LG-Traj: LLM Guided Pedestrian Trajectory Prediction

Pranav Singh Chib[1] Pravendra Singh[1]
[1]Department of Computer Science and Engineering, IIT Roorkee, India
{pranavs_chib, pravendra.singh}@cs.iitr.ac.in

Abstract

Accurate pedestrian trajectory prediction is crucial for various applications, and it requires a deep understanding of pedestrian motion patterns in dynamic environments. However, existing pedestrian trajectory prediction methods still need more exploration to fully leverage these motion patterns. This paper aims to enhance pedestrian trajectory prediction tasks by leveraging Large Language Models (LLMs) to induce motion cues. We introduce LG-Traj, a novel approach incorporating LLMs to generate motion cues present in pedestrian past/observed trajectories. Our approach also incorporates motion cues present in pedestrian future trajectories by clustering future trajectories of training data using a mixture of Gaussians. These motion cues, along with pedestrian coordinates, facilitate a better understanding of the underlying representation. Furthermore, we utilize singular value decomposition to augment the observed trajectories, incorporating them into the model learning process to further enhance representation learning. Our method employs a transformer-based architecture comprising a motion encoder to model motion patterns and a social decoder to capture social interactions among pedestrians. We demonstrate the effectiveness of our approach on popular pedestrian trajectory prediction benchmarks, namely ETH-UCY and SDD, and present various ablation experiments to validate our approach.

1. Introduction

Trajectory prediction is the process of anticipating pedestrian's future motions based on their past motion. This task is crucial for self-driving cars, behavioural analysis, robot planning, and other autonomous systems. Accurately predicting pedestrian movements is crucial for the safety of autonomous driving systems. These systems must understand the movements of nearby pedestrians to navigate complex traffic environments safely and avoid collisions. When forecasting a pedestrian's future trajectory, a wide range of trajectories can be possible, and learning such varied spatio-temporal representations of trajectories is a major challenge in pedestrian trajectory prediction. The trajectories of individual pedestrians are influenced by various factors, including their inherent motion characteristics and the social interactions of neighbouring pedestrians. Previous works have employed recurrent networks [2, 46] to model the underlying motion pattern of each pedestrian. Additionally, graph-based methods [4, 39, 50] predict the motion pattern of agents using graph structures, where vertices represent pedestrians and edges represent their interactions. Generative-based approaches, such as GANs [18, 20, 47] and VAEs [27, 33, 59, 60], model the distribution of plausible future motion. The above mentioned methods leverage the spatio-temporal information from the given data to understand pedestrian motion dynamics. Building on these efforts, we explore a novel approach to improving the pedestrian trajectory prediction task by leveraging motion cues generated by LLM.

In this work, we present a novel approach called **L**LM **G**uided **Traj**ectory prediction (LG-Traj). Our approach effectively incorporates motion cues, along with the past observed trajectories, to predict future trajectories (see Fig. 1). We use a motion encoder to integrate spatio-temporal motion patterns and a social decoder to capture social interactions among pedestrians for accurate trajectory prediction. We utilize LLM to generate past motion cues present in pedestrian past trajectories. Additionally, we utilize future motion cues present in pedestrian future trajectories by clustering future trajectories of training data using a mixture of Gaussians. Specifically, past motion cues, past observed trajectory, and future motion cues are utilized by the motion encoder of the transformer to model the motion patterns (see Fig. 1). Furthermore, the social decoder of the transformer uses the social interactions of neighbouring pedestrians along with the embedding generated by the motion encoder to generate socially plausible future trajectories. Additionally, to effectively model the past trajectories, we augment the observed trajectories by singular value decomposition (SVD) and incorporate them into the training process to further enhance representation learning. Prior work, Eigentrajectory [7], also used SVD, which involves converting trajectories to the EigenTrajectory space, training the model

to predict future trajectories in the EigenTrajectory space, and then converting the future trajectories back to Euclidean space. In contrast, our approach uses SVD for a different purpose, i.e., to augment the observed trajectories, and performs better than other techniques used in the trajectory prediction task. The novelty of our work lies in integrating the motion cues generated by the LLM, which, in turn, significantly improves the prediction performance. Our extensive experimentation on popular pedestrian benchmark datasets, namely ETH-UCY and SDD, demonstrates the effectiveness of our proposed approach. We also present various ablation experiments to validate our approach.

2. Related Work

2.1. Trajectory Prediction

Trajectory prediction involves forecasting the trajectory at future timestamps given past observations. Since future states can evolve from the current state, sequence-to-sequence modelling approaches can be used to model these trajectory sequences. Recurrent Neural Networks (RNNs) [2, 22] and Long Short-Term Memory networks (LSTM) [19] have made significant progress in sequence prediction tasks. These architectures have been utilized to learn the temporal patterns of pedestrian trajectories. Moreover, LSTM networks [23, 53] construct spatio-temporal networks capable of representing structured sequence data. However, it is worth noting that RNN-based models may encounter issues like gradient vanishing or explosion under specific circumstances. Since the movement of pedestrians is uncertain, there may exist variations in future trajectories. To capture this variation in future trajectories, deep generative models such as Generative Adversarial Network (GAN) [18, 20, 47], Variational Auto-Encoder (VAE) [27, 33, 59, 60], normalizing flow [9], and diffusion-based models [17, 35] are used. Transformers demonstrated satisfactory performance in trajectory prediction [15, 62, 64, 65] and have been frequently used to model long-range relationships. Some work [31, 49] uses graph neural network-based techniques by building a graph structure containing pedestrian nodes and interaction edges for trajectory prediction. The graph-structured pedestrian characteristics are updated using transformers [16, 41, 55, 57, 63], graph convolutional networks [4, 31], and graph attention networks [6, 22, 25, 50]. Despite significant progress in trajectory prediction, as mentioned above, there is a need for further exploration to leverage motion cues effectively. Our approach leverages motion cues from the LLM to move forward in this direction.

2.2. Large Language Model

Large language models [1, 29, 52] have started to be used in scene understanding tasks, including object localization [12], scene captioning [3], and visual question answering [37, 51]. For example, in autonomous driving, DriveLike-Human [14] uses LLMs to create a new paradigm that mimics how humans learn to drive. Similarly, GPT-Driver [34] uses GPT-3.5 to improve autonomous driving with reliable motion planning. Parallel to this, SurrealDriver [24] builds an LLM-based driver agent with memory modules that mimic human driving behaviour to comprehend driving scenarios, make decisions, and carry out safe actions. ADAPT [3] offers explanations in driving captions to understand every stage of the decision-making process involved in autonomous vehicle control. To explore the capabilities of LLMs without explicit prompt engineering, Traj-LLM [26] uses Sparse Context Joint Encoding to encode the spatial-temporal scene input, such as agent states and lanes, into a format that LLMs can understand. LLM planning capabilities are also being used in robot navigation [21, 42], where natural language commands are translated into navigation goals. In pedestrian trajectory prediction, there has not been much effort to utilize the capabilities of large language models. Recently, LMTraj [8] proposed a language-based multimodal trajectory predictor that uses the language model as a numerical regressor to predict future trajectories directly. In contrast, we leverage the capabilities of LLM by training the trajectory prediction model with motion cues generated by LLM to understand the underlying trajectory motion patterns better.

3. Method

3.1. Problem Definition

Formally, the observation trajectory with length T_{ob} can be represented as $\boldsymbol{X}_i = \{(x_i^t, y_i^t) \mid t \in [1, \ldots, T_{\text{ob}}]\}$, where (x_i^t, y_i^t) is the spatial coordinate of a pedestrian i at t^{th} time. Furthermore, ground truth future trajectory for prediction time length T_{pred} can be defined as $\boldsymbol{Y}_i = \{(x_i^t, y_i^t) \mid t \in [T_{\text{ob}}+1, \ldots, T_{\text{pred}}]\}$. The number of pedestrians in the scene is represented by N, and $i \in N$ denotes the pedestrian index. The goal of trajectory prediction is to generate future trajectories $(\hat{\boldsymbol{Y}}_{\boldsymbol{i}})$ that closely approximate the ground truth trajectory $(\boldsymbol{Y}_{\boldsymbol{i}})$.

3.2. Trajectories Augmentation

We first construct the trajectory matrix X by stacking all pedestrian observations from a training batch. We then utilize Singular Value Decomposition (Eq. 1) followed by the rank-*k* approximation (Eq. 2) to obtain the augmented trajectories.

$$\mathrm{X} = \boldsymbol{U}_{ob} \boldsymbol{S}_{ob} \boldsymbol{V}_{ob}^{\top} \tag{1}$$

where $\boldsymbol{U}_{ob} = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_L]$ and $\boldsymbol{V}_{ob} = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_N]$ are orthogonal matrices and $\boldsymbol{S}_{\boldsymbol{ob}}$ is a diagonal matrix, consisting of singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$. Here, $L =$

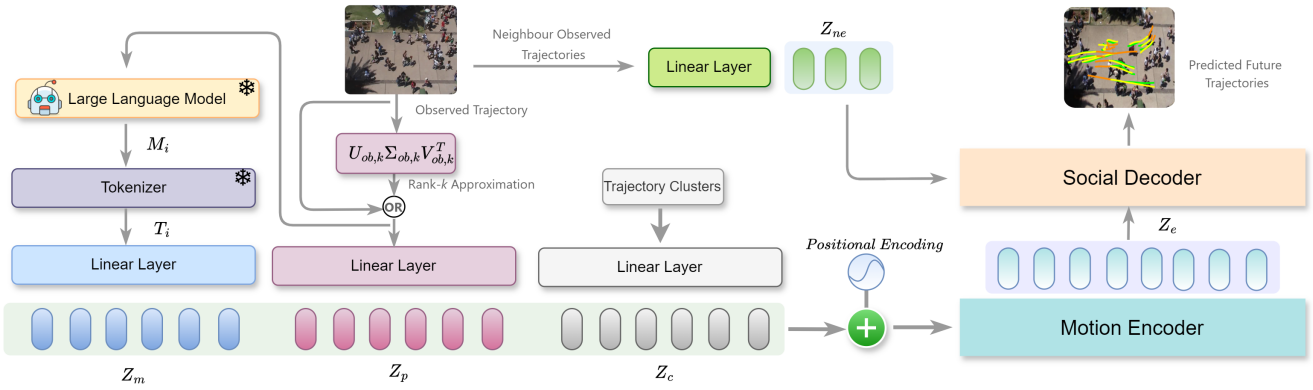

Figure 1. The overview of our proposed LG-Traj involves taking multiple inputs including past motion cues, past observed trajectory, and future motion cues. First, we augment the given observed trajectory using rank-k approximation via singular value decomposition (SVD). Then for the subsequent steps, we either use the original past observed trajectory or augmented past observed trajectory. Next, we generate past motion cues (M_i) from LLM using the past observed trajectory (X_i) of the i^{th} pedestrian. Tokenizer output (T_i) is generated from M_i by the tokenizer. Past motion cues embedding (Z_m) is obtained by a linear transformation of T_i. Past trajectory embedding (Z_p) is obtained by a linear transformation of X_i. Cluster embedding Z_c is obtained by a linear transformation of trajectory clusters. Trajectory clusters are generated by clustering future trajectories of training data using a mixture of Gaussians. Positional encoding is added to the concatenated embeddings (Z_m, Z_p, Z_c), and the result is passed as an input to the motion encoder to model the motion patterns. The embedding generated by the motion encoder (Z_e) along with neighbour embedding (Z_{ne}) is passed as an input to the social decoder to predict future trajectories. * represent frozen symbol for LLM and Tokenizer.

$2 \times T_{ob}$ for observed pedestrian trajectory. N is the number of pedestrians, and r is the rank of X. We use the rank-k approximation of X by using the first k singular vectors as shown below:

$$\tilde{\mathrm{X}} = \boldsymbol{U}_{ob,k}\, \boldsymbol{S}_{ob,k}\, \boldsymbol{V}^{\top}_{ob,k} \tag{2}$$

where $\boldsymbol{U}_{ob,k} = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_k]$, $\boldsymbol{S}_{ob,k} = \mathrm{diag}(\sigma_1, \ldots, \sigma_k)$, $\boldsymbol{V}_{ob,k} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]$. Also, $\tilde{\mathrm{X}}$ is the approximated matrices obtained from the best rank-k approximation of X, containing only the k most significant singular values. This approximation allows us to preserve critical information with minimal loss of information, as shown below:

$$\text{Information Loss} = 1 - \frac{\sum_{t=1}^{k} \sigma_t}{\sum_{t=1}^{r} \sigma_t} \tag{3}$$

Here, r is the total number of singular values, and σ_t represents the t^{th} singular value. We augment the given past observed trajectory using rank-k approximation via singular value decomposition (SVD). We also demonstrate experimentally how different k values impact information retention. We either use the original past observed trajectory or the augmented past observed trajectory, but not both simultaneously. Throughout the remainder of the paper, we exclusively present formulations using the original past observed trajectories for the sake of simplicity. However, it is important to note that the same formulation is applicable to augmented past observed trajectories.

3.3. Clustering using Mixture of Gaussians

We utilize the Gaussian Mixture Model to model the diverse future motion cues from the training data. Specifically, we construct mixed Gaussian clusters, which are a combination of $\mathcal{C}$ Gaussians ($\{\mathcal{N}(\mu_c, \sigma_c^2)\}, (1 \leq c \leq \mathcal{C})$). These multiple Gaussians represent different motion cues present in future trajectories (i.e., linear motion, curved motion, etc.). To generate clusters, we first preprocess the future trajectories by translating all starting coordinates to the origin, followed by rotation to the positive zero-degree direction. Then, we cluster similar motion behaviours of future trajectories into $\mathcal{C}$ clusters with their respective clusters means $\mu = \{\mu_{c_j}\}_{j=1,\ldots,|\mathcal{C}|}$. For instance, if there are 50 trajectory clusters, then $\mathcal{C} \in \mathbb{R}^{50 \times T_{\text{pred}} \times 2}$ and clusters $\{c_1, c_2, \ldots c_{50}\}$. The output clusters signify diverse motion cues in pedestrian future trajectories.

3.3.1. Soft Trajectory Probability

LG-Traj outputs the predicted trajectories along with their corresponding probabilities, which is beneficial for estimating the uncertainty associated with each prediction. For instance, lower probabilities indicate that the model is less confident about the prediction. Specifically, following the nearest neighbour hypothesis [11], the trajectory cluster c_j (from Gaussian mixture model M) closer to the ground truth is the most likely one, i.e., owning the maximum probability. The soft probability p_i for i^{th} pedestrian is computed by taking the negative squared Euclidean distance between the ground truth trajectory and the cluster centre, nor-

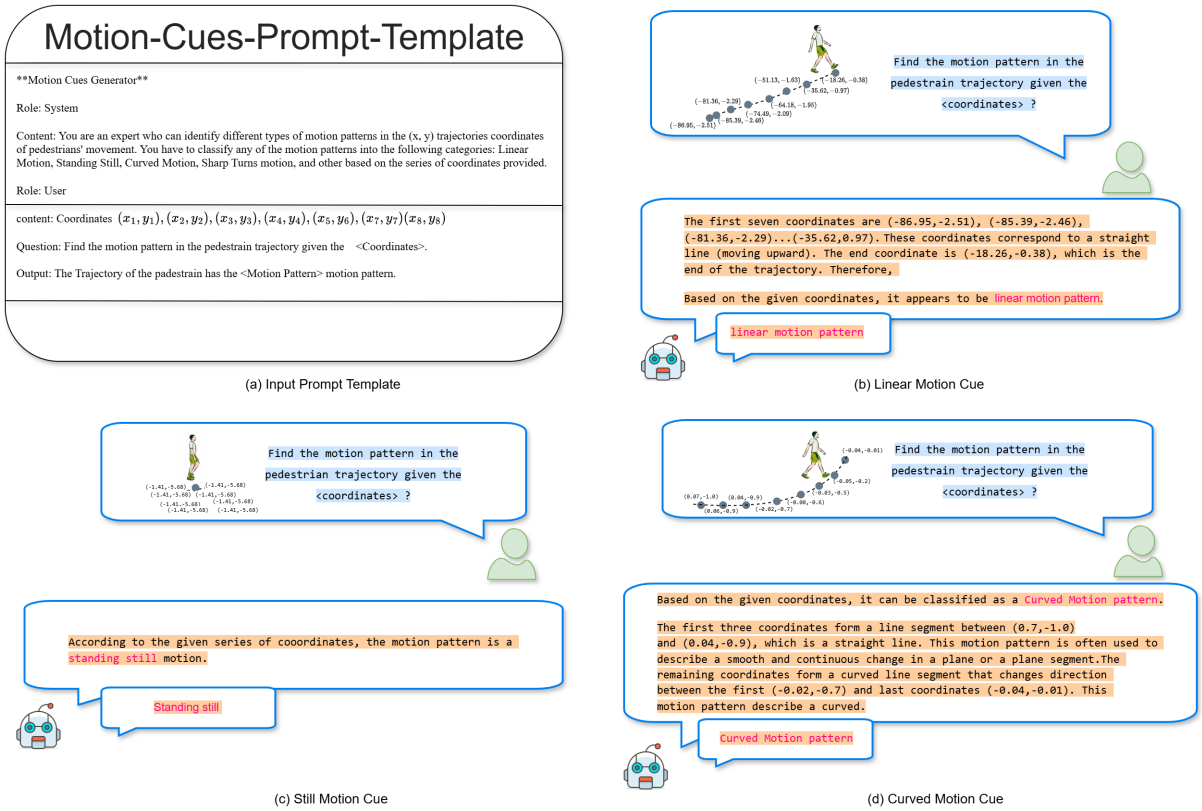

Figure 2. Illustration of input prompt and examples of motion cues generation from the LLM. We present three different examples where the LLM correctly identifies the underlying trajectory motion pattern, such as linear motion, curved motion, and standing still, based on the coordinates provided as input to the LLM.

malizing it, and then applying the softmax as given below:

$$p_i = \frac{e^{-\|Y_i - c_j\|_2^2}}{\sum_{j=1}^{|\mathcal{C}|} e^{-\|Y_i - c_j\|_2^2}} \tag{4}$$

Here, c_j represents the nearest cluster to the i^{th} pedestrian's ground truth trajectory, while p_i indicates the soft probability of the i^{th} pedestrian.

3.4. Past Motion Cues generated by LLM

3.4.1. Prompt Engineering

In this step, we guide the LLMs to generate the motion cues given the observed past trajectories of the pedestrians. We are not fine-tuning the LLM; instead, we are using a pre-trained LLM with frozen weights to generate motion cues. Our prompt design is shown in Fig. 2(a). Pedestrian trajectories [54] may follow any motion patterns, such as straight, curved, etc., to avoid collisions. To identify these patterns, we use the LLM, categorizing the motion patterns into linear, curved, standing still, and other patterns (see Fig. 2).

We adapted the chat template for our task to generate the motion cues. Our simple prompt can infer the motion cues when the trajectory coordinates sequence is fed to the LLM. We begin by specifying the system's role, where we provide a precise description of the system and its task, which is to identify motion patterns in pedestrian movements. Next, we specify the format of the user role and input data to the LLM. This simple template formatting ensures that the LLM correctly aligns the coordinates with the motion cues. The generated cues are shown in Fig. 2. The first example contains the linear motion trajectory of the pedestrian, and the LLM correctly identifies the pattern, followed by the stationary and curved motion of the pedestrian.

3.4.2. Generation

Given a set of observed coordinates for the i^{th} pedestrian trajectory X_i. A Language Model (f_{LLM}) generates past motion cues M_i containing the motion patterns of the pedestrian:

$$M_i = f_{\text{LLM}}(\boldsymbol{X}_i) \tag{5}$$

The past motion cues M_i for the i^{th} agent is transformed into an tokenized output embedding vector T_i using a tokenizer:

$$T_i = g_{\text{ST}}(M_i) \tag{6}$$

Here, M_i is the past motion cues (text description) generated by the LLM for the i^{th} pedestrian and g_{ST} is the tokenizer. Unlike a traditional classifier that relies on fixed labels/classes (e.g., 'walking,' 'standing'), our generated motion patterns are not fixed, allowing LLM to represent more complex behaviors. Following the prior NLP pipeline [43], we utilize a tokenizer to transform language-based motion cues into a format that the model can comprehend. This is achieved by directly employing pretrained tokenizers to convert motion cues into tokens specific to the sentences.

3.5. Motion Embedding

Our approach utilizes linear layers to embed the past motion cues (Eq. 7), future motion cues (Eq. 8), and past observed trajectory (Eq. 9). Additionally, we incorporate a positional encoding to model the temporal representation from the data to understand pedestrian motion dynamics. Positional encoding enables the model to capture long-term temporal dependencies within the underlying data. The embeddings from the LLM are tokenized and then transformed to yield the past motion cues embedding, as given below:

$$Z_m = \mathcal{F}_m(\mathbf{T}, \mathbf{W}_m) \tag{7}$$

Here, $\mathcal{F}_m(\cdot,\cdot)$ denotes a linear layer with trainable weight matrix $\mathbf{W}_m$. $\mathrm{T} \in \mathbb{R}^{B\times E_s}$, where B is the batch size and E_s is the output dimension of the tokenizer. The past motion cues embedding is $Z_m \in \mathbb{R}^{\mathrm{B}\times M_d}$, where M_d is the output dimension. Next, the cluster embedding is given as:

$$Z_c = \mathcal{F}_c(\mathcal{C}, \mathbf{W}_c) \tag{8}$$

Here, $Z_c \in \mathbb{R}^{|\mathcal{C}|\times O_c}$ represents cluster embedding. O_c is the output dimension of the linear layer and $|C|$ is the number of clusters. $\mathbf{W}_c$ denote the weight matrices for linear layers $\mathcal{F}_c(\cdot,\cdot)$. Finally, the past trajectory embedding is given as:

$$Z_p = \mathcal{F}_p(\mathbf{X}, \mathbf{W}_p) \tag{9}$$

Here, the embedding Z_p denotes the past trajectories embedding, and trajectory matrix $\mathbf{X} \in \mathbb{R}^{B\times T_{ob}\times 2}$. $\mathbf{W}_p$ denote the weight matrices for linear layers $\mathcal{F}_p(\cdot,\cdot)$, respectively. After obtaining all the embeddings, the past motion cues embedding, cluster embedding, and past trajectory embedding are concatenated and added with positional encoding to get the motion embedding $Z_f = concat(Z_m, Z_c, Z_p) + PE$, which is then passed as an input to the motion encoder. Here, $Z_f \in \mathbb{R}^{B\times|\mathcal{C}|\times M_e}$ and PE is a tensor containing positional encoding information. M_e is the output dimension.

3.6. Motion Encoder

The motion encoder is designed to model the spatio-temporal motion patterns in pedestrian trajectories. Specifically, the encoder consists of multiple identical layers (i.e., L number of layers), each layer consisting of a multi-head self-attention and feed-forward network. The final output of the encoder Z_e is obtained by passing the motion embedding Z_f through L encoder layers:

$$Z_e = \text{EncoderLayer}(Z_f) \tag{10}$$

where $Z_e \in \mathbb{R}^{B\times 1\times embed\_size}$, $embed\_size$ is the size of the embedding, B is the batch size.

3.7. Social Decoder

The social decoder combines the pedestrian's motion patterns with the social interactions of neighbour pedestrians. Neighbour embedding (Z_{ne}) is obtained by applying linear transformation of neighbour past observed trajectories. Neighbour embedding (Z_{ne}) along with the output Z_e from the motion encoder are fed into the decoder to forecast future trajectories. The query embedding represents the current pedestrian ($\mathbf{Q} \in \mathbb{R}^{B\times 1\times \text{embed\_size}}$). Key and value embeddings represent neighbouring pedestrians ($\mathbf{K}, \mathbf{V} \in \mathbb{R}^{B\times N\times \text{embed\_size}}$).

Through self-attention, the decoder weighs various neighbour interactions in relation to the current pedestrian. Furthermore, the resulting embeddings from the decoder represent the predicted future trajectory of the pedestrian. Along with the predicted future trajectory, we also predict the probability that estimates the uncertainty associated with each prediction.

$$Z_n,\ \hat{p}_n = \text{DecoderLayer}(Z_e, Z_{ne}) \tag{11}$$

Here, Z_e is the output of the encoder and Z_{ne} is the neighbour embedding. $Z_n \in \mathbb{R}^{B\times num\times T_{pred}\times 2}$ is the output of the decoder, where num is the number of trajectories to be predictions at inference time. $\hat{p}_n$ is the probabilities associated with predicted trajectories.

During test time, we generate past motion cues from LLM using the observed trajectory of test data. For future motion cues, we utilize the same trajectory clusters that were used during training. Finally, our model leverages past motion cues, the observed trajectory from test data, and future motion cues to predict the future trajectory for the test data.

3.8. Training Loss

Our training loss consists of two components: trajectory prediction loss ($\mathcal{L}_{\text{traj}}$), and loss for the corresponding trajectory probabilities ($\mathcal{L}_{\text{prob}}$). For trajectory prediction, we use

Table 1. Comparison of LG-Traj (Our) with other approaches on ETH, HOTEL, UNIV, ZARA1, and ZARA2 datasets in terms of ADE/FDE (lower values are better). All approaches use the observed 8-time steps and predict the future 12-time steps. The top performance is highlighted in **bold**, and the second-best performance is indicated with underline.

Model	PECNet	Trajectron++	SGCN	STGAT	CARPE	AgentFormer	GroupNet	GP-Graph
ETH	0.54/0.87	0.61/1.03	0.52/1.03	0.56/1.10	0.80/1.40	0.45/0.75	0.46/0.73	0.43/0.63
HOTEL	0.18/0.24	0.20/0.28	0.32/0.55	0.27/0.50	0.52/1.00	0.14/0.22	0.15/0.25	0.18/0.30
UNIV	0.35/0.60	0.30/0.55	0.37/0.70	0.32/0.66	0.61/1.23	0.25/0.45	0.26/0.49	0.24/0.42
ZARA1	0.22/0.39	0.24/0.41	0.29/0.53	0.21/0.42	0.42/0.84	0.18/0.30	0.21/0.39	0.17/0.31
ZARA2	0.17/0.30	0.18/0.32	0.25/0.45	0.20/0.40	0.34/0.74	0.14/0.24	0.17/0.33	0.15/0.29
AVG	0.29/0.48	0.31/0.52	0.37/0.65	0.31/0.62	0.46/0.89	0.23/0.39	0.25/0.44	0.23/0.39
Model	STT	Social-Implicit	BCDiff	Graph-TERN	FlowChain	EigenTrajectory	SMEMO	Our
ETH	0.54/1.10	0.66/1.44	0.53/0.91	0.42/0.58	0.55/0.99	0.36/0.56	0.39/0.59	0.38/0.56
HOTEL	0.24/0.46	0.20/0.36	0.17/0.27	0.14/0.23	0.20/0.35	0.14/0.22	0.14/0.20	0.11/0.17
UNIV	0.57/1.15	0.31/0.60	0.24/0.40	0.26/0.45	0.29/0.54	0.24/0.43	0.23/0.41	0.23/0.42
ZARA1	0.45/0.94	0.25/0.50	0.21/0.37	0.21/0.37	0.22/0.40	0.21/0.39	0.19/0.32	0.18/0.33
ZARA2	0.36/0.77	0.22/0.43	0.16/0.26	0.17/0.29	0.20/0.34	0.16/0.29	0.15/0.26	0.14/0.25
AVG	0.43/0.88	0.33/0.67	0.26/0.44	0.24/0.38	0.29/0.52	0.23/0.38	0.22/0.35	**0.20/0.34**

Table 2. Comparison of LG-Traj (Our) with other approaches on SDD dataset in terms of ADE/FDE (lower values are better). All approaches use the observed 8-time steps and predict the future 12-time steps.

Model	CAGN	STT	MID	SocialVAE	Graph-TERN	BCDiff	MRL	SMEMO	Our
ADE	9.42	9.13	9.73	8.88	8.43	9.05	8.22	8.11	**7.80**
FDE	15.93	15.42	15.32	14.81	14.26	14.86	13.39	13.06	**12.79**

the Huber loss between the predicted trajectory ($\hat{Y}_i$) and the ground truth trajectory (Y_i). Huber loss is chosen for its robustness to outliers, reducing the impact of large errors for stable training. For probabilities, we use Cross-Entropy loss between the ground truth probability and the predicted probability. The trajectory prediction loss is defined as:

$$\mathcal{L}_{\text{traj}} = \frac{1}{N}\sum_{i=1}^{N} \text{Huber}(Y_i, \hat{Y}_i) \tag{12}$$

where $\text{Huber}(Y_i, \hat{Y}_i)$ with δ threshold is defined as:

$$\text{Huber}(Y_i, \hat{Y}_i) = \begin{cases} \frac{1}{2}(Y_i - \hat{Y}_i)^2 & \text{if } |Y_i - \hat{Y}_i| \leq \delta \\ \delta(|Y_i - \hat{Y}_i| - \frac{1}{2}\delta) & \text{otherwise} \end{cases} \tag{13}$$

The loss for the trajectory probabilities is cross-entropy loss $\mathcal{L}_{\text{prob}}$ which is defined as:

$$\mathcal{L}_{\text{prob}} = -\frac{1}{N}\sum_{i=1}^{N} p_i \log(\hat{p}_i) \tag{14}$$

where p_i is the ground truth trajectory probability and $\hat{p}_i$ is the predicted probability. The overall loss function $\mathcal{L}_{total}$ is defined below.

$$\mathcal{L}_{total} = \mathcal{L}_{\text{traj}} + \mathcal{L}_{\text{prob}} \tag{15}$$

4. Experiments

4.1. Experimental Settings

4.1.1. Datasets

We conduct experiments on two benchmark datasets: the Stanford Drone Dataset (SDD) [46] and ETH-UCY dataset [28, 44]. ETH-UCY is a widely used benchmark dataset for predicting pedestrian trajectories, consisting of the trajectories of 1,536 pedestrians in four distinct scenarios split into five subsets: ETH, HOTEL, UNIV, ZARA1, and ZARA2. These scenarios include various scenes such as roads, intersections, and open areas. SDD is also a benchmark dataset providing bird's-eye-view perspectives of pedestrian trajectory prediction, with 5,232 trajectories from eight distinct scenarios. We used the same experimental settings for both ETH-UCY and SDD, with an observed trajectory length of 3.2 seconds (8 frames) and a predicted trajectory length of 4.8 seconds (12 frames) as used by the compared methods for fair comparison. We follow the standard split set as used by previous works [20, 33], employing a leave-one-out approach. In ETH-UCY, the agents' positions are labeled in meters, whereas in SDD, the annotations represent positions marked in pixels.

4.1.2. Evaluation Metrics

To assess the effectiveness of our method, we use widely used evaluation metrics for trajectory prediction [33], such

as Minimum Average Displacement Error (minADE) and Minimum Final Displacement Error (minFDE). ADE measures the average difference (l_2 distance) between the predicted and actual future positions of a pedestrian for all prediction time steps. FDE measures the difference (l_2 distance) between the predicted future endpoint position and the actual future endpoint position.

4.1.3. Implementation Details

We use different numbers of encoders in our experiments. We use a single encoder in ETH and HOTEL, while in UNIV, ZARA1, ZARA2, and SDD, we stack two encoders. The decoder remains one across all subsets. Additionally, we choose various Gaussian Mixtures for clustering: 50 for ETH, 90 for HOTEL, 50 for UNIV, 70 for ZARA1, 50 for ZARA2, and 100 for SDD. We utilize four singular values for trajectory augmentation, approximating the trajectories based on these singular values. We use $k = 1$ for ETH and HOTEL, and $k = 3$ for UNIV, ZARA1, ZARA2, and SDD. Our encoder and decoder each consist of 4 multi-head and 128-dimensional feed-forward networks. The tokenized output embedding vector T_i is of size 384. In motion embedding, M_e, the size is 128 for ETH and HOTEL, while for UNIV, ZARA1, and ZARA2, it is 64, and for SDD, it is also 64. The key and value embed size is 128. The batch size and learning rate are set to 128 and 1×10^{-4}, respectively, for ETH and Hotel. For UNIV, the batch size is 64, and the learning rate is 1×10^{-4}. In the Huber loss, we set the delta to 1.

For text generation, we utilize the open-source Llama-2-7b[1] model by Meta, and for tokenization, we use the Sentence Transformer[2], which outputs tokens in response to past motion cues. We use the following parameters for text generation: `max_new_tokens=32`, `temperature=0.7`, `top_k=50`, `top_p=0.95`. The `max_new_tokens` parameter determines the maximum number of new tokens the Language Model (LLM) can generate as output. The `temperature` parameter adjusts the randomness of the sampling process. Higher temperatures increase randomness, potentially leading to more diverse but less coherent outputs. We set it to 0.7 for a balanced output. The `top_k` parameter controls the number of tokens with the highest probability to consider during text generation. The `top_p` parameter sets a threshold for the cumulative probability mass for the model's distribution. Tokens with cumulative probability mass higher than this threshold are considered during text generation. We execute the experiments using Python 3.8.13 and PyTorch version 1.13.1+cu117. The training was conducted on NVIDIA RTX A5000 GPU with AMD EPYC 7543 CPU. Our text generation operates at a speed of approximately 19.71 tokens per second, and the trajectory prediction takes 39.96 milliseconds.

[1] https://llama.meta.com/llama2

[2] https://arxiv.org/abs/1908.10084

Figure 3. Illustration of predicted trajectories from ETH (first column), UNIV (second column), HOTEL (third column), and ZARA (fourth column) datasets. Predicted pedestrian trajectories are highlighted in yellow. The observed trajectories are indicated in orange, while the ground truth trajectories are depicted in green. Our method demonstrates the prediction of future trajectories (yellow), closely matching the ground truth trajectories.

Table 3. Effect of Motion Cues (MC), Position Encoding (PE), and Trajectories Augmentation (TA) on model performance. Results show that all three components are crucial for accurate trajectory prediction

Variants	ETH-UCY		SDD	
	ADE	FDE	ADE	FDE
LG-Traj	**0.20**	**0.34**	**7.80**	**12.79**
LG-Traj w/o MC	0.36	0.69	22.1	41.2
LG-Traj w/o PE	0.22	0.36	8.23	13.41
LG-Traj w/o TA	0.23	0.37	8.26	13.59

4.2. Comparison with State-of-art Methods

4.2.1. Quantitative Results

We compare our approach with recent methods. In Tables 1, we compare with SMEMO [36], FlowChain [32], Graph-TERN [4], EigenTrajectory [7], BCDiff [30], Social-Implicit [40], STT [41], GP-Graph [5], GroupNet [59], AgentFormer [64], CARPE [38], STGAT [22], SGCN [50], Trajectron++ [48], PECNeT [33], Social-STGCNN [39], and NMMP [20]. Our model outperforms all compared methods in terms of average ADE/FDE on ETH-UCY and achieves a 10%/3% relative improvement in average ADE/FDE compared to the recent method SMEMO [36]. Similarly, in Table 2, we present the experimental results on the SDD dataset in comparison with methods such as SMEMO [36], MRL [58], BCDiff [30], SocialVAE [61], MID [16], Graph-TERN [4], CAGN [13] and STT [41]. Our method outperforms all compared methods with ADE/FDE values of 7.80/12.79, respectively.

4.2.2. Qualitative Results

As shown in Fig. 3, our approach predicts future trajectories that closely align with ground truth trajectories. The model trained using our approach effectively captures pedestrian interactions, movements, and various motion

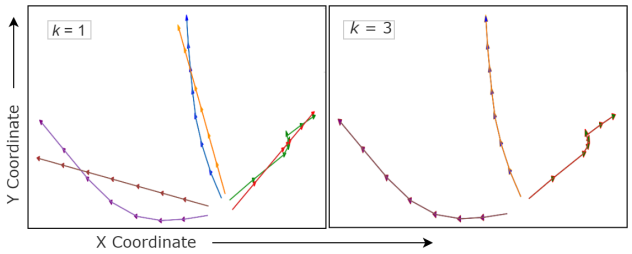

Figure 4. Visualization of augmented trajectories for three pedestrians sampled from SDD using different k values in rank-k approximation. Blue, green, and purple represent the original trajectories, while yellow, red, and brown denote their corresponding approximated variations.

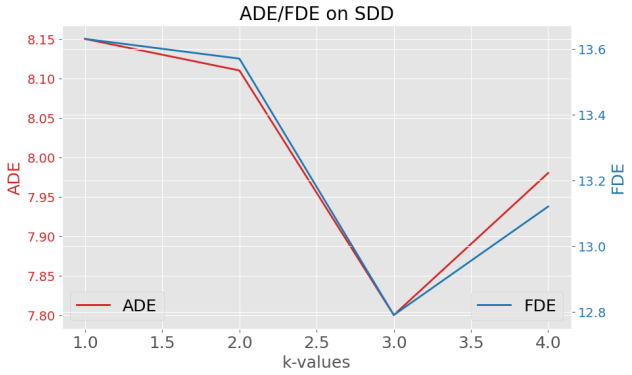

Figure 5. Visualization of ADE/FDE values obtained using our approach for different values of k over the SDD dataset. The best result is obtained for k=3.

patterns present in the scene.

4.3. Ablation Studies

In this section, we conduct extensive ablation studies to examine the effectiveness of each component of LG-Traj, along with the impact of prompt template, and rank-k approximation.

4.3.1. Effect of Various Components in LG-Traj on Model Performance

We investigate the impact of motion cues, position encoding, and trajectory augmentation on model performance. The results are presented in Table 3, with ADE/FDE values obtained by removing individual components from our approach. It is clear from the results that all three components are essential for our approach, with the most significant improvement achieved when motion cues are utilized. This demonstrates the significance of motion clues for the trajectory prediction task. Furthermore, many prior works, such as Graph-TERN, EigenTrajectory, and LMTraj, use data augmentations like scaling, flipping, and rotation. Our SVD-based augmentation, however, is novel and outperforms these methods.

Table 4. Using Different Prompt Templates on the ETH-UCY.

Prompt Templates	ADE	FDE
Zero-Shot Prompt Template	0.22	0.35
System Prompt Template	0.20	0.34
Few-Shot Prompt Template	0.20	0.34

4.3.2. Choice of Prompt Templates

We tested different prompt templates (three different templates) to generate motion cues and finally selected the *system prompt* [56] with a system role in identifying motion patterns, as shown in Fig. 2(a). Furthermore, we experimented with zero-shot [45] and few-shot prompts [10] to generate the motion cues. The results, in terms of ADE and FDE, are shown in Table 4. The zero-shot prompt template often generates text that is out of context. The few-shot prompt requires additional tokens to include examples, which significantly increases the text generation time and computational requirements. For this reason, we use the *system prompt* in our approach.

4.3.3. Effect of Using Different k Values

Fig. 4 shows augmented trajectories for three pedestrians sampled from SDD for different k values. Lower values of k result in a significant loss of information, where only the direction of the trajectory is preserved. As the value of k increases, more information is preserved in the augmented trajectories. For $k = 3$, original and augmented trajectories exhibit a similar motion pattern on SDD. Fig. 5 shows ADE/FDE values obtained using our approach for different values of k over the SDD dataset, and the best result is achieved when $k = 3$.

5. Conclusion

In this work, we propose LG-Traj, a novel approach that capitalizes on past motion cues derived from a large language model (LLM) utilizing pedestrian past/observed trajectories. Furthermore, our approach integrates future motion cues extracted from pedestrian future trajectories through clustering of training data's future trajectories using a mixture of Gaussians. Subsequently, the motion encoder utilizes both past motion cues and observed trajectories, along with future motion cues, to model motion patterns. Finally, the social decoder incorporates social interactions among neighbouring pedestrians, along with the embedding produced by the motion encoder, to generate socially plausible future trajectories. Our experimental findings illustrate the feasibility of integrating LLM into trajectory prediction tasks. We showcase the effectiveness of our approach on widely used pedestrian trajectory prediction benchmarks, including ETH-UCY and SDD, and present various ablation experiments to validate our approach.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 2

[2] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016. 1, 2

[3] Görkay Aydemir, Adil Kaan Akan, and Fatma Güney. Adapt: Efficient multi-agent trajectory prediction with adaptation. *arXiv preprint arXiv:2307.14187*, 2023. 2

[4] Inhwan Bae and Hae-Gon Jeon. A set of control points conditioned pedestrian trajectory prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6155–6165, 2023. 1, 2, 7

[5] Inhwan Bae, Jin-Hwi Park, and Hae-Gon Jeon. Learning pedestrian group representations for multi-modal trajectory prediction. In *Proceedings of the European Conference on Computer Vision*, 2022. 7

[6] Inhwan Bae, Jin-Hwi Park, and Hae-Gon Jeon. Non-probability sampling network for stochastic human trajectory prediction. In *CVPR*, 2022. 2

[7] Inhwan Bae, Jean Oh, and Hae-Gon Jeon. Eigentrajectory: Low-rank descriptors for multi-modal trajectory forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10017–10029, 2023. 1, 7

[8] Inhwan Bae, Junoh Lee, and Hae-Gon Jeon. Can language beat numerical regression? language-based multimodal trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 753–766, 2024. 2

[9] Apratim Bhattacharyya, Michael Hanselmann, Mario Fritz, Bernt Schiele, and Christoph-Nikolas Straehle. Conditional flow variational autoencoders for structured sequence prediction. *arXiv preprint arXiv:1908.09008*, 2019. 2

[10] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 8

[11] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967. 3

[12] Vikrant Dewangan, Tushar Choudhary, Shivam Chandhok, Shubham Priyadarshan, Anushka Jain, Arun K Singh, Siddharth Srivastava, Krishna Murthy Jatavallabhula, and K Madhava Krishna. Talk2bev: Language-enhanced bird's-eye view maps for autonomous driving. *arXiv preprint arXiv:2310.02251*, 2023. 2

[13] Jinghai Duan, Le Wang, Chengjiang Long, Sanping Zhou, Fang Zheng, Liushuai Shi, and Gang Hua. Complementary attention gated network for pedestrian trajectory prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 542–550, 2022. 7

[14] Daocheng Fu, Xin Li, Licheng Wen, Min Dou, Pinlong Cai, Botian Shi, and Yu Qiao. Drive like a human: Rethinking autonomous driving with large language models. *arXiv preprint arXiv:2307.07162*, 2023. 2

[15] Roger Girgis, Florian Golemo, Felipe Codevilla, Martin Weiss, Jim Aldon D'Souza, Samira Ebrahimi Kahou, Felix Heide, and Christopher Pal. Latent variable sequential set transformers for joint multi-agent motion prediction. In *International Conference on Learning Representations*, 2022. 2

[16] Tianpei Gu, Guangyi Chen, Junlong Li, Chunze Lin, Yongming Rao, Jie Zhou, and Jiwen Lu. Stochastic trajectory prediction via motion indeterminacy diffusion. In *CVPR*, 2022. 2, 7

[17] Tianpei Gu, Guangyi Chen, Junlong Li, Chunze Lin, Yongming Rao, Jie Zhou, and Jiwen Lu. Stochastic trajectory prediction via motion indeterminacy diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17113–17122, 2022. 2

[18] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2255–2264, 2018. 1, 2

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2

[20] Yue Hu, Siheng Chen, Ya Zhang, and Xiao Gu. Collaborative motion prediction via neural motion message passing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6319–6328, 2020. 1, 2, 6, 7

[21] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10608–10615. IEEE, 2023. 2

[22] Yingfan Huang, Huikun Bi, Zhaoxin Li, Tianlu Mao, and Zhaoqi Wang. Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6272–6281, 2019. 2, 7

[23] Boris Ivanovic and Marco Pavone. The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2375–2384, 2019. 2

[24] Ye Jin, Xiaoxi Shen, Huiling Peng, Xiaoan Liu, Jingli Qin, Jiayang Li, Jintao Xie, Peizhong Gao, Guyue Zhou, and Jiangtao Gong. Surrealdriver: Designing generative driver agent simulation framework in urban contexts based on large language model. *arXiv preprint arXiv:2309.13193*, 2023. 2

[25] Vineet Kosaraju, Amir Sadeghian, Roberto Martín-Martín, Ian Reid, Hamid Rezatofighi, and Silvio Savarese. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[26] Zhengxing Lan, Lingshan Liu, Bo Fan, Yisheng Lv, Yilong Ren, and Zhiyong Cui. Traj-llm: A new exploration for

empowering trajectory prediction with pre-trained large language models. *IEEE Transactions on Intelligent Vehicles*, 2024. 2

[27] Mihee Lee, Samuel S Sohn, Seonghyeon Moon, Sejong Yoon, Mubbasir Kapadia, and Vladimir Pavlovic. Muse-vae: multi-scale vae for environment-aware long term trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2221–2230, 2022. 1, 2

[28] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. In *Computer graphics forum*, pages 655–664. Wiley Online Library, 2007. 6

[29] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019. 2

[30] Rongqing Li, Changsheng Li, Dongchun Ren, Guangyi Chen, Ye Yuan, and Guoren Wang. Bcdiff: Bidirectional consistent diffusion for instantaneous trajectory prediction. *Advances in Neural Information Processing Systems*, 36, 2024. 7

[31] Pei Lv, Wentong Wang, Yunxin Wang, Yuzhen Zhang, Mingliang Xu, and Changsheng Xu. Ssagcn: social soft attention graph convolution network for pedestrian trajectory prediction. *IEEE transactions on neural networks and learning systems*, 2023. 2

[32] Takahiro Maeda and Norimichi Ukita. Fast inference and update of probabilistic density estimation on trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9795–9805, 2023. 7

[33] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 759–776. Springer, 2020. 1, 2, 6, 7

[34] Jiageng Mao, Yuxi Qian, Hang Zhao, and Yue Wang. Gpt-driver: Learning to drive with gpt. *arXiv preprint arXiv:2310.01415*, 2023. 2

[35] Weibo Mao, Chenxin Xu, Qi Zhu, Siheng Chen, and Yanfeng Wang. Leapfrog diffusion model for stochastic trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5517–5526, 2023. 2

[36] Francesco Marchetti, Federico Becattini, Lorenzo Seidenari, and Alberto Del Bimbo. Smemo: social memory for trajectory forecasting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 7

[37] Ana-Maria Marcu, Long Chen, Jan Hünermann, Alice Karnsund, Benoit Hanotte, Prajwal Chidananda, Saurabh Nair, Vijay Badrinarayanan, Alex Kendall, Jamie Shotton, et al. Lingoqa: Video question answering for autonomous driving. *arXiv preprint arXiv:2312.14115*, 2023. 2

[38] Matías Mendieta and Hamed Tabkhi. Carpe posterum: A convolutional approach for real-time pedestrian path prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2346–2354, 2021. 7

[39] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14424–14432, 2020. 1, 7

[40] Abduallah Mohamed, Deyao Zhu, Warren Vu, Mohamed Elhoseiny, and Christian Claudel. Social-implicit: Rethinking trajectory prediction evaluation and the effectiveness of implicit maximum likelihood estimation. In *European Conference on Computer Vision*, pages 463–479. Springer, 2022. 7

[41] Alessio Monti, Angelo Porrello, Simone Calderara, Pasquale Coscia, Lamberto Ballan, and Rita Cucchiara. How many observations are enough? knowledge distillation for trajectory forecasting. In *CVPR*, 2022. 2, 7

[42] Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. In *International Conference on Machine Learning*, pages 17359–17371. PMLR, 2022. 2

[43] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization, 2017. 5

[44] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th international conference on computer vision*, pages 261–268. IEEE, 2009. 6

[45] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 8

[46] Alexandre Robicquet, Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Learning social etiquette: Human trajectory understanding in crowded scenes. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*, pages 549–565. Springer, 2016. 1, 6

[47] Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1349–1358, 2019. 1, 2

[48] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, pages 683–700. Springer, 2020. 7

[49] Jasmine Sekhon and Cody Fleming. Scan: A spatial context attentive network for joint multi-agent intent prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6119–6127, 2021. 2

[50] Liushuai Shi, Le Wang, Chengjiang Long, Sanping Zhou, Mo Zhou, Zhenxing Niu, and Gang Hua. Sgcn: Sparse graph

convolution network for pedestrian trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8994–9003, 2021. 1, 2, 7

[51] Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Ping Luo, Andreas Geiger, and Hongyang Li. Drivelm: Driving with graph visual question answering. *arXiv preprint arXiv:2312.14150*, 2023. 2

[52] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 2

[53] Anirudh Vemula, Katharina Muelling, and Jean Oh. Social attention: Modeling attention in human crowds. In *2018 IEEE international Conference on Robotics and Automation (ICRA)*, pages 4601–4607. IEEE, 2018. 2

[54] Honghui Wang, Weiming Zhi, Gustavo Batista, and Rohitash Chandra. Pedestrian trajectory prediction using dynamics-based deep learning. *arXiv preprint arXiv:2309.09021*, 2023. 4

[55] Song Wen, Hao Wang, and Dimitris Metaxas. Social ode: Multi-agent trajectory forecasting with neural ordinary differential equations. In *ECCV*, 2022. 2

[56] Jason Weston and Sainbayar Sukhbaatar. System 2 attention (is something you might need too). *arXiv preprint arXiv:2311.11829*, 2023. 8

[57] Conghao Wong, Beihao Xia, Ziming Hong, Qinmu Peng, Wei Yuan, Qiong Cao, Yibo Yang, and Xinge You. View vertically: A hierarchical network for trajectory prediction via fourier spectrums. In *ECCV*, 2022. 2

[58] Yuxuan Wu, Le Wang, Sanping Zhou, Jinghai Duan, Gang Hua, and Wei Tang. Multi-stream representation learning for pedestrian trajectory prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2875–2882, 2023. 7

[59] Chenxin Xu, Maosen Li, Zhenyang Ni, Ya Zhang, and Siheng Chen. Groupnet: Multiscale hypergraph neural networks for trajectory prediction with relational reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6498–6507, 2022. 1, 2, 7

[60] Chenxin Xu, Yuxi Wei, Bohan Tang, Sheng Yin, Ya Zhang, and Siheng Chen. Dynamic-group-aware networks for multi-agent trajectory prediction with relational reasoning. *arXiv preprint arXiv:2206.13114*, 2022. 1, 2

[61] Pei Xu, Jean-Bernard Hayet, and Ioannis Karamouzas. Socialvae: Human trajectory prediction using timewise latents–supplemental material. 7

[62] Cunjun Yu, Xiao Ma, Jiawei Ren, Haiyu Zhao, and Shuai Yi. Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*, pages 507–523. Springer, 2020. 2

[63] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *ICCV*, 2021. 2

[64] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9813–9823, 2021. 2, 7

[65] Zikang Zhou, Jianping Wang, Yung-Hui Li, and Yu-Kai Huang. Query-centric trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17863–17873, 2023. 2