International Journal of Geo-Information

MDPI

Article

End-to-End Pedestrian Trajectory Forecasting with Transformer Network

Hai-Yan Yao [1,2,*], Wang-Gen Wan [1] and Xiang Li [1]

[1] School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China; wanwg@staff.shu.edu.cn (W.-G.W.); shdxlix@shu.edu.cn (X.L.)
[2] School of Electronic Information and Electrical Engineering, Anyang Institute of Technology, Anyang 455000, China
* Correspondence: yaohywqh@shu.edu.cn

Abstract: Analysis of pedestrians' motion is important to real-world applications in public scenes. Due to the complex temporal and spatial factors, trajectory prediction is a challenging task. With the development of attention mechanism recently, transformer network has been successfully applied in natural language processing, computer vision, and audio processing. We propose an end-to-end transformer network embedded with random deviation queries for pedestrian trajectory forecasting. The self-correcting scheme can enhance the robustness of the network. Moreover, we present a co-training strategy to improve the training effect. The whole scheme is trained collaboratively by the original loss and classification loss. Therefore, we also achieve more accurate prediction results. Experimental results on several datasets indicate the validity and robustness of the network. We achieve the best performance in individual forecasting and comparable results in social forecasting. Encouragingly, our approach achieves a new state of the art on the Hotel and Zara2 datasets compared with the social-based and individual-based approaches.

Keywords: trajectory forecasting; transformer; random deviation query

Citation: Yao, H.-Y.; Wan, W.-G.; Li, X. End-to-End Pedestrian Trajectory Forecasting with Transformer Network. *ISPRS Int. J. Geo-Inf.* **2022**, *11*, 44. https://doi.org/10.3390/ijgi11010044

Academic Editor: Wolfgang Kainz

Received: 19 November 2021
Accepted: 7 January 2022
Published: 9 January 2022

1. Introduction

Analysis of pedestrians' motion is one of the core problems for many autonomous systems in public scenes, such as surveillance, crowd simulation, mobile robot navigation, and autonomous driving. It is also essential for urban safety, and city planning. Trajectory prediction relies on the past observed human motion to predict the future locations of the pedestrians. Perceiving the crowd behavior and understanding the future behavior of agents are crucial abilities. Trajectory prediction is a challenging task. The pedestrian trajectory can be influenced by multiple factors, including individual moving style, the underlying destination, the motion of other agents, the environment topology structure, etc. Moreover, the agents velocity, the interactions with other moving pedestrians also affect the walking behavior of an individual. An efficient and effective end-to-end trainable framework is expected to improve pedestrian trajectory prediction performance.

Most current methods of pedestrian action prediction are sequence prediction. Since the Long-short term memory networks (LSTM) show the ability to learn and reproduce long sequences effectively, some LSTM-based approaches [1] are proposed to learn social behaviors. Then, some researchers integrate rich information into the standard LSTM, such as current intention of neighbors [2], group coherent motion patterns [3], scene information [4,5], high-level road-agent behavior [6], dynamic and static context-aware motion [7]. By adding this information to human movement trajectories in the prediction process, the performance can be improved significantly. In addition, some researchers try to use classical algorithms in AI-based methods. For example, combining Kalman filter algorithm and support vector machine algorithm to predict the trajectory prediction of fast flight ping-pong in the research of ping-pong robot [8], and using a Deep Kalman Filtering

Network (DKFN) for traffic prediction [9]. The results show that the prediction accuracy is obviously improved compared with the single algorithm.

This problem of trajectory prediction can be viewed as a sequence generation task, where researchers are interested in predicting the future trajectory of people based on their past positions. Some efforts [10–14] have been made to tackle social interactions and produce socially acceptable trajectories through generative adversarial networks (GANs). They introduce the adversarial loss along with sequence generation. Note that the notorious instability of training is a challenge, so integration needs to be done carefully. This strategy can improve the traditional trajectory prediction performance obviously.

Applying attention in sequence learning tasks has proved its effectiveness in the overall algorithm performance [11,15–17], and in pedestrian trajectory prediction methods [18–21]. It is helpful to draw more plausible trajectories. Some researchers [11,15,16] use the soft attention modules to evaluate social interactions. Fernando et al. [17] applied hard attention to assign weights based on pedestrians' distance, they also introduced additional soft attention to evaluate the interaction salience in a scene region. So, their trajectory prediction drew conclusions about which region was more likely for a pedestrian to navigate through. In the proposed work, we use the transformer network adopting an attention mechanism with Query-Key-Value (QKV) model to predict the pedestrian future location.

Inspired by the great success of transformer in natural language processing [18,19], computer vision [20,22,23], and audio processing [21,24,25], we aim to implement trajectory forecasting with an end-to-end transformer model. Thanks to their better capability to learn non-linear patterns, we argue that transformer networks are suitable for sequence modeling and trajectories forecasting. Giuliari et al. [26] utilize a simple Transformer model to achieve good performance on the single trajectory forecasting task. They feed the current and prior positions sequence to the encoder-decoder transformer network (TF), and predict the future track positions. However, the decoder is trained with ground truth, but tested with predicted positions. The predicted points are often biased, and affect the following prediction, resulting in inevitable error cascades. To overcome this problem, we propose a transformer model embedding the random deviation query. During training, disturbance error is embedded in the query, and the decoder is still required to predict accurate trajectory. In this way, the network has the ability of self-correction, alleviates the prediction deviation caused by error cascade, and improves the robustness.

In this work our main contributions are twofold:

- First, we propose an effective and end-to-end trainable framework built upon the transformer framework which is embedded with a random deviation query for trajectory forecasting. Taking advantage of the self-correcting ability introduced by the random deviation query, the robustness of the existing transformer network is enhanced. For detail, we design an attention mask to solve the assignment problems between parallel input queries and sequentially predict.
- Second, we present a co-training strategy based on a classification branch to improve the training effect. The whole scheme is trained collaboratively by the original loss and classification loss, which can improve the accuracy of the results. Experimental results compared with the state of the art methods show that the proposed method can predict plausible trajectory with higher accuracy.

The remainder of this paper is organized as follows. In Section 2, we briefly review the related work. Then the materials and methods are proposed in Section 3. Experimental results on datasets are presented in Section 4. Finally, the conclusion and prospect are presented in Section 5.

2. Related Work

Analysis of pedestrians' motion has been studied for many years. Some researchers have dedicated their works to model pedestrian dynamics [27,28], develop software for pedestrian trajectories extraction [29], and estimate pedestrian safety level [30]. We give a brief review of the literature on pedestrian trajectory forecasting, especially approaches

that compared with the proposed method. For the purpose of this paper, we distinguish two main trends for related work: social-based and attention-based trajectory forecasting.

social-based trajectory forecasting: Many existing methods employ social information into trajectory prediction. Early works include LSTM-based methods [1,2,4,5,7] that utilize LSTM network to learn context information. Then, a variety of social information are taken into account, such as current intention of neighbors [2,31], group coherent motion patterns [3], scene context [4,5,32,33], high-level road-agent behavior [6], dynamic and static context-aware motion [7], and social interactions modelling [34]. Then, following the success of the Generative Adversarial Network (GANs), some efforts [10–14] have been made to tackle social interactions. The social GAN model (abbreviated as SGAN) attemps to generate multiple trajectory predictions for each observed trajectory. Their network includes a pooling module to expand the neighborhood around each person of interest (POI) to cover the whole scene so all the pedestrians can be considered in the training and prediction processes. This effectively expands the local neighborhood context to a global level. Similarly, Sophie [11] uses GAN to generate multiple future paths for each trajectory.

attention-based trajectory forecasting: Attention-based models have been widely used in many tasks, such as natural language processing [18,19], computer vision [20,22,23], and audio processing [21,24,25]. In the area of trajectory prediction, attention mechanisms have been used to draw more plausible trajectories [11,15–17,35,36]. Sadeghian et al. [11] focus on the problem of wider scope: one that involves both pedestrians and vehicles. Giuliari et al. [26] utilize a simple transformer model to achieve good performance on a single trajectory forecasting task. They feed the current and prior positions sequence to the encoder-decoder transformer network (TF) and predict the future track positions. The SoPhie method [11] uses two attention modules to deal with scene context and social interactions.

Different from the methods reviewed above, an effective and end-to-end trainable transformer-based framework is used to predict the pedestrian trajectory. The temporal and spatial information of trajectories is embedded in queries. Moreover, embedding random deviation query is helpful for robustness and generalizability.

3. Materials and Methods

In this work, we address the problem of future positions prediction by processing their current and prior positions. Encoder-decoder transformer learns the pedestrian trajectory embeddings. The random deviation query and co-training strategy are beneficial to boost network performance. We will present the details of the proposed method in the following. The proposed approach is visualized in Figure 1. Symbols and their representations in this section are shown in Table 1.

3.1. Problem Formulation

Assume there are total of N pedestrians involved in a scene, and t is the current time stamp (frame). The trajectory of a pedestrian $P_i (i \in [1, N])$ from time stamp t_s to t_e is denoted as $T_i^{t_s:t_e} = [(x_i^{t_s}, y_i^{t_s}), \ldots, (x_i^{t_e}, y_i^{t_e})]$. The spatial location at time t of the pedestrian P_i is denoted as $p_i^t = (x_i^t, y_i^t)$. If we let 0 be the current time stamp, the current and prior positions observed in Cartesian coordinates are denoted as $\mathcal{T}_{obs} = T_i^{t_s:0}$, and the predicted positions are $\mathcal{T}_{pre} = T_i^{1:t_e}$. The proposed network aims to generate predicted trajectories $\hat{\mathcal{T}}_{pre}$ that match the ground truth future trajectories $\mathcal{T}_{gt}$.

We sequentially predict the points in each future frame. Since some works [37,38] have shown that the displacement prediction is easier in sequential estimation, therefore we predict the location displacement corresponding to a current frame for each pedestrian. A 2M-dimensional displacement vector $\mathcal{D}_{obs} = D_i^{t_s:0} = [p_i^{-(M-1)} - p_i^{-M}, p_i^{-(M-2)} - p_i^{-(M-1)}, \ldots, p_i^{0} - p_i^{-1}] \in \mathcal{R}^2$ is used to discribe pedestrian P_i 's walking path in the past M time stamps with respect to $t = 0$. When feeding to the transformer, the input vector is embedded onto a higher D-dimensional space by a linear projection with a matrix of

weights $e_{obs}^{(i,t)} = \mathcal{D}_{obs} W_i$. Similarly, the output of the transformer is back-projected to the Cartesian coordinates.

Table 1. Related mathematical symbols and their representation.

Symbol	Representation
t	time stamp (frame)
N	total number of pedestrians
P_i	the trajectory of a pedestrian ($i \in [1, N]$)
t_s, t_e	the start time stamp and end time stamp
$T_i^{t_s:t_e}$	the trajectory of a pedestrian $P_i (i \in [1, N])$ from t_s to t_e
$p_i{}^t = (x_i{}^t, y_i{}^t)$	the spatial location of P_i
$\mathcal{T}_{obs}$	observed trajectory
$\hat{\mathcal{T}}_{pre}$	predicted trajectory
$\mathcal{T}_{gt}$	ground truth future trajectories
$\mathcal{D}_{obs}$	displacement vector
W_i	weights matrix
$E_{obs}^{(i,t)}$	input embedding data
P^t	temporal position embedding
$e_{obs}^{(i,t)}$	spatial trajectory embedding
Q	query embedding inputs
K	key embedding inputs
V	value embedding inputs
W_i^Q, W_i^K, W_i^V, W^O	weight matrices in queries, keys, values, and output

Positional encoding: The transformer network discards the sequential nature of time-series data used in the LSTM-based model, but models temporal dependencies with the self-attention mechanism. Thus, the input embedding data $E_{obs}^{(i,t)}$ consists of spatial trajectory embedding $e_{obs}^{(i,t)}$ and temporal position embedding P^t. By following the same setting as in [18], the position embedding P^t is defined by sine and cosine functions. Each dimension of the positional encoding varies in time according to a sinusoid of different frequency, from 2π to $10{,}000 \cdot 2\pi$, which ensures a unique time stamp for the error.

$$
\begin{aligned}
E_{obs}^{(i,t)} &= e_{obs}^{(i,t)} + P^t \\
P^t &= p(t,d)_{d=1}^{D} \\
p(t,d) &= \begin{cases} sin(\frac{t}{10{,}000^{d/D}}) & \text{for } d \text{ even} \\ cos(\frac{t}{10{,}000^{d/D}}) & \text{for } d \text{ odd} \end{cases}
\end{aligned} \tag{1}
$$

Position embeddings are useful in our model, since they give the network a sense of which position of the sequence it is currently dealing with. There is no position information in self-attention. Through position embedding, each position has a unique position vector. In other words, each $e_{obs}^{(i,t)}$ appends a one-hot vector p^t.

3.2. Encoder-Decoder Transformer

Figure 1 shows the overall framework of the proposed approach for pedestrian trajectory prediction. In general, the framework is a transformer-based network assisted with random deviation queries and a classify branch to enhance performance. The observed pedestrian positions are fed to the network, and the network predicts the future trajectory. The detail information of the encoder and decoder in the transformer are also shown.

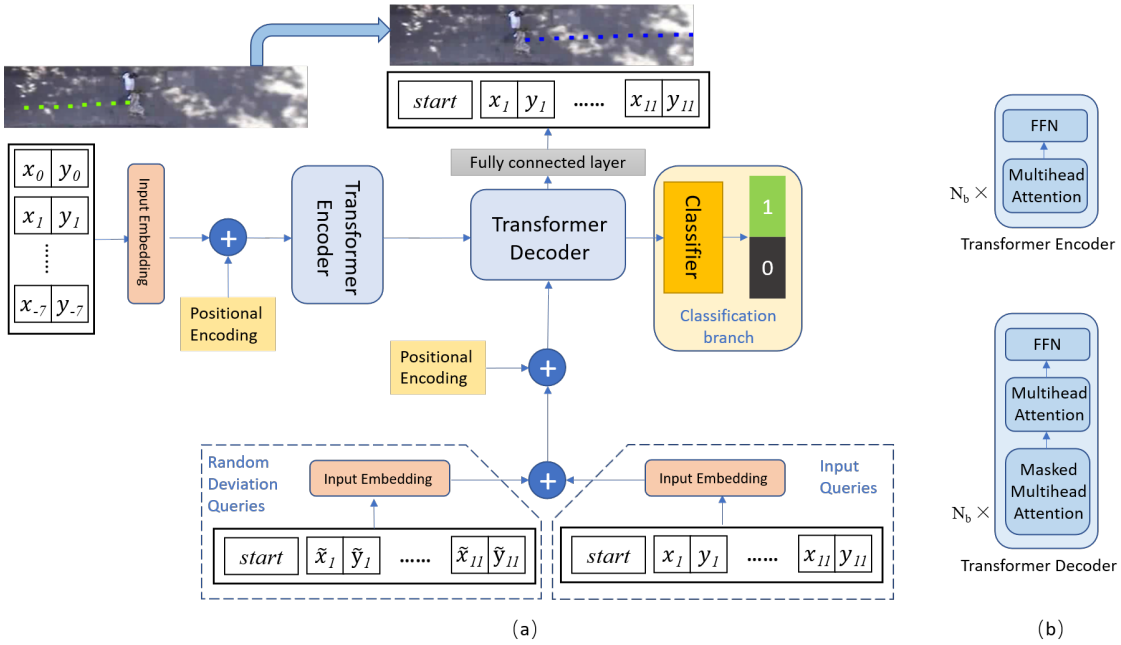

Figure 1. (**a**) The architecture of the overall framework of the proposed approach for pedestrian trajectory prediction. The framework is a transformer-based network assisted with random deviation queries and a classify branch to enhance performance. The observed pedestrian positions are fed to the network, and the network predicts the future trajectory. (**b**) The detail information of the encoder and decoder in the transformer.

Firstly, an end-to-end transformer is used as the base network to learn the function from prior trajectory sequence to future sequence. In this work, the encoder-decoder network is composed of six encoder blocks and six decoder blocks. The encoder block includes a multi-head attention and a feed forward network (FFN), and the decoder block contains two multi-head attention (one of which uses masked) and an FFN. Each is followed by a residual add to prevent degradation and a layer normalization to accelerate convergence.

The transformer is a mechanism in which a model learns to make predictions by selectively focusing on given data. While self-attention is an attention mechanism in which the model uses the observed portion of the sample and makes predictions of the remaining. The inputs of an attention module consist of Q (query embedding inputs), K (key embedding inputs) and V (value embedding inputs). Thus, the output is the weighted sum of the Value vectors, where the weight assigned to each V is determined by the scaled dot product of Q and the corresponding K.

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{2}$$

Multi-head attention is the core component of Transformer. Unlike simple attention, the multi-head mechanism breaks the input into many small chunks, computes the scaled dot product of each subspace in parallel, and finally concatenates all the attention output.

$$\begin{aligned} \text{MultiHead Attention}(Q, K, V) &= \text{concat}(head_1, \ldots, head_h)W^O \\ head_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{3}$$

where W_i^Q, W_i^K, W_i^V, W^O are the weight matrices in queries, keys, values, and output. All these weights can be trained.

In each sub-layer, multi-head attention has a feed forward network (FFN). The order in FFN is a linear transformation [18], Relu activation [39], and another linear transformation. The dropout is used to reduce overfitting, speed up training and enhance performance.

$$\text{FeedForward}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{4}$$

During the training procedure, the encoder is fed current and previous positions embeddings. Then the encoder outputs are passed to the decoder as memory to make an attention operation. In addition, the decoder is fed object queries which are referred to as position embedding of predicted position vectors, and the object queries are ground truth points in training. The decoder generates positions of future time stamps when testing.

In addition, we design an attention mask to prevent positions from attending to subsequent positions. This masking ensures that the predictions for position i can depend only on the known outputs at positions prior to i. The mask matrix $Mask_{i,j}$ is added to the softmax layer of self-attention in the decoder, i.e, $softmax(\frac{QK^T}{\sqrt{d_k}} + Mask_{i,j})V$.

$$Mask_{i,j} = \begin{cases} 0 & i \geqslant j \\ -\infty & \text{otherwise} \end{cases} \tag{5}$$

where $Mask_{i,j}$ determines whether the query q_i is prior to the query q_j. The queries after the current time stamp are masked so that the prediction is only based on the current and prior data.

3.3. Random Deviation Query

Encoder-decoder transformer learns the pedestrian trajectory embeddings. However, we observe that the output of the decoder block is not parallel in testing, but is predicted sequentially. Especially, the input of the decoder is ground truth in training, but predict queries with bias in testing. This deviation would cascade, leading to worsening predicted position along with the time stamp. Introducing appropriate deviation in training may improve the robustness and the performance of the network. With this motivation, random deviation queries is added as the input of the decoder for training to simulate the biased input. The random deviation query is defined as "deviation query + ground truth query". Even though one predicted point deviates from the ground truth, subsequent frames can still predict accurate position.

To improve the training effect, we also introduce a co-training strategy, which is based on the classification task. In the training process, the random deviation query containing real positions information can assist the decoder training. While in testing, the decoder input contains predicted position information which is deviated most of the time. Considering the decoder receives the predicted value rather than the ground truth in testing, a classification branch is added to judge whether the predicted sequence is accurate or not. As Figure 1 shows, a fully connected layer is set as a classifier.

The labels of the deviation query are defined based on the random deviation value. In the classification branch, the accuracy discrimination distance (ADD)is defined to measure the degree of deviation. $ADD = \sqrt{(\mathbf{x}_i{}^t - x_i{}^t)^2 + (\mathbf{y}_i{}^t - y_i{}^t)^2}$, where $\mathbf{x}_i$ and $\mathbf{y}_i$ are random deviation queries, $x_i{}^t$ and $y_i{}^t$ are ground truth queries. If the ADD is greater than the threshold value (which is set as 0.3), the corresponding label is 0. Similarly, the label is 1 when ADD is less than the threshold. The reason is that we assume the network can predict an accurate future trajectory. Theoretically, if the current position deviates far from the real point, then the next position predicted based on it would not be accurate, and vice versa.

3.4. Final Objective

For the loss calculation, the original trajectory loss L_p is the distance between the predicted position displacement $\hat{D}_i$ and the ground truth position displacement D_i at

time stamp t_j, i.e., $L_p = \frac{1}{M_p}\sum_{j=1}^{M_p} \text{distance}(D_i^{t_j}, \hat{D}_i^{t_j})$. M_p denotes the total frames of predictions. The classification loss is the cross-entropy of the output of the classifer $\hat{l}_i$ and the corresponding labels l_i, i.e., $L_{cls} = CrossEntropy(l_i, \hat{l}_i)$

The final loss for the network is defined as:

$$L_{final} = L_p + \lambda L_{cls} \tag{6}$$

where λ keeps the balance of the two objectives. λ is set as 50 for training.

4. Results

We evaluate the end-to-end transformer model on several trajectory forecasting datasets, and achieve comparable performance compared to methods in the literature. Implementation of the proposed model and its training are based on the PyTorch deep learning framework, using NVIDIA Geforce RTX 3090 GPU.

4.1. Experiment Setup

The detail information of the datasets and metrics are listed as follows:

- Datasets; Following the related prior research, we evaluate the proposed method on two public datasets: ETH [40] and UCY [41]. These datasets contain 5 video sequences (Hotel, ETH, UCY, ZARA1, and ZARA2) consisting of 1536 pedestrians in total with different movement patterns and social interactions. People walk in parallel, moving in groups, turning in the corner, avoiding collisions when they walk face-to-face. These are common scenarios that involve social behaviors. These sequences are recorded in 25 frames/second (fps) and contain 4 different scene backgrounds.
- Metrics; Average Displacement Error (ADE), mean square error overall estimated points in the predicted trajectory and ground-truth trajectory. Final Displacement Error (FDE), the distance between the predicted final destination and the ground-truth final destination. They can be mathematically defined as follows:

$$\begin{aligned} ADE &= \frac{1}{N}\sum_{i}\sum_{t=1}^{t_e} ||\mathcal{T}_{gt} - \hat{\mathcal{T}}_{pre}||_2 \\ FDE &= \frac{1}{N}\sum_{t=1}^{n} ||\mathcal{T}_{gt} - \hat{\mathcal{T}}_{pre}||_2 \end{aligned} \tag{7}$$

4.2. Experiment on ETH and UCY Dataset

First we compare the proposed methods to the state of the art methods on ETH and UCY datasets following the single trajectory deterministic protocol. Table 2 summarizes these results and shows that the proposed model achieves comparable performance. The results are separated into two categories: social version and individual version, listed in the top and bottom parts of the table, respectively (separated with a line). We achieve the best performance in the individual version (blue data). For the social version, comparable results are achieved. The average error result also indicates that adding more mapping information can significantly improve performance. Comparison with some approaches with a best-of-20 protocol [14] are shown in Table 3. Several models are trained, drawing 20 samples during both training and testing, and the best model is selected. So far, the best-of-20 protocol is a kind of upper-bound. The proposed method yields the best performance compared with other individual methods on ETH and UCY datasets (blue data). Furthermore, we achieve the best performance in the ADE and FDE on the Hotel dataset and FDE on Zara2 (black bold data), compared with the social-based and individual-based approaches. Figure 2 shows the qualitative comparison of the predicting results between the proposed method with TF [26], a classical transformer method. Some examples successfully predict the trajectories with small errors are shown on the first two rows. The last row

shows some suboptimal cases. For example, the pedestrian took a linear path while the real trajectory is curved. Even so, the proposed method predicts a plausible path.

We observe that most individual-based methods perform worse than social-based methods. The reason is that social information is considered, e.g., the influence of other agents. For example, when a pedestrian comes in the opposite direction or stops on the planning path, the original plan should be changed to avoid a collision. Besides, when traveling with a partner, the agent should keep the speed and distance constant. Another factor affecting performance is abnormal trajectory. For example, a pedestrian making a sudden turn. The observed information cannot reflect the pedestrian's intention to turn, but the phenomenon of turning in the prediction time stamp. The predictive function of the model is useless in this case, and the prediction path is still straight. The reasons are as follows: firstly, the destination of a pedestrian is unknown, future turns cannot be predicted according to the previous trajectory information. Secondly, the pedestrians' history behavior habits are unknown. If combining the historical track of the agents and the observed trajectories, accurate predictions may still be made even when there are turns or speed changes.

Table 2. Result comparison with the state of the art methods on ETH and UCY datasets following the single trajectory deterministic protocol. The social-based and individual-based methods are shown in the top and bottom parts of the table, respectively. We achieve the best performance in the individual version.

Method	Performance (ADE/FDE)					
	ETH	**Hotel**	**UCY**	**Zara1**	**Zara2**	**Average**
SGAN [14]	1.13/2.21	1.01/2.18	0.60/1.28	0.42/0.91	0.52/1.11	0.74/1.54
Social LSTM citeS-gan18	1.09/2.35	0.79/1.76	0.67/1.40	0.47/1.00	0.56/1.17	0.72/1.54
S-Attention [32]	0.39/3.74	0.29/2.64	0.20/0.52	0.30/2.13	0.33/3.92	0.30/2.35
Trajectron++ [33]	0.50/1.19	0.24/0.59	0.36/0.89	0.29/0.72	0.27/0.67	0.34/0.84
LSTM [14]	1.09/2.94	0.86/1.91	0.61/1.31	0.41/0.88	0.52/1.11	0.70/1.62
TF [26]	1.03/2.10	0.36/0.71	0.53/1.32	0.44/1.00	0.34/0.76	0.54/1.17
Ours	0.98/2.00	0.33/0.65	0.53/1.16	0.40/0.88	0.31/0.68	0.51/1.07

Table 3. Result comparison with the state of the art methods on ETH and UCY datasets following the best-of-20 protocol. The social-based and individual-based methods are shown in the top and bottom parts of the table, respectively. We achieve the best performance in the ADE and FDE on the Hotel dataset and FDE on Zara2, compared with the social-based and individual-based approaches.

Method	Performance (ADE/FDE)					
	ETH	**Hotel**	**UCY**	**Zara1**	**Zara2**	**Average**
SGAN [14]	0.87/1.62	0.67/1.37	0.76/1.52	0.35/0.68	0.42/0.84	0.61/1.21
Sophie [11]	0.70/1.43	0.76/1.67	0.54/1.24	0.30/0.63	0.38/0.78	0.54/1.15
Social-bigat [32]	0.69/1.29	0.49/1.01	0.55/1.32	0.30/0.62	0.36/0.75	0.48/1.00
Trajectron++ [33]	0.35/0.77	0.18/0.38	0.22/0.48	0.14/0.28	0.14/0.30	0.21/0.45
SGAN-ind [14]	0.81/1.52	0.72/1.61	0.60/1.26	0.34/0.69	0.42/0.84	0.58/1.18
TF [26]	0.61/1.12	0.18/0.30	0.35/0.65	0.22/0.38	0.17/0.32	0.31/0.55
Ours	0.49/0.82	**0.17**/**0.27**	0.34/0.61	0.22/0.38	**0.13**/**0.30**	0.27/0.48

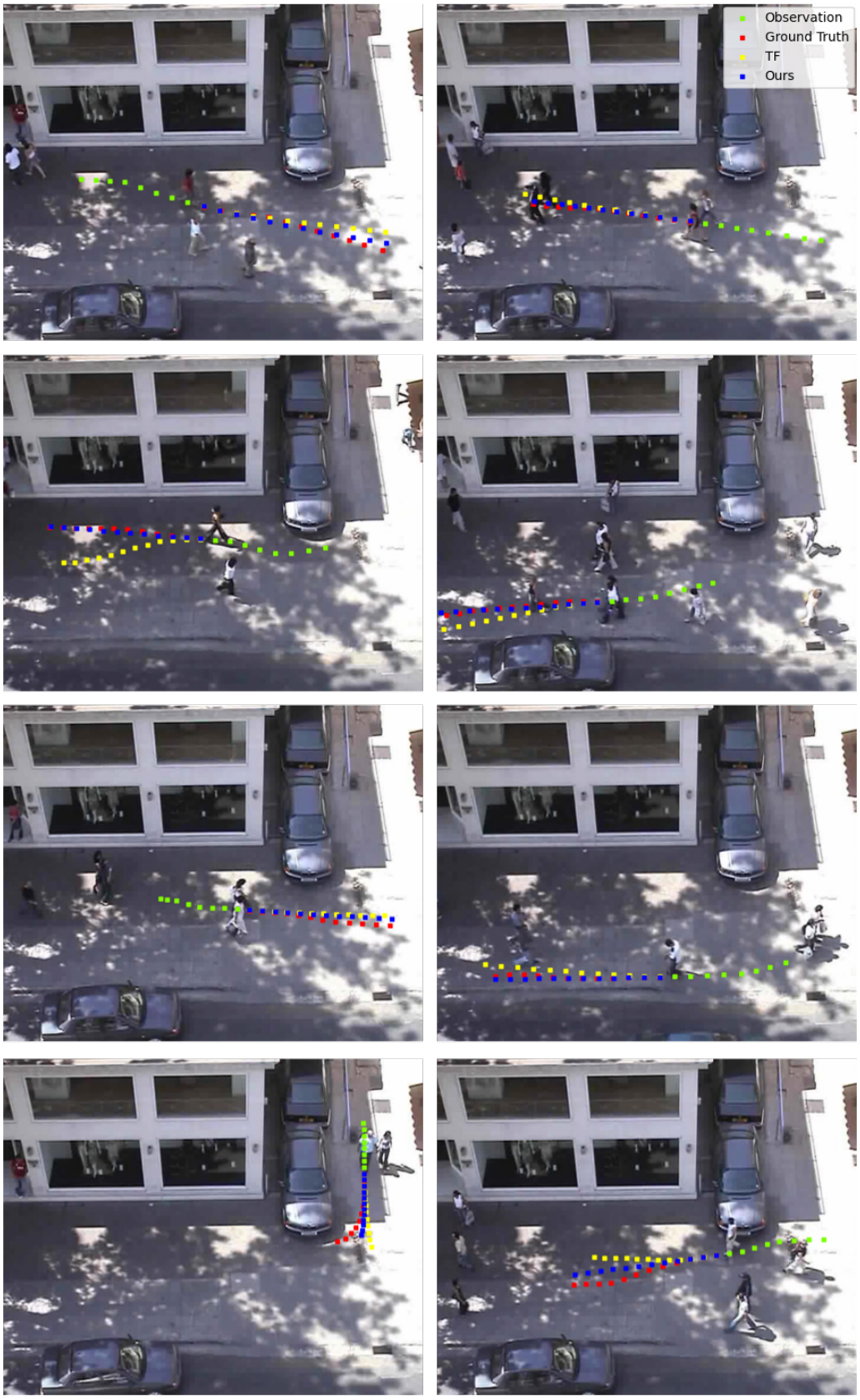

Figure 2. Qualitative comparison between the proposed method with TF predicting trajectories. The results are shown on Zara1 dataset. The first 3 rows show examples where the proposed method successfully predicts the trajectories with small errors. The last row shows some suboptimal cases, e.g., a person took a linear path. Even so, the proposed method predicts a plausible path.

4.3. Ablation Study

A number of ablation studies are performed to show the details of the proposed method. The results and discussions are shown following.

4.3.1. Effect on Different Numbers of Encoder-Decoder Blocks

Table 4 shows ADE and FDE results when the number of blocks in the encoder and decoder is changed. For the encoder, each block contains a multi-head attention module, an FFN, and two following residual connections. For the decoder, each block contains two multi-head attention (one of which uses masked), an FFN, and two following residual connections too. Since the encoder and decoder architectures play the same important role, we set the same numbers of blocks in both of them. We can see that with the increase of blocks, ADE and FDE gradually decreased. ADE and FDE tend to be stable when the number of blocks increases to 6. Thus, considering the balance between performance and computation, the number of blocks layers is set to 6 in the modular transformer. The comparison visualization results of average ADE and FDE can be seen in Figure 3.

Table 4. Result comparison on ETH and UCY datasets with different numbers of blocks N_b in encoder-decoder architecture. Considering the balance between performance and computation, the number of blocks layers is set to 6 in the modular transformer.

N_b	Performance (ADE/FDE)					
	ETH	**Hotel**	**UCY**	**Zara1**	**Zara2**	**Average**
2	1.021/2.092	0.336/0.645	0.554/1.224	0.418/0.927	0.329/0.746	0.532/1.127
4	1.023/2.121	0.348/0.668	0.543/1.199	0.414/0.916	0.321/0.717	0.530/1.125
6	0.987/2.005	0.337/0.650	0.537/1.168	0.404/0.886	0.311/0.688	0.515/1.079
8	0.976/1.953	0.338/0.654	0.535/1.156	0.408/0.898	0.314/0.687	0.514/1.070
10	0.981/1.916	0.316/0.593	0.540/1.164	0.406/0.886	0.321/0.711	0.513/1.054

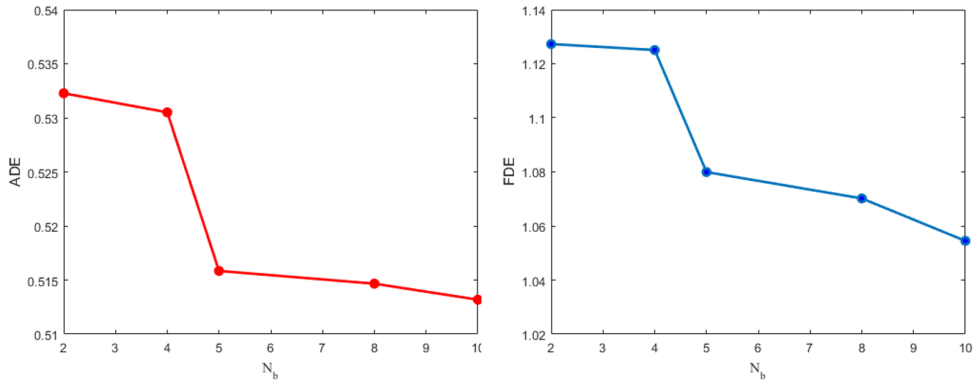

Figure 3. Results comparison of average ADE (**left**) and FDE (**right**) on different numbers of layers N_b. With the increase of blocks, ADE and FDE gradually decreased. ADE and FDE tend to be stable when N_b increases to 6. Thus, considering the balance between performance and computation, the number of blocks layers is set to 6 in the modular transformer.

4.3.2. Effect on Different Key Parameter λ

A comparison experiment is taken on ETH and UCY datasets to find the optimum value of super parameter λ, which is used to keep the balance of the loss. With the increase of λ, the performance keeps getting better steadily until up to 50 when starts to deteriorate. Especially when adding ADE and FDE to make the error superposition, the effect is more obvious (black bold data). Since the ADE and FDE are very small, adding them together can better reveal the differences. This means that a suitable equilibrium loss function can encourage the network to predict a more accurate trajectory. Although similar results can be achieved by other values, Table 5 shows that the parameter of 50 is often sufficient to achieve very good results on pedestrian forecasting. Eventually, the parameter λ is set to 50.

This result shows that it is necessary to choose a suitable key parameter, and it can balance the different loss functions in the training process. The comparison visualization results of average ADE and FDE can be seen in Figure 4.

Table 5. Result comparison on ETH and UCY datasets with different values of the key parameter λ, which is used to keep the balance of the loss functions. λ is set to 50 in training.

λ	Performance (ADE/FDE)						
	ETH	**Hotel**	**UCY**	**Zara1**	**Zara2**	**Average**	**ADE + FDE**
1	1.021/2.005	0.380/0.788	0.536/1.161	0.429/0.922	0.316/0.704	0.536/1.116	1.64
10	1.016/2.113	0.342/0.660	0.531/1.153	0.406/0.885	0.313/0.686	0.522/1.100	1.62
30	0.991/2.037	0.352/0.685	0.538/1.176	0.413/0.911	0.314/0.694	0.521/1.101	1.62
50	0.987/2.005	0.337/0.650	0.537/1.168	0.404/0.886	0.311/0.688	0.515/1.079	**1.58**
70	1.008/2.083	0.340/0.668	0.547/1.178	0.423/0.938	0.336/0.732	0.531/1.120	1.65
100	1.038/2.178	0.337/0.661	0.570/1.219	0.427/0.940	0.323/0.712	0.539/1.142	1.67

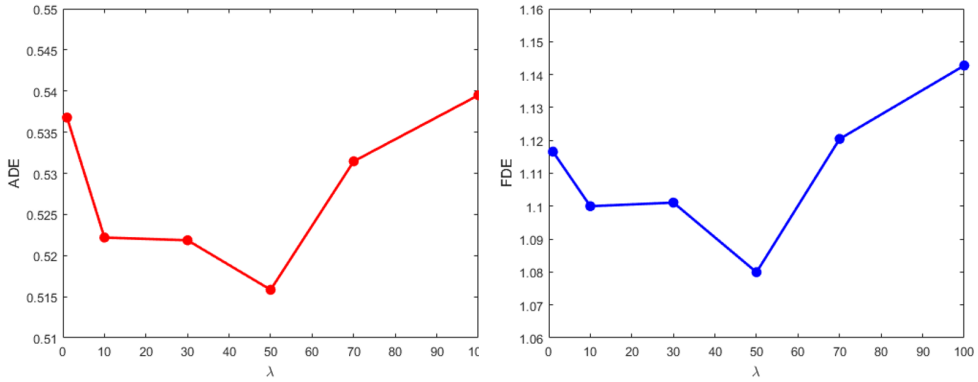

Figure 4. Results comparison of average ADE (**left**) and FDE (**right**) on different values of the key parameter λ. With the increase of λ, the performance keeps getting better steadily until up to 50 when starts to deteriorate. λ is set as 50, which is often sufficient to achieve very good results on pedestrian forecasting.

4.3.3. Effect on Different Accuracy Discrimination Distance

Accuracy discrimination distance (ADD) is defined to measure the degree of deviation. It is helpful to enhance the robustness of the network. If ADD is too large, the network would be insensitive to deviations, resulting in largely biased prediction. On the contrary, too small ADD invalidates the classifier branch, leading to deviation correction failure. The comparison results on ETH and UCY datasets with different ADD are shown in Table 6, and the visualization results of average ADE and FDE can be seen in Figure 5. To balance these two aspects, the threshold of ADD is set as 0.3 to pursue the best performance in average ADE and FDE.

Table 6. Result comparison on ETH and UCY datasets with different Accuracy Discrimination Distance ADD. The threshold of ADD is set as 0.3 to pursue the best performance in average ADE and FDE.

ADD	Performance (ADE/FDE)					
	ETH	**Hotel**	**UCY**	**Zara1**	**Zara2**	**Average**
0.01	1.029/2.099	0.350/0.698	0.532/1.158	0.464/1.028	0.329/0.721	0.541/1.141
0.03	1.007/2.040	0.344/0.679	0.538/1.178	0.471/1.031	0.315/0.696	0.535/1.125
0.05	1.036/2.097	0.374/0.770	0.533/1.171	0.400/0.878	0.332/0.738	0.535/1.131
0.1	1.002/2.016	0.358/0.702	0.527/1.148	0.402/0.880	0.319/0.697	0.522/1.089
0.3	0.987/2.005	0.337/0.650	0.537/1.168	0.404/0.886	0.311/0.688	**0.515/1.079**
0.5	0.998/2.024	0.337/0.651	0.540/1.174	0.405/0.888	0.321/0.721	0.520/1.092
0.7	1.037/2.153	0.332/0.646	0.581/1.237	0.410/0.905	0.330/0.740	0.538/1.136
1	1.033/2.141	0.346/0.677	0.569/1.246	0.413/0.917	0.325/0.726	0.537/1.141

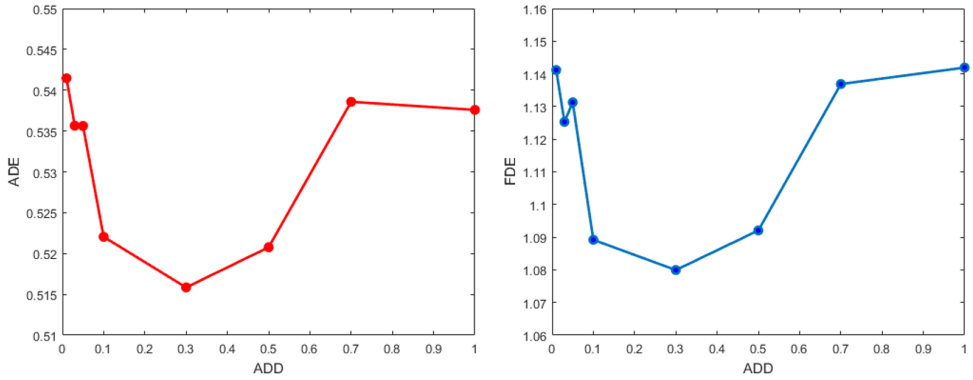

Figure 5. Results comparison of average ADE (**left**) and FDE (**right**) on different accuracy discrimination distance ADD. When ADD is 0.3, the network can achieve better performance, so ADD is set to 0.3.

5. Conclusions

We present an effective end-to-end transformer network for trajectory forecasting. The random deviation query is embedded in the classical transformer network and enhances the performance obviously. A co-training strategy based on a classification branch is utilized to improve the training effect. We achieve the best performance in individual forecasting and comparable results in social forecasting. Encouragingly, the proposed approach achieves a new state of the art on the Hotel and Zara2 datasets compared with the social-based and individual-based approaches.

In future work, we would like to apply social information and map information to improve performance. Moreover, this framework is not limited to trajectory forecasting. It can also be applied to more sequential prediction-related tasks.

Author Contributions: Conceptualization, Hai-Yan Yao and Wang-Gen Wan; methodology, Hai-Yan Yao; figures/tables preparation, Hai-Yan Yao and Xiang Li; data curation, Hai-Yan Yao and Xiang Li; writing—original draft preparation, Hai-Yan Yao and Xiang Li; writing—review and editing, Hai-Yan Yao; visualization, Hai-Yan Yao; supervision, Wang-Gen Wan. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by the International cooperation project of Shanghai Science and Technology Commission (Grant 18510760300) and Anyang science and technology program (grant No.2021C01SF052).

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Alahi, A.; Goel, K.; Ramanathan, V.; Robicquet, A.; Li, F.; Savarese, S. Social LSTM: Human Trajectory Prediction in Crowded Spaces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016.
2. Zhang, P.; Ouyang, W.; Zhang, P.; Xue, J.; Zheng, N. SR-LSTM: State Refinement for LSTM towards Pedestrian Trajectory Prediction. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019.
3. Bisagno, N.; Zhang, B.; Conci, N. *Group LSTM: Group Trajectory Prediction in Crowded Scenarios*; Springer: Cham, Switzerland, 2018.
4. Huynh, M.; Alaghband, G. Trajectory Prediction by Coupling Scene-LSTM with Human Movement LSTM. In Proceedings of the International Symposium on Visual Computing, Lake Tahoe, NV, USA, 7–9 October 2019.
5. Manh, H.; Alaghband, G. Scene-LSTM: A Model for Human Trajectory Prediction. *arXiv* **2018**, arXiv:1808.04018.
6. Chandra, R.; Guan, T.; Panuganti, S.; Mittal, T.; Bhattacharya, U.; Bera, A.; Manocha, D. Forecasting Trajectory and Behavior of Road-Agents Using Spectral Clustering in Graph-LSTMs. *arXiv* **2019**, arXiv:1912.01118.

7. Tao, C.; Jiang, Q.; Duan, L.; Luo, P. Dynamic and Static Context-aware LSTM for Multi-agent Motion Prediction. In Proceedings of the Europeon Conference on Computer Vision, Glasgow, UK, 23–28 August 2020.
8. Cheng, Q.; Wang, C. A Method of Trajectory Prediction Based on Kalman Filtering Algorithm and Support Vector Machine Algorithm. In Proceedings of the 2017 Chinese Intelligent Systems Conference (CISC), Mudanjiang, China, 14–15 October 2017; pp. 495–504.
9. Chen, F.; Chhen, Z.; Biswas, S.; Lei, S.; Ramakrishnan, N.; Lu, C. Graph Convolutional Networks with Kalman Filtering for Traffic Prediction. In Proceedings of the 28th International Conference on Advances in Geographic Information Systems (SIGSPATIAL), Seattle, WA, USA, 3–6 November 2020.
10. Dendorfer, P.; Ošep, A.; Leal-Taixé, L. Goal-GAN: Multimodal Trajectory Prediction Based on Goal Position Estimation. In Proceedings of the Asian Conference on Computer Vision, Kyoto, Japan, 30 November–4 December 2020.
11. Sadeghian, A.; Kosaraju, V.; Sadeghian, A.; Hirose, N.; Savarese, S. SoPhie: An Attentive GAN for Predicting Paths Compliant to Social and Physical Constraints. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
12. Fernando, T.; Denman, S.; Sridharan, S.; Fookes, C. GD-GAN: Generative Adversarial Networks for Trajectory Prediction and Group Detection in Crowds. In Proceedings of the Asian Conference on Computer Vision, Perth, Australia, 2–6 December 2018.
13. Javad, A.; Jean-Bernard, H.; Julien, P. Social Ways: Learning Multi-Modal Distributions of Pedestrian Trajectories with GANs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Long Beach, CA, USA, 16–20 June 2019.
14. Gupta, A.; Johnson, J.; Fei-Fei, L.; Savarese, S.; Alahi, A. Social gan: Socially acceptable trajectories with generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018.
15. Haddad, S.; Wu, M.; Wei, H.; Lam, S.K. Situation-Aware Pedestrian Trajectory Prediction with Spatio-Temporal Attention Model. In Proceedings of the 24th Computer Vision Winter Workshop (CVWW), Stift Vorau, Austria, 6–8 February 2019.
16. Yu, J.; Zhou, M.; Wang, X.; Pu, G.; Cheng, C.; Chen, B. A Dynamic and Static Context-Aware Attention Network for Trajectory Prediction. *ISPRS Int. J. Geo-Inf.* **2020**, *10*, 336. [CrossRef]
17. Fernando, T.; Denman, S.; Sridharan, S.; Fookes, C. Soft+ hardwired attention: An lstm framework for human trajectory prediction and abnormal event detection. *Neural Netw.* **2018**, *108*, 466–478. [CrossRef] [PubMed]
18. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, L.; Polosukhin, I. Attention Is All You Need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 6000–6010.
19. Fan, Z.; Gong, Y.; Liu, D.; Wei, Z.; Wang, S.; Jiao, J.; Duan, N.; Zhang, R.; Huang, X. Mask Attention Networks: Rethinking and Strengthen Transformer. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), Online, 6–11 June 2021; pp. 1692–1701.
20. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-End Object Detection with Transformers. In Proceedings of the Europeon Conference on Computer Vision (ECCV), Glasgow, UK, 23–28 August 2020; pp. 213–229.
21. Chen, X.; Wu, Y.; Wang, Z.; Liu, S.; Li, J. Developing Real-time Streaming Transformer Transducer for Speech Recognition on Large-scale Dataset. *arXiv* **2020**, arXiv:2010.11395.
22. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image is Worth 16 × 16 Words: Transformers for Image Recognition at Scale. *arXiv* **2020**, arXiv:2010.11929.
23. Parmar, N.; Vaswani, A.; Uszkoreit, J.; Kaiser, L.; Shazeer, N.; Ku, A.; Tran, D. Image Transformer. In Proceedings of the International Conference on Machine Learning (ICML), Stockholm, Sweden, 10–15 July 2018; pp. 4052–4061.
24. Dong, L.; Xu, S.; Xu, B. Speech-Transformer: A No-Recurrence Sequence-to-Sequence Model for Speech Recognition. In Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018; pp. 5884–5888. [CrossRef]
25. Gulati, A.; Qin, J.; Chiu, C.; Parmar, N.; Zhang, Y.; Yu, J.; Han, W.; Wang, S.; Zhang, Z.; Wu, Y.; et al. Conformer: Convolution-augmented Transformer for Speech Recognition. *Proc. Interspeech* **2020**, *2020*, 5036–5040. [CrossRef]
26. Giuliari, F.; Hasan, I.; Cristani, M.; Galasso, F. Transformer Networks for Trajectory Forecasting. In Proceedings of the 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021.
27. Seitz, M.J.; Dietrich, F.; Köster, G. The effect of stepping on pedestrian trajectories. *Phys. A Stat. Mech. Its Appl.* **2015**, *421*, 594–604. [CrossRef]
28. Caramuta, C.; Collodel, G.; Giacomini, C.; Gruden, C.; Longo, G.; Piccolotto, P. Survey of detection techniques, mathematical models and simulation software in pedestrian dynamics. *Transp. Res. Procedia* **2017**, *25*, 551–567. [CrossRef]
29. Boltes, M.; Seyfried, A. Collecting pedestrian trajectories. *Neurocomputing* **2013**, *100*, 127–133. [CrossRef]
30. Gruden, C.; Campisi, T.; Canale, A.; Tesoriere, G.; Sraml, M. A cross-study on video data gathering and microsimulation techniques to estimate pedestrian safety level in a confined space. *IOP Conf. Ser. Mater. Sci. Eng.* **2019**, *603*, 042008. [CrossRef]
31. Ma, W.C.; Huang, D.A.; Lee, N.; Kitani, K.M. Forecasting interactive dynamics of pedestrians with fictitious play. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
32. Kosaraju, V.; Sadeghian, A.; Martin, R.; Reid, I.; Rezatofighi, H.; Savarese, S. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. *arXiv* **2019**, arXiv:1907.03395.

33. Salzmann, T.; Ivanovic, B.; Chakravarty, P.; Pavone, M. Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control. In Proceedings of the Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16–18 June 2020.
34. Parth, K.; Kreiss, S.; Alahi, A. Human trajectory forecasting in crowds: A deep learning perspective. *IEEE Trans. Intell. Transp. Syst*. **2021**. [CrossRef]
35. Xue, H.; Huynh, D.Q.; Reynolds, M. A location-velocity-temporal attention LSTM model for pedestrian trajectory prediction. *IEEE Access* **2020**, *8*, 44576–44589. [CrossRef]
36. Yu, C.; Ma, X.; Ren, J.; Zhao, H.; Yi, S. Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In Proceedings of the European Conference on Computer Vision, Virtual, 23–28 August 2020.
37. Xu, Y.; Piao, Z.; Gao, S. Encoding Crowd Interaction with Deep Neural Network for Pedestrian Trajectory Prediction. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018.
38. Yi, S.; Li, H.; Wang, X. Understanding Pedestrian Behaviors from Stationary Crowd Groups. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015.
39. Glorot, X.; Bordes, A.; Bengio, Y. Deep Sparse Rectifier Neural Networks. In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), Ft. Lauderdale, FL, USA, 11–13 April 2011.
40. Pellegrini, S.; Ess, A.; Schindler, K.; Van Gool, L. You'll never walk alone: Modeling social behavior for multi-target tracking. In Proceedings of the 2009 IEEE 12th International Conference, Kyoto, Japan, 27 September–4 October 2009; pp. 261–268.
41. Lerner, A.; Chrysanthou, Y.; Lischinski, D. Crowds by example. In *Computer Graphics Forum*; Wiley: Hoboken, NJ, USA, 2007; Volume 26, pp. 655–664.