*, human parts, to refine the human parsing results. Also, GPNN [27] iteratively inferred a Human-Object Interaction graph with a message passing inference framework. 2) Inter-frame. By treating each individual frame as a node, AGNN [42] built a fully connected graph to efficiently capture and mined much richer relations between frames for zero-shot video object segmentation. PMPNet [43] adaptively encoded point cloud information by graph-based message passing mechanism with a relatively huge receptive field. To summarize, the GNNs are suitable for modeling a wide variety of spatio-temporal topology structures.

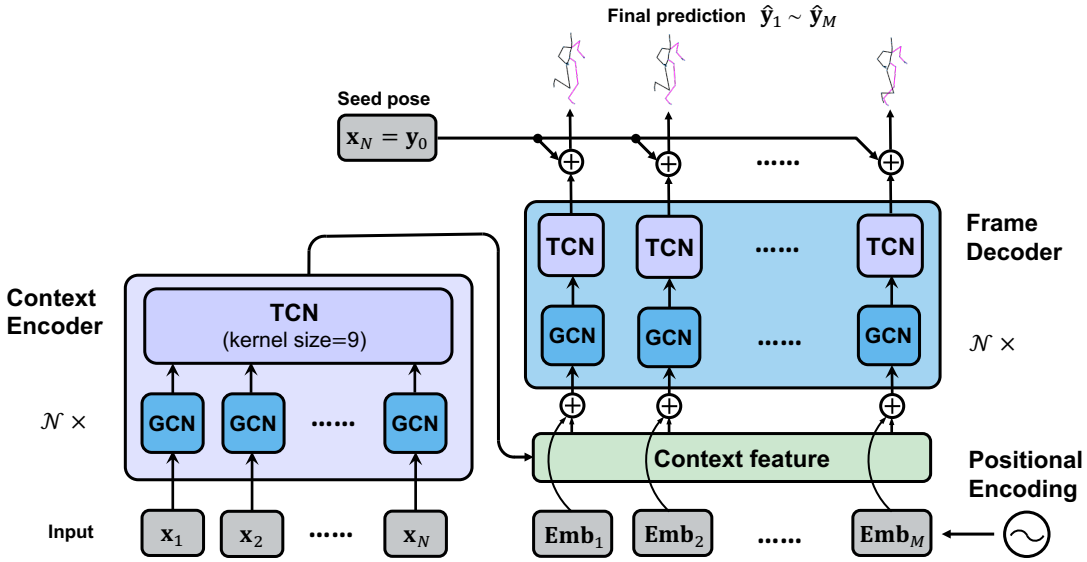


Fig. 4. **Illustration of NAT pipeline.** Given the prefix human motion sequence $\mathbf{x}_1 \sim \mathbf{x}_N$, context encoder stacks GCN module and TCN module multiple times to encode it as context feature. The context feature is further added by a series of sinusoidal positional-related signal generated by positional encoding module. The features are then sent to frame decoder to generate final predicted human motion sequence $\hat{\mathbf{y}}_1 \sim \hat{\mathbf{y}}_M$ in a non-autoregressive scheme. "$\mathcal{N}\times$" means that $\mathcal{N}$ "GCN-TCN" modules are stacked in the context encoder and the frame decoder, respectively. Note that the illustration of context encoder denotes feature flow, *i.e.*, multiple features of different time steps are passed into the TCN module.

III. Problem Formulation

As shown in Fig. 3 (a), the human motion skeleton is usually represented as a skeletal kinematic tree. A kinematic tree is composed of one root joint and several other joints as child nodes. Each child node possesses only one parent joint, forming a tree structure. Hence, the human motion sequence is constructed by stacking multiple human skeletons through time horizon, as shown in Fig. 3 (b).

Specifically, we consider to be given a length-N observed sequence $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N) \in \mathbb{R}^{N\times J\times K}$, where each of the frames $\mathbf{x}_n = \{\mathbf{x}_n^j\}_{j=1}^J$ represents the single skeleton, containing J joints data. $\mathbf{x}_n^j \in \mathbb{R}^K$ is a minimal per-joint representation at the n-th frame and the j-th joint, of which K is the feature dimension which represents human joint data. In this paper, we adopt $K = 4$ for quaternion as this format is free of discontinuity and singularity [10]. Our goal is to predict consecutive length-M target sequence $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_M)$. Note that decoding always starts from the last frame of the given sequence $\mathbf{X}$. For simplicity, we name it as "seed pose": $\mathbf{x}_{\text{seed}} = \mathbf{x}_N = \mathbf{y}_0$.

In the following, we first explain the reason why autoregressive model leads to error accumulation problem. We then discuss each part of our NAT model in detail. In particular, we introduce context encoder, positional encoding, and frame decoder in Sec IV-A, IV-B, IV-C, respectively. In Sec V, we introduce the multitask training pipeline for our NAT model.

Usually, the human motion is typically viewed as an inertial model, where only small changes happen in two consecutive frames. Therefore, RNNs are adopted to model this temporal continuity in an autoregressive paradigm. In this paper, we argue that this inertance still exists in short term (less than one second). Hence, instead of predicting the target skeleton in a frame-by-frame manner, we directly regress the residual item between each target pose and seed pose.

In detail, assume that the conventional RNN-based encoder-decoder framework acts as

$$\hat{\mathbf{y}}_t = \hat{\mathbf{y}}_{t-1} + \mathbb{D}\big(\{\hat{\mathbf{y}}_i\}_{i=1}^{t-1}, \mathbb{E}(\mathbf{X})\big), \tag{1}$$

where $\mathbb{E}(\cdot)$ denotes the encoder function, $\mathbb{D}(\cdot)$ is the decoder function, $\hat{\mathbf{y}}_t$ is the predicted pose at time t, and $\{\hat{\mathbf{y}}_i\}_{i=1}^{t-1}$ denotes a set of skeletons ranging from 1 to $t-1$.

Note that the above equation defines the recursive formula between two consecutive frames. We then expand this equation by

$$\begin{aligned}\hat{\mathbf{y}}_t &= \hat{\mathbf{y}}_{t-1} + \mathbb{D}\big(\{\hat{\mathbf{y}}_i\}_{i=1}^{t-1}, \mathbb{E}(\mathbf{X})\big) \\ &= \hat{\mathbf{y}}_{t-2} + \mathbb{D}\big(\{\hat{\mathbf{y}}_i\}_{i=1}^{t-2}, \mathbb{E}(\mathbf{X})\big) + \mathbb{D}\big(\{\hat{\mathbf{y}}_i\}_{i=1}^{t-1}, \mathbb{E}(\mathbf{X})\big) \\ &= \mathbf{y}_0 + \sum_{j=0}^{t-1} \mathbb{D}\big(\{\hat{\mathbf{y}}_i\}_{i=1}^{j}, \mathbb{E}(\mathbf{X})\big),\end{aligned} \tag{2}$$

where $\mathbf{y}_0$ is the seed pose, which is the last frame of the given motion sequence as we discuss above. For consistence of equation form, we assume $\{\hat{\mathbf{y}}_i\}_{i=1}^{0} = \mathbf{y}_0$ when we calculate $\hat{\mathbf{y}}_1 = \mathbf{y}_0 + \mathbb{D}\big(\mathbf{y}_0, \mathbb{E}(\mathbf{X})\big)$.

We argue that the aforementioned error accumulation problem could be derived from Eq. 2. First of all, Eq. 2 constructs the relation between the target pose $\hat{\mathbf{y}}_t$ and seed pose $\mathbf{y}_0$. The residual item between $\hat{\mathbf{y}}_t$ and $\mathbf{y}_0$ is a sum of multiple predictions. Suppose an initial error item δ_1 happens between $\hat{\mathbf{y}}_1$ and $\mathbf{y}_1$. Then we have $\hat{\mathbf{y}}_2 = \mathbf{y}_1 + \delta_1 + \mathbb{D}\big(\mathbf{y}_1 + \delta_1, \mathbb{E}(\mathbf{X})\big)$, which means the initial error term δ_1 propagates to δ_2. Therefore, the initial error term δ_1 would spread to each time item up to δ_t repeatedly. Hence, the error accumulation takes

places when the sum item increases rapidly with the evolution of the time t. This per-frame prediction error grows so fast that the long-term prediction soon becomes implausible to use.

Inspired by this observation, we model the residual term between $\hat{\mathbf{y}}_t$ and $\mathbf{y}_0$ directly, rather than modeling it in an autoregressive way. Specifically, $\hat{\mathbf{y}}_t$ no longer depends on the accuracy of previous generated poses in our design. We directly obtain each target pose based on the last available ground truth skeleton $\mathbf{y}_0$ only, which largely alleviates the error accumulation problem.

IV. Non-autoregressive Model

In this section, we describe the details of our NAT structure. Unlike complicatedly designed models in natural language processing, we find out that non-autoregressive architecture could be simply implemented by three components: a context encoder $\mathbb{E}(\cdot)$, a frame decoder $\mathbb{D}(\cdot)$, and a positional encoding module, as shown in Fig. 4. We introduce each part in detail below.

A. Context Encoder

Usually, a human motion sequence is represented as a spatio-temporal graph. Therefore, the encoder needs to simultaneously model both the joint-wise dependency in the spatial domain and frame-wise dependency in the temporal domain. To this end, we propose to stack multiple GCN-TCN blocks to form the context encoder which is capable of generating context feature being representative of the whole given sequence.

To encode the spatial dependency of human skeletons, we make use of GCNs [17]. GCNs are a class of models which are specially designed for non-Euclidean data. To make our paper self contained, we briefly introduce how GCNs work here. As mentioned, each frame of human skeleton sequence contains J joints. The bone connection is thus formulated as an adjacency matrix $\mathbf{A} \in \mathbb{R}^{J\times J}$, where $\mathbf{A}_{ij} = 1$ if and only if joint i connects with joint j (each joint connects with itself). Assume that in layer l, we have input feature as $\mathbf{h}^{(l)} \in \mathbb{R}^{J\times K}$, where K denotes input dimension. Following Kipf *et al.* [17], we adopt first order approximation and output $\mathbf{h}^{(l+1)}$ as

$$\mathbf{h}^{(l+1)} = \sigma\Big(\mathrm{BN}\big(\tilde{\mathbf{A}} \cdot \mathbf{h}^{(l)} \cdot \mathbf{W}^{(l)}\big)\Big), \tag{3}$$

where $\tilde{\mathbf{A}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ is the normalized adjacency matrix and $\mathbf{D}_{ii} = \sum_j \mathbf{A}_{ij}$ is the corresponding degree matrix, $\mathbf{W}^{(l)} \in \mathbb{R}^{K\times K}$ is a trainable weight matrix, $\mathrm{BN}(\cdot)$ denotes Batch Normalization [44], and $\sigma(\cdot)$ is the Leaky ReLU [45].

To encode the temporal dependency, we make use of TCN [16] where RNNs are replaced by 1D CNNs. To capture the long-term dependency, multiple CNN layers are stacked to increase the receptive field. In the TCN model, the receptive field drastically grows in a linear speed.

In practice, we find a large kernel size is necessary to ensure that the final feature covers the whole input sequence. However, once reaching an appropriate receptive field, no benefit is found by using a larger kernel size. We conduct experiments in Sec VI to verify the effects of different kernel sizes of TCN.

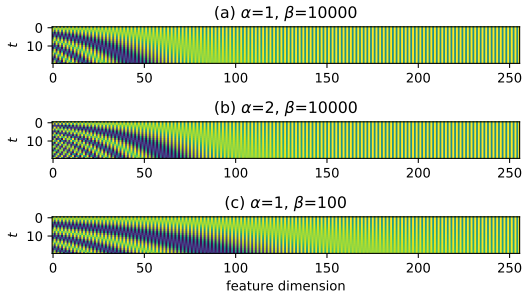


Fig. 5. Illustration of the Positional Embedding. This embedding is generated by sine and cosine functions of different frequencies. We also explore the influence of hyper-parameter α and β.

We also add skip-connection [46] between each two blocks as it makes easier to propagate the gradients and accelerates training. The number of channels is 64, 64, 128, 128, 256, 256 in total 6 blocks, respectively, which map the input 4-d quaternion to a 256-d feature. In the end, we perform global average pooling in both temporal dimension and spatial dimension to obtain a single context feature $\mathbf{c} = \mathbb{E}(\mathbf{X}) \in \mathbb{R}^{256}$.

B. Positional Encoding Module

Compared with the autoregressive models, which implicitly encode temporal order in a frame-by-frame way, the non-autoregressive model faces a problem on representing time. To this end, we propose to inject the explicit temporal signal into decoder directly. We further rewrite Eq. 1 as follows

$$\hat{\mathbf{y}}_t = \mathbf{y}_0 + \mathbb{D}\big(\mathbf{p}(t), \mathbb{E}(\mathbf{X})\big), \tag{4}$$

where t represents the time index and $\mathbf{p}(\cdot)$ is a function that maps input scalar index into a vector form embedding. In our case, we call this mapping function $\mathbf{p}(\cdot)$ as positional encoding module.

Following the former success of Transformer [18], we adopt the sinusoidal functions of different frequencies as our positional encoding. In particular, we aim to map relative time index into a vector form feature. For time index $t = 1, ..., M$, the positional encoding function is expressed as

$$\begin{aligned} p_{2i}(t) &= \sin\left(\alpha \cdot t/\beta^{2i/d_{\mathrm{model}}}\right) \\ p_{2i+1}(t) &= \cos\left(\alpha \cdot t/\beta^{2i/d_{\mathrm{model}}}\right), \end{aligned} \tag{5}$$

where $p_{2i}(t)$, $p_{2i+1}(t)$ represent the even and odd dimension of $\mathbf{p}(t)$, d_{model} denotes the positional embedding dimension, α is a scale factor which controls the difference across time indexes, β controls the wavelength for each dimension, and i is the dimension index ranging from 1 to $\lfloor d_{\mathrm{model}}/2 \rfloor$.

The benefit of sinusoidal positional encoding module is two-fold. Firstly, compared with trivial one-hot encoding, this encoding outputs a continuous form vector, carrying more information. Secondly, the sinusoidal positional embedding is highly correlated, *i.e.*, the closer two time indexes t_1, t_2 are, the more similar $\mathbf{p}(t_1)$, $\mathbf{p}(t_2)$ are. Consequently, the positional embedding could be seen as a disturbance item added to

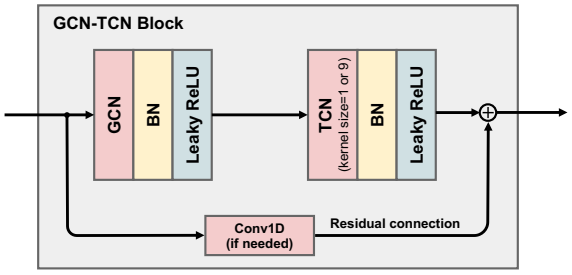


Fig. 6. Detailed structure of the GCN-TCN Block. Both GCN and TCN operation are followed by a BN and Leaky ReLU. Note that the below Conv1D only appears when input channel is not equal to output channel.

the encoded feature, ensuring the smoothness of generated sequence.

Note that in the original paper, the α and β are set to fixed values ($\alpha = 1$, $\beta = 10000$). We argue that due to the domain difference, *i.e.*, the predicted length is usually 10 or 25 in human motion prediction while hundreds of tokens might be involved in neural machine translation, the default setting could be sub-optimal in this task. As shown in Fig. 5, with the growth of α, the difference between each embedding increases (see (a), (b)). With the decrease of β, more dimensions are involved to distinguish different embeddings. We conduct extensive experiments on exploring the influence of α and β.

C. Frame Decoder

As mentioned above, the frame decoder is responsible for generating each frame independently. To generate M future frames, we first obtain a series of positional embeddings $\mathbf{P} = \{\mathbf{p}(1), ..., \mathbf{p}(M)\}$, each of which is also a 256-d vector. These embeddings are further added to the context feature $\mathbf{c}$ to form the input of the frame decoder $\mathbf{F} = \{\mathbf{f}_t\}_{t=1}^{M}$, where each $\mathbf{f}_t = \mathbf{c} + \mathbf{p}(t)$.

As shown in Fig. 6, in each GCN-TCN block, GCN module and TCN module are adopted sequentially. Besides, Batch Normalization [44], Leaky ReLU activation [45], and skip-connection [46] are also introduced. The GCN-TCN blocks have a kernel size of 9 in the context encoder and a kernel size of 1 in the frame decoder. Under this setting, we rewrite Eq. 4 as: $\hat{\mathbf{y}}_t = \mathbf{y}_0 + \mathbb{D}(\mathbf{f}_t)$, where each frame generation process is strictly limited to the single frame, avoiding to be affected by subsequent predicted frames, also shown in Fig. 4. Similar to context encoder, the number of channels is 256, 128, 128, 64, 64, 4 with total 6 blocks, respectively, which map the input 256-d feature back to 4-d quaternion.

V. Multitask Training

A. Action Recognition Classifier

As another classic task in skeleton-based activity understanding, skeleton-based action recognition also attracts lots of attention recently. However, few work has explored the relation of these two tasks. Martinez *et al.* [4] proposed Res-GRU MA (Multi-Action) by simply concatenating one-hot vectors with 15 action classes of Human3.6M dataset. The result shows limited performance gain. Therefore, most of the subsequent works report the SA (Single-Action) result without the action category information.

Algorithm 1 NAT Multitask Training

Input: Training set $\mathcal{D} = \{(\mathbf{X}^i, \mathbf{Y}^i, \mathbf{y}_{\text{cls}}^i)\}_{i=1}^{\#sample}$ with C classes; Training iterations τ.
Output: The parameters θ of NAT and ARC

for iteration = $1, ..., \tau$ **do**
  Randomly sample $(\mathbf{X}^i, \mathbf{Y}^i, \mathbf{y}_{\text{cls}}^i)$ from $\mathcal{D}$
  Compute context feature $\mathbf{c}$ with context encoder
  Obtain the predicted sequence $\hat{\mathbf{Y}}^i$ using Eq. 4
  Compute $\mathbf{o}_1$ from $\mathbf{c}$ with ARC module
  Compute $\mathbf{c}'$ by feeding $\hat{\mathbf{Y}}^i$ back into context encoder
  Compute $\mathbf{o}_2$ from $\mathbf{c}'$ wit ARC module
  Calculate $\mathcal{L}_{\text{recst}}$, $\mathcal{L}_{\text{pnlty}}$, $\mathcal{L}_{\text{cls1}}$, and $\mathcal{L}_{\text{cls2}}$ using Eq. 6 $\sim$ 9
  Update the parameters θ using Eq. 10 by ADAM [47]
end for
return The parameters θ of NAT and ARC

TABLE I
COMPARISON OF MEAN JOINT ERROR OF ANGLE SPACE ON AVERAGE OF ALL 15 ACTIONS OF HUMAN3.6M DATASET

millseconds	Average 80	160	320	400	Conference
Zero-velocity [4]	0.42	0.74	1.12	1.20	CVPR2017
Res-GRU [4]	0.39	0.72	1.08	1.22	CVPR2017
ConvSeq2Seq [5]	0.38	0.68	1.01	1.13	CVPR2018
QuaterNet [10]	0.35	0.64	1.07	1.23	BMVC2018
AGED w/ adv [12]	0.33	0.58	0.94	1.01	ECCV2018
SkelNet [6]	0.36	0.64	0.99	1.02	AAAI2019
TD-DCT [9]	**0.27**	0.51	0.83	0.95	ICCV2019
NAT (Ours)	**0.27**	0.50	0.79	0.91	-
mNAT (Ours)	**0.27**	**0.48**	**0.74**	**0.85**	-

In this paper, we argue that, as human beings, the high-level human action category guides the low-level human skeletons and the existing literatures rarely investigate these two related tasks. To this end, we propose an action recognition classifier (ARC) on the top of the context feature $\mathbf{c}$, as shown in Fig. 2. The ARC module is implemented with a three-layer MLP, where FC, Dropout, and LeakyReLU are included. Let the ground truth label be $\mathbf{y}_{\text{cls}} = \{0, 1\}^C$, where C denotes the number of actions in total, and the corresponding classification result be $\mathbf{o}_1 = \text{Softmax}(\text{ARC}(\mathbf{c}))$. The classification loss is formulated as

$$\mathcal{L}_{\text{cls1}} = -\mathbf{y}_{\text{cls}}^{\top} \log(\mathbf{o}_1). \tag{6}$$

Inspired by self-supervised learning [48], [49], we also add a cycle consistency classification loss. We expect that the predicted human motion sequence $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_t\}_{t=1}^{M}$ not only is close to the ground truth sequence, but also represents the high-level action category. Therefore, $\hat{\mathbf{Y}}$ is sent to the same context encoder and ARC to obtain the classification result $\mathbf{o}_2$. The cycle consistency classification loss is

$$\mathcal{L}_{\text{cls2}} = -\mathbf{y}_{\text{cls}}^{\top} \log(\mathbf{o}_2). \tag{7}$$

TABLE II
COMPARISON OF MEAN JOINT ERROR OF ANGLE SPACE BETWEEN OUR MODEL AND THE STATE-OF-THE-ART METHODS ON ALL 15 ACTIONS OF HUMAN3.6M DATASET

	Walking				**Eating**				**Smoking**				**Discussion**				**Direction**			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Zero-velocity [4]	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	0.39	0.59	0.79	0.89
Res-GRU [4]	0.27	0.47	0.70	0.78	0.25	0.43	0.71	0.87	0.33	0.61	1.04	1.19	0.31	0.69	1.03	1.12	0.26	0.47	0.72	0.84
ConvSeq2Seq [5]	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	0.39	0.60	0.80	0.91
AGED w/ adv [12]	0.22	0.36	0.55	0.67	0.17	0.28	0.51	0.64	0.27	0.43	0.82	0.84	0.27	0.56	0.76	0.83	**0.23**	**0.39**	0.63	0.69
SkelNet [6]	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	0.36	0.58	0.77	0.86
TD-DCT [9]	0.18	0.31	0.49	0.56	**0.16**	0.29	0.50	0.62	**0.22**	0.41	0.86	0.80	**0.20**	**0.51**	0.77	0.85	0.26	0.45	0.71	0.79
NAT (Ours)	0.19	**0.28**	**0.45**	**0.51**	**0.16**	**0.25**	**0.44**	0.58	0.23	0.42	0.82	0.84	0.22	0.54	0.79	0.89	0.26	0.42	0.62	0.72
mNAT (Ours)	**0.17**	0.29	**0.45**	0.53	0.17	0.31	0.48	**0.54**	**0.22**	**0.40**	**0.81**	**0.78**	0.23	0.54	**0.72**	**0.80**	0.27	0.43	**0.58**	**0.67**
	Greeting				**Phoning**				**Posing**				**Purchases**				**Sitting**			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Zero-velocity [4]	0.54	0.89	1.30	1.49	0.64	1.21	1.65	1.83	0.28	0.57	1.13	1.37	0.62	0.88	1.19	1.27	0.40	1.63	1.02	1.18
Res-GRU [4]	0.75	1.17	1.74	1.83	0.23	0.43	0.69	0.82	0.36	0.71	1.22	1.48	0.51	0.97	1.07	1.16	0.41	1.05	1.49	1.63
ConvSeq2Seq [5]	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
AGED w/ adv [12]	0.56	0.81	1.30	1.46	**0.19**	**0.34**	**0.50**	**0.68**	0.31	0.58	1.12	1.34	0.46	0.78	1.01	1.07	0.41	0.76	1.05	1.19
SkelNet [6]	0.50	0.84	1.28	1.45	0.58	1.12	1.52	1.64	0.29	0.62	1.19	1.44	0.58	0.84	1.17	1.24	0.40	0.61	1.01	1.15
TD-DCT [9]	0.36	0.60	0.95	1.13	0.53	1.02	1.35	1.48	0.19	0.44	1.01	1.24	0.43	0.65	1.05	1.13	**0.29**	**0.45**	**0.80**	**0.97**
NAT (Ours)	0.36	0.59	0.93	1.08	0.55	0.96	1.28	1.42	**0.18**	0.43	0.93	1.16	0.46	0.67	0.96	1.03	**0.29**	0.46	**0.80**	0.98
mNAT (Ours)	**0.33**	**0.51**	**0.79**	**0.94**	0.53	0.92	1.15	1.28	**0.18**	**0.38**	**0.81**	**1.00**	**0.40**	**0.55**	**0.85**	**0.89**	**0.29**	0.46	0.84	1.04
	Sitting Down				**Taking Photo**				**Waiting**				**Walking Dog**				**Walking Together**			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Zero-velocity [4]	0.39	0.74	1.07	1.19	0.25	0.51	0.79	0.92	0.34	0.67	1.22	1.47	0.60	0.98	1.36	1.50	0.33	0.66	0.94	0.99
Res-GRU [4]	0.39	0.81	1.40	1.62	0.24	0.51	0.90	1.05	0.28	0.53	1.02	1.14	0.56	0.91	1.26	1.40	0.31	0.58	0.87	0.91
ConvSeq2Seq [5]	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
AGED w/ adv [12]	0.33	0.62	0.98	1.10	0.23	0.48	0.81	0.95	0.24	0.50	1.02	1.13	0.50	0.81	1.15	1.27	0.23	0.41	0.56	0.62
SkelNet [6]	0.37	0.72	1.05	1.17	0.24	0.47	0.78	0.93	0.30	0.63	1.17	1.40	0.54	0.88	1.20	1.35	0.27	0.53	0.68	0.74
TD-DCT [9]	**0.30**	**0.61**	**0.90**	**1.00**	**0.14**	0.34	0.58	0.70	0.23	0.50	0.91	1.14	0.46	0.79	1.12	1.29	**0.15**	0.34	0.52	0.57
NAT (Ours)	0.31	0.63	0.92	1.05	0.17	0.37	0.59	0.71	0.23	**0.48**	0.87	1.07	**0.40**	**0.69**	1.00	1.14	**0.15**	0.31	**0.45**	**0.51**
mNAT (Ours)	0.34	0.66	0.94	1.06	**0.14**	**0.32**	**0.54**	**0.68**	**0.22**	**0.48**	**0.79**	**0.97**	0.43	0.70	**0.86**	**1.03**	0.16	**0.28**	0.51	0.62

B. Training

We now summarize the whole training process. Given the predicted human motion sequence $\hat{\mathbf{Y}} = \{\hat{\mathbf{y}}_t\}_{t=1}^{M}$ and ground truth, we apply the average $L1$ distance as the reconstruction loss

$$\mathcal{L}_{\text{recst}} = \frac{1}{J \times M} \sum_{j=1}^{J} \sum_{t=1}^{M} \|\hat{\mathbf{y}}_t^j - \mathbf{y}_t^j\|_1, \tag{8}$$

where $\hat{\mathbf{y}}_t^j$ denotes the predicted skeleton of the j-th joint in the t-th frame, and $\mathbf{y}_t^j$ is the corresponding ground truth.

Since that we use quaternion as the joint representation, we must ensure the output $\hat{\mathbf{y}}_t$ has unit length as only unit quaternion represents a valid 3D rotation [50], [51]. To this end, we also add a penalty loss for each of the prediction to ensure this property

$$\mathcal{L}_{\text{pnlty}} = \frac{1}{J \times M} \sum_{j=1}^{J} \sum_{t=1}^{M} \left(\|\hat{\mathbf{y}}_t^j\|_2^2 - 1\right)^2. \tag{9}$$

To summarize, our overall objective is

$$\mathcal{L} = \mathcal{L}_{\text{recst}} + \lambda_{\text{pnlty}} \mathcal{L}_{\text{pnlty}} + \lambda_{\text{cls}} (\mathcal{L}_{\text{cls1}} + \mathcal{L}_{\text{cls2}}), \tag{10}$$

where λ_{pnlty} and λ_{cls} control the relative importance of each loss item. The pseudo code of NAT Multitask Training is shown in Algorithm 1.

VI. EXPERIMENTS

In this section, we first introduce two popular motion capture benchmarks: Human3.6M [24] and CMU Motion Capture dataset [52] (CMU-Mocap) as well as the implementation details and evaluation metrics. We then demonstrate our results compared with the current state-of-the-arts and ablation studies.

A. Datasets

Human3.6M [24] is the largest publicly available dataset for human motion research so far, which contains 3.6 million 3D poses recorded by Vicon motion capture system. It contains 15 activity scenarios including walking, eating, smoking, and discussion. Seven subjects are involved in the dataset, each of which performs two sequences for each action. In total, each sequence contains about 3000 to 5000 frames. Each frame consists of 34 rows of data, including a global translation, a global rotation and 32 joint rotations with respect to its parent joint. Each joint is represented as an exponential map (axis-angle) form. Following the standard protocol [4], [5], [10], all sequences are downsampled to a frame rate of 25fps; global translation and global rotation are discarded. The Subject 5 (S5) is used in testing while the others are used in training.

CMU Motion Capture [52] is a large dataset including actions such as walking, running, dancing. Different from Human3.6M, the CMU-Mocap dataset has 38 joints in total. Therefore, it has a different skeleton configuration. Li *et al.* [5] first conduct experiments on CMU-Mocap with selected eight actions. We follow their experiment setting with 86293 frames in total. Five subjects are used for training while one subject is used for testing. Similar to Human3.6M, all sequences are also downsampled to 25fps. Global translation and global rotation are discarded.

B. Implementation Details

1) Network and Training Details: Our model is based on QuaterNet [10] and quaternion is used as input representation of joints. Similar to the previous works [4], [5], [10], our model is also trained on all actions. Both context encoder and frame decoder stack 6 GCN-TCN building blocks with residual connections, which indicates $\mathcal{N}$ is set to 6 in the

TABLE III
COMPARISON OF MEAN JOINT ERROR OF ANGLE SPACE BETWEEN OUR MODEL AND THE STATE-OF-THE-ART METHODS ON ALL 15 ACTIONS IN LONG-TERM PREDICTION OF HUMAN3.6M DATASET.

	Walking		**Eating**		**Smoking**		**Discussion**		**Direction**		**Greeting**		**Phoning**		**Posing**	
milliseconds	560	1000	560	1000	560	1000	560	1000	560	1000	560	1000	560	1000	560	1000
Zero-velocity [4]	1.35	1.32	1.04	1.38	1.02	1.69	1.41	1.96	1.02	1.50	1.79	1.80	1.81	2.04	1.81	2.78
ERD [2]	2.00	2.38	2.36	2.41	3.68	3.82	3.47	2.92	-	-	-	-	-	-	-	-
SRNN [3]	1.81	2.20	2.49	2.82	3.24	2.42	2.48	2.93	-	-	-	-	-	-	-	-
Res-GRU [4]	0.93	1.03	0.95	1.08	1.25	1.50	1.43	1.69	0.96	1.42	1.68	1.76	1.56	1.77	1.78	2.29
ConvSeq2Seq [5]	0.86	0.92	0.89	1.24	0.97	1.62	1.44	1.86	0.93	1.42	1.57	1.79	1.66	1.83	1.75	2.78
AGED w adv [12]	0.78	0.91	0.86	0.93	1.06	**1.21**	1.25	1.30	-	-	-	-	-	-	-	-
SkelNet [6]	0.79	0.83	**0.84**	1.06	**0.98**	**1.21**	**1.39**	1.75	-	-	-	-	-	-	-	-
NAT (Ours)	0.56	0.58	0.71	0.96	0.78	1.39	**1.13**	1.35	0.73	1.18	1.23	1.30	**1.14**	1.44	1.35	1.92
mNAT (Ours)	**0.54**	**0.50**	**0.64**	**0.87**	**0.73**	**1.26**	1.18	**1.22**	**0.70**	**1.15**	**1.17**	**1.24**	1.15	**1.40**	**1.29**	**1.88**
	Purchases		**Sitting**		**Sitting Down**		**Taking Photo**		**Waiting**		**Walking Dog**		**Walking Together**			
milliseconds	560	1000	560	1000	560	1000	560	1000	560	1000	560	1000	560	1000		
Zero-velocity [4]	1.64	2.45	1.26	1.63	1.36	1.80	1.03	1.27	1.89	2.63	1.74	1.96	1.10	1.52		
ERD [2]	-	-	-	-	-	-	-	-	-	-	-	-	-	-		
SRNN [3]	-	-	-	-	-	-	-	-	-	-	-	-	-	-		
Res-GRU [4]	1.41	2.30	1.24	1.51	1.28	1.72	0.95	1.17	1.64	2.30	1.69	1.73	0.80	1.43		
ConvSeq2Seq [5]	1.44	2.38	1.15	1.48	1.26	1.75	0.92	1.23	1.70	2.37	1.62	1.78	0.79	1.45		
AGED w adv [12]	-	-	-	-	-	-	-	-	-	-	-	-	-	-		
SkelNet [6]	-	-	-	-	-	-	-	-	-	-	-	-	-	-		
NAT (Ours)	1.13	1.96	**1.19**	1.55	1.17	1.64	**0.85**	1.07	1.27	1.65	1.18	1.50	0.60	1.09		
mNAT (Ours)	**1.06**	**1.81**	1.21	**1.46**	**1.09**	**1.58**	0.86	**1.04**	**1.26**	**1.58**	**1.16**	**1.44**	**0.59**	**1.07**		

TABLE IV
COMPARISON OF MEAN JOINT ERROR OF ANGLE SPACE BETWEEN OUR MODEL AND THE STATE-OF-THE-ART METHODS ON 8 ACTIONS AS WELL AS AVERAGE RESULT OF CMU-MOCAP DATASET.

	Basketball				**Basketball Signal**				**Directing Traffic**				**Jumping**				**Running**			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Zero-velocity [4]	0.48	0.82	1.40	1.64	0.24	0.44	0.75	0.86	0.30	0.57	0.90	1.01	0.36	0.62	1.48	1.68	0.56	1.00	1.38	1.46
Res-GRU [4]	0.50	0.80	1.27	1.45	0.41	0.76	1.32	1.54	0.33	0.59	0.93	1.10	0.56	0.88	1.77	2.02	0.33	0.50	0.66	0.75
ConvSeq2Seq [5]	0.37	0.62	1.07	1.18	0.32	0.59	1.04	1.24	0.25	0.56	0.89	1.00	0.39	0.60	1.36	1.56	0.28	**0.41**	**0.52**	0.57
NAT (Ours)	**0.34**	0.52	0.88	1.03	0.19	0.28	0.49	**0.61**	0.22	0.44	0.67	0.79	**0.38**	**0.56**	**1.27**	1.47	0.26	0.49	**0.52**	**0.56**
mNAT (Ours)	**0.34**	**0.49**	**0.86**	**1.01**	**0.15**	**0.24**	**0.48**	**0.61**	**0.20**	**0.41**	**0.65**	**0.77**	**0.38**	**0.56**	1.29	**1.45**	**0.24**	0.43	0.53	**0.56**
	Soccer				**Walking**				**Wash Window**				**Average**							
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400				
Zero-velocity [4]	0.27	0.48	0.92	1.10	0.41	0.60	0.83	0.95	0.34	0.57	0.90	1.10	0.37	0.64	1.07	1.22				
Res-GRU [4]	0.29	0.51	0.88	0.99	0.35	0.47	0.60	0.65	0.30	0.46	0.72	0.91	0.38	0.62	1.02	1.18				
ConvSeq2Seq [5]	0.26	0.44	0.75	0.87	0.35	0.44	0.45	0.50	0.30	0.47	0.80	1.01	0.32	0.52	0.86	0.99				
NAT (Ours)	0.23	0.34	0.61	0.73	0.33	0.39	0.42	0.48	0.27	0.40	0.72	0.93	0.28	0.43	0.70	0.83				
mNAT (Ours)	**0.20**	**0.33**	**0.59**	**0.72**	**0.31**	**0.37**	**0.40**	**0.46**	**0.23**	**0.36**	**0.66**	**0.86**	**0.26**	**0.40**	**0.68**	**0.80**				

Figure 4. When the dimension changes, an extra 1D Conv is performed to transform dimension. The number of channels in the context encoder is 64, 64, 128, 128, 256, 256 for each building block, respectively. The number of channels in the frame decoder is 256, 128, 128, 64, 64, 4, respectively. We use a pre-defined graph with relation of joints for GCN based on specific dataset. The kernel size ks is set to 9 for TCN in context encoder while set to 1 in frame decoder. Each TCN is performed with the same padding mode to ensure that the temporal size stays unchanged. LeakyReLU is utilized as the non-linear activation with a rate of 0.01. We apply $\alpha = 10$ and $\beta = 500$ for positional encoding as this setting achieves optimal performance in our observation. The dropout rate is set to 0.5 for ARC module. The whole model is lightweight with 2.28 MB.

For both datasets, ADAM [47] is selected as the optimizer in our experiment. The initial learning rate is 0.001 with a 0.9995 decay in every epoch. The gradient clip norm is set to 0.1 and the mini-batch is composed of 60 samples. Following the previous works [10], λ_{pnlty} is set to 0.01. Also, λ_{cls} is set to 0.01 to balance two tasks. We conduct an ablation study on the effects of λ_{cls} on final results. Our model is trained with PyTorch [53] framework for 3000 epochs on a single NVIDIA 1080TI GPU.

2) Evaluation Metrics and Baselines: For Human3.6M, we report our results on both short-term ($80 \sim 400$ ms) and long-term ($80 \sim 1000$ ms). For CMU-Mocap, due to the space limit, we only report short-term results. For all datasets, 50 frames (2000 ms) are given.

Following the previous evaluation protocol [4], we report the comparison results of the mean joint error of **angle space**, *i.e.*, Euler angle, between predicted joints and ground truth. Besides, we also evaluate the mean joint error of **3D space** [9], [14] in millimeter. The mean 3D error could be measured either by converting the predicted angles to 3D space, or directly train on 3D coordinates of the skeleton sequence.

To evaluate the performance of our model, we compare it with five state-of-the-art human motion prediction approaches, namely, Res-GRU [4], ConvSeq2Seq [5], AGED (w/ adv) [12], SkelNet [6], TD-DCT [9], as well as one baseline method Zero-velocity [4]. Note that the results of SkelNet are based on their open source project[2] since they do not provide results for all 15 actions. All the other results are referred from their original papers.

[2]https://github.com/CHELSEA234/SkelNet_motion_prediction

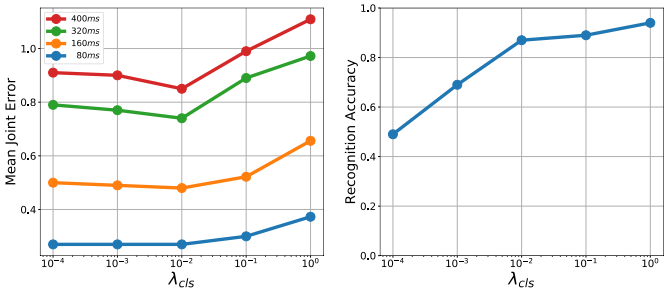
TABLE V
COMPARISON OF MEAN JOINT ERROR OF 3D SPACE BETWEEN OUR MODEL AND THE STATE-OF-THE-ART METHODS ON ALL 15 ACTIONS OF HUMAN3.6M DATASET.

	Walking				Eating				Smoking				Discussion				Direction			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
ConvSeq2Seq [5]	21.8	37.5	55.9	63.0	13.3	24.5	48.6	60.0	15.4	25.5	39.3	44.5	23.6	43.6	68.4	74.9	26.7	43.3	59.0	72.4
ConvSeq2Seq 3D [5]	17.1	31.2	53.8	61.5	13.7	25.9	52.5	63.3	11.1	21.0	33.4	38.3	18.9	39.3	67.7	75.7	22.0	37.2	59.6	73.4
TD-DCT [9]	11.1	19.0	32.0	39.1	9.2	19.5	40.3	48.9	9.2	16.6	26.1	29.0	11.3	23.7	41.9	46.6	11.2	23.2	52.7	64.1
TD-DCT 3D [9]	8.9	15.7	29.2	33.4	8.8	18.9	39.4	47.2	7.8	14.9	25.3	28.7	9.8	22.1	39.6	44.1	12.6	24.4	48.2	58.4
LDR-GCN [14]	9.7	17.7	28.3	32.2	10.2	17.4	38.7	49.3	8.9	14.1	25.9	26.7	**7.6**	23.4	36.6	**39.9**	**10.4**	24.1	44.7	51.3
LDR-GCN 3D [14]	8.9	14.9	25.4	29.9	7.6	**15.9**	37.2	**41.7**	8.1	**13.4**	24.8	**24.9**	9.4	20.3	**35.2**	41.2	13.1	23.7	44.5	50.9
Ours	9.4	13.7	23.6	28.0	8.2	17.1	35.0	44.5	**7.6**	15.5	27.4	32.1	10.2	**19.7**	38.5	43.3	12.1	**20.9**	39.2	49.4
Ours 3D	**8.4**	**13.4**	**23.3**	**27.9**	**7.0**	16.3	**33.8**	42.2	**7.6**	14.5	**23.8**	27.2	9.7	22.2	41.5	50.2	11.5	**20.9**	**30.7**	**36.7**
	Greeting				**Phoning**				**Posing**				**Purchases**				**Sitting**			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
ConvSeq2Seq [5]	30.4	58.6	110.0	122.8	22.4	38.4	65.0	75.4	22.4	42.1	87.3	106.1	28.4	53.8	82.1	93.1	24.7	50.0	88.6	100.4
ConvSeq2Seq 3D [5]	24.5	46.2	90.0	103.1	17.2	29.7	53.4	61.3	16.1	35.6	86.2	105.6	29.4	54.9	82.2	93.0	19.8	42.4	77.0	88.4
TD-DCT [9]	14.2	27.7	67.1	82.9	13.5	22.5	45.2	52.4	11.1	27.1	69.4	86.2	20.4	42.8	69.1	78.3	11.7	27.0	55.9	66.9
TD-DCT 3D [9]	14.5	30.5	74.2	89.0	11.5	20.2	37.9	43.2	9.4	23.9	66.2	82.9	19.6	38.5	64.4	72.2	10.7	24.6	50.6	62.0
LDR-GCN [14]	13.4	31.2	69.3	86.1	11.7	18.3	**32.8**	44.1	8.6	19.2	59.4	84.2	18.2	39.1	63.2	75.2	9.8	25.2	48.9	59.4
LDR-GCN 3D [14]	**9.6**	27.9	66.3	78.8	**10.4**	**14.3**	33.1	**39.7**	8.7	21.1	58.3	81.9	**16.2**	36.1	62.8	76.2	**9.2**	23.1	47.2	57.7
Ours	12.9	**24.6**	62.1	76.5	12.0	21.6	38.8	46.3	13.6	26.5	65.4	81.8	16.8	36.1	63.6	71.3	9.8	25.9	55.1	67.3
Ours 3D	13.5	25.7	**46.0**	**57.1**	12.7	20.4	35.5	42.1	**7.2**	**18.4**	**50.7**	**64.0**	18.6	**35.5**	**50.1**	**58.0**	9.3	**22.2**	**44.0**	**52.2**
	Sitting Down				**Taking Photo**				**Waiting**				**Walking Dog**				**Walking Together**			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
ConvSeq2Seq [5]	23.9	39.9	74.6	89.8	18.4	32.1	60.3	72.5	24.9	50.2	101.6	120.0	56.4	94.9	136.1	156.3	21.1	38.5	61.0	70.4
ConvSeq2Seq 3D [5]	17.1	34.9	66.3	77.7	14.0	27.2	53.8	66.2	17.9	36.5	74.9	90.7	40.6	74.7	116.6	138.7	15.0	29.9	54.3	65.8
TD-DCT [9]	11.5	25.4	53.9	65.6	8.3	15.8	38.5	49.1	12.1	27.5	67.3	85.6	35.8	63.6	106.7	126.8	11.7	23.5	46.0	53.5
TD-DCT 3D [9]	11.4	27.6	56.4	67.6	6.8	15.2	38.2	49.6	9.5	22.0	57.5	73.9	32.2	58.0	102.2	122.7	8.9	18.4	35.3	44.3
LDR-GCN [14]	10.8	24.2	49.7	61.4	**6.5**	14.3	32.3	46.7	**9.1**	21.5	50.9	68.7	26.5	54.3	94.7	119.2	10.3	20.6	34.9	45.3
LDR-GCN 3D [14]	**9.3**	**21.4**	46.3	59.3	7.1	13.8	**29.6**	44.2	9.2	**17.6**	**47.2**	71.6	25.3	56.6	87.9	99.4	8.2	18.1	31.2	39.4
Ours	15.3	24.9	53.6	63.0	7.4	14.4	34.8	**43.5**	12.8	25.2	53.9	63.8	24.9	50.5	89.6	107.6	**7.7**	17.9	33.8	43.7
Ours 3D	13.5	23.0	**34.8**	**40.3**	7.7	**13.6**	35.0	45.9	10.6	23.7	50.5	**61.3**	**21.1**	**34.5**	**55.3**	**72.5**	8.3	**15.8**	**28.0**	**33.3**

C. Comparison with State-of-the-Art

We evaluate our model on two popular datasets, Human3.6M and CMU-Mocap. We post results for NAT and the NAT variant which is equipped with multitask training paradigm (mNAT). Table I reports for the average of the mean joint error of angle space of Human3.6M dataset for all 15 human actions. With the help of our non-autoregressive setting, both our NAT and mNAT outperform all the state-of-the-art approaches in average of 15 actions. Although our results keep the same level with the latest state-of-the-art (TD-DCT [9]) in 80 and 160 ms, our methods surpass a large margin (0.09 and 0.10) in 320ms and 400ms, respectively. Note that with the help of multitask learning, the average performance gains 0.02, 0.05, 0.06 for 160, 320, and 400ms compared with NAT. The interesting part lies in that the ARC module helps improve performance in the long run. This is because, in the short term, the class information is not important since the inertance is the key factor. However, in the long term, it is not easy to predict since much more factors are taken into consideration. Therefore, to predict target pose and the action label simultaneously helps a lot by introducing the guidance from activity, leading to the convincing performance gain.

Table II shows the results for all 15 human actions in detail. NAT and mNAT also achieve the best performance on most actions. Of all the action categories, we also notice the abnormal performance in action "Phoning". After examination, we finally realize that the high error is due to the discontinuity of the test data. We observe that the finger joint data are abnormal, which largely affects the performance of the mean joint error. Since this is an end-effector joint with little influence on others, the performance in 3D space is not affected, which is also discussed in [9]. As for the results of other state-of-the-art approaches, we faithfully report their performances from their original papers.



(a) Effect of λ_{cls} for Mean Joint Error. (the lower the better)

(b) Effect of λ_{cls} for Recognition Accuracy. (the higher the better)

Fig. 7. Illustration of effects of λ_{cls} for both mean joint error and recognition accuracy of Human3.6M dataset.

In Table III, we also report the performance of long-term prediction for all 15 actions of Human3.6M dataset. For baselines with open source code, we run their codes to get the results, for baselines without public code, we just left it blank. We observe an obvious improvement for actions such as "Walking", "Eating". For example, the performance gain reaches 0.25 and 0.33 for 560 and 1000 ms of "Walking". This proves that the inertance still exists in long term such that the non-autoregressive model could be utilized. The results also show the existence of error accumulation of the previous RNN-based autoregressive methods, in which the mean joint error grows fast with the evolution of time.

Table IV reports the results on the CMU-Mocap dataset of the mean joint error. Our model also achieves the best performance on all 8 actions and the lower average error than the previous baselines, which verifies what we discuss above.

In Table V, we report the results of 3D joint error for short-term on Human 3.6M. To be consistent with previous works [9], [14], we conduct experiments in two ways: converting the

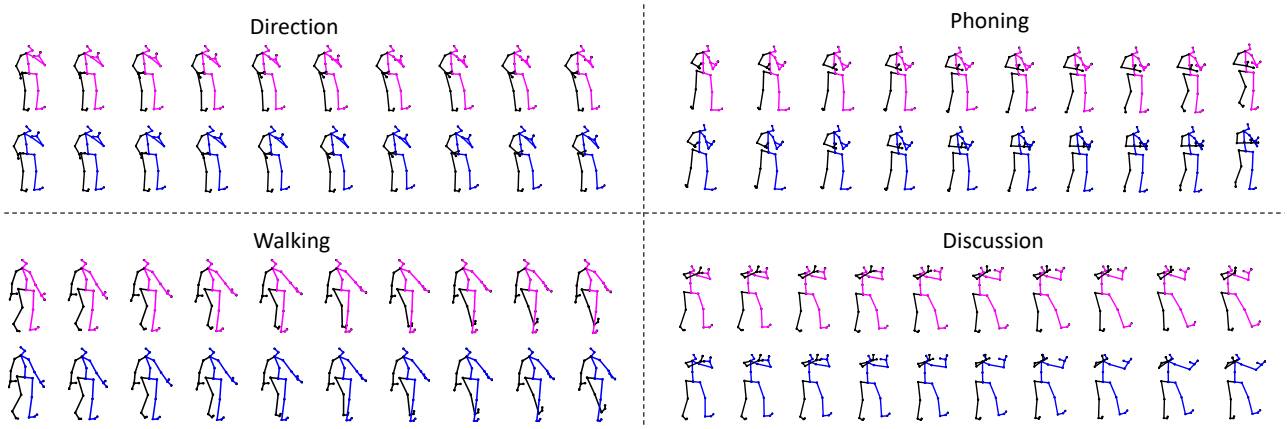


Fig. 8. Qualitative results based on Human3.6M dataset. Starting from top left, we demonstrate for four actions: direction, phoning, walking, and discussion. For each action, the top (in red color) and the bottom (in blue color) are the ground truth and our prediction respectively.

TABLE VI
THE AVERAGE MEAN JOINT ERROR BETWEEN PERFORMANCE WITH AND WITHOUT CYCLE CONSISTENCY LOSS OF HUMAN3.6M DATASET.

	Average (ms)				
	80	160	320	400	1000
mNAT w/o $\mathcal{L}_{\text{cls2}}$	0.28	**0.48**	0.75	0.88	1.38
mNAT w/ $\mathcal{L}_{\text{cls2}}$	**0.27**	**0.48**	**0.74**	**0.85**	**1.30**

TABLE VII
THE AVERAGE MEAN JOINT ERROR OF DIFFERENT VALUES OF λ_{pnlty} OF HUMAN3.6M DATASET.

	Average (ms)			
λ_{pnlty}	80	160	320	400
0.4	0.28	0.51	0.81	0.93
0.6	**0.27**	0.51	0.81	0.92
0.8	**0.27**	**0.49**	0.80	0.92
1.0	**0.27**	0.50	**0.79**	**0.91**
1.5	0.28	0.51	0.82	0.93
2.0	0.29	0.52	0.82	0.94

predicted Euler angles to 3D (ours), or directly training the models on 3D space (ours 3D). We observe that our method are on par with the current state-of-the-art method on 3D joint error metric, which verifies the generalization of our method.

We also present the qualitative results for Human 3.6M dataset in Fig. 8. From the visualization, our prediction is quite close to the ground truth. Note that even for the "phoning" action, our model still gives plausible and reliable results.

D. Ablation Study

1) Balance of losses: In this model, we present the ARC module which makes use of multitask learning, leading to convincing improvement. However, the performance gains for prediction and classification are traded off by the hyper-parameter λ_{cls}. We study the effects of λ_{cls} on two tasks.

We evaluate our model and present the mean joint error and recognition accuracy for different λ_{cls}. Fig. 7 illustrates the performances for both tasks. From the result, we observe that: 1) The recognition accuracy grows with the improvement of λ_{cls}. 2) For human motion prediction task, the performance is robust when λ_{cls} is low. However, an obvious performance drop is observed when λ_{cls} gets larger. This is mainly because motion prediction is more difficult than classification task. When we pay too much attention to classification, the model becomes unbalanced. Therefore, we choose $\lambda_{\text{cls}} = 0.01$ in our paper.

TABLE VIII
THE AVERAGE MEAN JOINT ERROR OF DIFFERENT TYPES OF EMBEDDING OF HUMAN3.6M DATASET.

	Average (ms)			
Embedding type	80	160	320	400
Random embedding	0.31	0.54	0.82	0.94
One-hot embedding	0.32	0.53	0.83	0.96
Learned embedding	0.28	0.52	0.83	0.94
Positional encoding	**0.27**	**0.50**	**0.79**	**0.91**

2) Effects of cycle consistency loss: In the multi-task setting, it encourages the predicted human motion sequence not only is close to the ground truth sequence and also represents the high-level action category. To investigate the effects of the cycle consistency classification loss, we have conducted experiments to evaluate cycle consistency loss, *i.e.*, $\mathcal{L}_{\text{cls2}}$, and the results are shown in Table VI. It is observed that the performance with the cycle consistency achieves lower prediction error than that without it, particularly in the long-term prediction (3.4% improvement on 400ms, 5.8% improvement on 1000ms), which verifies the necessity of the cycle consistency classification loss.

3) Effects of λ_{pnlty}: To investigate the effects of the balance controlling factor λ_{pnlty}, we have conducted experiments on different values of λ_{pnlty}. As shown in Table VII, λ_{pnlty}=1.0 achieves the best performance in most of the experimental settings. Also, the performance is stable in a relatively wide range from 0.4 to 2.0. In real scenario, we did not tune this hyper-parameter directly and set it as default value 1.0 for all cases.

4) Effects of positional embedding: We have conducted two experiments to evaluate the merit of positional encoding. Our methods are firstly compared with the one-hot embedding and

TABLE IX
THE AVERAGE MEAN JOINT ERROR OF DIFFERENT SETTINGS OF α, β OF HUMAN3.6M DATASET.

		Average (ms)			
α	β	80	160	320	400
1	10000	0.28	0.51	0.84	0.96
0.1	10000	0.28	0.52	0.84	0.94
10	10000	0.28	0.50	0.80	0.92
1	500	0.28	0.51	0.82	0.94
1	20000	0.29	0.54	0.85	0.99
10	500	**0.27**	**0.50**	**0.79**	**0.91**

TABLE X
THE AVERAGE MEAN JOINT ERROR OF DIFFERENT SETTINGS OF KERNEL SIZE IN TCN OF HUMAN3.6M DATASET.

	Average (ms)			
kernel size	80	160	320	400
3	0.40	0.70	1.06	1.19
5	0.40	0.71	1.07	1.20
7	0.40	0.67	0.98	1.11
9	**0.27**	**0.50**	**0.79**	0.91
11	0.28	**0.50**	0.81	**0.90**

TABLE XI
THE AVERAGE MEAN JOINT ERROR OF DIFFERENT SETTINGS OF GRAPH TYPE IN GCN OF HUMAN3.6M DATASET.

	Average (ms)			
graph type	80	160	320	400
no graph	0.28	0.53	0.85	0.98
random graph	0.29	0.52	0.82	0.96
forward	**0.27**	0.51	0.82	0.93
backward	0.28	0.52	0.83	0.95
bi-directional	**0.27**	**0.50**	**0.79**	**0.91**

the embedding which is randomly initialized. As shown in Table VIII, an apparent performance improvement is observed in random embedding (0.04 in 80ms and 0.03 in 400ms) and one-hot embedding (0.05 in 80ms and 400ms). The advantage of the positional encoding is to capture the positional-sensitive information of skeletons sequences along the time dimension and the periodic information of human motion. The positional encoding utilizes the periodic functions with different frequencies to capture the human movements of various positions and amplitudes, which is naturally tailored for the human motion prediction task.

Besides, we compare the positional encoding with the learned embedding. The Table VIII suggests an improvement in positional encoding (0.01 in 80ms and 0.03 in 400ms), which indicates the positional encoding is designed for the human motion sequences with positional and periodical information, while the learned embedding directly model the weights and lack the prior constraints of periodicity and interpretability.

5) Effects of α and β: In Sec IV, we discuss that the original setting of positional encoding module might be sub-optimal due to the domain gap between NMT and motion prediction. In Table IX, we study the influence of different settings of α and β. From the results, we observe that the performance gains with the increase of α and the decrease of β. We explain this observation with the function of two parameters. On one hand, α controls the magnitude of time index t. When α becomes larger, the positional embedding vectors become more distinguishable, leading to a high-quality prediction. On the other hand, β controls the frequency of each dimension. As can be easily observed from Fig. 5, with the high value of β, lots of dimensions are wasted due to the limited number of poses to be predicted. In conclusion, our ablation study shows that a large value of α and a small value of β lead to an optimal performance.

6) Effects of kernel size for TCN: We study the effects of kernel size for TCN in this section. In Table X, we evaluate the mean joint error for different kernel sizes ranging from 3 to 11. From the results, we find an obvious boundary between 7 and 9. When the kernel size is less than 7, the performance becomes poor. However, limited improvement is found when we further enlarge the kernel size. Since multiple GCN-TCN blocks are stacked in our model, the choice of kernel size has a direct influence for the receptive field. Therefore, the receptive field leads to the above observation. On one hand, when the kernel size is less than 7, the overall receptive field cannot cover the whole skeleton sequence. Thus, the performance drops due to the large information loss. On the other hand, all the frames are average pooled in the last, simply increasing the kernel size would not make an obvious difference. In conclusion, we choose to use kernel size 9 due to the computational efficiency.

7) Effects of graph type for GCN: We also study the effects of multiple graphs on GCN. From Table XI, "no graph" denotes that each joint is connected with itself only. In practice, we replace the adjacency matrix with an identity matrix I. Similarly, "random graph" means that the connection of joints is random. From the results, both "no graph" and "random graph" are slightly worse than GCN with people skeleton graph.

Further, we also explore the effects of the graph direction. On one hand, "forward" denotes that the connection of joints is from the central joint to all its end-effectors. On the other hand, "backward" denotes a converse direction. From the results, GCN with bi-directional graph obtains a better performance, which means that both parent joint and child joint are equally important in human motion prediction task.

E. Discussion

In this section, we discuss some potential research points for future work. First of all, to make the generated skeletons more realistic, we would add constraints to ensure the continuity and smoothness by TV-norm [54]. Secondly, due to the fact that the deep GCNs lead to a relatively high computational cost, we would accelerate the process of encoding and decoding by methods such as knowledge distillation.

As for other applications, our method is related to a broad area where the inertial human motion exists, including motion generation [55], motion re-targetting [56], and motion recovery [57]. We leave these for our future work.

VII. Conclusion

In this paper, we analyze the error accumulation problem in human motion prediction is mainly due to the recurrent decoding scheme. To remedy this issue, we present a novel human motion prediction framework based on a non-autoregressive method. The framework takes an encoder-decoder model where a simple yet effective non-autoregressive pipeline is adopted in decoding stage while multiple GCN-TCN blocks are performed so as to fully explore the spatio-temporal relation. In addition, we also find that by predicting human action category, the prediction becomes more feasible and reliable. In experiments, our approach surpasses all the recent state-of-the-art autoregressive methods.

References

[1] D. Holden, J. Saito, and T. Komura, "A deep learning framework for character motion synthesis and editing," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 138, 2016.

[2] K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik, "Recurrent network models for human dynamics," in *Proc. ICCV*, December 2015, pp. 4346–4354.

[3] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-rnn: Deep learning on spatio-temporal graphs," in *Proc. CVPR*, June 2016, pp. 5308–5317.

[4] J. Martinez, M. J. Black, and J. Romero, "On human motion prediction using recurrent neural networks," in *Proc. CVPR*, July 2017, pp. 4674–4683.

[5] C. Li, Z. Zhang, W. Sun Lee, and G. Hee Lee, "Convolutional sequence to sequence model for human dynamics," in *Proc. CVPR*, June 2018, pp. 5226–5234.

[6] X. Guo and J. Choi, "Human motion prediction via learning local structure representations and temporal dependencies," in *Proc. AAAI*, February 2019, pp. 2580–2587.

[7] E. Aksan, M. Kaufmann, and O. Hilliges, "Structured prediction helps 3d human motion modelling," in *Proc. ICCV*, October 2019, pp. 7143–7152.

[8] Z. Liu, S. Wu, S. Jin, Q. Liu, S. Lu, R. Zimmermann, and L. Cheng, "Towards natural and accurate future motion prediction of humans and animals," in *Proc. CVPR*, June 2019, pp. 9996–10 004.

[9] W. Mao, M. Liu, M. Salzmann, and H. Li, "Learning trajectory dependencies for human motion prediction," in *Proc. ICCV*, October 2019, pp. 9488–9496.

[10] D. Pavllo, D. Grangier, and M. Auli, "Quaternet: A quaternion-based recurrent model for human motion," in *Proc. BMVC*, 2018.

[11] D. Pavllo, C. Feichtenhofer, M. Auli, and D. Grangier, "Modeling human motion with quaternion-based neural networks," *Int. J. Comput. Vis.*, pp. 1–18, 2019.

[12] L.-Y. Gui, Y.-X. Wang, X. Liang, and J. M. F. Moura, "Adversarial geometry-aware human motion prediction," in *Proc. ECCV*, September 2018, pp. 823–842.

[13] A. Hernandez, J. Gall, and F. Moreno-Noguer, "Human motion prediction via spatio-temporal inpainting," in *Proc. ICCV*, October 2019, pp. 7133–7142.

[14] Q. Cui, H. Sun, and F. Yang, "Learning dynamic relationships for 3d human motion prediction," in *Proc. CVPR*, June 2020, pp. 6518–6526.

[15] Y. Zhou, Z. Li, S. Xiao, C. He, Z. Huang, and H. Li, "Auto-conditioned recurrent networks for extended complex human motion synthesis," in *Proc. ICLR*, 2018.

[16] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[17] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, 2017.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NeurIPS*, 2017, pp. 5998–6008.

[19] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proc. AAAI*, February 2018, pp. 7444–7452.

[20] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Skeleton-based action recognition with directed graph neural networks," in *Proc. CVPR*, June 2019, pp. 7904–7913.

[21] C. Si, W. Chen, W. Wang, L. Wang, and T. Tan, "An attention enhanced graph convolutional lstm network for skeleton-based action recognition," in *Proc. CVPR*, June 2019, pp. 1227–1236.

[22] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *Proc. CVPR*, June 2019, pp. 12 018–12 027.

[23] B. Li, X. Li, Z. Zhang, and F. Wu, "Spatio-temporal graph routing for skeleton-based action recognition," in *Proc. AAAI*, February 2019, pp. 8561–8568.

[24] 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 Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1325–1339, 2013.

[25] T. Zhou, H. Fu, C. Gong, J. Shen, L. Shao, and F. Porikli, "Multi-mutual consistency induced transfer subspace learning for human motion segmentation," in *Proc. CVPR*, June 2020, pp. 10 274–10 283.

[26] T. Zhou, W. Wang, S. Qi, H. Ling, and J. Shen, "Cascaded human-object interaction recognition," in *Proc. CVPR*, June 2020, pp. 4262–4271.

[27] S. Qi, W. Wang, B. Jia, J. Shen, and S.-C. Zhu, "Learning human-object interactions by graph parsing neural networks," in *Proc. ECCV*, September 2018, pp. 407–423.

[28] T. Li, Z. Liang, S. Zhao, J. Gong, and J. Shen, "Self-learning with rectification strategy for human parsing," in *Proc. CVPR*, June 2020, pp. 9260–9269.

[29] F. S. Grassia, "Practical parameterization of rotations using the exponential map," *Journal of Graphics Tools*, vol. 3, no. 3, pp. 29–48, 1998.

[30] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[31] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[32] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, Oct. 2014, pp. 103–111.

[33] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[34] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. NeurIPS*, 2014, pp. 3104–3112.

[35] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Proc. NeurIPS*, 2015, pp. 1171–1179.

[36] A. M. Lamb, A. G. A. P. Goyal, Y. Zhang, S. Zhang, A. C. Courville, and Y. Bengio, "Professor forcing: A new algorithm for training recurrent networks," in *Proc. NeurIPS*, 2016, pp. 4601–4609.

[37] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *Proc. ICML*. JMLR. org, 2017, pp. 1243–1252.

[38] J. Gu, J. Bradbury, C. Xiong, V. O. Li, and R. Socher, "Non-autoregressive neural machine translation," *arXiv preprint arXiv:1711.02281*, 2017.

[39] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[40] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. NeurIPS*, 2016, pp. 3844–3852.

[41] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, "Spectral networks and locally connected networks on graphs," in *Proc. ICLR*, 2014.

[42] W. Wang, X. Lu, J. Shen, D. J. Crandall, and L. Shao, "Zero-shot video object segmentation via attentive graph neural networks," in *Proc. ICCV*, October 2019, pp. 9235–9244.

[43] J. Yin, J. Shen, C. Guan, D. Zhou, and R. Yang, "Lidar-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention," in *Proc. CVPR*, June 2020, pp. 11 492–11 501.

[44] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[45] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, vol. 30, no. 1, 2013, p. 3.

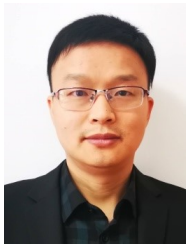
[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, June 2016, pp. 770–778.

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[48] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proc. ICCV*, December 2015, pp. 1422–1430.
[49] L. Gomez, Y. Patel, M. Rusinol, D. Karatzas, and C. V. Jawahar, "Self-supervised learning of visual features through embedding images into text topic spaces," in *Proc. CVPR*, July 2017, pp. 2017–2026.
[50] E. Pervin and J. A. Webb, "Quaternions in computer vision and robotics," Carnegie-Mellon Univ Pittsburgh PA DEPT of Computer Science, Tech. Rep. AD-A-125076/0, 1982.
[51] K. Shoemake, "Animating rotation with quaternion curves," in *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, 1985, pp. 245–254.
[52] "CMU Graphics Lab motion capture database." http://mocap.cs.cmu.edu.
[53] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. NeurIPS*, 2019, pp. 8024–8035.
[54] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.
[55] S. Yan, Z. Li, Y. Xiong, H. Yan, and D. Lin, "Convolutional sequence generation for skeleton-based action synthesis," in *Proc. ICCV*, October 2019, pp. 4393–4401.
[56] R. Villegas, J. Yang, D. Ceylan, and H. Lee, "Neural kinematic networks for unsupervised motion retargetting," in *Proc. CVPR*, June 2018, pp. 8639–8648.
[57] Q. Cui, H. Sun, Y. Li, and Y. Kong, "A deep bi-directional attention network for human motion recovery," in *Proc. of IJCAI*, 7 2019, pp. 701–707.

Bin Li received the bachelor's degree in information and communication engineering from Zhejiang University, China, in 2015. He is currently pursuing the Ph.D. degree with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China, under the supervision of Prof. Z. Zhang and Prof. X. Li. His current research interests are primarily in computer vision and deep learning.

Jian Tian received the bachelor's degree in computer science and technology from Zhejiang University, China, in 2018. He is currently pursuing the master's degree with the College of Software Technology, Zhejiang University, Hangzhou, China, under the supervision of Prof. X. Li. His current research interests are primarily in computer vision and deep learning.

Zhongfei Zhang received B.S. (Hons.) in electronics engineering and M.S. in information sciences both from Zhejiang University, China, and Ph.D. in computer science from the University of Massachusetts at Amherst, USA. He is currently a Professor at Computer Science Department, Binghamton University, State University of New York, USA. His research interests include machine learning, knowledge discovery, artificial intelligence, computer vision, and pattern recognition. He is an IEEE Fellow.



Hailing Feng Received his Ph.D. degree in computer science from the University of Science and Technology of China in June 2007. Since 2007, he worked in the school of information engineering of Zhejiang A&F University. From 2013 to 2014, he was a visiting professor at Forest Products Laboratory, USDA. He is currently a professor in the school of information engineering. His main interest areas include computer vision, intelligent information processing, and Internet of Things.

Xi Li received the Ph.D. degree from the National Laboratory of Pattern Recognition, Chinese Academy of Sciences, Beijing, China, in 2009. From 2009 to 2010, he was a Post-Doctoral Researcher with CNRS, Telecomd ParisTech, France. He is currently a Full Professor with Zhejiang University, China. Prior to that, he was a Senior Researcher with the University of Adelaide, Australia. His research interests include visual tracking, motion analysis, face recognition, web data mining, and image and video retrieval.