Pose Transformers (POTR): Human Motion Prediction with Non-Autoregressive Transformers

Angel Martínez-González*†, Michael Villamizar†and Jean-Marc Odobez*†
*École Polytechnique Fédéral de Lausanne, Switzerland
†Idiap Research Institute, Martigny, Switzerland
angel.martinez@idiap.ch, michael.villamizar@idiap.ch, odobez@idiap.ch

Abstract

We propose to leverage Transformer architectures for non-autoregressive human motion prediction. Our approach decodes elements in parallel from a query sequence, instead of conditioning on previous predictions such as in state-of-the-art RNN-based approaches. In such a way our approach is less computational intensive and potentially avoids error accumulation to long term elements in the sequence. In that context, our contributions are fourfold: (i) we frame human motion prediction as a sequence-to-sequence problem and propose a non-autoregressive Transformer to infer the sequences of poses in parallel; (ii) we propose to decode sequences of 3D poses from a query sequence generated in advance with elements from the input sequence; (iii) we propose to perform skeleton-based activity classification from the encoder memory, in the hope that identifying the activity can improve predictions; (iv) we show that despite its simplicity, our approach achieves competitive results in two public datasets, although surprisingly more for short term predictions rather than for long term ones.

1. Introduction

An important ability of an artificial system aiming at human behaviour understanding resides in its capacity to apprehend the human motion, including the possibility to anticipate motion and activities (e.g. reaching towards objects). As such, human motion prediction finds applications in visual surveillance or human-robot interaction (HRI) and has been a hot topic researched for decades.

With the recent popularity of deep learning, Recurrent Neural Networks (RNN) have replaced conventional methods that relied on Markovian dynamics [12] and smooth body motion [20] and instead learn these sequence properties from data. However, motion prediction remains a challenging task due to the non-linear nature of the articulated body structure. Although the different motions of the body

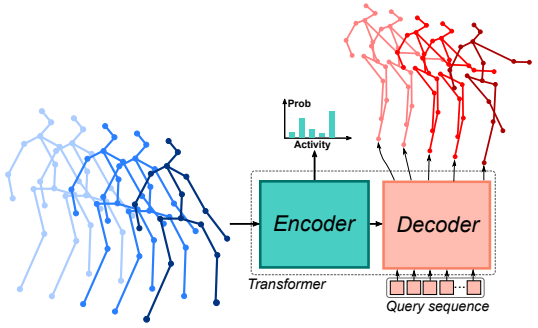

Figure 1: Proposed approach for non-autoregressive motion prediction approach with Transformers. The input sequence is encoded with the Transformer encoder. The decoder works in a non-autoregressive fashion generating the predictions of all poses in parallel. Finally, the encoder embeddings are used for skeleton-based activity classification.

landmarks are highly correlated, their relations and temporal evolution are hard to model in learning systems.

Recently, a family of RNN-based approaches have proposed to frame the task of human motion prediction as a sequence-to-sequence problem. These methods usually rely on stacks of LSTM or GRU modules and solve the task with autoregressive decoding: generating predictions one at a time conditioned on previous predictions [17, 2]. This practice has two major shortcomings. First, autoregressive models are prone to accumulate prediction errors over time: predicted elements are conditioned to previous predictions, containing a degree of error, thus potentially increasing the error for new predictions. Second, autoregressive modelling is not parallelizable which may cause deep models to be more computationally intensive since predicted elements are generated sequentially one at time.

Since their breakthrough in machine translation [21], Transformer neural network have been adopted in other research areas for different sequence-to-sequence tasks such as automatic speech recognition [11] and object detection [3]. These methods leverage the long range memory of

the attention modules to identify specific entries in the input sequence which are relevant for prediction, a shortcoming of RNN models. During training, Transformers allows parallelization with *look ahead* masking. Yet, at testing time, they use an autoregressive setting which makes it difficult to leverage the parallelization capabilities. Hence, autoregressive Transformers exhibit large inference processing times hampering their use in applications that require real-time performance such as in HRI.

In this paper, we thus investigate the use of non-autoregressive human motion prediction aiming to reduce computational cost of autoregressive inference with the Transformer neural network and potentially avoid error propagation. Our work is inline with recent methods [3, 7] that perform non-autoregressive (parallel) decoding with Transformers. Contrary to state-of-the-art methods that rely only in a Transformer encoder for human motion prediction [1, 22], our approach uses as well a Transformer decoder architecture with self- and encoder-decoder attention. Inspired by recent research in non-autoregressive machine translation [7], we generate the inputs to the decoder in advance with elements from the input sequence. We show that this strategy, though simple, is effective and helps reducing the error in short and long term horizons.

In addition, we explore the inclusion of activity information by predicting as well activity from the input sequences. Modelling motion and activity prediction jointly has not often been investigated by previous works, though these topics are highly related. Indeed, a better ability at identifying an activity may improve the selection of the dynamics to be applied to the sequence. Hence, we propose a skeleton-based activity classification by classifying activities using the encoder self-attention predictions. We train our models jointly for activity classification and motion prediction and study the potential of this multi-task framework. Code and models will be made public [1].

The rest of the paper is organized as follows. Section 2 presents relevant state-of-the-art methods to our work. Section 3 introduces our approach for non-autoregressive motion prediction with Transformers. Experimental protocol and results are presented in Section 4 and Section 5 concludes our work.

2. Related work

Deep autoregressive methods. Early deep learning approaches used stacks of RNN units to model human motion. For example, the work in [6] introduces an encoder-recurrent-decoder (ERD) network with a stack of LSTM units. The approach prevents of error accumulation and catastrophic drift by including a schedule to add Gaussian noise to the inputs and increase the model robustness. However, this scheduling is hard to tune in practice. The work presented in [17] uses a encoder-decoder RNN architecture with a single GRU unit. The architecture includes a residual connection between decoder inputs and outputs as a way of modeling velocities in the predicted sequence. This connectivity reduces discontinuity between input sequences and predictions and adds robustness at long time horizons. Along this line, the approach in [2] introduce a decoder that explicitly models the spatial dependencies between the different body parts with small specialized networks, each predicting a specific body part (e.g. elbow). Final predictions are decoded following the hierarchy of the body skeleton which reduces the drift effect. Recently, a family of methods prevent the drift issue by including adversarial losses and enhance prediction quality with geodesic body measurements [8] or by framing motion prediction as an pose inpainting process with GANs [9]. However, training with adversarial loses is difficult and hard to stabilize.

Attention-based approaches have recently gained interest for modeling human motion. For example, the work presented in [22] exploits a self-attention module to attend the input sequence with a sliding window of small subsequences from the input. Ideally, attention should be larger in elements of the input sequence that repeat with time. Prediction works in an autoregressive fashion using a Graph Convolutional Network (GCN) to decode attention embeddings to 3D skeletons. Along the same line [1] introduces an spatio-temporal self-attention module to explicitly model the spatial components of the sequence. Input sequences are processed by combining two separate self-attention modules: a spatial module to model body part relationships and a temporal module to model temporal relationships. Predictions are generated by aggregating attention embeddings with feedforward networks in an autoregressive fashion.

Our work differs from these works. First, our architecture is a encoder-decoder Transformer, with self- and encoder-decoder attention. This allows us to exploit the Transformer decoder to identify elements in the input sequence relevant for prediction. Secondly, our architecture works in non-autoregressive fashion to prevent the overhead of autoregressive decoding.

Non-autoregressive modelling. Most neural network-based models for sequence-to-sequence modelling use autoregressive decoding: generating entries in the sequence one at a time conditioned on previous predicted elements. This approach is not parallelizable causing deep learning models to be more computationally intensive, as in the case of machine translation with Transformers [21, 18]. Although in principle Transformers are paralellizable, autoregressive decoding makes impossible to leverage this property during inference. Therefore, recent efforts have sought to parallelize decoding with transformers in machine translation using *fertilities* [7] and in visual object detection by

[1] https://github.com/idiap/potr

decoding sets [3].

Non-autoregressive modeling has also been explored in the human motion prediction literature. Clearly, the most challenging aspect is to represent the temporal dependencies for decoding predictions. Most of the solutions in the literature provide additional information to the decoder that account for the temporal correlations in the target sequence. Different methods have been proposed relying in decoder architectures that exploit temporal convolutions [14], feeding the decoder with learnable embeddings [13], or relying in a representation of the sequence in the frequency domain [23]. The work presented in [23] represents the temporal dependencies using the Discrete Cosine Transform (DCT) of the sequence. During inference a GCN predicts the DCT coefficients of the target sequence. However, to account for smoothness, during training, the GCN is trained to predict both input and target sequence DCT coefficients. The approach in [14] performs a similar approach, modelling separately short term and long term dependencies with temporal convolutions. Their decoder is composed of a short term and long term temporal encoders that move in a sliding window. Short and long term information are then processed by a spatial decoder to produce pose frames.

Our approach contrast from these methods in different ways. First, we do not incorporate any prior information of the temporality of the sequences and let the Transformer learn these from sequences of skeletons. Additionally, our decoding process relies in a simple strategy to generate query sequences from the inputs rather than relying in learnable query embeddings.

3. Method

The goal of our study is to explore solutions for human motion prediction leveraging the parallelism properties of Transformers during inference. In the following sections we introduce our Pose Transformer (POTR), a non-autoregressive Transformer for motion prediction and skeleton-based activity recognition.

3.1. Problem Formulation

Given a sequence $\mathbf{X} = \{\mathbf{x}_{1:T}\}$ of 3D poses, we seek to predict the most likely immediate following sequence $\mathbf{Y} = \{\mathbf{y}_{1:M}\}$, where $\mathbf{x}_t, \mathbf{y}_t \in R^N$ are N-dimensional pose vectors (skeletons). This problem is strongly related with conditional sequence modelling where the goal is to model the probabilities $P(\mathbf{Y}|\mathbf{X}; \theta)$ with model parameters θ. In our work, θ are the parameters of a Transformer.

Given its temporal nature, motion prediction has been widely addressed as an autoregressive approach in an encoder-decoder configuration: the encoder takes the conditioning motion sequence $\mathbf{x}_{1:T}$ and computes a representation $\mathbf{z}_{1:T}$ (memory). The decoder then generates pose vectors $\mathbf{y}_t$ one by one taking $\mathbf{z}_{1:T}$ and its previous generated vectors $\mathbf{y}_{\tau<t}$. While this autoregressive approach explicitly models the temporal dependencies of the predicted sequence $\mathbf{y}_{1:M}$, it requires to execute the decoder M times. This becomes computationally expensive for very large Transformers, which in principle have the property of parallelization (exploited during training). Moreover, autoregressive modelling is prone to propagate errors to future predictions: predicting pose vector $\mathbf{y}_t$ relies in predictions $\mathbf{y}_{\tau<t}$ which in practice contain a degree of error. We address these limitations by modelling the problem in a non-autoregressive fashion as we describe in the following.

3.2. Pose Transformers

The overall architecture of our POTR approach is shown in Figure 2. Similarly to the original Transformer [21], our encoder and decoder modules are composed of feed forward networks and multi-head attention modules. While the encoder architecture stays unchanged, the decoder works in a non-autoregressive fashion to avoid error accumulation and reduce computational cost.

Our POTR comprises three main components: a pose encoding neural network ϕ that computes pose embeddings for each 3D pose vector in the input sequence, a non-autoregressive Transformer, and a pose decoding neural network ψ that computes a sequence of 3D pose vectors. While the Transformer learns the temporal dependencies, the networks ϕ and ψ shall identify spatial dependencies between the different body parts for encoding and decoding pose vector sequences.

More specifically, our architecture works as follows. First, the pose encoding network ϕ computes an embedding of dimension D for each pose vector in the input sequence $\mathbf{x}_{1:T}$. The Transformer encoder takes the sequence of pose embeddings (agreggated with positional embeddings) and computes the representation $\mathbf{z}_{1:T}$ with a stack of L multi-head self-attention layers. The Transformer decoder takes the encoder outputs $\mathbf{z}_{1:T}$ as well as a *query sequence* $\mathbf{q}_{1:M}$ and computes an output embedding with a stack of L multi-head self- and encoder-decoder attention layers. Finally, pose predictions are generated in parallel by the network ψ from the decoder outputs and a residual connection with $\mathbf{q}_{1:M}$. We detail each component in the following.

Transformer Encoder. It is composed of L layers, each with a standard architecture consisting of multi-head self-attention modules and a feed forward networks. The encoder receives as input the sequence of pose embeddings of dimension D added with positional encodings and produces a sequence of embeddings $\mathbf{z}_{1:T}$ of the same dimensionality.

Transformer Decoder. Our Transformer decoder follows the standard architecture: it comprise L layers of multi-head self- and encoder-decoder attention modules and feed forward networks. In our work, every layer in the decoder gen-

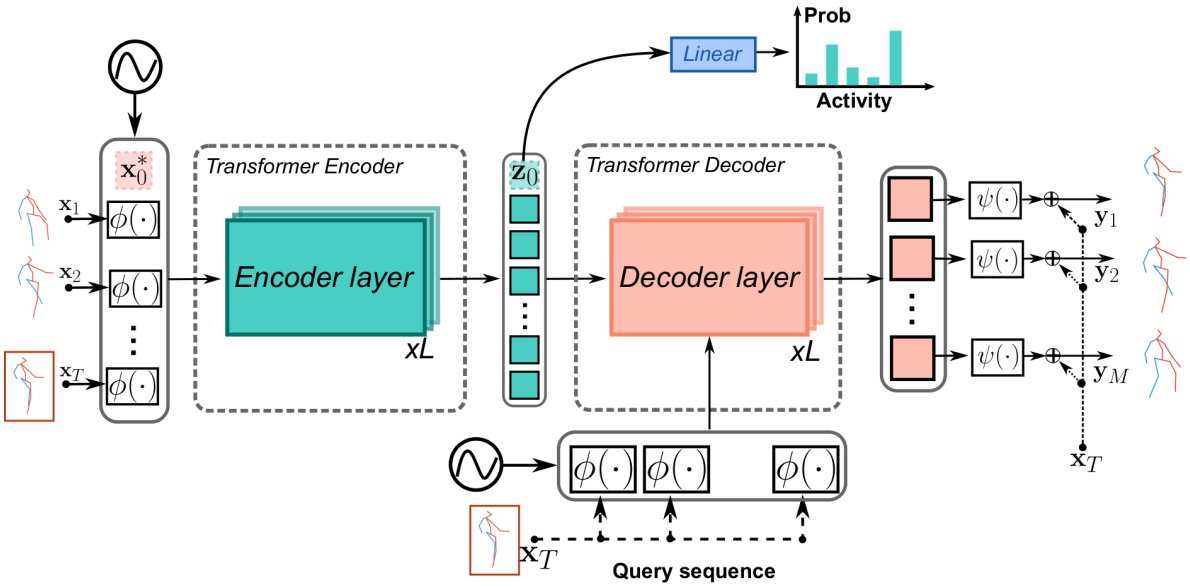

Figure 2: Overview of our approach for non-autoregressive human motion prediction. Our model is composed of networks ϕ and ψ, and a non-autoregressive Transformer built on feed forward networks and multi-head attention layers as in [21]. First, a network ϕ computes embeddings for each pose in the input sequence. Then, the Transformer processes the sequence and decodes attention embeddings in parallel. Finally, the predicted sequence is generated with network ψ in a residual fashion. Activity classification is performed by adding a learnable *class token* $\mathbf{x}_0$ to the input sequence.

erates predictions. The decoder receives a query sequence $\mathbf{q}_{1:M}$ and encoder outputs $\mathbf{z}_{1:T}$ and produces M output embeddings in a single pass. These are then decoded by the network ψ into 3D body skeletons.

The decoding process starts by generating the input to the decoder $\mathbf{q}_{1:M}$. As remarked in [7] given that non-autoregressive decoding exhibits complete conditional independence between predicted elements $\mathbf{y}_t$, the decoder inputs should account as much as possible for the time correlations between them. Additionally, $\mathbf{q}_{1:M}$ should be easily inferred. Inspired by non-autoregressive machine translation [7], we use a simple approach filling $\mathbf{q}_{1:M}$ using copied entries from the encoder inputs. More precisely, each entry $\mathbf{q}_t$ is a copy of a selected *query pose* from the encoder inputs $\mathbf{x}_{1:T}$. We select the last element of the sequence $\mathbf{x}_T$ as the query pose and fill the query sequence with this entry. Given the residual learning setting, predicting motion can be seen as predicting the necessary pose offsets from last conditioning pose $\mathbf{x}_T$ to each element $\mathbf{y}_t$.

3.3. Pose Encoding and Decoding

Input and output sequences are processed from and to 3D pose vectors with networks ϕ and ψ respectively. The network ϕ is shared by the Transformer encoder and decoder. It computes a representation of dimension D for each of the 3D skeletons in the input and query sequences. The decoding network ψ transforms the M decoder predictions of dimension D to 3D skeletons residuals independently at every decoder layer.

The aim of the ϕ and ψ networks is to model the spatial relationships between the different elements of the body structure. To do this, we investigated two approaches. In the first one we consider a simple approach setting ϕ and ψ with single linear layers. In the second approach we follow [22] and use Graph Convolutional Networks (GCN) that densely learn the spatial connectivity in the body.

To make our manuscript self contained, we briefly introduce how GCNs work in our human motion prediction approach. Given a feature representation of the human body with K nodes, a GCN learns the relationships between nodes with the strength of the graph edges represented by the adjacency matrix $\mathbf{A} \in R^{K \times K}$. Examples of representations are body skeletons or embeddings. A GCN layer l takes as input a matrix of node features $\mathbf{H}_{l-1} \in R^{K \times F}$ with F features per node, and a set of learnable weights $\mathbf{W}_l \in R^{F \times O}$. Then, the layer computes output features

$$\mathbf{H}_l = \sigma(\mathbf{A}_l \mathbf{H}_{l-1} \mathbf{W}_l), \tag{1}$$

where σ is an activation function. A network is composed by stacking layers which aggregates features of the vicinity of the nodes.

Our GCN architecture is shown in Figure 3. It is inspired in the architecture presented in [22], where matrices $\mathbf{A}_l$ and weights $\mathbf{W}_l$ are learnt. It is composed of a stack of S residual modules of graph convolution layers followed by batch normalization, *tanh* activations and dropout layers.

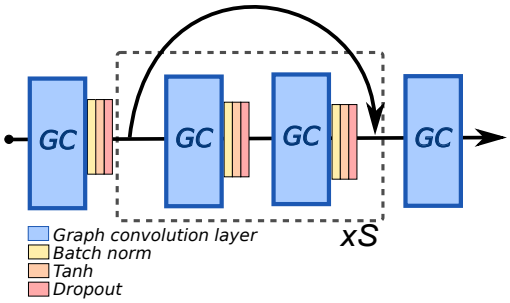

Figure 3: Our Graph Convolutional Network architecture. It comprises graph convolution layers followed by *tanh* activations, batch normalization, and dropout layers. As in [22], our architecture has S residual connections.

We set the internal feature dimension to $F = 512$ features per node until the output layer that generates pose embeddings. Though we can normally squeeze as many inner layers, we set $S = 1$.

3.4. Activity Recognition

Activity can normally be understood as a sequence of motion of the different body parts in interaction with the scene context (objects or people). In our method, the Transformer encoder encodes the body motion with a series of self-attention layers. We explore the use of encoder outputs $\mathbf{z}_{1:T}$ for activity classification (as a second task) and train a classifier to determine the action corresponding to the motion sequence presented as input to the Transformer.

We explore two approaches. The first approach consist on using the entire Transformer encoder outputs $\mathbf{z}_{1:T}$ as input to the classifier. However, these normally contain many zeroed entries suppressed by the probability maps normalization in the multi-head attention layers. Naively using these for activity classification might lead our classifier to struggle in discarding these many zero elements. Therefore, similar to [4], we include a specialized *class token* in the input sequence to store information about the activity of the sequence. The class token $\mathbf{x}_0$ is a learnable embedding that is padded to input sequence to form $\mathbf{x}_{0:T}$. In the output of encoder embeddings $\mathbf{z}_{0:T}$, $\mathbf{z}_0$ works as the activity representation of the encoded motion sequence. To perform activity classification we feed $\mathbf{z}_0$ to a single linear layer to predict class probabilities for C activity classes (see Figure 2).

3.5. Training

We train our model in a multi-task fashion to jointly predict motion and activity. Let $\hat{\mathbf{y}}^l_{1:M}$ be the predicted sequence of N-dimensional pose vectors at layer l of the Transformer decoder. We compute the layerwise loss

$$L_l = \frac{1}{M \cdot N} \sum_{t=1}^{M} ||\hat{\mathbf{y}}^l_t - \mathbf{y}^*_t||_1, \tag{2}$$

where $\mathbf{y}^*_t$ is the ground truth skeleton at target sequence entry t. The overall motion prediction loss L_{motion} is computed by averaging the losses over all decoder layers L_l. Finally, we train our POTR with the loss

$$L_{POTR} = L_{motion} + \lambda L_{activity}, \tag{3}$$

where $L_{activity}$ is the multi-class cross entropy loss and $\lambda = 1$.

4. Experiments

This section presents the experiments we conducted to evaluate our approach.

4.1. Data

Human 3.6M [10]. We used the Human 3.6M dataset in our experiments for human motion prediction. The dataset depicts seven actors performing 15 activities, e.g. walking, eating, sitting, etc. We followed standard protocols for training and testing [17, 2, 23]. Subject 5 is used for testing while the others for training. Input sequences are 2 seconds long and testing is performed over the first 400 ms of the predicted sequence. Evaluation is done in a total of 120 sequences (8 seeds) across all activities by computing the Euler angle error between predictions and ground truth.

NTU Action Dataset [19]. The NTU-RGB+D dataset is one of the biggest benchmark datasets for human activity recognition. It is composed of 58K Kinect 2 videos of 40 different actors performing 60 different actions. We followed the cross subject evaluation protocol provided by the authors using 40K sequences for training and 16.5K for testing. Given the small length of the sequences, we set input and output sequences length to 1.3 seconds (40 frames) and 660 ms (20 frames) respectively.

4.2. Implementation details

Data Preprocessing. We apply standard normalization to the input and ground truth skeletons by substracting the mean and dividing by the standard deviation computed over the whole training set. For the H3.6M dataset we remove global translation of the skeletons and represent the skeletons with rotation matrices. Skeletons in the NTU dataset are represented in 3D coordinates and are centred by subtracting the spine joint.

Training. We use Pytorch as our deep learning framework in all our experiments. Our POTR is trained with AdamW [15] setting the learning rate to 10^{-04} and weight decay to 10^{-05}. POTR models for the H3.6M dataset are trained during 100K steps with warmup schedule during 10K steps. For the NTU dataset we train POTR models during 300K steps with warmup schedule during 30K.

Models. We set the dimension of the embeddings in our

milliseconds	80	160	320	400	560	1000
POTR-AR	0.23	0.57	0.99	1.14	1.37	1.81
POTR	0.23	**0.55**	**0.94**	1.08	1.32	1.79
POTR-GCN (enc)	**0.22**	0.56	**0.94**	**1.01**	**1.30**	**1.77**
POTR-GCN (dec)	0.24	0.57	0.96	1.10	1.33	1.77
POTR-GCN (full)	0.23	0.57	0.96	1.10	1.33	1.80

Table 1: **H3.6M** prediction performance in terms of the Euler angle error. Top: autoregressive (POTR-AR) and non-autoregressive POTR models using linear layers for networks ϕ and ψ. Bottom: non-autoregressive models with GCNs for network ϕ (enc), network ψ (dec) and both (full).

POTR models to $D = 128$. The multi-head attention modules are set with pre-normalization and four attention heads and four layers in encoder and decoder.

4.3. Evaluation metrics

Euler Angle Error. We followed standard practices to measure the error of pose predictions in the H3.6M dataset by computing the euclidean norm between predictions and ground truth Euler angle representations.

Mean Average Precision (mAP). We use mAP@10cm to measure the performance of predictions in the NTU dataset. A successful detection is considered when the predicted 3D body landmark falls within a distance less than 10 cm from the ground truth.

Mean Per Joint Position Error (MPJPE). We use the MPJPE to evaluate error in the NTU dataset. MPJPE measures the average error in Euclidean distance between the predicted 3D body landmarks and the ground truth.

4.4. Results

4.4.1 Evaluation on H3.6M Dataset

In this section, we validate our proposed approach for motion prediction in the H3.6M dataset.

Non-Autoregressive Prediction. Table 1 compares the performance in terms of the Euler angle error of our POTR with its autoregressive version (POTR-AR). Lower values are better. The autoregressive version do not use the query pose and predicts pose vectors one at a time from its own predictions. Our non-autoregressive approach shows lower error than its counter part in most of the time intervals.

Pose Encoding and Decoding. We experimented with the networks ϕ and ψ using either linear layers or GCNs. Table 1 reports the results (bottom part). We indicate when models are trained with GCN in the encoder (enc), decoder (dec) or in both (full). We observe that the use of GCN reduces the errors when it is applied exclusively to the encoder. Using a shallow GCN ($S = 1$) ψ might be a weak attempt to decode pose vectors. However, we observed that the small size of the H3.6M dataset might not be enough to learn deeper GCN architectures.

Comparison with the State-Of-The-Art. Tables 2 and 3 compares our best performing model with the state-of-the-art in terms of angle error for all the activities in the dataset. Our POTR often obtains the first and second lower errors in in the short term, and the lowest average error in the 80ms range. The use of the last input sequence entry as the query pose most probably helps to significantly reduce the error in the immediate horizons. However, this strategy introduces larger errors for longer horizons where the difference between further pose vectors in the sequence and the query pose is larger (see Figure 4 for some examples). In such a case, it appears that autoregressive approaches perform better as a result of updating the conditioning decoding distribution to elements closer in time.

Attention Weights Visualization. In Figure 5(a) we visualize the encoder-decoder attention maps for one prediction instance of four activities in the dataset. Figure 5(b) further shows the attention between elements of the input and predicted sequences for the *walking* action. Due to the continuity within such activity, we can notice a high dependency (attention) between the last elements of the input and the firts elements of the predicted sequences, while the prediction of further elements also pay attention to other input elements of the input matching the same phase of the walking cycle. A similar behavior is observed for the direction example. For the eating and discussion activities involving less body motion, we can notice that while the approach slightly attends to the last elements of the input, it also strongly attends to other specific segments. Further analysis would be needed to analyse the behavior of these weight matrices.

Computational Requirements. We measured the computational requirements of models POTR and POTR-AR by the number of sequences per second (SPS) of their forward pass in a single Nvidia card GTX 1050. We tested models with 4 layers in encoder and decoder, and 4 heads in their attention layers. We input sequences of 50 elements and predict sequences of 25 elements. POTR runs at 149.2 SPS while POTR-AR runs at 8.9 SPS. Therefore, the non-autoregressive approach is less computationally intensive.

4.4.2 Evaluation on NTU Dataset

This section presents our results on jointly predicting motion and activity on the NTU dataset.

Motion Prediction Performance. Table 4 compares our POTR with the different decoding settings using the mAP. Notice that removing the activity loss ($\lambda = 0$) slightly drops the performance for the longer horizons. The non-autoregressive setting shows higher mAP than the autoregressive setting, specially in long term. However, setting

	Walking				Eating				Smoking				Discussion			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Zero Velocity [17]	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
Seq2seq. [17]	0.28	0.49	0.72	0.81	0.23	0.39	0.62	0.76	0.33	0.61	1.05	1.15	0.31	0.68	1.01	1.09
AGED [8]	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**
RNN-SPL [2]	0.26	0.40	0.67	0.78	0.21	0.34	0.55	0.69	0.26	0.48	0.96	0.94	0.30	0.66	0.95	1.05
DCT-GCN (ST) [22]	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
ST-Transformer [1]	0.21	0.36	0.58	0.63	0.17	0.30	**0.49**	**0.60**	0.22	0.43	0.88	0.82	0.19	0.52	0.79	0.88
POTR-GCN (enc)	**0.16**	0.40	0.62	0.73	**0.11**	0.29	0.53	0.68	**0.14**	**0.39**	0.84	0.82	**0.17**	0.56	0.85	0.96

Table 2: **H3.6M** performance comparison with the state-of-the-art in terms of the Euler angle error for the common *walking*, *eating*, *smoking* and *discussion* across different horizons.

	Directions				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	80	160	320	400
Seq2seq [17]	0.26	0.47	0.72	0.84	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
AGED [8]	0.23	0.39	**0.63**	**0.69**	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
DCT-GCN (ST) [22]	0.26	0.45	0.71	0.79	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**
ST-Transformer [1]	0.25	**0.38**	0.75	0.86	0.35	0.61	1.10	1.32	0.53	1.04	1.41	1.54	0.61	0.68	1.05	1.28	0.43	0.77	1.30	1.37	0.29	0.46	0.84	1.01
POTR-GCN (enc)	**0.20**	0.45	0.79	0.91	**0.29**	0.69	1.17	1.30	0.50	1.10	1.50	1.65	**0.18**	0.52	1.18	1.47	**0.33**	**0.63**	1.04	1.09	**0.25**	0.47	0.92	1.09

	Sitting down				Taking photos				Waiting				Walking Dog				Walking Together				Average			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Seq2seq. [17]	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	0.36	0.67	1.02	1.15
AGED [8]	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	0.31	0.54	0.85	0.97
DCT-GCN (ST) [22]	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**	0.27	**0.52**	**0.83**	**0.95**
ST-Transformer [1]	0.32	0.66	0.98	1.10	0.15	0.38	0.64	0.75	0.22	0.51	0.98	1.22	0.43	**0.78**	1.15	1.30	0.17	0.37	0.58	0.62	0.30	0.55	0.90	1.02
POTR-GCN (enc)	**0.25**	0.63	1.00	1.12	**0.12**	0.41	0.71	0.86	**0.17**	0.56	1.14	1.37	**0.35**	0.79	1.21	1.33	**0.15**	0.44	0.63	0.70	**0.22**	0.56	0.94	1.01

Table 3: Euler angle error results for the reminder of the 11 actions in the **H3.6M** dataset with our main non-autoregressive transformer.

milliseconds	80	160	320	400	500	660	avg	accuracy
POTR-AR	0.96	0.92	0.85	0.83	0.80	0.76	0.76	0.32
POTR	0.96	**0.93**	**0.89**	**0.87**	**0.86**	**0.84**	**0.84**	**0.38**
POTR ($\lambda = 0$)	0.96	**0.93**	**0.89**	**0.87**	0.85	0.83	0.83	-
POTR (memory)	0.96	0.92	0.88	**0.87**	0.85	0.83	0.83	0.30
POTR-GCN (enc)	0.96	0.92	0.88	**0.87**	0.85	0.83	0.83	0.27
POTR-GCN (dec)	0.96	0.92	0.88	0.86	0.85	0.83	0.83	0.34
POTR-GCN (full)	0.95	0.90	0.85	0.84	0.82	0.79	0.79	0.30

Table 4: **NTU** motion prediction performance in terms of the mAP@10cm for different time horizons. Higher values are better. Model marked with *memory* replace the class token with the encoded memory for activity classification.

the networks ϕ and ψ with GCNs does not bring many benefits compared to using linear layers.

Figure 6 compares their per body part mAP and MPJPE using linear layers for ϕ and ψ. POTR-AR shows larger MPJPE and lower mAP than POTR specially for the body extremities (arms and legs).

Activity Recognition. Table 4 compares the classification accuracy for the different POTR configurations. Using a specialized activity token shows better performance than using the encoder memory $\mathbf{z}_{1:T}$. Given that the memory embeddings contain many non-informative zeroed values the classifier could get stuck in an attempt to ignore them.

Table 5 compares the classification accuracy with state-of-the-art methods from sequences of 3D skeletons or color images. We can see that our approach only performs inline with the state-of-the-art method with the lowest accuracy, but can note that methods using only skeletal information perform worse. Among this category, the method presented by [19] achieves the largest accuracy. It relies on a stack of LSTM modules with specialized part-based cells that processes groups of body parts (arms, torso and legs). Such an explicit scheme could potentially improve our approach which is a simpler modeling of the overall body motion, especially given the size of the training set. The best performance overall is obtained by [16] which combines color images and skeleton modalities. In such case, including image context provides extra information that cannot be extracted from skeletal data, e.g. objects of interaction.

5. Conclusions

In this paper we addressed the problem of non-autoregressive human motion prediction with Transformers. We proposed to decode predictions in parallel from a query sequence generated in advance with elements from the input sequence. Additionally, we analyzed different settings to encode and decode pose embeddings. We leveraged the encoder memory embeddings to perform activity classification with an activity token. Our non-autogressive method

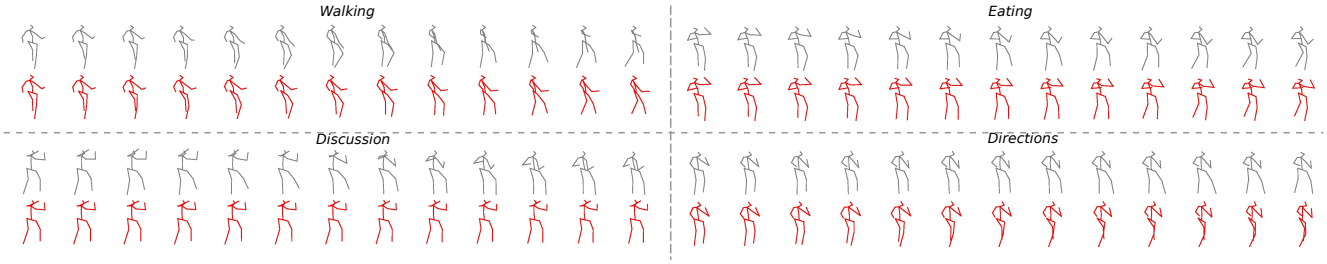

Figure 4: Qualitative results for the **H36M** dataset. We show results for four actions and show ground truth and predicted elements coloured in gray and red respectively.

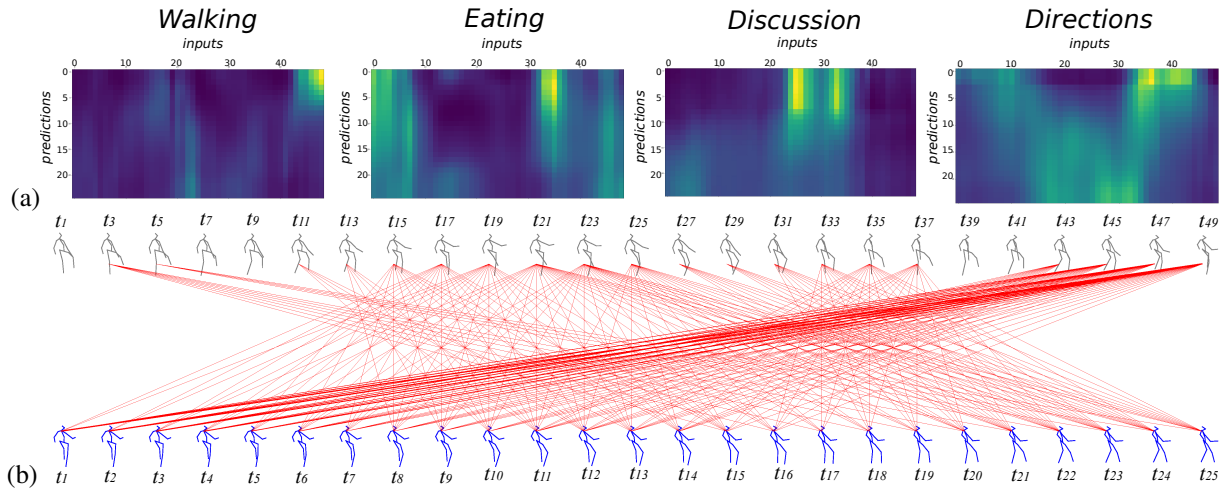

Figure 5: **H3.6M** datasest encoder-decoder attention weight visualization. (a) Raw encoder-decoder attention maps. Input and predicted entries are represented by columns and rows respectively. (b) Attention weights between input (gray) and predicted (blue) skeleton sequences of the *walking* action. Only weights larger than the median are visualized. The thickness of the lines are proportional to the attention weights. For visualization purposes we show only half of the input sequence;

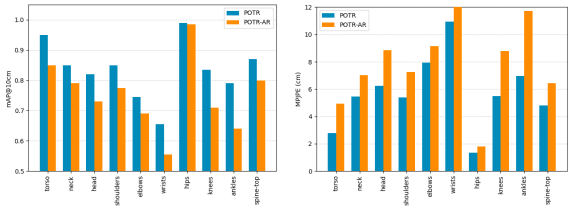

Figure 6: **NTU** per body part motion prediction performance in terms of (a) mAP@10cm. Higher is better and, (b) MPJPE. Lower is better.

Method	Skeletons	RGB	Accuracy
Skeletal quads [5]	√	-	38.62 %
2 Layer P-LSTM [19]	√	-	62.93 %
Multi-task [16]	√	√	**85.5** %
Multi-task [16]	-	√	84.6 %
Ours POTR	√	-	38.0 %
Ours POTR (memory)	√	-	30.0 %

Table 5: Activity classification performance comparison with the state-of-the-art in the **NTU** dataset. We specify if methods work with skeleton sequences or color images.

outperforms its autoregressive version in long term horizons and is less computationally intensive. Finally, despite the simplicity of our approach we have obtained competitive results on motion prediction in two public datasets.

Our work opens the door for more research. One of the main drawbacks in our method is the increased error at long term horizons as a consequence of non-autoregressive decoding and relying in a single pose vector as query sequence. A more suitable strategy to explore would be to rely in a set of M query poses by sampling from the input or selected using the encoder self-attention embeddings by position modelling such as in [7].

Acknowledgments: This work was supported by the European Union under the EU Horizon 2020 Research and Innovation Action MuMMER (MultiModal Mall Entertainment Robot), project ID 688146, as well as the Mexican National Council for Science and Technology (CONACYT) under the PhD scholarships program.

References

[1] Emre Aksan, Peng Cao, Manuel Kaufmann, and Otmar Hilliges. Spatio-temporal transformer for 3d human motion prediction. *arXiv:2004.08692v2*, 2021. 2, 7

[2] Emre Aksan, Manuel Kaufmann, and Otmar Hilliges. Structured prediction helps 3d human motion modelling. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019. First two authors contributed equally. 1, 2, 5, 7

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 3

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 5

[5] Georgios Evangelidis, Gurkirt Singh, and Radu Horaud. Skeletal quads: Human action recognition using joint quadruples. In *2014 22nd International Conference on Pattern Recognition*, pages 4513–4518, 2014. 8

[6] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, page 4346–4354, USA, 2015. IEEE Computer Society. 2

[7] Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K. Li, and Richard Socher. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*, 2018. 2, 4, 8

[8] Liang-Yan Gui, Yu-Xiong Wang, Xiaodan Liang, and Jose M. F. Moura. Adversarial geometry-aware human motion prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 2, 7

[9] Alejandro Hernandez, Jürgen Gall, and Francesc Moreno. Human motion prediction via spatio-temporal inpainting. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7133–7142, 2019. 2

[10] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural en vironments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, jul 2014. 5

[11] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020. 1

[12] Andreas M. Lehrmann, Peter V. Gehler, and Sebastian Nowozin. Efficient nonlinear markov models for human motion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1314–1321, 2014. 1

[13] Bin Li, Jian Tian, Zhongfei Zhang, Hailin Feng, and Xi Li. Multitask non-autoregressive model for human motion prediction. *IEEE Transactions on Image Processing*, 30:2562–2574, 2021. 3

[14] Chen Li, Zhen Zhang, Wee Sun Lee, and Gim Hee Lee. Convolutional sequence to sequence model for human dynamics. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 3

[15] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 5

[16] Diogo C. Luvizon, David Picard, and Hedi Tabia. 2d/3d pose estimation and action recognition using multitask deep learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5137–5146, 2018. 7, 8

[17] Julieta Martinez, Michael J. Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *CVPR*, 2017. 1, 2, 5, 7

[18] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. 2

[19] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2016. 5, 7, 8

[20] L. Sigal, M. Isard, H. Haussecker, and M. J. Black. Loose-limbed people: Estimating 3D human pose and motion using non-parametric belief propagation. *International Journal of Computer Vision*, 98(1):15–48, May 2011. 1

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 1, 2, 3, 4

[22] Mao Wei, Liu Miaomiao, and Salzemann Mathieu. History repeats itself: Human motion prediction via motion attention. In *European Conference on Computer Vision (ECCV)*, 2020. 2, 4, 5, 7

[23] Mao Wei, Liu Miaomiao, Salzemann Mathieu, and Li Hongdong. Learning trajectory dependencies for human motion prediction. In *ICCV*, 2019. 3, 5