DE-TGN: Uncertainty-Aware Human Motion Forecasting using Deep Ensembles

Kareem A. Eltouny, Wansong Liu, Sibo Tian, Minghui Zheng, *Member, IEEE*, and Xiao Liang, *Member, IEEE*

***Abstract*—Ensuring the safety of human workers in a collaborative environment with robots is of utmost importance. Although accurate pose prediction models can help prevent collisions between human workers and robots, they are still susceptible to critical errors. In this study, we propose a novel approach called deep ensembles of temporal graph neural networks (DE-TGN) that not only accurately forecast human motion but also provide a measure of prediction uncertainty. By leveraging deep ensembles and employing stochastic Monte-Carlo dropout sampling, we construct a volumetric field representing a range of potential future human poses based on covariance ellipsoids. To validate our framework, we conducted experiments using three motion capture datasets including Human3.6M, and two human-robot interaction scenarios, achieving state-of-the-art prediction error. Moreover, we discovered that deep ensembles not only enable us to quantify uncertainty but also improve the accuracy of our predictions.**

***Index Terms*—Human Motion Prediction, Deep Learning, Deep Ensembles, Human-Robot Collaboration (HRC)**

I. INTRODUCTION

THE integration of automated robots into various industries has revolutionized repetitive task execution. As the demand for environmentally conscious manufacturing grows, there has been a surge in research on human-robot collaboration (HRC) to address electronic waste management tasks [1, 2]. In an HRC environment, accurate human motion prediction plays a pivotal role in ensuring the safety of human workers. It empowers robots to anticipate human movement, enabling them to adjust their motion plans and avoid collisions [3]. Extensive studies have been conducted on 3D human motion forecasting, primarily leveraging motion capture technology. With the rapid advancements in artificial intelligence and its applications, machine learning methods have emerged for human motion prediction. These include recurrent neural networks (RNNs) [4, 5], convolutional neural networks (CNNs) [6-9], graph convolutional networks (GCNs) [10-13], and transformers [14]. While RNNs can struggle with computational demands and long-term forecasting tasks, CNNs are hampered by their limited receptive fields due to kernel sizes. Transformers present a promising solution for long-term forecasting, as they can process entire sequences, leveraging the self-attention mechanism to focus on relevant parts in the input sequence, regardless of their distance. Nevertheless, the computational and memory requirements of transformers for handling long input/output sequences can be significant compared to alternative methods. This inefficiency hampers their practicality for real-time applications [15].

Human motion is highly intricate, and accurately forecasting it entails dealing with a significant degree of uncertainty. In a collaborative robot setting, it is crucial for robots to recognize and account for such uncertain behaviors, allowing them to take appropriate actions when confidence level decreases. Several studies have been conducted to offer probabilistic outputs instead of deterministic ones, primarily through approximate variational inference and generative models [16-19]. However, conventional variational inference methods tend to generate samples from a local mode within the solution space, capturing only local uncertainty while often imposing training constraints such as prior distributions. To address these limitations, deep ensembles have emerged as a potential solution. Deep ensembles encompass a collection of deep learning models that generate samples derived from distinct training trajectories [20]. By leveraging this ensemble approach, deep ensembles tackle the issue of local uncertainty by providing a broader range of potential predictions.

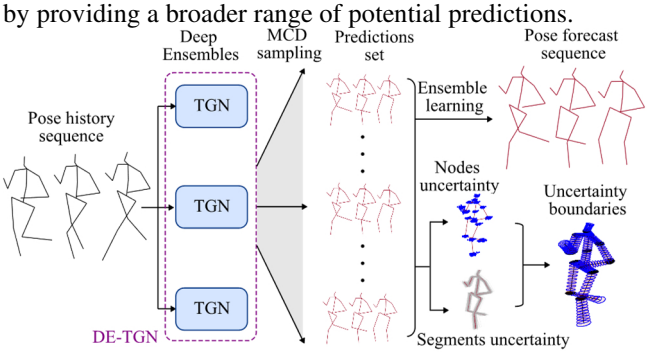

Fig. 1. Overview of the proposed deep ensembles of temporal graph neural networks.

In this study, we present an innovative approach called **D**eep **E**nsembles of **T**emporal-**G**raph **N**eural Networks (DE-TGN) for precise 3D human motion forecasting based on motion capture sequence data (Fig. 1). The contributions of this work are as follows: 1) We develop a novel deep learning architecture for human motion prediction, incorporating a blend of temporal convolutional networks (TCN) and graph

Manuscript received: July 5, 2023; Revised October 24, 2023; Accepted: January 2, 2024.

This paper was recommended for publication by Editor Gentiane Venture upon evaluation of the Associate Editor and Reviewers' comments.

This research was supported by the National Science Foundation under Grant 2026533. (Corresponding authors: M. Zheng and X. Liang.)

K. A. Eltouny and X. Liang are with the Civil, Structural and Environmental Engineering Department, University at Buffalo, Buffalo, NY 14260 USA (e-mail: keltouny@buffalo.edu; liangx@buffalo.edu).

W. Liu, S. Tian, and M. Zheng are with the Mechanical and Aerospace Engineering Department, University at Buffalo, Buffalo, NY 14260 USA (e-mail: wansongl@buffalo.edu; sibotian@buffalo.edu; mhzheng@buffalo.edu).

Digital Object Identifier (DOI): see top of this page.

attention networks (GAT) with residual connections. 2) We employ deep ensembles in combination with Monte Carlo (MC) dropout sampling [21] to generate a diverse set of plausible motions. Notably, this marks the first instance of utilizing deep ensembles for human motion forecasting in existing research. 3) We propose a technique to construct 3D uncertainty boundaries using covariance ellipsoids derived from the probabilistic output. These boundaries offer valuable insights into the reliability of the model's predictions in an HRC environment. 4) Our method surpasses state-of-the-art human motion prediction models in long-term human motion forecasting benchmarks, showcasing its superior performance.

II. Related Work

A. GCN

In the past decade, the field of human motion forecasting has been dominated by RNNs, with several groundbreaking RNN-based methods proposed [4, 5]. However, RNN-based methods exhibited noticeable discontinuities at the beginning of the forecast. To address this issue, Martinez et al. [5] proposed a sequence-to-sequence model with residual connections which predicts velocities instead of poses. Despite these advancements, long-term predictions remain challenging for these methods due to their one-step-ahead prediction mode, leading to error accumulation and increased computational cost. Feedforward networks attempt to solve many of the inherited issues in RNN-based methods. Earlier methods, however, relied on the predefined human kinematic tree [6], overlooking the need for coordinated motion between the different body parts, even those that are distant. Some methods have turned to CNN architectures to address these limitations [7]. Nevertheless, the challenges persist due to the reliance on kernel size for the temporal receptive field and the treatment of data as an image-like structure when modeling human motion.

In recent years, there has been growing interest in using GCNs for human pose forecasting [10-13]. GCNs have shown promise in processing non-grid-like structures, such as the human pose, making them suitable for capturing inter-joint spatial correlations. Mao et al. [10] proposed a sequential, feed-forward network of GAT layers with fully connected graphs. This approach enables the learning of global spatial connectivity among joints through attention mechanisms in the trajectory space. In another study, Mao et al. [11] introduced motion attention layers to capture the similarity between the current motion and historical motion, resulting in more accurate predictions. To gain a deeper understanding of the spatiotemporal dynamics of joints, Sofianos et al. [12] proposed the use of depth-wise separable GCNs with trainable spatiotemporal adjacency matrices. Zhong et al. [13] took a mixture-of-experts approach in their GCN-based motion forecasting technique, where a gating network applies importance factors to a set of adjacency matrices. What distinguishes the GCN layers in our DE-TGN model from prior literature is the design of a GAT residual block. This block incorporates multiple GAT, normalization, dropout, and non-linear activation layers, along with a skip connection, making our model deeper and easier to train.

B. TCN

TCNs have gained attention as an efficient and effective alternative to RNN- and attention-based techniques for human motion forecasting, offering advantages such as reduced error accumulation and improved computational efficiency. However, the exploration of TCNs in this context has been limited compared to other time-series modeling methods. In a comparative study by Pavllo et al. [22], a GRU-based motion forecasting model was pitted against a WaveNet-based model, with the former demonstrating superior performance. Cui et al. [8] proposed a forecasting network consisting of GCN blocks that incorporated TCN layers to capture time dependencies. Li et al. [9] presented a similar approach but with the additional inclusion of a positional encoding module, allowing the network to predict action types alongside motion forecasting. Overall, despite the promising results and advantages offered by TCNs and dilated causal convolution in general, their applications in human motion forecasting remain relatively unexplored. Our TCN residual blocks are designed akin to TCN's original architecture with minor adjustments [23]. Although a handful of previous researchers have explored GCN-TCN hybrids [8, 9], our design is the first of its kind, to the best of our knowledge. It encompasses distinct stages of spatial feature learning employing GAT, succeeded by temporal feature learning through TCNs.

C. Probabilistic learning

Several studies have put forth generative methods for human motion forecasting that aim to provide probabilistic output, allowing for diverse predictions without compromising accuracy. Barsoum et al. [16] introduced HP-GAN, drawing inspiration from generative adversarial networks, which employs a sequence-to-sequence generator to predict a set of plausible human motion predictions. Aliakbarian et al. [17] noted that HP-GAN begins to disregard stochastic components the longer it is trained and proposed a recurrent-based conditional variational autoencoder (CVAE) with a mix-and-match strategy to address this issue. Another study by Yuan and Kitani [18] focused on diversifying generated samples and introduced the diversifying latent flows (DLow) sampling method, which utilizes a CVAE network. In a different approach, Salzmann et al. [19] proposed a typed graph-GRU hybrid to directly predict motion distributions, providing a probabilistic perspective. To the best of our knowledge, deep ensembles have not been investigated as a viable option for generating probabilistic output in human motion predictions. Furthermore, previous literature on probabilistic human motion forecasting did not emphasize the utilization of the probabilistic output to establish uncertainty boundaries for ensuring safe human-robot interactions.

III. Network Architecture

In this section, we introduce the TGN architecture (Fig. 2) along with the Bayesian inference approximation using deep ensembles. Let us define $X_{1:N} = [x_1, x_2, \dots, x_N]^T$ as the historical motion sequence consisting of N 3D human poses. If the collaborating robot's motion is available, we denote its

sequence of N 3D poses as $Y_{1:N} = [y_1, y_2, \dots, y_N]^T$. The vectors $x_i \in \mathbb{R}^{C_x}$ and $y_i \in \mathbb{R}^{C_y}$ contains C_x and C_y parameters, respectively, that describe the poses. We posit that incorporating the robot's motion history into the input data holds valuable information for the predictive model. Consequently, adding this information enhances the accuracy of the forecast predictions. The input to our network is the concatenation of these two sequences: $[X_{1:N}, Y_{1:N}]$. Our goal is to provide a forecast of the human motion poses for T time steps, represented by the sequence $X_{N+1:N+T}$. We propose TGN, a TCN-GAT hybrid network, to predict the future sequence based on the provided input. To provide a measure of uncertainty, we rely on deep ensembles and MC dropout sampling to obtain a diverse set of predictions.

A. GAT

GCNs are types of neural networks specifically designed to handle graph-structured data. Unlike CNNs, which operate on grid-like structured data with fixed local connectivity, GCNs consider varying connections for each node and its neighbors in the graph, as defined by an adjacency matrix. Among GCNs, GAT stands out as it utilizes self-attention mechanisms to assign varying importance to neighboring nodes, thereby adjusting the adjacency matrix [24]. In our model, we employ a GAT encoder to extract representative features that capture the spatial relations between pose nodes. Inspired by Mao et al. [10], we establish full connectivity among all nodes in the graph, allowing GAT to adapt the connections based on the available training data. In our case, we assume that both human and robot nodes form a fully-connected graph with a total of $C = C_x + C_y$ nodes. The edges of this graph can be represented by an adjacency matrix, denoted as $\boldsymbol{A} \in \mathbb{R}^{C \times C}$. To transform the input into the trajectory space, we utilize the Discrete Cosine Transform (DCT). Consequently, each node is associated with a matrix $\boldsymbol{H} \in \mathbb{R}^{C \times F}$, where F represents the number of DCT coefficients. The graph convolutional layer estimates the output $\boldsymbol{H'}$ using the following formula, acting as input for the subsequent layer:

$$\boldsymbol{H'} = \sigma(\boldsymbol{A}\,\boldsymbol{H}\,\boldsymbol{W}) \tag{1}$$

where $\sigma(\cdot)$ is an activation function and $\boldsymbol{W} \in \mathbb{R}^{F \times F}$ is the trainable weight matrix. Using self-attention mechanisms, GAT applies attention weights to the entries in $\boldsymbol{A}$ resulting in a learned adjacency matrix $\boldsymbol{A}^*$ that replaces $\boldsymbol{A}$ in Eq. (1):

$$\boldsymbol{A}^* = \boldsymbol{\alpha} \cdot \boldsymbol{A} \tag{2}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{C \times C}$ contains the edgewise attention weights obtained by averaging the output of multi-head attention.

To incorporate GAT layers into our model, we adopt a residual architecture, as depicted in Fig. 2b. Within a GAT block, two GAT layers are used, each followed by layer normalization [25], rectified linear unit (ReLU) activation [26], and dropout regularization [27]. A skip connection is then employed to merge the input with the output by means of element-wise addition. This residual architecture enables the blocks to focus on learning the relative changes in the feature maps, rather than the entire transformations, which can facilitate deep learning.

B. TCN

TCNs are a specific type of CNNs designed to effectively handle sequential data [23]. Unlike RNNs, TCNs do not rely on recurrent connections to capture the temporal dependencies. Instead, they employ dilated causal convolution on the input sequence. This allows for the easy attainment of large receptive fields, making TCNs capable of efficiently processing very long sequences while mitigating the risk of vanishing gradients. For a one-dimensional sequence input $x \in \mathbb{R}^N$ and a kernel $k: \{0, \dots, w-1\} \to \mathbb{R}$, the output $F(t)$ of a dilated convolution operation at step t is defined by:

$$F(t) = \sum_{i=0}^{w-1} k(i)\,x(t - d \cdot i) + b \tag{3}$$

where w represents the kernel size, d is the dilation factor, and b is the bias term. The dilation factor is often chosen with a base (e.g., 2) that doubles as the network gets deeper. Increasing the dilation factor d and the kernel size w in Equation (3) allows for the expansion of the receptive field.

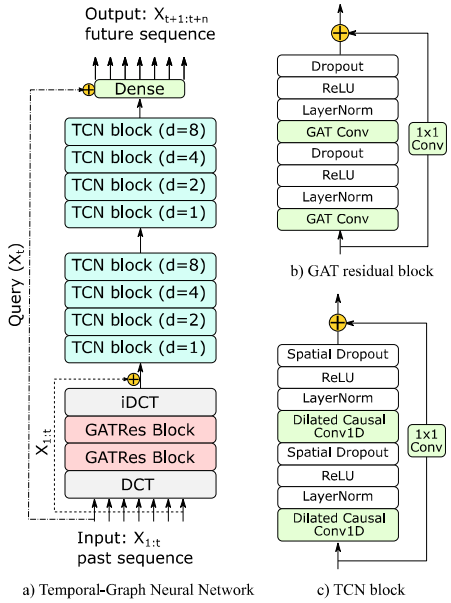

Fig. 2. Temporal graph neural network architecture.

Similar to the GAT residual blocks, the TCN modules in our model utilize a residual architecture, as illustrated in Fig. 2c. Each TCN block contains two sets of 1D dilated causal convolution layers, each followed by layer normalization, ReLU activation, and spatial dropout regularization [28]. Finally, the receptive field r of n successive TCN blocks can be defined as:

$$r = 1 + 2 \cdot (w - 1) \cdot \sum_{i=0}^{n-1} d^i \tag{4}$$

C. Combined Architecture

The TGN architecture facilitates spatio-temporal feature learning through the integration of three key modules: the GAT encoder, TCN encoder, and TCN decoder (Fig. 2a). The GAT encoder is dedicated to capturing the spatial features of poses and nodal connectivity, while the TCN encoder-decoder learns the temporal dependencies within sequences. First, the data undergoes DCT transformation and then passes through two GAT-Res blocks. Subsequently, the transformed data is converted back to the time domain using the inverse DCT operation. Both the TCN encoder and decoder consist of four TCN blocks, each corresponding to dilation values of 1, 2, 4,

and 8. The TCN blocks utilize a kernel size of 3. The decoder concludes with a fully-connected layer that generates output corresponding to the number of desired forecasting steps. Additionally, a global residual connection is established between the last input step and the output, allowing the model's output to represent the relative position with respect to a query, which is typically the last known position [5].

D. Deep Ensembles

Deep ensembles are machine learning techniques that entail training multiple neural networks with diverse initializations, architectures, or data subsets, and amalgamating their predictions to enhance accuracy [20]. This ensemble strategy enhances model robustness and uncertainty estimation by mitigating the influence of random initialization and optimization on the model's performance. Deep ensembles offer advantages over traditional Bayesian neural networks as they are easier to implement, require fewer computational resources, and involve minimal hyperparameter tuning. Deep ensembles are believed to excel because they can sample from distinct functions or modes within the function (solution) space, a capability not shared by variational methods, as illustrated in Fig. 3 [29]. In addition, subsampling techniques may sample from a local optimum based on the training loss, but there is no guarantee that it corresponds to a local optimum of the validation loss. Deep ensembles have been shown to be effective across various applications, including image classification, natural language processing, and time-series forecasting [30].

In our model, we utilize deep ensembles to enhance prediction accuracy and quantify uncertainty. Specifically, we train three models with the same architecture but different parameter initializations. The predictions for future poses are obtained by averaging the node-wise outputs of these models. Although this may increase training time and memory requirements, there is minimal to no increase in inference time compared to variational inference methods. To quantify uncertainty, we combine deep ensembles with MC dropout sampling. This combination allows for an increased number of samples, enabling the construction of more robust distributions without significant computational overhead.

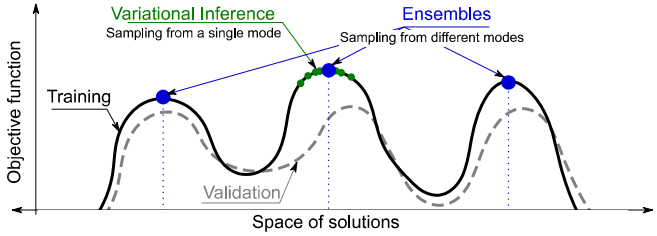

Fig. 3. The unique functions sampling hypothesis.

IV. Uncertainty Boundary

The stochastic output generated by the deep ensembles and the MC dropout sampling offers various possibilities for creating boundaries around pose estimates to indicate prediction uncertainty. In this section, we present an example of estimating uncertainty boundaries, which consists of two parts: 1) estimating uncertainty around the joints, and 2) estimating uncertainty along the segments.

A. Joints Uncertainty

To establish an uncertainty boundary around the estimated joint position in an ensemble of predictions, we construct a covariance (error) ellipsoid. After performing an eigenvalue decomposition of a joint's position vector, we derive three uncorrelated principal axes that represent the joint's position. By treating the position vector components along these axes as random variables, following a Gaussian distribution, we can construct a confidence boundary using a three-degree-of-freedom Chi-square distribution.

Given the global joint position vector $P = \{x, y, z\}^T$, the local position vector for the same joint is represented by $P' = \{x', y', z'\}^T$ where x', y', and z' are positions along the joint principal axes. The equation of the error ellipsoid in the local coordinates can be expressed as:

$$\left(\frac{x'}{\lambda_1}\right)^2 + \left(\frac{y'}{\lambda_2}\right)^2 + \left(\frac{z'}{\lambda_3}\right)^2 = \chi^2_{3,\alpha} \tag{5}$$

where λ_1, λ_2, and λ_3 are the eigenvalues of the position vector ensemble, while $\chi^2_{3,\alpha}$ represents the third-degree Chi-square value at a significance level α. To determine if a point falls outside the error ellipsoid, the left-hand side must be greater than the right-hand side (the critical Chi-square value). In global coordinates, the general formulas for the error ellipsoid are as follows:

$$\begin{bmatrix} x(\theta,\phi) \\ y(\theta,\phi) \\ z(\theta,\phi) \end{bmatrix} = \sqrt{\chi^2_{3,\alpha}} \cdot \boldsymbol{V}\boldsymbol{\Lambda}^{1/2} \begin{bmatrix} cos(\theta)\sin(\phi) \\ sin(\theta)\,sin(\phi) \\ \cos(\phi) \end{bmatrix} \tag{6}$$

where θ and ϕ are the local azimuth and zenith, $\boldsymbol{V}$ is the eigenvectors matrix, and $\boldsymbol{\Lambda}^{1/2}$ is a diagonal matrix containing the square roots of the eigenvalues. The following general form can be used to determine if a point falls inside or outside the ellipsoid:

$$[x \quad y \quad z]\, \boldsymbol{V}\boldsymbol{\Lambda}^{-1/2}\boldsymbol{V}^T \begin{bmatrix} x \\ y \\ Z \end{bmatrix} = \chi^2_{3,\alpha} \tag{7}$$

where $\boldsymbol{\Lambda}^{-1/2}$ represents a diagonal matrix containing the reciprocals of the square root of the eigenvalues.

B. Segments Uncertainty

As our motion prediction model generates an ensemble of joint position predictions, connecting these joints according to the human kinematic tree results in a set of segment predictions in the 3D space. For a segment connecting two nodes, the model produces two groups of prediction points, one for each node. We construct the uncertainty boundary for the segment based on the mean segment connecting the mean points of the two groups, along with a dynamic 2D error ellipse that depends on the longitudinal position along the mean line.

There are various forms of the 3D line equation, including the symmetric form defined by three line parameters and a line intersect. The parameters can be obtained given two points lying on the line: $p_1 = [x_1, y_1, z_1]$ and $p_2 = [x_2, y_2, z_2]$. The symmetric form can be rearranged in vector form, expressing x and y as functions of z:

$$\begin{Bmatrix} x(z) \\ y(z) \end{Bmatrix} = \begin{bmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{bmatrix} \begin{Bmatrix} z \\ 1 \end{Bmatrix} \tag{9}$$

where $\beta_{11} = \frac{x_2 - x_1}{z_2 - z_1}, \beta_{12} = x_1 - \left(\frac{x_2 - x_1}{z_2 - z_1}\right) z_1, \beta_{21} = \frac{y_2 - y_1}{z_2 - z_1}$, and

$\beta_{22} = y_1 - \left(\frac{y_2 - y_1}{z_2 - z_1}\right) z_1$. We can define $P_h = [x(z), y(z)]^T$, $z_h = [z,\ 1]^T$, $\beta_1 = [\beta_{11}, \beta_{21}]^T$, $\beta_2 = [\beta_{12}, \beta_{22}]^T$, and then $\boldsymbol{B} = [\beta_1,\ \beta_2]$. Equation (9) can be rewritten as:

$$\boldsymbol{P}(z) = B z_h \tag{10}$$

Since we have an ensemble of point pairs, $\boldsymbol{B}$ and consequently $\boldsymbol{P}(z)$, contain random variables and possess a variance that we exploit to construct the segment uncertainty boundary. Taking the variance of Eq. (10):

$$Var(\boldsymbol{P}) = z_h^T Var(\boldsymbol{B}) z_h \tag{11}$$

where

$$Var(\boldsymbol{B}) = \begin{bmatrix} Var(\beta_1) & Cov(\beta_1, \beta_2) \\ Cov(\beta_2, \beta_1) & Var(\beta_2) \end{bmatrix} \tag{12}$$

is a symmetric 4×4 matrix. $Var(\beta_1)$ and $Var(\beta_2)$ are regular covariance matrices while $Cov(\beta_1, \beta_2) = Cov(\beta_2, \beta_1)^T$ represents the cross-covariance matrices of the two random vectors β_1 and β_2. With some rearrangement, Eq. (11) becomes:

$$Var(\boldsymbol{P}) = z^2 Var(\beta_1) + z\big(Cov(\beta_1, \beta_2) + Cov(\beta_2, \beta_1)\big) + Var(\beta_2) \tag{13}$$

which is the 2×2 covariance matrix of the points on the lines intersecting with the z plane. The covariance matrix can be used to construct a dynamic 2D error ellipse at any plane z and is defined as follows in local coordinates:

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \sqrt{\chi^2_{\alpha,n}} \cdot \boldsymbol{V} \boldsymbol{\Lambda}^{1/2} \begin{bmatrix} \cos(t) \\ \sin(t) \end{bmatrix} \tag{14}$$

where $t \in [0, 2\pi]$. The general form for testing is:

$$[x \quad y]. \boldsymbol{V}\, \boldsymbol{\Lambda}^{-1/2} \boldsymbol{V}^T \begin{bmatrix} x \\ y \end{bmatrix} = \chi^2_{\alpha,n} \tag{15}$$

The following formulas can be used to transform the positional vectors of the points from global to local coordinates at the z plane:

$$P_{local} = \boldsymbol{V}^{-1}(P_{global} - P_{origin}) \tag{16}$$

where P_{global} is the point in the global coordinates, while P_{origin} is the center of the error ellipse. The local axes are represented by three orthonormal dimensions, with the z-axis aligned with the mean segment's longitudinal dimension. Additionally, the local origin, P_{origin}, is chosen as the point on the mean segment at the plane of interest. The uncertainty boundary of the segments can be utilized to evaluate the proximity of robot segments by determining the points with the shortest distance between the robot and human segments.

C. Using the Uncertainty Boundary

The primary aim of quantifying uncertainty is to enhance the collision avoidance strategies employed by collaborative robots during trajectory planning. A major application of HRC is in assembly and disassembly processes. However, a crucial concern arises when robots operate in shared environments with human workers – ensuring the worker's safety is of utmost importance. By integrating an uncertainty-aware human motion prediction model, robots can utilize real-time motion planning methods to dynamically adjust their routes, effectively avoiding collisions and ensuring a safer working environment. In typical robot trajectory planning scenarios, optimization revolves around collision avoidance and efficiency, guided by a predetermined probability of collision. Fortunately, the uncertainty boundary estimation method detailed in this section allows for control over human motion prediction confidence through the adjustment of the significance level α. Consequently, the uncertainty boundary can function as a danger zone that robots must navigate around while executing their tasks.

V. EXPERIMENTS

To evaluate our model, we utilize three motion capture datasets: Human3.6M [31], the Arm Motion dataset [3], and the Reaching Motion dataset. We first provide an overview of these datasets, followed by details on the model implementation, evaluation metrics, and finally, the results.

A. Datasets

Human3.6M: Human3.6M is a widely used publicly available dataset for motion capture data, particularly for human pose forecasting. It comprises motion capture recordings of seven actors performing 15 different actions, such as walking, eating, and engaging in discussions. Each pose includes the 3D Cartesian coordinates of 32 joints. We consider 17 joints after excluding joints with constant readings or close proximity to others. Following the approach in the literature [5], we use subject 5 for testing and subject 11 for validation. The remaining subjects (1, 6-9) are used for training. Additionally, we remove the global rotations and translation from each sequence and downsample all motions to 25 frames per second.

Arm motion dataset: This dataset focuses on the arm motion of human workers who grasp and relocate screwdrivers while being captured by the Vicon camera system. Only the trajectories of three nodes representing the arm motion are recorded. Three types of motions are performed, resulting in a total of 429 trajectories captured at a frequency of 25 Hz. The data is split into training, validation, and test sets using a ratio of 75/12.5/12.5, respectively, for each motion type.

Reaching motion: In this dataset, a human worker attempts to collect screws from different locations while a robot is moving in the shared space. This scenario represents an HRC environment and introduces complexities such as collision risks. The dataset includes 463 motion sequences recorded in 3D Cartesian coordinates, comprising six worker arm nodes and eight robotic arm nodes. The data is split into training, validation, and test sets using a ratio of 80/10/10, respectively.

B. Evaluation Metrics

We employ the mean per joint position error (MPJPE) to assess prediction accuracy, a metric suitable for motion datasets represented in 3D Cartesian coordinates [31]. We also rely on three commonly used metrics to evaluate both the diversity and accuracy of the probabilistic output [18]. For measuring sample diversity, we use the Average Pairwise Distance (APD), calculated as the average Euclidean distance between all pairs in the motion ensemble. The other two metrics employed are the Average Displacement Error (ADE) and the Final Displacement Error (FDE). ADE represents the lowest average Euclidean distance over all time steps between the ground truth motion forecast and the predicted samples, while FDE indicates the lowest Euclidean distance in the last time step between the ground truth and the predictions.

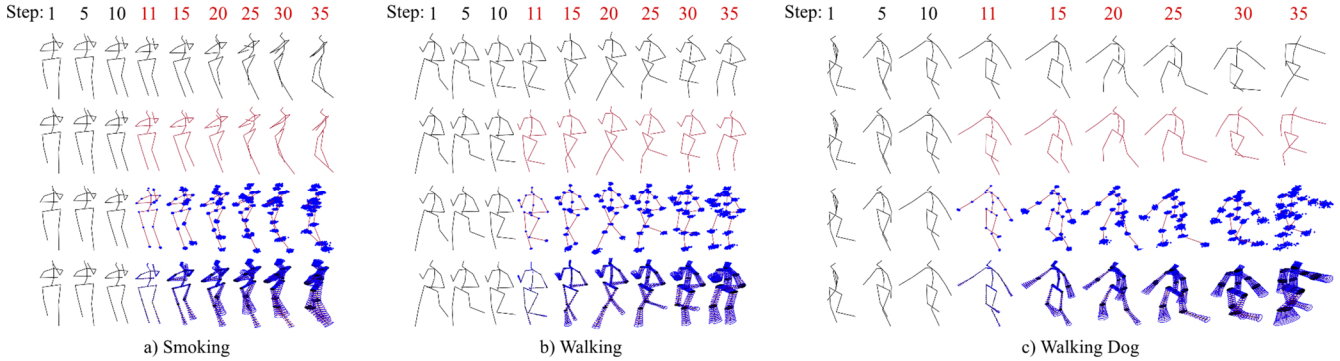

Fig. 4. Examples of DE-TGN 10-25 (1000 ms) predictions on Human3.6m including the uncertainty boundary. From top to bottom, we show the ground truth, DE-TGN mean predictions, DE-TGN + MC-dropout generated samples, constructed uncertainty boundary.

TABLE I. Human3.6M MPJPE values (mm) on the 15 action types at different forecasting steps for our proposed method (DE-TGN) trained on forecasting 10, 25, and 50 steps (400 ms, 1000 ms, and 2000 ms). Results of other methods in the literature are also provided (reported from [11] and [13]). *Our models.

	Directions						**Discussion**						**Eating**						**Greeting**					
milliseconds	80	160	320	400	1000	2000	80	160	320	400	1000	2000	80	160	320	400	1000	2000	80	160	320	400	1000	2000
Res-GRU 25-25 [5]	21.6	41.3	72.1	84.1	129		25.7	47.8	80.0	91.3	132		16.8	31.5	53.5	61.7	98.0		31.2	58.4	96.3	109	154	
ConvS2S 50-25 [7]	13.5	29.0	57.6	69.7	116		17.1	34.5	64.8	77.6	129		11.0	22.4	40.7	48.4	87.1		22.0	45.0	82.0	96.0	147	
LTD 10-25 [10]	9.2	20.6	46.9	58.8	109		12.2	25.8	53.9	66.7	119		7.7	15.8	30.5	37.6	74.1		16.7	33.9	67.5	81.6	140	
HRI 50-10 [11]	7.4	18.4	44.5	56.5	107		10.2	23.4	52.1	65.4	120		7.0	14.9	29.9	36.4	75.7		13.7	30.1	63.8	78.1	139	
GAGCN 10-25[13]	7.3	12.8	30.3	34.5	69.9		9.7	17.1	31.4	38.9	76.9		6.4	11.5	21.7	25.2	51.4		11.8	20.1	40.5	48.4	87.7	
DE-TGN 10-10*	**3.1**	**3.8**	**4.0**	**5.9**			**3.6**	**4.9**	**5.4**	**7.1**			**3.8**	**4.9**	**5.2**	**7.4**			**3.8**	**4.2**	**4.3**	**5.8**		
DE-TGN 10-25*	5.9	8.6	8.3	8.2	**12.9**		5.5	8.6	10.3	10.2	**14.9**		5.9	8.9	10.3	10.5	**15.4**		6.0	8.7	9.0	9.0	**13.5**	
DE-TGN 10-50*	8.0	12.9	15.4	15.4	13.8	**19.4**	7.6	12.7	15.7	16.6	16.0	**23.1**	7.6	12.7	15.7	16.0	18.4	**25.6**	8.8	14.3	15.5	14.9	16.4	**20.8**
	Phoning						**Posing**						**Purchases**						**Sitting**					
milliseconds	80	160	320	400	1000	2000	80	160	320	400	1000	2000	80	160	320	400	1000	2000	80	160	320	400	1000	2000
Res-GRU 25-25 [5]	21.1	38.9	66.0	76.4	126		29.3	56.1	98.3	114	183		28.7	52.4	86.9	1001	154		23.8	44.7	78.0	91.2	153	
ConvS2S 50-25 [7]	13.5	26.6	49.9	59.9	114		16.9	36.7	75.7	92.9	187		20.3	41.8	76.5	89.9	152		13.5	27.0	52.0	63.1	121	
LTD 10-25 [10]	10.2	20.2	40.9	50.9	105		12.5	27.5	62.5	79.6	172		15.5	32.3	63.6	77.3	136		10.4	21.4	45.4	57.3	119	
HRI 50-10 [11]	8.6	18.3	39.0	49.2	105		10.2	24.2	58.5	75.8	178		13.0	29.2	60.4	73.9	134		9.3	20.1	44.3	56.0	116	
GAGCN 10-25[13]	8.8	13.5	25.5	28.7	66.0		10.1	17.0	35.5	45.1	99.1		11.9	20.7	41.8	47.6	85.1		9.3	14.4	29.6	38.5	71.1	
DE-TGN 10-10*	**4.0**	**5.2**	**5.5**	**7.3**			**3.4**	**3.9**	**4.0**	**5.4**			**3.7**	**4.3**	**4.4**	**6.0**			**3.4**	**4.6**	**5.0**	**6.9**		
DE-TGN 10-25*	6.4	9.5	10.7	10.9	**16.4**		6.1	8.3	8.1	8.5	**12.9**		6.0	8.4	8.3	8.3	**13.2**		5.3	8.1	9.7	9.8	**15.3**	
DE-TGN 10-50*	8.6	14.2	17.1	17.3	19.0	**29.4**	8.9	14.1	15.0	14.7	14.4	**17.3**	10.8	15.9	16.6	17.3	16.4	**19.3**	7.6	12.4	16.6	17.0	17.0	**24.4**
	Sitting Down						**Smoking**						**Taking Photo**						**Waiting**					
milliseconds	80	160	320	400	1000	2000	80	160	320	400	1000	2000	80	160	320	400	1000	2000	80	160	320	400	1000	2000
Res-GRU 25-25 [5]	31.7	58.3	96.7	112	187		18.9	34.7	57.5	65.4	102		21.9	41.4	74.0	87.6	154		23.8	44.2	75.8	87.7	135	
ConvS2S 50-25 [7]	20.7	40.6	70.4	82.7	150		11.6	22.8	41.3	48.9	81.7		12.7	26.0	52.1	63.6	128		14.6	29.7	58.1	69.7	118	
LTD 10-25 [10]	17.0	33.4	61.6	74.4	144		8.4	16.8	32.5	39.5	73.6		9.9	20.5	43.8	55.2	120		10.5	21.6	45.9	57.1	107	
HRI 50-10 [11]	14.9	30.7	59.1	72.0	144		7.0	14.9	29.9	36.4	69.5		8.3	18.4	40.7	51.5	116		8.7	19.2	43.4	54.9	108	
GAGCN 10-25[13]	14.1	24.8	40.0	47.4	84.1		7.1	11.8	21.7	24.3	48.7		8.5	13.9	28.8	35.1	70.0		8.5	14.1	29.8	33.8	69.3	
DE-TGN 10-10*	**3.8**	**5.1**	**5.5**	**7.5**			**3.3**	**4.5**	**4.9**	**6.7**			**3.0**	**3.7**	**4.0**	**5.4**			**3.3**	**4.2**	**4.4**	**6.1**		
DE-TGN 10-25*	5.7	9.0	10.2	10.5	**17.3**		5.4	8.2	9.6	9.7	**14.6**		5.3	7.5	7.9	7.8	**11.5**		5.5	8.1	8.8	8.7	**13.7**	
DE-TGN 10-50*	8.4	14.0	17.7	18.4	18.1	**26.8**	7.2	11.8	14.6	14.8	17.1	**23.8**	7.4	12.0	13.5	13.0	13.0	**21.1**	8.2	13.7	15.9	15.4	16.3	**23.6**
	Walking						**Walking Dog**						**Walk Together**						**Average**					
milliseconds	80	160	320	400	1000	2000	80	160	320	400	1000	2000	80	160	320	400	1000	2000	80	160	320	400	1000	2000
Res-GRU 25-25 [5]	23.2	40.9	61.0	66.1	79.1		36.4	64.8	99.1	111	166		20.4	37.1	59.4	67.3	98.2		25.0	46.2	77.0	88.3	137	
ConvS2S 50-25 [7]	17.7	33.5	56.3	63.6	82.3		27.7	53.6	90.7	103	162		15.3	30.4	53.1	61.2	87.4		16.6	33.3	61.4	72.7	124	
LTD 10-25 [10]	12.6	23.6	39.4	44.5	60.9		22.9	43.5	74.5	86.4	142		10.8	21.7	39.6	47.0	65.7		12.4	25.2	49.9	60.9	113	
HRI 50-10 [11]	10.0	19.5	34.2	39.8	58.1		20.1	40.3	73.3	86.3	147		8.9	18.4	35.1	41.9	69.6		10.4	22.6	47.1	58.3	112	
GAGCN 10-25[13]	10.3	16.1	28.8	23.4	51.1		17.0	28.8	50.1	59.4	91.3		8.8	13.8	26.2	29.9	50.4		10.1	16.9	32.5	38.5	77.3	
DE-TGN 10-10*	**5.4**	**6.9**	**7.9**	**10.6**			**5.7**	**6.5**	**7.1**	**9.3**			**4.2**	**5.0**	**5.8**	**7.7**			**3.9**	**5.0**	**5.4**	**7.3**		
DE-TGN 10-25*	8.7	12.6	13.8	14.0	**21.6**		9.2	12.8	13.3	13.9	**21.8**		7.3	10.3	11.0	11.2	**17.1**		6.3	9.3	10.3	10.4	**16.0**	
DE-TGN 10-50*	12.2	19.9	23.1	22.7	24.2	**37.0**	14.3	23.0	21.9	21.6	23.7	**33.8**	9.9	15.6	16.5	15.9	18.2	**27.4**	8.9	14.5	17.1	17.2	18.0	**26.0**

C. Implementation

Using fixed-length windows, all motions are divided into segments. For Human3.6M and Arm Motion datasets, three separate experiments are conducted, each with a different output length. In all experiments, the input sequences (history) consist of 10 steps (400 ms), while the output sequences (forecasts) are 10, 25, and 50 steps long, respectively. An additional experiment on Human3.6M is conducted to evaluate the probabilistic performance of our model using ADP, ADE, and FDE metrics. In this experiment, the motion frame rate is set at 50Hz with a 25-step input sequence (500 ms) and 100-step output sequence (2000 ms), enabling a meaningful comparison with existing probabilistic modeling literature. Furthermore, eight experiments are conducted on the reaching motion dataset using 10- and 25-step input sequences with 25- and 50-step output sequences. These experiments are conducted both with and without including robot history motion as part of the input sequence.

D. Results

Human3.6M: Table I presents the MPJPE values for the three DE-TGN models. trained on the Human3.6M dataset. The values are provided for each of the 15 actions and different forecasting steps. Additionally, results from other models in the literature are included for comparison. Our proposed DE-TGN models outperform all other models across all actions. However, it is worth noting that models trained to produce long-term forecasts perform worse in shorter-term predictions compared to models focused on short-term forecasts (e.g., DE-TGN 10-50 vs. DE-TGN 10-10). As a result, there are a few instances where other models outperform our long-term model (DE-TGN 10-50) in the 80 milliseconds forecast range (e.g., Walk Together). This trade-off indicates that long-term forecast models sacrifice short-term forecast accuracy to achieve exceptional accuracy in long-term predictions. This observation is supported by significant improvements in long-term predictions when compared to state-of-the-art models. Fig. 4 showcases examples of DE-TGN predictions and the estimation of uncertainty boundary at multiple time steps.

Table II presents the average MPJPE values for each individually trained model in the deep ensembles, covering the 10, 25, and 50 steps variants. Notably, the deep ensembles technique achieves lower MPJPE values compared to all individual TGN models used to construct DE-TGN. Similarly, Table III demonstrates that deep ensembles offer enhanced diversity along with improved accuracy. This is evident from the higher APD values and the lower ADE and FDE values compared to all individual TGN models. In comparison to other studies, note that the existing methods in Table III are particularly designed to diversify the predictions, serving different purposes than DE-TGN. Therefore, our models exhibit lower APD values but notably superior ADE and FDE results. This aspect renders our models' predictions more accurate and realistic with a certain variation level, a crucial factor for HRC tasks.

TABLE II. Human3.6M average MPJPE values (mm) over all actions for individual TGNs and their deep ensembles. All models use 0.4s input.

Forecast length	400 ms	1000 ms	2000 ms
TGN #1	6.24	12.7	20.8
TGN #2	6.65	12.7	23.2
TGN #3	6.18	13.9	20.7
DE-TGN	**5.02**	**10.6**	**17.7**

TABLE III. Human3.6M APD, ADE, and FDE results (mm) for DE-TGN and exiting methods (reported from [18] and [19]).

Model	APD ↑	ADE ↓	FDE ↓
HP-GAN [16]	7214	858	867
MT-VAE [32]	403	457	595
DLow [18]	**11741**	425	518
Motron [19]	7168	375	488
TGN #1 MC-dropout*	1252	107	139
TGN #2 MC-dropout*	1210	99	128
TGN #3 MC-dropout*	1258	104	134
DE-TGN + MC-dropout*	1395	**98**	**122**

Arm Motion Dataset: The MPJPE results for the Arm Motion dataset are displayed in Table IV. For comparison, we include the results of a residual sequence-to-sequence (Seq2Seq) GRU-based model with input and output lengths of 25 steps each. The DE-TGN 10-25 variant not only outperforms the Seq2Seq model but also utilizes a shorter input length and requires fewer computational resources due to the efficiency of the convolutional layers. Moreover, the DE-TGN 10-50 variant achieves slightly improved predictions compared to the Seq2Seq model while offering double the forecast length. The results also demonstrate that deep ensembles reduce prediction errors in all models.

TABLE IV. Arm Motion Dataset average MPJPE values (mm) over all actions for individual models and their deep ensembles.

Model type	TGN 10-10	TGN 10-25	TGN 10-50	Seq2Seq 25-25
Model #1	1.94	4.21	7.83	8.27
Model #2	1.89	4.23	7.99	8.17
Model #3	1.91	4.18	7.99	8.24
Deep Ensembles	1.83	4.06	7.61	7.89

Reaching Motion Dataset: Table V presents the performance metric values for the eight DE-TGN models using the Reaching Motion dataset. Consistent with previous experiments, we observe enhancements in all metrics due to the utilization of the deep ensembles and an increase in errors as the forecast length grows. Moreover, Table V examines the impact of including robot motion in the input sequence. While there is no significant impact on the metrics when including robot motion in the 25-step output models, we note slight improvements in both accuracy- and diversity-based metrics for the 50-step output models. This suggests that the forecasting model could benefit from incorporating robot motion in long-term predictions. Our analysis also indicates that these improvements remain consistent even when increasing the input sequence length, as shown by the metrics differences observed in models with the same output length but varying input lengths, both with and without appending the robot history motion to the input. Fig. 5 showcases example predictions on the reaching motion dataset.

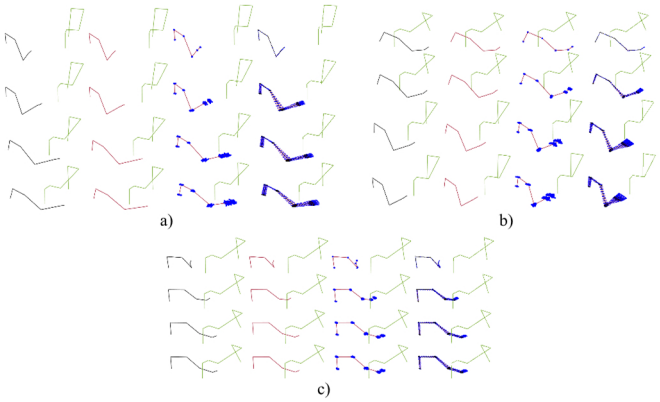

Fig. 5. Three Examples (a-c) of DE-TGN predictions for the Reaching Motion Dataset at time steps 11, 30, 50, and 60. From left to right, we show the ground truth, mean predictions, generated samples, and constructed uncertainty boundary. The robot arm is shown in green.

TABLE V. Reaching Motion Dataset evaluation metrics (mm) using multiple DE-TGN variants, with and without robot motion input.

Model type	Including robot input				Not including robot input			
	10-25	25-25	10-50	25-50	10-25	25-25	10-50	25-50
MPJPE ↓	7.18	7.08	16.62	16.45	7.22	7.2	17.54	17.16
APD ↑	91.1	92.8	259	251	89.9	96.4	278	256
ADE ↓	14.3	14.3	34.4	32.9	14.3	14.6	34.5	35.1
FDE ↓	25.6	23.9	53.7	48.7	26.1	24.5	54.1	49

V. Conclusion

This study highlights the advantages of employing deep ensembles for human motion forecasting. We have introduced the DE-TGN architecture, which surpasses the current state-of-the-art methods in human motion prediction for the Human3.6M benchmark, while also offering longer-term forecasts. Furthermore, our models have demonstrated low prediction errors in two HRC datasets that capture the motions of human workers engaged in collaborative tasks with a robotic arm. We have also proposed a statistical method for estimating uncertainty boundaries of human body nodes and segments utilizing deep ensembles and MC dropout sampling. Leveraging convolutional layers, our approach proves to be highly efficient compared to traditional sequence-to-sequence models.

By providing accurate predictions and assessing the reliability of the models through uncertainty estimation, our framework lays a solid foundation for safer HRC. Nevertheless, the method has several potential limitations. Deep ensembles might demand extra computational resources to achieve real-time performance comparable to individual models. Furthermore, this method was not evaluated in scenarios involving sensors and tracking errors (e.g., occlusions) and other environmental disturbances. It is advisable for future studies to address these limitations through further investigation.

References

[1] S. Sajedi, W. Liu, K. Eltouny, S. Behdad, M. Zheng, and X. Liang, "Uncertainty-assisted image-processing for human-robot close collaboration," *IEEE Robotics and Automation Letters,* vol. 7, no. 2, pp. 4236-4243, 2022.

[2] X. Zhang, K. Eltouny, X. Liang, and S. Behdad, "Automatic Screw Detection and Tool Recommendation System for Robotic Disassembly," *Journal of Manufacturing Science and Engineering,* vol. 145, no. 3, p. 031008, 2023.

[3] W. Liu, X. Liang, and M. Zheng, "Dynamic model informed human motion prediction based on unscented kalman filter," *IEEE/ASME Transactions on Mechatronics,* vol. 27, no. 6, pp. 5287-5295, 2022.

[4] K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik, "Recurrent network models for human dynamics," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4346-4354.

[5] J. Martinez, M. J. Black, and J. Romero, "On human motion prediction using recurrent neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2891-2900.

[6] J. Butepage, M. J. Black, D. Kragic, and H. Kjellstrom, "Deep representation learning for human motion prediction and classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 6158-6166.

[7] C. Li, Z. Zhang, W. S. Lee, and G. H. Lee, "Convolutional sequence to sequence model for human dynamics," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5226-5234.

[8] Q. Cui, H. Sun, and F. Yang, "Learning dynamic relationships for 3d human motion prediction," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6519-6527.

[9] B. Li, J. Tian, Z. Zhang, H. Feng, and X. Li, "Multitask non-autoregressive model for human motion prediction," *IEEE Transactions on Image Processing,* vol. 30, pp. 2562-2574, 2020.

[10] W. Mao, M. Liu, M. Salzmann, and H. Li, "Learning trajectory dependencies for human motion prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9489-9497.

[11] W. Mao, M. Liu, and M. Salzmann, "History repeats itself: Human motion prediction via motion attention," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, 2020: Springer, pp. 474-489.

[12] T. Sofianos, A. Sampieri, L. Franco, and F. Galasso, "Space-time-separable graph convolutional network for pose forecasting," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 11209-11218.

[13] C. Zhong, L. Hu, Z. Zhang, Y. Ye, and S. Xia, "Spatio-temporal gating-adjacency GCN for human motion prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6447-6456.

[14] E. Aksan, M. Kaufmann, P. Cao, and O. Hilliges, "A spatio-temporal transformer for 3d human motion prediction," in *2021 International Conference on 3D Vision (3DV)*, 2021: IEEE, pp. 565-574.

[15] H. Zhou *et al.*, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, 2021, vol. 35, no. 12, pp. 11106-11115.

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

[17] S. Aliakbarian, F. S. Saleh, M. Salzmann, L. Petersson, and S. Gould, "A stochastic conditioning scheme for diverse human motion prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5223-5232.

[18] Y. Yuan and K. Kitani, "Dlow: Diversifying latent flows for diverse human motion prediction," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, 2020: Springer, pp. 346-364.

[19] T. Salzmann, M. Pavone, and M. Ryll, "Motron: Multimodal probabilistic human motion forecasting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6457-6466.

[20] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in neural information processing systems,* vol. 30, 2017.

[21] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, 2016: PMLR, pp. 1050-1059.

[22] D. Pavllo, C. Feichtenhofer, M. Auli, and D. Grangier, "Modeling human motion with quaternion-based neural networks," *International Journal of Computer Vision,* vol. 128, pp. 855-872, 2020.

[23] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271,* 2018.

[24] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," in *International Conference on Learning Representation*, 2018.

[25] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer Normalization," in *Neural Information Processing Systems (NIPS)*, 2016.

[26] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *The 27th international conference on machine learning (ICML-10)*, 2010, pp. 807-814.

[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research,* vol. 15, no. 1, pp. 1929-1958, 2014.

[28] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 648-656.

[29] S. Fort, H. Hu, and B. Lakshminarayanan, "Deep ensembles: A loss landscape perspective," *arXiv preprint arXiv:1912.02757,* 2019.

[30] Y. Ovadia *et al.*, "Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift," *Advances in neural information processing systems,* vol. 32, 2019.

[31] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *IEEE transactions on pattern analysis and machine intelligence,* vol. 36, no. 7, pp. 1325-1339, 2013.

[32] X. Yan *et al.*, "MT-VAE: Learning motion transformations to generate multimodal human dynamics," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 265-281.