

# ControlTraj: Controllable Trajectory Generation with Topology-Constrained Diffusion Model

Yuanshao Zhu<sup>\*,1,2,3</sup>, James Jianqiao Yu<sup>4,†</sup>, Xiangyu Zhao<sup>2,†</sup>, Qidong Liu<sup>2</sup>, Yongchao Ye<sup>2</sup>

Wei Chen<sup>3</sup>, Zijian Zhang<sup>2</sup>, Xuetao Wei<sup>1</sup>, Yuxuan Liang<sup>3,†</sup>

<sup>1</sup>Southern University of Science and Technology, <sup>2</sup>City University of Hong Kong

<sup>3</sup>The Hong Kong University of Science and Technology (Guangzhou), <sup>4</sup>York University

zhuys2019@mail.sustech.edu.cn, james.yu@york.ac.uk, xianzhao@cityu.edu.hk, liuqidong@stu.xjtu.edu.cn

yongchao.ye@my.cityu.edu.hk, onedeanxxx@gmail.com, zhangjz2114@mails.jlu.edu.cn

weixt@sustech.edu.cn, yuxliang@outlook.com

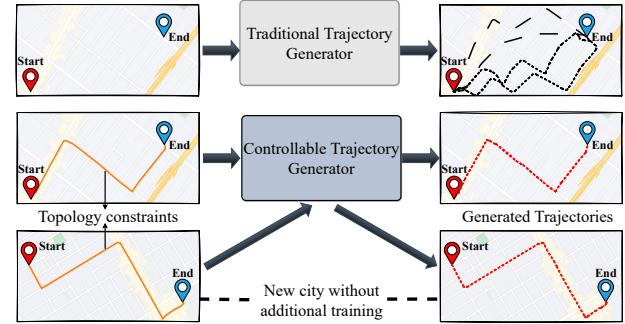
## ABSTRACT

Generating trajectory data is among promising solutions to addressing privacy concerns, collection costs, and proprietary restrictions usually associated with human mobility analyses. However, existing trajectory generation methods are still in their infancy due to the inherent diversity and unpredictability of human activities, grappling with issues such as fidelity, flexibility, and generalizability. To overcome these obstacles, we propose **ControlTraj**, a **Controllable Trajectory** generation framework with the topology-constrained diffusion model. Distinct from prior approaches, ControlTraj utilizes a diffusion model to generate high-fidelity trajectories while integrating the structural constraints of road network topology to guide the geographical outcomes. Specifically, we develop a novel road segment autoencoder to extract fine-grained road segment embedding. The encoded features, along with trip attributes, are subsequently merged into the proposed geographic denoising UNet architecture, named GeoUNet, to synthesize geographic trajectories from white noise. Through experimentation across three real-world data settings, ControlTraj demonstrates its ability to produce human-directed, high-fidelity trajectory generation with adaptability to unexplored geographical contexts.

## 1 INTRODUCTION

The proliferation of GPS technology has revolutionized our understanding of human mobility patterns, yielding profound implications for urban planning [32], location-based services [37], and beyond [22, 40]. However, acquiring real-world GPS trajectory data frequently confronts formidable obstacles, including privacy concerns [54], data collection costs [52], proprietary restrictions [2, 10], and a myriad of regulatory barriers [3, 47]. These challenges substantially hinder researchers' ability to access data that faithfully captures the complexity of human mobility. In this context, trajectory generation emerges as a compelling solution to the problem, offering the potential to create synthetic yet realistic GPS trajectories to bypass these limitations [55]. By generating synthesized trajectories, this solution can propel research without real datasets and adhere to the stringent privacy and data ownership requirements, thereby broadening research and applications in a series of fields [5, 13, 19, 53].

To generate trajectories that closely reflect the complexities of real-world scenarios, researchers have pursued various strategies.



**Figure 1: Comparison of trajectory generation models. Our controllable trajectory generator can provide high-fidelity trajectories and adhere to given conditions. Also, it adapts to new urban environments without retraining.**

Early attempts, grounded in rule-based or statistical models, managed to offer a degree of interpretability but often fell short in mirroring the stochastic behavior of human mobility [1, 33]. Advancements in machine learning, particularly through the use of Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs) [16, 29, 45], have led to more sophisticated modeling of complex trajectory distributions. Nevertheless, these approaches typically resort to transforming trajectories into simplified data formats like grids [11, 27, 49] or images [2, 38], compromising the fidelity and granularity of the resulting data [24]. Furthermore, such techniques primarily focus on group mobility simulation and lack the precision required for reproducing individual paths at the detailed level of GPS data points. Recent advances endeavor to utilize the generative power of diffusion models to create finer-grained trajectories [55]. Despite their potential, such approaches are unable to control the routing of trajectory generation in accordance to the road network or apply the model directly to a new city without retraining. As depicted in Fig. 1, the current landscape reveals a gap in generating fine-grained trajectories and a lack of control over the outcomes influenced by the road network topology.

Therefore, a desirable trajectory generation solution shall satisfy the following requirements to effectively address the shortcomings of current approaches: (i) **High-fidelity**: The model shall be capable of generating trajectories that are fine-grained and retain the original spatial-temporal properties. (ii) **Flexibility**: The framework shall be adaptable, enabling users to steer trajectory generation towards simulating specific mobility patterns or complying with distinct conditions, such as constraints imposed by road network

\*Work was done at Hong Kong University of Science and Technology (Guangzhou)

†Corresponding authors.

topology or varying times of day. (iii) **Generalizability**: The model shall possess the ability to generalize beyond the geographical context of training data. This quality ensures that the generated trajectories remain realistic and applicable when introduced to new environments or used to simulate conditions not directly encountered.

To meet these essential criteria and address the existing gaps, we propose a **Controllable Trajectory (ControlTraj)** generation framework with the topology-constrained diffusion model. Fig. 1 illustrates the superiority of the proposed trajectory generator compared to traditional methods. By incorporating road network information, ControlTraj can follow its topology structure according to human guidance faithfully, and generate higher resolution trajectories. In addition, our model showcases remarkable adaptability and generalizability, enabling its application to new cities without the need for retraining. Firstly, in pursuit of reliable conditional control, we creatively introduce a Masked Road Autoencoder (**RoadMAE**). This novel structure allows for capturing fine-grained embedded representations of road segments with real-world scenarios. Secondly, for the flexibility of condition generation and the transferability across different environments, we develop a geographic attention-based UNet architecture, **GeoUNet**. Finally, ControlTraj fuses the conditional topology constraints from RoadMAE into GeoUNet to achieve controlled high-fidelity trajectory generation.

In summary, the contributions of our research are as follows:

- We design a masked road segment Autoencoder equipped with the ability to encode robust embedding representations based on the given road segment topology.
- We introduce a geographic attention-based denoising UNet structure, which seamlessly integrates topological constraints into the diffusion process. This enables the generation of trajectories that are both flexible and controllable.
- We validate ControlTraj on three real-world trajectory datasets. Quantification and visualization analyses demonstrate that the proposed method can produce high-fidelity trajectories with competitive utility. Further, the generated trajectories can be human-controlled and exhibit strong generalizability to unexplored road network topology.

## 2 PRELIMINARY

### 2.1 Problem Definition

**Definition 1 (Trajectory).** We denote a trajectory as a sequence of continuously sampled GPS points by  $\mathbf{x} = \{p_i \mid i = 1, 2, \dots, n\}$  with length  $n$ . Each GPS point is represented as  $p_i = [\text{lng}_i, \text{lat}_i, t_i]$ , which is a triple of longitude, latitude, and timestamp.

**Definition 2 (Trip Attributes).** Trip attributes are the inherent properties that accompany the entire trajectory. They generally include departure time, travel time, total distance, average speed, etc.

**Definition 3 (Road Segments).** A road segment is conceptualized as a series of lines formed by two or more connecting GPS points. Specifically, we denote road segments as  $\mathbf{r} = \{l_j \mid j = 1, 2, \dots, m\}$ , where each  $l_j = [\text{lng}_j^1, \text{lat}_j^1, \text{lng}_j^2, \text{lat}_j^2]$  represents a line segment formed by two consecutive GPS points, and  $m$  indicates the number of lines. For a city, various road segments are interconnected to

form the urban road network, i.e., the road topology structure. In practice, the representation data for such road segments can be obtained from public data sources, e.g., OpenStreetMap [26].

**Problem Statement (Controllable Trajectory Generation).** The primary objective of this research is to develop a controllable trajectory generator  $G$  capable of generating multiple trajectories that closely align with given road topology constraints. Formally, the problem is defined as follows: Given a set of road segments  $\mathcal{R} = \{r_1, r_2, \dots, r_m\}$ , the goal for  $G$  is to produce a set of generated trajectories  $\tilde{\mathcal{X}} = \{\tilde{\mathbf{x}}^1, \tilde{\mathbf{x}}^2, \dots, \tilde{\mathbf{x}}^n\}$ , where each trajectory  $\tilde{\mathbf{x}}^i = \{p_1^i, p_2^i, \dots, p_n^i\}$  is a sequence of GPS points. The generated trajectories are topology-constrained, realistic, and generalizable.

### 2.2 Conditional Diffusion Probabilistic Model

Diffusion probabilistic models represent a cutting-edge class of generative models renowned for their efficacy in generating complex data distributions [17, 23, 35]. These models are particularly effective when conditioned on external information, enabling them to produce data that adheres to specific constraints [18, 41]. Typically, these models comprise two Markov chain processes with step length  $T$ : the forward (noising) process, which incrementally adds noise to the original data  $\mathbf{x}_0$ , and the reverse (denoising) process, which aims to recover the original distribution from a Gaussian noise state  $\mathcal{N}(0, \mathbf{I})$ .

**Forward process.** In the forward process, noise is sequentially added to a dataset of original samples  $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ , where  $q(\mathbf{x}_0)$  represents the distribution of the clean data. This incremental noising is mathematically defined as a Markov chain that transitions the data from its original state  $\mathbf{x}_0$  to a noise-dominated state  $\mathbf{x}_T$ :

$$q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) = \prod_{t=1}^T q(\mathbf{x}_t \mid \mathbf{x}_{t-1}), \quad (1)$$

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I}\right), \quad (2)$$

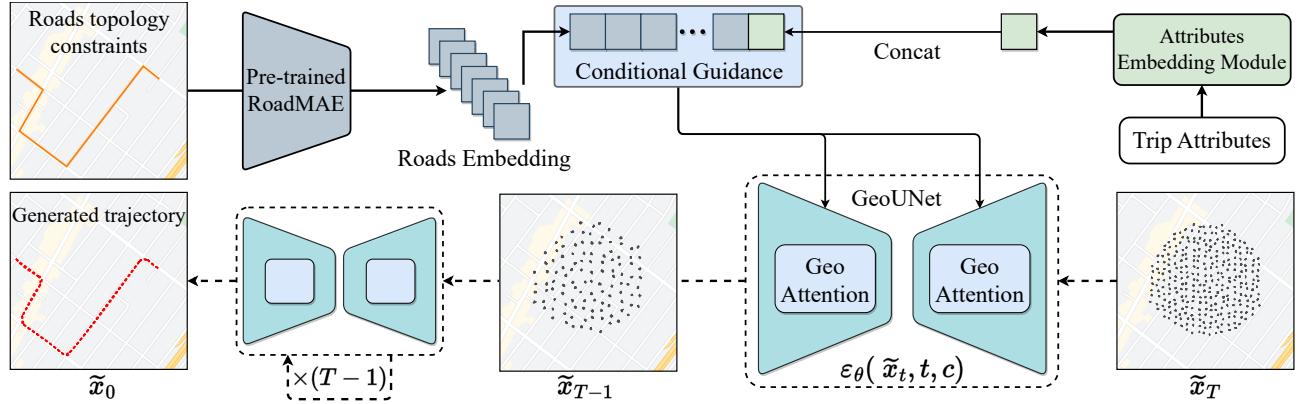
where  $\beta_t$  represents the noise schedule. For practical gradient-based optimization, a reparameterization approach [17] is employed such that  $\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}$ , and  $\bar{\alpha}_t = \prod_{i=1}^t (1 - \beta_i)$ .  $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$  is sampled from a Gaussian distribution: when  $T$  is large enough, it ensures that  $\mathbf{x}_T$  obeys a Gaussian distribution.

**Reverse process.** The reverse process of our conditional diffusion probabilistic model is designed to reconstruct the original trajectory data from the noise-corrupted state  $\tilde{\mathbf{x}}_T$ . This denoising process depends on the previous noise data and conditional information  $c$ . Within the trajectory generation domain, this conditional information encompasses elements such as road segments and trajectory attributes, thereby facilitating the generation of trajectories that are not only noise-free but also conformal to specific control requirements. Formally, the reverse process is defined by:

$$p_\theta(\tilde{\mathbf{x}}_{0:T-1} \mid \tilde{\mathbf{x}}_T, c) = \prod_{t=1}^T p_\theta(\tilde{\mathbf{x}}_{t-1} \mid \tilde{\mathbf{x}}_t, c), \quad (3)$$

$$p_\theta(\tilde{\mathbf{x}}_{t-1} \mid \tilde{\mathbf{x}}_t, c) := \mathcal{N}\left(\tilde{\mathbf{x}}_{t-1}; \mu_\theta(\tilde{\mathbf{x}}_t, t, c), \sigma_\theta(\tilde{\mathbf{x}}_t, t, c)^2 \mathbf{I}\right), \quad (4)$$

where  $\mu_\theta(\tilde{\mathbf{x}}_t, t, c)$  and  $\sigma_\theta(\tilde{\mathbf{x}}_t, t, c)$  represent the mean and variance of the reverse process at each step  $t$ , respectively. Both are parameterized by  $\theta$  and modulated by the conditional information



**Figure 2: The overview of the proposed ControlTraj framework. The pre-trained RoadMAE encodes road embedding based on the topology constraints of the road segments. Road embedding is then concatenated with trip attributes and merged into the diffusion model via geographic attention.**

c. Building upon the work of Ho *et al.* [17], we adopt an effective parameterization of  $\mu_\theta$  and  $\sigma_\theta$ , which are defined as:

$$\begin{cases} \mu_\theta(\tilde{x}_t, t, c) = \frac{1}{\sqrt{\alpha_t}} \left( \tilde{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \varepsilon_\theta(\tilde{x}_t, t, c) \right) \\ \sigma_\theta(\tilde{x}_t, t, c) = (\tilde{\beta}_t)^{\frac{1}{2}}, \text{ where } \tilde{\beta}_t = \begin{cases} \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \beta_t & t > 1 \\ \beta_1 & t = 1 \end{cases} \end{cases}. \quad (5)$$

Here,  $\varepsilon_\theta(\tilde{x}_t, t, c)$  denotes the estimated noise level under conditions  $t$  and  $c$ , as predicted by the neural network model uniquely tailored for this task. Although the diffusion model has been applied in the field of trajectory data generation, prior models did not consider topology constraints of the road network in principle.

### 3 METHODOLOGY

In this section, we introduce the workflow of our ControlTraj framework, designed to produce trajectories that adhere to specific constraints and preferences. Fig. 2 illustrates the entire process, beginning with the generation of road topology constraint guidance through the pre-trained RoadMAE. This road embedding, in conjunction with the attribute embedding, informs GeoUNet to ensure the controlled generation of trajectories. GeoUNet, equipped with geo-attention mechanisms, synthesizes the final trajectory by iteratively refining noise estimations conditioned on the embedded road and trip information.

#### 3.1 Extracting Topology Constraint

When constructing the controllable trajectory generation framework, extracting the conditional guidance is one important prerequisite. For most diffusion model-based CV or NLP tasks, this is a trivial step easily performed with pre-trained encoders such as Bert [6] or CLIP [28]. However, trajectory generation poses unique challenges due to annotated road network data scarcity and complex geography, such as irregularities and diverse topologies. In response, we design a masked road Autoencoder that learns efficiently in a self-supervised manner while capturing essential features of the road network. Fig. 3 showcases the architecture of

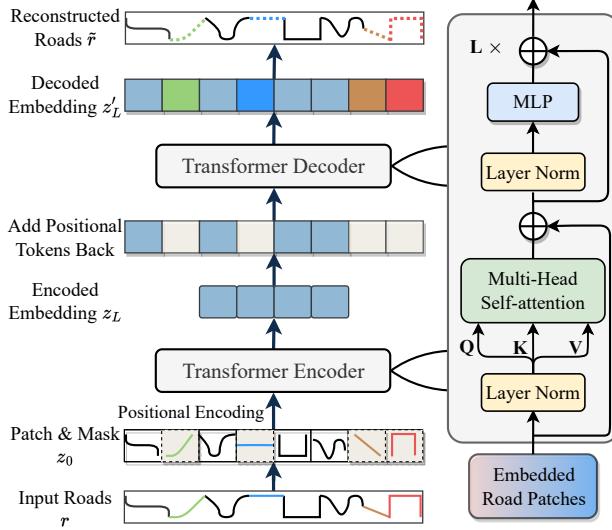
the proposed RoadMAE, comprising a transformer encoder that processes road patches with positional encoding and a transformer decoder for reconstructing the road segments. This self-supervised learning framework efficiently distills spatial context into diffusion model-compatible embedding, providing fine-grained guidance for trajectory generation.

**3.1.1 Road Segments Patching and Masking.** As defined in Sec. 2.1, road segments are represented by a series of connected GPS coordinates and maintain geometric and geographic attributes such as shape and location. Considering the semantic sparsity at the individual point level, we adopt a patching strategy [8] to methodically decompose these segments into discrete, information-rich patches. This approach facilitates capturing enough spatial context within patches while mitigating computational demands for lengthy input sequences. Specifically, we reshape a road segment  $r \in \mathbb{R}^{2 \times L}$  into a sequence of patches  $r_p \in \mathbb{R}^{N \times (2 \cdot P)}$ , where  $(2, L)$  represents the original spatial resolution of the road segment, and  $P$  denotes the designated patch length.  $N = \lceil L/P \rceil$  signifies the total count of patches. Then, we embed each patch into a  $D$ -dimensional space, and positional embedding are added to retain the sequentiality of the patches. The operations above can be formalized as:

$$z_0 = \text{Embed}(\text{Patch}(r)) + E_{pos}, \quad (6)$$

where  $z_0 \in \mathbb{R}^{N \times D}$  is the resulting embedded patch sequence and  $E_{pos} \in \mathbb{R}^D$  is the positional embedding tokens.

In anticipation of potential irregularities and inconsistencies between trajectories and road segments, we introduce a random masking strategy to enhance the robustness of our model, where a subset of patch representations are randomly masked during training [15]. This operation forces the encoder to learn a comprehensive understanding of spatial structure, ensuring that it does not overfit specific road segments and enhancing its ability to reconstruct missing information. Formally, we mathematically represent the masking procedure with a binary mask matrix  $M \in \{0, 1\}$ , where  $M_i$  corresponds to the masking state of the  $i$ -th patch, controlled by a predetermined ratio  $r_o$ . Before fed into transformer, the positioned



**Figure 3: The pipeline of RoadMAE.**

patch embedding  $z_0$  is element-wise multiplied by  $M$ , producing the masked sequence  $z_1 = z_0 \odot M$ . This preparatory step ensures that the encoder can cope with road segment missing situations, facilitating a self-supervised learning paradigm.

**3.1.2 Transformer-based Autoencoder.** Upon preparing the masked sequence of patches, the encoder emerges as a pivotal element in our model, tasked with crafting a contextualized encoding of the road segments. The encoder employs a transformer-based architecture (renowned for its efficacy in capturing long-range dependencies) to process sequences through blocks consisting of multi-headed self-attention (MSA), multi-layer perceptron (MLP), and layer normalization (LN) components [36] (details of the structure can be found in Fig. 3). This architecture empowers the encoder to consider the entire road context when encoding each patch, resulting in a richer, more context-aware representation. The process can be formalized as:

$$z'_\ell = \text{MSA}(\text{LN}(z_{\ell-1})) + z_{\ell-1}, \quad \ell = 1 \dots L, \quad (7)$$

$$z_\ell = \text{MLP}(\text{LN}(z'_\ell)) + z'_\ell, \quad \ell = 1 \dots L, \quad (8)$$

where  $L$  denotes the number of stacked blocks and output of the encoder  $z_L$  is a sequence of coded tensors representing the learned features of the road segments. It is used by the decoder to reconstruct the road network during the training road segment encoder phase, and as a geo-context condition to control trajectory generation during the trajectory generation phase.

The primary objective of the decoder is to interpret the encoded sequences  $z_L$  and reconstruct the segments that were masked during the encoding process. It uses the same architecture as the encoder (the number of stacked blocks can be different), which ensures a coherent and symmetric processing flow that facilitates efficient reconstruction of road segments. Specifically, we first construct a learnable tensor  $E_{tmp} \in \mathbb{R}^{1 \times D}$ , which is merged into the encoded vector based on the position tokens. Then, the compacted information is effectively decoded back into spatially coherent segments

by inverting the transforms, which are expressed as follows:

$$z'_L = z_L \odot M + E_{tmp} \odot (1 - M), \quad (9)$$

$$\tilde{r}_i = \text{Decoder}(z'_L). \quad (10)$$

The efficacy of RoadMAE is optimized through the reconstruction loss  $\mathcal{L}_{ssl}$ , calculated as the squared error between the original road segments  $r_i$  and the reconstructed segments  $\tilde{r}_i$ :

$$\mathcal{L}_{ssl} = \| (r - \tilde{r}) \odot M \|^2. \quad (11)$$

This function ensures that the Autoencoder focuses on accurately reconstructing the occluded portion of roads, thus enhancing its ability to infer missing information based on learned contextual cues. Meanwhile, the encoder is empowered to generate robust geographic representations based on the masked road segments.

### 3.2 GeoUNet Architecture

As discussed in Sec. 2.2, we need to meticulously design a neural network that can accurately predict the noise level  $\epsilon_\theta(\tilde{x}_t, t, c)$  at each reverse diffusion step. While the UNet architecture has been validated in a range of diffusion models [17, 31], it still lacks geo-context awareness of trajectory data. To address this challenge, the proposed GeoUNet integrates geo-attention mechanisms that enable accurate modeling of complex road networks and spatial-temporal dynamics. Specifically, the designed GeoUNet equips the two primary parts, i.e., downsampling and upsampling, with a devised Geo-Attention mechanism, which is tailored for trajectory processing. Each part consists of multiple stacked neural network blocks designed to extract and process features at various scales. This architecture enables the fusion of multi-scale features through skip connections and allows for efficient capture of both local and global contextual information (More details about GeoUNet can be found in Fig. 9 of Appendix A.1).

Then, the geo-attention is integrated into GeoUNet, which augments the model with geo-context embedding at each sampling block. In practice, the geo-attention mechanism is implemented by cascading two attention mechanisms:

$$Q = W_{sq} \cdot h^i, \quad K = W_{sk} \cdot h^i, \quad V = W_{sv} \cdot h^i, \quad (12)$$

$$Q = W_{cq} \cdot h^i, \quad K = W_{ck} \cdot c, \quad V = W_{cv} \cdot c, \quad (13)$$

where  $h \in \mathbb{R}^{L \times d}$  is the feature embedding of sampling blocks,  $c$  is the attribute embedding, and all  $W_{s\_} \in \mathbb{R}^{d \times d}$  and  $W_{c\_} \in \mathbb{R}^{N \times d}$  are learnable matrices. Here, the first formula represents the self-attention mechanism, which enables the model to internalize and emphasize fine-grained features in the sequence of trajectories. Meanwhile, the cross-attention mechanism (second formulation) critically integrates external conditional information such as road segment embedding, trajectory attributes, etc. This aligns the features with this additional context to render the final generated trajectories more realistic and context-specific. Then, the outcomes of both attention mechanisms are computed by  $h^{i+1} = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$ . Finally, the output features are further processed by the standard residual network block in UNet.

In addition, we recognize the critical role of various external attributes (e.g., departure time and travel distance) in shaping the

**Algorithm 1** The main processes of ControlTraj**Training Process:**


---

```

1: for  $i = 1, 2, \dots, n$ , do
2:   Road segments  $r$ , Mask ratio  $r_o$ , Trajectory Attributes
3:   Get masked embedding  $z_L$  by encoder of RoadMAE
4:   Get conditional guidance  $c = \text{Concat}(z_{attr}, z_L)$ 
5:   Sample  $x_0 \sim q(x)$ ,
6:   Sample  $t \sim \text{Uniform}(\{1, \dots, T\})$ ,  $\epsilon \sim \mathcal{N}(0, I)$ 
7:    $x_t = x_0 + \sqrt{1 - \alpha_t} \epsilon$ 
8:   Updating the gradient  $\nabla_\theta \|\epsilon - \epsilon_\theta(x_t, t, c)\|_2^2$ 
9: end for

```

**Generating Process:**

```

10: Road segments  $r$ , Trajectory Attributes
11: Get road segments embedding  $z_L$ 
12: Get conditional guidance  $c = \text{Concat}(z_{attr}, z_L)$ 
13: Sample  $\tilde{x}_T \sim \mathcal{N}(0, I)$ 
14: for  $t = T, T - S, \dots, 1$  do
15:   Compute  $\mu_\theta(\tilde{x}_t, t, c)$  according to Eq. (5)
16:   Compute  $p_\theta(\tilde{x}_{t-1} | \tilde{x}_t, c)$  according to Eq. (3)
17: end for
18: return  $\tilde{x}_0$ 

```

---

motion patterns and properties of real-world trajectories. To capture these trajectory properties, we follow the design in literature [4, 55] by integrating the Attribute embedding module. In practice, these attribute embedding will be concatenated with the road segment embedding to serve as the conditional information ( $c = \text{Concat}(z_{attr}, z_L)$ ) to guide trajectory generation.

### 3.3 Control Trajectory Generation

Built on the conditional embedding module and the GeoUNet noise estimation model, the diffusion model allows for controlled trajectory generation based on Eq. (5). This target is implemented through the training process and the generation process (see Algorithm 1 for details).

**Training Process.** During the training process, the framework first utilizes the encoder of pre-trained RoadMAE and attribute embedding module to convert road segment topology constraints and trip attributes into guideline embedding. Then, the GeoUNet model estimates the noise level during the reverse process based on the conditional embedding  $c$  and the diffusion step  $t$ . In this way, the goal of training a diffusion model is to minimize the mean square error between model predicted noise and the Gaussian noise  $\epsilon$ :

$$\min_{\theta} \mathcal{L}(\theta) = \min_{\theta} \mathbb{E}_{c, t, x_0 \sim q(x), \epsilon \sim \mathcal{N}(0, I)} \|\epsilon - \epsilon_\theta(x_t, t, c)\|_2^2. \quad (14)$$

**Generating Process.** As described in Algorithm 1, the model is guided by the conditional information to gradually generate trajectories from  $\tilde{x}_T \sim \mathcal{N}(0, I)$  in the generating process. Within this phase, the desired trajectories are generated from the noise step by step according to Eq. (3) and Eq. (5). In addition, we utilize the method proposed in [35] to speed up the generating process. Note that the additional computational overhead of integrating RoadMAE for model training and generation is negligible since

we pre-train RoadMAE in advance to obtain topological constraint embedding.

## 4 EXPERIMENTS

In this section, we first describe the dataset, baseline, and evaluation metrics setup of the experiment. Then, we evaluate the ControlTraj framework by comprehensive experiments on real-world datasets to answer the following research questions:

- **RQ1:** Can the trajectory generated by ControlTraj provide superior spatial-temporal fidelity compared to several state-of-the-art baselines?
- **RQ2:** How does the realism and utility of the trajectory generated by ControlTraj?
- **RQ3:** Does ControlTraj facilitate controllable trajectory generation, allowing for precise manipulation of output trajectories?
- **RQ4:** How does the RoadMAE affect the ControlTraj?
- **RQ5:** How do the transferability and generalizability features of ControlTraj perform across different geographical contexts?

### 4.1 Experimental Setups

**Datasets.** We conduct our experiments on three real-world GPS trajectory datasets: Chengdu, Xi'an, and Porto, accessed with appropriate permissions. These datasets are rich sources of GPS data, representing diverse urban mobility patterns. A comprehensive summary is provided in Appendix B.1 for reference.

**Baselines.** To benchmark the performance of our model, we compare it against a suite of leading trajectory generation methods: VAE [45], TrajGAN [44], DP-TrajGAN [29, 51], Diffwave [20], and DiffTraj [55]. Moreover, two ablation studies are conducted to evaluate specific components of our model: 1) ControlTraj w/o  $c$ , which examines the contribution of conditional embedding, and 2) ControlTraj-AE, where a conventional CNN-based autoencoder substitutes our proposed RoadMAE to assess the impact of different road segment embedding strategies on model efficacy. The implementation details are presented in Appendix B.2.

**Evaluation Metrics.** We follow the common practice of [9, 55] and use three metrics to quantitatively measure the fidelity of the generated trajectories: **Density error**, **Travel error**, and **Length error**. These metrics collectively evaluate the generated trajectories' geographic distribution, human activity representation, and intra-trajectory coherence, offering insights from both global and local perspectives. Given that ControlTraj can modify trajectory patterns, direct comparison of pattern similarity is deemed inapplicable [55]. Instead, we compute the Jensen-Shannon divergence between distributions of generated and real trajectories, averaging the results over 10 iterations for reliability. Further details are available in Appendix B.3.

**Implement Details.** The ControlTraj implementation involves various general neural network components and parameter settings. See Appendix A.2 for details.

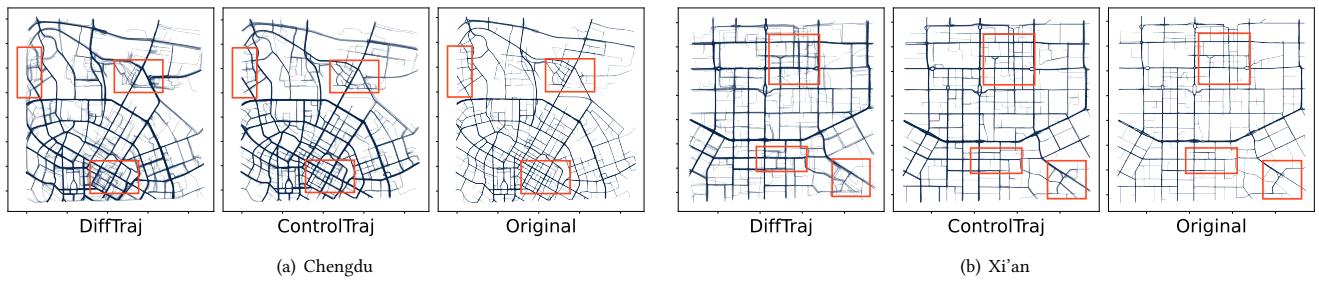
### 4.2 Overall Performance (RQ1)

**Quantitative analysis.** Table 1 summarizes the empirical performance for a suite of baseline methods on diverse real-world datasets. The results show a clear general trend that distinctly positions ControlTraj as a superior solution for trajectory generation

**Table 1: Performance comparison of ControlTraj and generative baselines.**

Methods	Chengdu			Xi'an			Porto		
	Density (↓)	Trip (↓)	Length (↓)	Density (↓)	Trip (↓)	Length (↓)	Density (↓)	Trip (↓)	Length (↓)
VAE [45]	0.0139	0.0502	0.0368	0.0237	0.0608	0.0497	0.0121	0.0224	0.0382
TrajGAN [44]	0.0137	0.0488	0.0329	0.0220	0.0512	0.0386	0.0101	0.0268	0.0332
DP-TrajGAN [51]	0.0127	0.0438	0.0234	0.0207	0.0498	0.0436	0.0109	0.0237	0.0295
Diffwave [20]	0.0145	0.0253	0.0315	0.0213	0.0343	0.0321	0.0106	0.0193	0.0266
DiffTraj [55]	0.0066	0.0143	0.0174	0.0126	0.0165	0.0203	0.0087	0.0132	0.0242
ControlTraj w/o c	0.0079	0.0264	0.0306	0.0141	0.0213	0.0295	0.0114	0.0163	0.0267
ControlTraj-AE	0.0047	0.0153	0.0162	0.0127	0.0151	0.0211	0.0071	0.0104	0.0213
<b>ControlTraj</b>	<b>0.0039</b>	<b>0.0106</b>	<b>0.0117</b>	<b>0.0104</b>	<b>0.0125</b>	<b>0.0168</b>	<b>0.0052</b>	<b>0.0096</b>	<b>0.0167</b>

**Bold** indicates the best performance over the baselines. ↓: lower is better.

**Figure 4: Visualization for different trajectory generation methods. Parameter size: DiffTraj 61.8 MB, ControlTraj 40.7 MB.**

across diverse scenarios. This satisfactory result can be largely attributed to the innovative architecture of ControlTraj, which effectively combines geo-attention with road and attribute embedding to enable a fine-grained understanding of spatial-temporal dynamics. This observation is further validated by a comparison with the most competitive baseline DiffTraj, where the trip error metrics show a substantial improvement. This remarkable difference suggests that ControlTraj, which utilizes detailed road segment embedding, reflects more consistent real-world trajectory behaviors especially for modeling trip dynamics. Further dissection of ControlTraj performance through ablation studies emphasizes the importance of each component. The performance drop in ControlTraj w/o c highlights the critical role of conditional embedding in enhancing the ability to generate contextually accurate trajectories. Without the road topology constraints, the ability of the model to forge realistic spatial-temporal patterns is diminished. Similarly, the results for ControlTraj-AE are slightly degraded compared to the full ControlTraj setup. This suggests that while the CNN-based autoencoder can capture road segment information, the tailored RoadMAE structure delivers a more robust representation of the road topology.

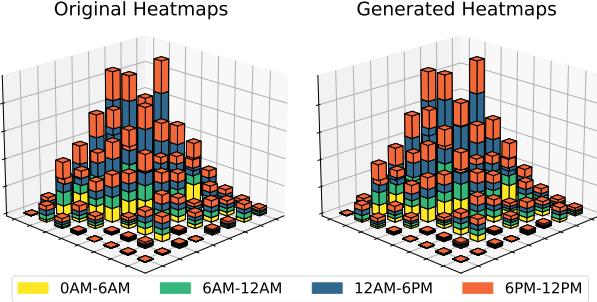
**Visualization analysis.** In addition, we also visualize the generation results of DiffTraj and ControlTraj to better compare the actual generation capabilities. Fig. 4 shows the distribution of trajectories generated in Xi'an and Chengdu context (Please refer to Appendix C.1 for a larger view and results on other cities). From the visualization, we can see that both models clearly show a geographic overview of the city. However, ControlTraj presents more fine-grained details. In particular, ControlTraj generates trajectories

that are more closely aligned with the underlying road network and conform to the physical constraints within the city for complex or sparse scenarios (marked by red boxes in the figures). These visual results further confirm the quantitative finding that ControlTraj is a superior solution for high-fidelity trajectory generation over existing methods.

### 4.3 Generated Trajectories Analysis (RQ2)

Since the generated trajectories serve to analyze human mobility patterns, their realism and utility are crucial for investigating the data generation method. In this section, we first present heatmaps of the trajectory distribution within a day for both the generated and original data. Then, we evaluate the utility of the generated data through a well-known traffic flow prediction task.

The heatmap shown in Fig. 5 vividly represents the pattern of human activity at different day periods. Specifically, colors are used to separate time, bar locations indicate urban areas, and heights reflect activity frequency that provides direct visualization of human activities over the day. Both heatmaps display prominent activity peaks during expected rush hours, which align with common commuting behaviors. For example, the increased activities during the 6 AM–12 PM and 6 PM–12 AM periods reflect the typical morning and evening peaks in urban traffic flow. Moreover, the consistency of activity levels within different areas further confirms the realism of the generated trajectories. To summarize, the generated trajectories closely mirror the original distribution in terms of spatial and temporal, suggesting that the generative model has successfully captured the human activity dynamics.

**Figure 5: Activity heatmap for periods within a day.**

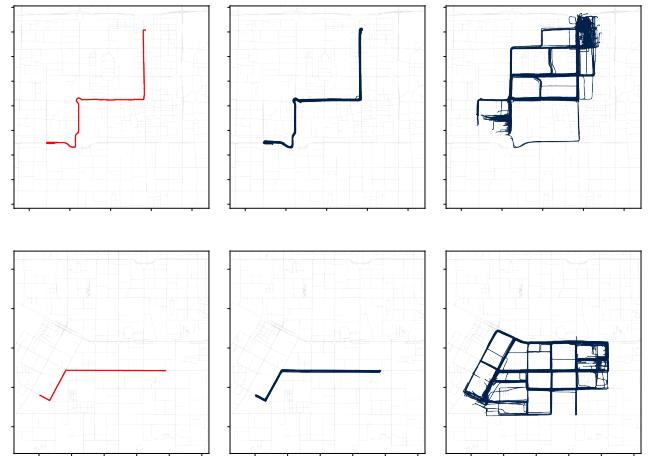
The utility of the generated data from our ControlTraj framework is further substantiated through its downstream application in traffic flow prediction. We employed a series of sophisticated neural network models, including ASTGCN [14], GWNet [43], MTGNN [42], and DCRNN [21], to compare the performance metrics when utilizing the original and the generated data. Detailed experimental settings and configurations for these tasks are comprehensively documented in **Appendix C.2**. The results, summarized in Table 2, are reported as pairs (original / generated) for three evaluation metrics: RMSE, MAE, and MAPE. Generally, these metrics reveal a welcomed similarity between the results obtained from the original and the generated data, indicating that the model-generated trajectories retain the spatial-temporal properties required for downstream traffic flow analysis. For example, the ASTGCN-based models have nearly identical results, and the difference ratios of the performance metrics are usually kept within a moderate range. These findings emphasize the potential utility of the data generated by ControlTraj, evidencing that the generated trajectories exhibit realistic movement patterns that can be exploited in complex predictive models.

**Table 2: Data utility comparison by traffic flow prediction (Chengdu). Results are expressed as (original / generated).**

Methods	ASTGCN	GWNet	MTGNN	DCRNN
RMSE	5.76 / 5.75	6.76 / 6.48	6.82 / 6.40	7.29 / 6.76
MAE	3.47 / 3.53	4.58 / 4.43	4.59 / 4.34	4.88 / 4.55
MAPE	25.47 / 25.58	29.39 / 31.70	29.73 / 30.79	32.40 / 31.94

#### 4.4 Controllable Generation (RQ3)

The capability of controlled trajectory generation is a signature of ControlTraj, enabling humans to directly interfere with the traveled geographic route of the trajectory. Figure 6 illustrates this by comparing the trajectories generated by ControlTraj (middle image) and DiffTraj (right image) against the given route planning (left image), with visualizations based on 256 generated trajectories for each model. The visualization clearly shows that the trajectories generated by the ControlTraj adhere closely to predefined routes, showcasing its ability to integrate human-specified instructions. On the other hand, DiffTraj exhibits limitations in its road guidance capabilities, as it relies solely on the start and end areas for direction [55], leading to a marked variance in the resulting travel routes from the intended path. Such divergence from the

**Figure 6: Comparison of controlled trajectory generation. Left: route planning, Middle: ControlTraj, Right: DiffTraj.**

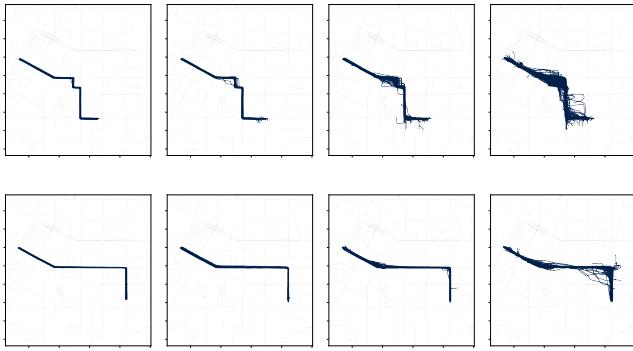
planned route underscores the shortcomings of DiffTraj in accommodating specific route preferences. In contrast, ControlTraj not only captures the physical road network accurately and aligns with human-defined routing preferences but also highlights its potential for applications that request custom trajectory planning, from urban planning to navigation systems. More controlled generation results are presented in **Appendix C.3**.

#### 4.5 Component Analysis (RQ4)

As we described in Sec. 3.1.1, the proposed RoadMAE integrates a masking strategy to enhance the robustness to cope with the missing part of the road segment. To evaluate the performance of RoadMAE for fine-grained road embedding, we investigate the influence of mask ratios on the generation capability of ControlTraj. As illustrated in Fig. 7, we test varying mask ratio {0, 0.25, 0.5, 0.75} and generate 256 trajectories for each scenario, while keeping the rest of the settings the same. The results indicate that with no masking (0 ratio), the model generates trajectories with high fidelity, closely following the intended path. As the mask ratio increases to 0.25 and 0.5, RoadMAE demonstrates uncertainty but still produces coherent trajectories that largely respect the road network topology. However, at an even higher mask ratio at 0.75, the trajectories exhibit deviations from the planned route, weakening the ability of the model to infer the correct road structure. Nonetheless, the above results still firmly demonstrate the robustness of RoadMAE, which produces satisfactory topological guidance with half of the road segment details missing. More results are presented in **Appendix C.4**

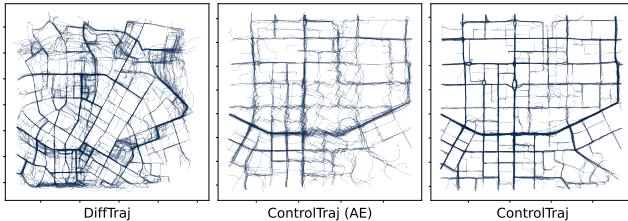
#### 4.6 Zero-shot Learning Test (RQ5)

Generalizability and transferability are paramount in evaluating the effectiveness of trajectory generation frameworks, particularly when applied to diverse realistic urban scenarios. These attributes ensure that models trained with data from one city can seamlessly adapt to new city environments without additional training – a



**Figure 7: Controllable generation with different mask ratios {0, 0.25, 0.5, 0.75} from left to right.**

concept similar to zero-shot learning. In our experiments, we train the model in one city and then generate trajectories under the guidance of a new city. We compare the performance of three models, DiffTraj, ControlTraj-AE, and ControlTraj, to evaluate their zero-shot learning capabilities. The results, as visualized in Fig. 8, reveal that ControlTraj outperforms the other models regarding both transferability and generalizability. DiffTraj struggles to maintain structural integrity when applied to a new city, as evidenced by its chaotic and unrealistic trajectory patterns. ControlTraj-AE shows improved performance but still exhibits some discrepancies from the road layouts. In stark contrast, ControlTraj maintains a high level of accuracy in the generated trajectories, closely reflecting the road network of the new city. Such performance demonstrates the generalizability and confirms its potential for practical deployment across diverse urban landscapes. Results on Xi'an → Chengdu is shown in Appendix C.5



**Figure 8: Generalizability of Chengdu → Xi'an. (Density error: DiffTraj: 0.0806, ControlTraj (AE): 0.0544, ControlTraj: 0.0171)**

## 5 RELATED WORK

**Mobility data generation.** Broadly speaking, trajectory data synthesis falls into two categories: non-generative and generative [24]. Existing research endeavors to generate trajectories that can realistically reflect the spatial-temporal properties and human behavior. The traditional approach is implemented by statistical models [33], simulations [33], or perturbations [50]. These methods rely on historical data and assumptions about mobility patterns and can provide some level of analysis of human mobility behavior.

With the widespread application of deep neural models such as Generative Adversarial Networks (GANs) [12] and Variational Autoencoders (VAEs)[45], recent research has focused on leveraging them for more complex trajectory modeling. For example, a number of methods map the trajectory data to a spatial grid and employ a GAN to generate matrixed trajectory data, thus generating trajectories [11, 27, 38, 46, 49]. However, there is a trade-off between its generation accuracy and grid size, only generating coarse-grained behavioral patterns. Furthermore, some researchers leveraged the image generation ability of GAN by transforming trajectories into images for time-series generation [2, 38], but the transformation between trajectories and images requires extra computation. In addition, researchers utilize the image generation capability of GAN to convert trajectories to images for generating [2, 38], but the translation between trajectories and images requires additional computation and fails to guarantee fidelity. A most recent work, DiffTraj [55], firstly adopted the diffusion model for trajectory synthesizing; however, it lacks controllability and generalizability.

**Diffusion model.** As a type of probabilistic generative model, the diffusion model was first introduced by Sohl-Dickstein *et al.* [34]. It is characterized by two sequential processes, i.e., a forward process that incrementally perturbs the data distribution by adding multi-scale noise, and a reverse process that learns about the data distribution for recovery [48]. Numerous attempts have focused on improving the quality of the generative samples and accelerating the sampling speed. DDPM [17] enhanced it from the aspect of optimization. Song *et al.* proposed a non-Markovian diffusion process to cut down the number of sampling steps [35], while Nichol *et al.* worked on learning the variance of reverse processes to achieve fewer steps [25]. Furthermore, Dhariwal *et al.* [7] focused on optimizing the structure of the reverse denoising neural network to enhance the sampling quality. Meanwhile, to generate an ideal object, control of the generation process is necessary. The typical method is the conditional diffusion model. For example, DDPM [17] proposed to add the conditional signal to the attention block in U-Net architecture. Furthermore, Stable Diffusion [30] introduced a cross-attention mechanism to guide the generation based on conditional information, such as texts. In this paper, we notably design a masked road segment Autoencoder for real road networks to enable controllable guidance in trajectory generation.

## 6 CONCLUSION

This paper introduces ControlTraj, a novel framework for generating high-fidelity, controllable trajectories through a topology-constrained diffusion model. Our novel approach includes the development of a RoadMAE for capturing detailed embeddings of road segments and a GeoUNet architecture that integrates topological constraints into the diffusion process. Extensively evaluated on three real-world trajectory datasets, ControlTraj has proven its ability to produce trajectories with high fidelity, flexibility, and generalizability. The results highlight the utility and its potential to simulate realistic human mobility patterns under various constraints. For future work, we aim to explore the integration of more complex topological dynamics into the ControlTraj framework to further enhance the realism and applicability of the generated trajectories in a wider range of scenarios.

## REFERENCES

- [1] Hugo Barbosa, Marc Barthelemy, Gourab Ghoshal, Charlotte R James, Maxime Lenormand, Thomas Louail, Ronaldo Menezes, José J Ramasco, Filippo Simini, and Marcello Tomasini. 2018. Human mobility: Models and applications. *Physics Reports* 734 (2018), 1–74.
- [2] Chu Cao and Mo Li. 2021. Generating Mobility Trajectories with Retained Data Utility. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2610–2620.
- [3] Wei Chen, Yuxuan Liang, Yuanshao Zhu, Yanchuan Chang, Kang Luo, Haomin Wen, Lei Li, Yanwei Yu, Qingsong Wen, Chao Chen, et al. 2024. Deep Learning for Trajectory Data Management and Mining: A Survey and Beyond. *arXiv preprint arXiv:2403.14151* (2024).
- [4] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Irshii Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.
- [5] Jian Dai, Bin Yang, Chenjun Guo, and Zhiming Ding. 2015. Personalized route recommendation using big trajectory data. In *2015 IEEE 31st international conference on data engineering*. IEEE, 543–554.
- [6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
- [7] Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems* 34 (2021), 8780–8794.
- [8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).
- [9] Yuntao Du, Yujia Hu, Zhikun Zhang, Ziquan Fang, Lu Chen, Baihua Zheng, and Yunjun Gao. 2023. LDPTTrace: Locally Differentially Private Trajectory Synthesis. *arXiv preprint arXiv:2302.06180* (2023).
- [10] Ziquan Fang, Lu Chen, Yunjun Gao, Lu Pan, and Christian S Jensen. 2021. Dra-goon: a hybrid and efficient big trajectory management system for offline and online analytics. *The VLDB Journal* 30 (2021), 287–310.
- [11] Jie Feng, Zeyu Yang, Fengli Xu, Haisu Yu, Mudan Wang, and Yong Li. 2020. Learning to simulate human mobility. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 3426–3433.
- [12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, Vol. 27.
- [13] Chenjuan Guo, Bin Yang, Jilin Hu, and Christian Jensen. 2018. Learning to route with sparse trajectory sets. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 1073–1084.
- [14] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 922–929.
- [15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 16000–16009.
- [16] Nils Henke, Shimon Wonsak, Prasenjit Mitra, Michael Nolting, and Nicolas Tempelmeier. 2023. Condtraj-gan: Conditional sequential gan for generating synthetic vehicle trajectories. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 79–91.
- [17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems* 33 (2020), 6840–6851.
- [18] Junfeng Hu, Xu Liu, Zhencheng Fan, Yuxuan Liang, and Roger Zimmermann. 2023. Towards Unifying Diffusion Models for Probabilistic Spatio-Temporal Graph Learning. *arXiv preprint arXiv:2310.17360* (2023).
- [19] Christian S Jensen. 2022. Digitalization in the Service of Society: The Case of Big Vehicle Trajectory Data. In *Proceedings of the 34th International Conference on Scientific and Statistical Database Management*. 1–1.
- [20] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. 2020. DiffWave: A Versatile Diffusion Model for Audio Synthesis. In *International Conference on Learning Representations*.
- [21] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926* (2017).
- [22] Yuxuan Liang, Kun Ouyang, Hanshu Yan, Yiwei Wang, Zekun Tong, and Roger Zimmermann. 2021. Modeling Trajectories with Neural Ordinary Differential Equations. In *IJCAI*. 1498–1504.
- [23] Yuxuan Liang, Haomin Wen, Yuqi Nie, Yushan Jiang, Ming Jin, Dongjin Song, Shirui Pan, and Qingsong Wen. 2024. Foundation Models for Time Series Analysis: A Tutorial and Survey. *arXiv preprint arXiv:2403.14735* (2024).
- [24] Massimiliano Luca, Gianni Barlaachi, Bruno Lepri, and Luca Pappalardo. 2021. A survey on deep learning for human mobility. *ACM Computing Surveys (CSUR)* 55, 1 (2021), 1–44.
- [25] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. 2021. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741* (2021).
- [26] OpenStreetMap contributors. 2017. Planet dump retrieved from <https://planet.osm.org> . <https://www.openstreetmap.org>.
- [27] Kun Ouyang, Reza Shokri, David S Rosenblum, and Wenzhuo Yang. 2018. A non-parametric generative model for human trajectories. In *IJCAI*, Vol. 18. 3812–3817.
- [28] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR, 8748–8763.
- [29] Jinmeng Rao, Song Gao, Yuhan Kang, and Qunying Huang. 2020. LSTM-TrajGAN: A Deep Learning Approach to Trajectory Privacy Protection. In *11th International Conference on Geographic Information Science (GIScience 2021)*.
- [30] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10684–10695.
- [31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 234–241.
- [32] Sijie Ruan, Cheng Long, Jie Bao, Chunyang Li, Zisheng Yu, Ruiyuan Li, Yuxuan Liang, Tianfu He, and Yu Zheng. 2020. Learning to generate maps from trajectories. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 890–897.
- [33] Filippo Simini, Gianni Barlaachi, Massimiliano Luca, and Luca Pappalardo. 2021. A deep gravity model for mobility flows generation. *Nature communications* 12, 1 (2021), 6576.
- [34] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*. PMLR, 2256–2265.
- [35] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502* (2020).
- [36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).
- [37] Sheng Wang, Zhifeng Bao, J Shane Culpepper, and Gao Cong. 2021. A survey on trajectory data management, analytics, and learning. *ACM Computing Surveys (CSUR)* 54, 2 (2021), 1–36.
- [38] Xingrui Wang, Xinyu Liu, Ziteng Lu, and Hanfang Yang. 2021. Large scale GPS trajectory generation using map based on two stage GAN. *Journal of Data Science* 19, 1 (2021), 126–141.
- [39] Zheng Wang, Kun Fu, and Jieping Ye. 2018. Learning to Estimate the Travel Time. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. Association for Computing Machinery, London, United Kingdom, 858–866.
- [40] Zheng Wang, Cheng Long, and Gao Cong. 2021. Trajectory simplification with reinforcement learning. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 684–695.
- [41] Haomin Wen, Youfang Lin, Yutong Xia, Huaiyu Wan, Qingsong Wen, Roger Zimmermann, and Yuxuan Liang. 2023. DiffSTG: Probabilistic Spatio-Temporal Graph Forecasting with Denoising Diffusion Models. *arXiv preprint arXiv:2301.13629* (2023).
- [42] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 753–763.
- [43] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph Wavenet for Deep Spatial-Temporal Graph Modeling. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (Macao, China) (IJCAI'19)*. 1907–1913.
- [44] Liu Xi, Chen Hanzhou, and Andris Clio. 2018. trajGANs: using generative adversarial networks for geo-privacy protection of trajectory data. *Vision paper* (2018).
- [45] Tianqi Xia, Xuan Song, Zipei Fan, Hiroshi Kanasugi, QuanJun Chen, Renhe Jiang, and Ryosuke Shibasaki. 2018. DeepRailway: A Deep Learning System for Forecasting Railway Traffic. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. 51–56.
- [46] Nan Xu, Loc Trinh, Sirisha Rambhatla, Zhen Zeng, Jiahao Chen, Samuel Assefa, and Yan Liu. 2021. Simulating continuous-time human mobility trajectories. In *Proc. 9th Int. Conf. Learn. Represent.* 1–9.
- [47] Yuan Xu, Jiajie Xu, Jing Zhao, Kai Zheng, An Liu, Lei Zhao, and Xiaofang Zhou. 2022. MetaPTP: an adaptive meta-optimized model for personalized spatial trajectory prediction. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2151–2159.

- [48] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Yingxia Shao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. 2022. Diffusion models: A comprehensive survey of methods and applications. *arXiv preprint arXiv:2209.00796* (2022).
- [49] Yuan Yuan, Jingtao Ding, Huandong Wang, Depeng Jin, and Yong Li. 2022. Activity Trajectory Generation via Modeling Spatiotemporal Dynamics. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4752–4762.
- [50] Paul A Zandbergen. 2014. Ensuring confidentiality of geocoded health data: assessing geographic masking strategies for individual-level data. *Advances in medicine* 2014 (2014).
- [51] Jing Zhang, Qihan Huang, Yirui Huang, Qian Ding, and Pei-Wei Tsai. 2022. DP-TrajGAN: A privacy-aware trajectory generation model with differential privacy. *Future Generation Computer Systems* (2022).
- [52] Yu Zheng. 2015. Trajectory data mining: an overview. *ACM Transactions on Intelligent Systems and Technology (TIST)* 6, 3 (2015), 1–41.
- [53] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. 2014. Urban computing: concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 5, 3 (2014), 1–55.
- [54] Yuanshao Zhu, Yi Liu, James J. Q. Yu, and Xingliang Yuan. 2021. Semi-supervised federated learning for travel mode identification from gps trajectories. *IEEE Transactions on Intelligent Transportation Systems* 23, 3 (2021), 2380–2391.
- [55] Yuanshao Zhu, Yongchao Ye, Shiya Zhang, Xiangyu Zhao, and James Jianqiao Yu. 2023. DiffTraj: Generating GPS Trajectory with Diffusion Probabilistic Model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

## A DETAILS OF CONTROLTRAJ FRAMEWORK

In this section, we provide a detailed implementation of ControlTraj, including the GeoUNet architecture, parameter settings.

### A.1 GeoUNet architecture

As illustrated in Fig. 9, ControlTraj is divided into two parts, down-sampling and upsampling, and each part consists of multiple stacked Geo Down/Up Blocks with integrated geographic attention and residual modules. To learn road segment topological constraints and form attributes, ControlTraj integrates an attribute module to embed attribute information, which is then concatenated with the road segment embedding and collectively fed to each block. Specifically, each sampling module consists of a collection of components such as residual blocks, transformers, etc. Among them, for down-sampling the max pooling layer is used and linear interpolation is applied for upsampling. For trip attribute information, such as distance, speed, departure time, travel time, etc., we follow the practice in [39, 55] and integrate a Wide and Deep module. In addition, road topology constraints are incorporated into GeoUNet through transformer layers, each of which includes a series of group normalization, attention layer, and fully connected layers. Notably, two cascaded attention mechanisms are included here, i.e., self-attention and cross-attention.

### A.2 Implementation Details

**Table 3: General setting for RoadMAE.**

Parameter	Setting value	Refer range
Mask ratio	0.5	0 ~ 0.75
Patch length	5	5 ~ 10
Encoder Blocks	8	≥ 2
Decoder Blocks	4	≥ 2
Heads	4	≥ 1
Encode Dim	128	≥ 64
Parameter size	9.2 MB	–

**Table 4: General setting for GeoUNet.**

Parameter	Setting value	Refer range
Diffusion Steps	500	300 ~ 500
Skip steps	5	1 ~ 10
Embedding dim	128	≥ 64
$\beta$ (linear schedule)	0.0001 ~ 0.05	–
Batch size	1024	≥ 256
Sampling blocks	4	≥ 3
Resnet blocks	2	≥ 1
Input Length	200	120 ~ 256
Parameter size	31.5 MB	–

In addition, we summarize the list of hyperparameters and the specific implementation settings used in this paper. Specifically, we summarize the parameter settings for RoadMAE in Table 3 and

GeoUNet in Table 4. Meanwhile, we provide reference ranges for the settings of these hyperparameters based on the results of the current work and general practical experience.

## B EXPERIMENT AND SETUP

In this section, we describe the experimental setup of this paper in detail, including the dataset, baseline implementation and evaluation metrics. Please kindly note that all experiments are implemented with PyTorch 3.9 and conducted on a single NVIDIA A100 40GB GPU.

### B.1 Dataset

We evaluate the model performance of ControlTraj and baselines on three datasets with different cities, **Chengdu**, **Xi'an**<sup>1</sup>, and **Porto**<sup>2</sup>. Table 5 shows the statistic of these datasets and Fig. 10 show the distribution, respectively. Deeper colors indicate more frequent trajectories in the region, and lighter ones indicate sparse trajectories. For all datasets, we filtered trajectories with lengths less than 120 and normalized them to a fixed length using linear interpolation.

**Table 5: Statistics of Real-world Trajectory Datasets.**

Dataset	Trajectory Number	Avg. Length	Avg. Distance
Chengdu	5 773 525	175.9	7.42 km
Xian	3 044 828	243.8	5.73 km
Porto	1 710 670	48.9	4.12 km

### B.2 Baselines

- **VAE** [45] We initially encode trajectory data into a concealed distribution using a sequence of two convolutional layers followed by a linear layer. Subsequent to this encoding, the trajectory generation is accomplished through a decoder that incorporates a linear layer and two deconvolutional layers. To maintain consistency in the dimensions of both input and output trajectories, we establish the dimensionality of the convolution and deconvolution kernels at 4.
- **TrajGAN** [44] Here, the trajectory data is amalgamated with stochastic noise before being processed through a generator, which is composed of two linear layers and two convolution layers. Following this, the discrimination process utilizes a convolutional layer coupled with a linear layer. Training of the generator and discriminator proceeds in a sequential, alternating fashion, optimizing their performance iteratively.
- **Dp-TrajGAN** [29, 51] Since adding differential privacy affects the performance of the model, here we remove the differential privacy operation. Therefore, the model is practically very similar to TrajGAN, with the difference that the backbone layer of the model is an LSTM instead of a CNN.
- **Diffwave** [20] This model is built upon the Wavenet architecture, and is tailored for synthesizing sequences like timeseries, and speech voice. This model makes significant use of dilated convolution techniques. In our implementation, we incorporate

<sup>1</sup><https://outreach.didichuxing.com/>

<sup>2</sup><https://www.kaggle.com/datasets/crailltap/taxi-trajectory/>

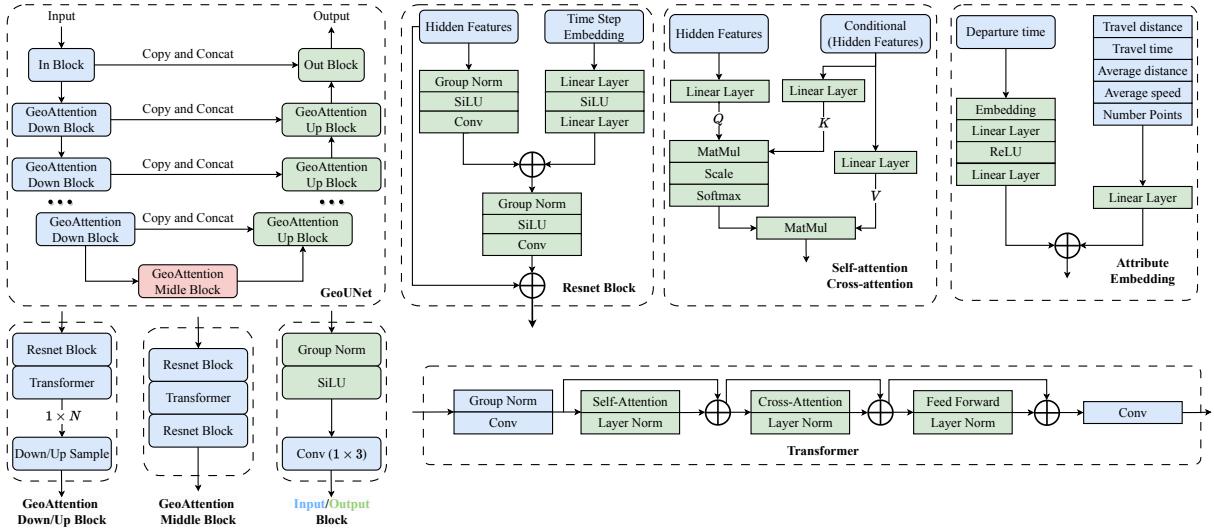


Figure 9: The main architecture and components of GeoUNet.

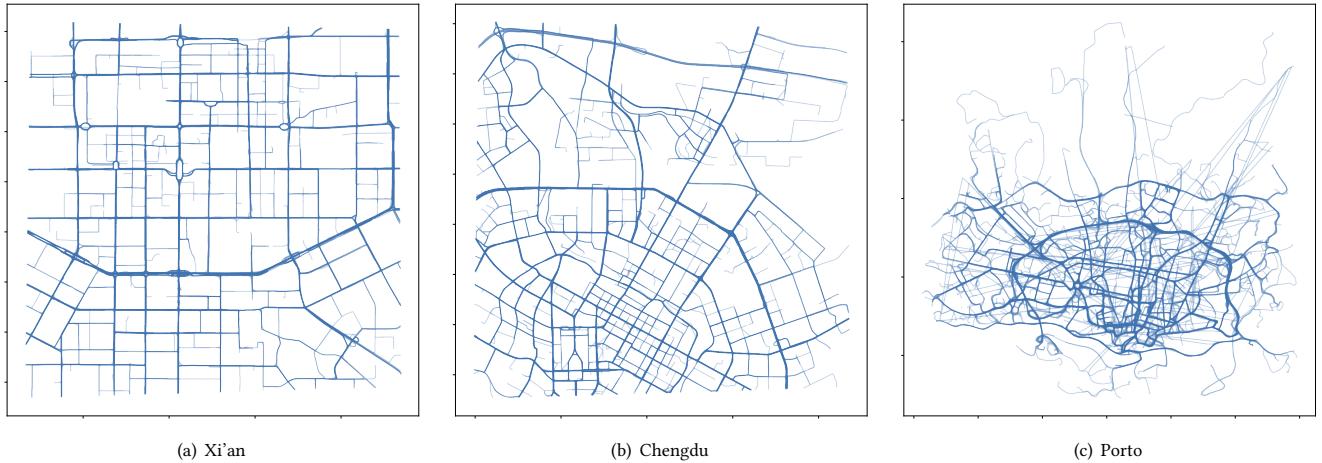


Figure 10: Origin trajectory distribution of three cities.

16 blocks connected through residual links, where each block features a bidirectional dilated convolution. The outputs from these blocks are integrated using sigmoid and tanh activation functions, respectively, before being processed by a one-dimensional Convolutional Neural Network (1D CNN).

- **DiffTraj** [55] DiffTraj is an advanced trajectory generation framework based on diffusion models. Here, we follow the model and setup in code repository<sup>3</sup>. Note that since the diffTraj does not feature topological guidance for road segments, we simply embed the regions to which the start and end areas belong as conditions.
- **ControlTraj w/o c** This baseline has the same settings and GeoUNet structure as ControlTraj. The difference is that this baseline removes the section topology constraints and attribute

information. It can be used to analyze the effect of condition information on ControlTraj.

- **ControlTraj- AE** This baseline replaces the road network topology encoding in ControlTraj, RoadMAE with a valina Autoencoder. It is used to explore the learning ability of the proposed RoadMAE for road topology constraints. Finally, the encoder embeds the road segment topology in the same dimension as RoadMAE

### B.3 Evaluation Metrics

The objective of trajectory generation is to create trajectories that mimic real-world movements to support and enhance downstream applications. To assess the degree of resemblance between the generated and actual trajectories, it is crucial to evaluate their "similarity". In alignment with methodologies adopted in prior research, such

<sup>3</sup><https://github.com/Yasoz/DiffTraj>

as highlighted by Du et al. (2023) [9, 55], we utilize the Jenson-Shannon divergence (JSD) to gauge the quality of the trajectories produced. JSD serves as a metric for comparing the distributions of real and synthesized trajectories, where a reduced JSD value signifies a closer approximation to the original statistical characteristics. Let  $P$  represent the probability distribution of the original data and  $Q$  denote the probability distribution of the generated data; the JSD is determined by the following formula:

$$\text{JSD}(P\|Q) = \frac{1}{2}\mathbb{D}(P\|M) + \frac{1}{2}\mathbb{D}(Q\|M), \quad (15)$$

where  $M = \frac{1}{2}(P + Q)$  is the mixture distribution of  $P$  and  $Q$ .

In the evaluation process, we divide each city into grids of  $16 \times 16$  size and count the frequency of trajectory points associated with each grid. Then, we can calculate each evaluation metric based on the calculated matrix. Specially, we evaluate the performance of different trajectory generation methods with following metrics:

- **Density error:** This metric counts the geographic occurrences of all trajectory points in the city and serves as a global metric to assess the similarity of geographic distribution between the entire generated trajectory and the original counterpart. Assuming that  $\mathcal{M}(G)$  is the distribution matrix of the generated trajectories and  $\mathcal{M}(P)$  is the distribution of the original trajectories, this evaluation metric is formulated as:

$$\text{Density Error} = \text{JSD}(\mathcal{M}(G), \mathcal{M}(P)), \quad (16)$$

where  $\text{JSD}(\cdot)$  represents the Jenson-Shannon divergence calculation between two distributions.

- **Trip error:** This metric evaluates the trajectory at a userlevel by analyzing the correlation between the starting and ending points of a travel trajectory. To conduct this assessment, we calculate the probability distribution of the start and end points for both the original and the generated trajectories. The JSD is then employed to measure the similarity between these distributions. By using JSD in this context, we aim to quantify the degree to which the generated trajectories accurately reflect the spatial patterns observed in the original data, with a focus on how closely the start and end points of these trips match.
- **Length error:** This metric is designed to evaluate the distribution of travel distances at the trajectory level. It involves calculating the geographical distance between consecutive points within a trajectory. By analyzing these distances, we can assess the overall travel distance distribution of the trajectories, providing insight into the spatial characteristics and movement patterns encapsulated by the data. This approach allows for a detailed examination of how closely the generated trajectories replicate the travel distances observed in the real-world trajectories they aim to mimic. Specifically, we compute the distance distributions for each of the two trajectories and then compute the JSD between the two distributions.

## C SUPPLEMENTARY EXPERIMENTS

In this section, we provide a detailed presentation and setup of the experimental effects.

### C.1 Visualization

Here, we show the trajectories generated by different trajectory generation methods and a large view comparison of the one-on-one geographic results with the original trajectories. In particular, Fig. 11 shows Xian, Fig. 12 shows Chengdu, and Fig. 13 shows Porto. The main differences between the two methods are labeled through red boxes. From the refined visualization, we can see that the trajectories generated by the ControlTraj show higher consistency with the original trajectories, especially for those complex trajectory scenarios, or regions with sparse trajectories.

### C.2 Generated Trajectories Analysis

**Table 6: Data utility comparison by traffic flow prediction (Xi'an). Results are expressed as (original / generated).**

Methods	ASTGCN	GWNet	MTGNN	DCRNN
RMSE	5.39 / 4.91	6.69 / 6.19	6.34 / 5.58	6.52 / 5.76
MAE	3.26 / 3.00	4.53 / 4.03	4.29 / 3.82	4.48 / 3.97
MAPE	23.07 / 23.83	29.42 / 34.37	28.70 / 30.95	32.82 / 32.15

In this section, we provide heat maps of the all-day trajectory distribution for the cities of Xi'an (Fig. 14) and Chengdu (Fig. 15). In addition, Fig. 16 shows the change in trip volumes and average speed over the day. Note that here we divide the city areas into smaller grids ( $12 \times 12$  and  $16 \times 16$ ) to provide a better and more detailed comparison. In addition, we did the same data availability experiments for the trajectories generated in Xi'an city, and the results are summarized in Table 6. To evaluate the performance of these tasks and the model, we adopt three metrics:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_i^n (y_i - \hat{y}_i)^2}, \quad (17)$$

$$\text{MAE} = \frac{1}{n} \sum_i^n |y_i - \hat{y}_i|, \quad (18)$$

$$\text{MAPE} = \frac{1}{n} \sum_i^n \left| \frac{y_i - \hat{y}_i}{y_i} \right|, \quad (19)$$

where  $y_i$  represents the actual traffic value and  $\hat{y}_i$  denotes the predicted traffic value for each observation.

### C.3 Controllable Trajectories Generation

Here, we provide a series of comparisons of the effects of trajectory generation using ControlTraj and ControlTraj for a variety of road segment topology constraint scenarios. The selected results are displayed in Fig. 17.

### C.4 RoadMAE Analysis

In this section, we provide more results for RoadMAE trajectory generation at different mask ratios in Fig. 18.

### C.5 Generalizability

In this section, we provide visualizations of the results of Chengdu transfer to Xi'an, and Xi'an transfer to Chengdu in a zero-shot learning scenario. The large view result are shown in Fig. 19 and Fig. 20.

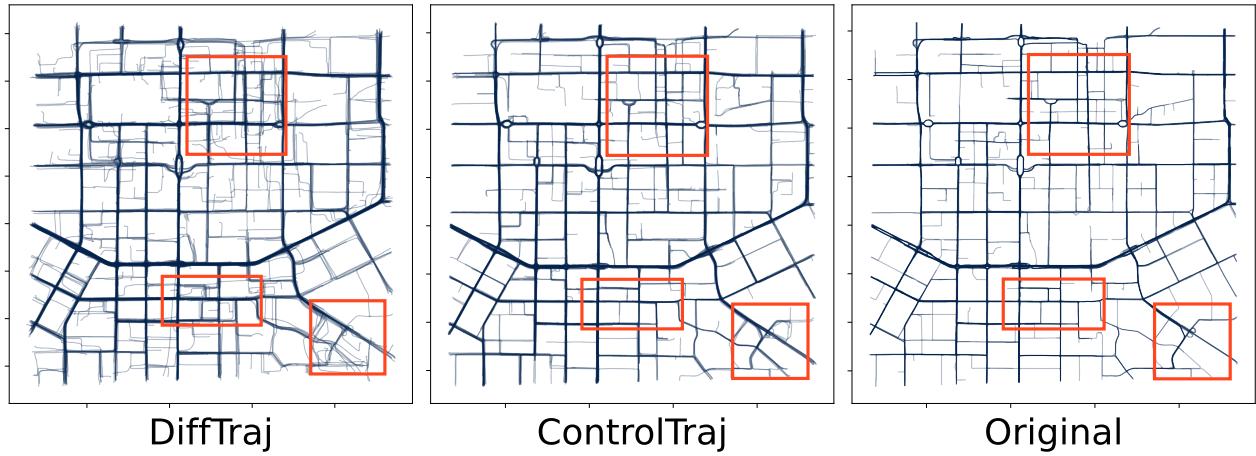


Figure 11: Generated trajectories comparison with Visualization.(Xi'an)

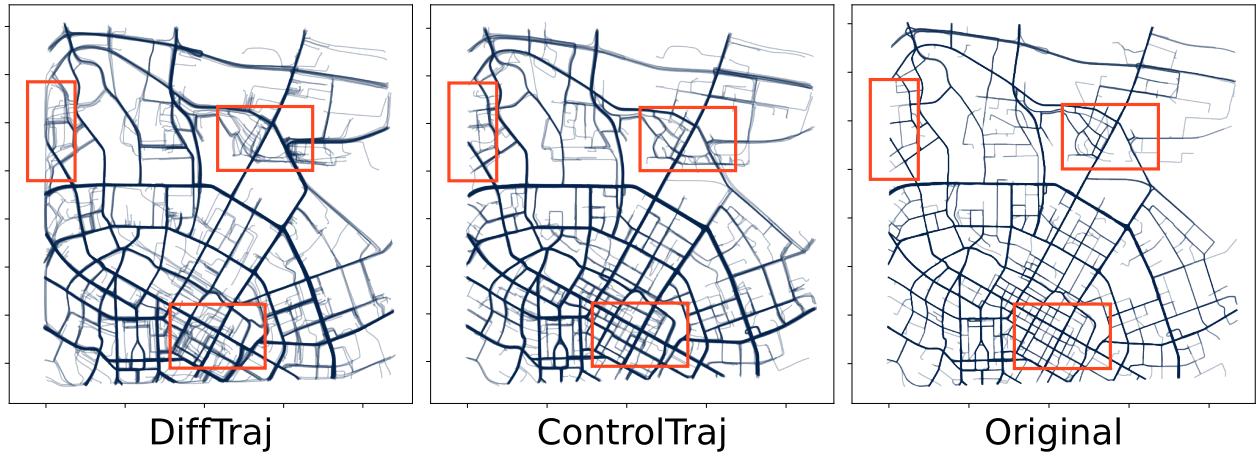


Figure 12: Generated trajectories comparison with Visualization.(Chengdu)

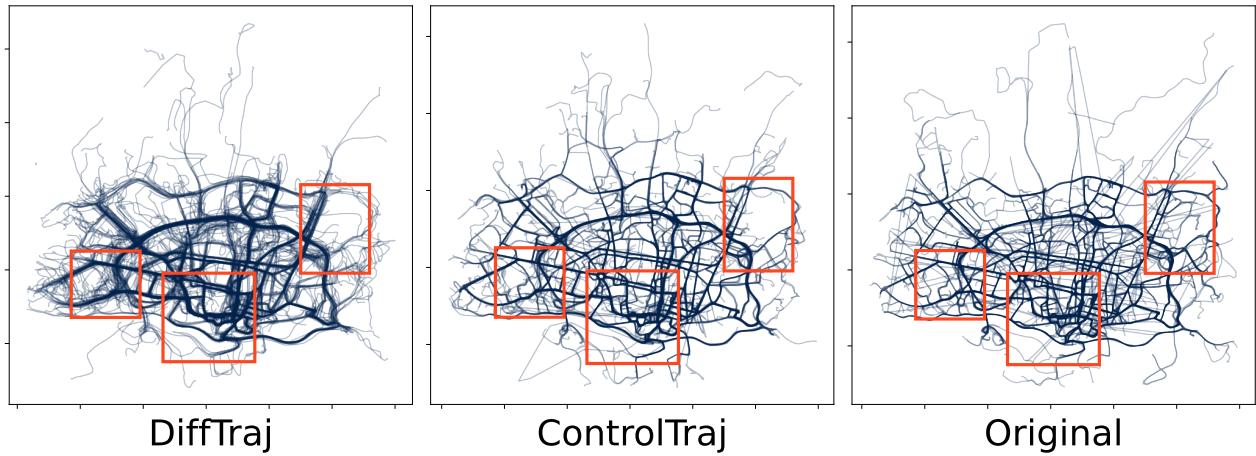
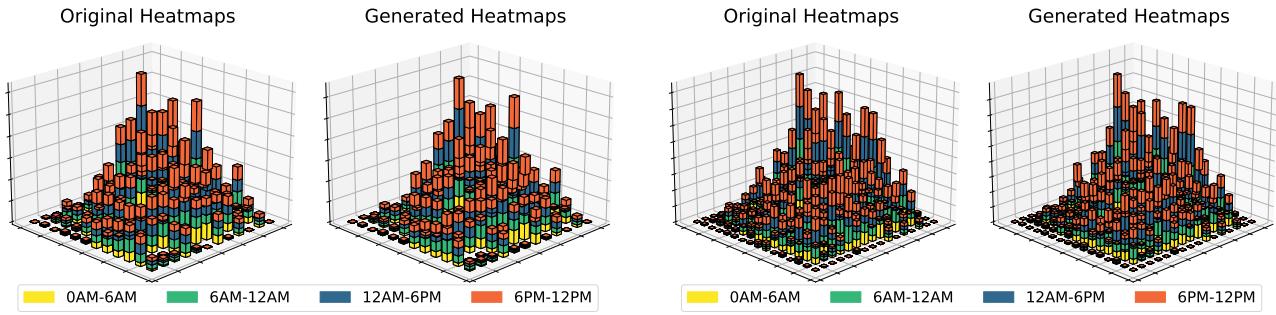
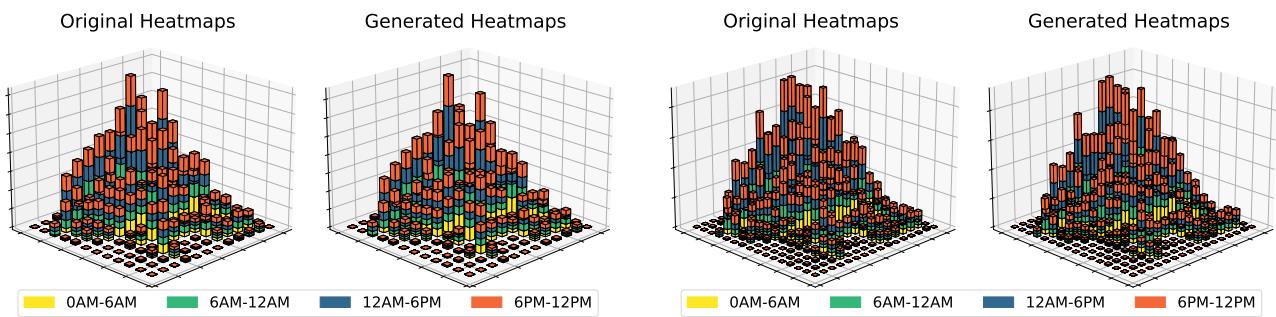


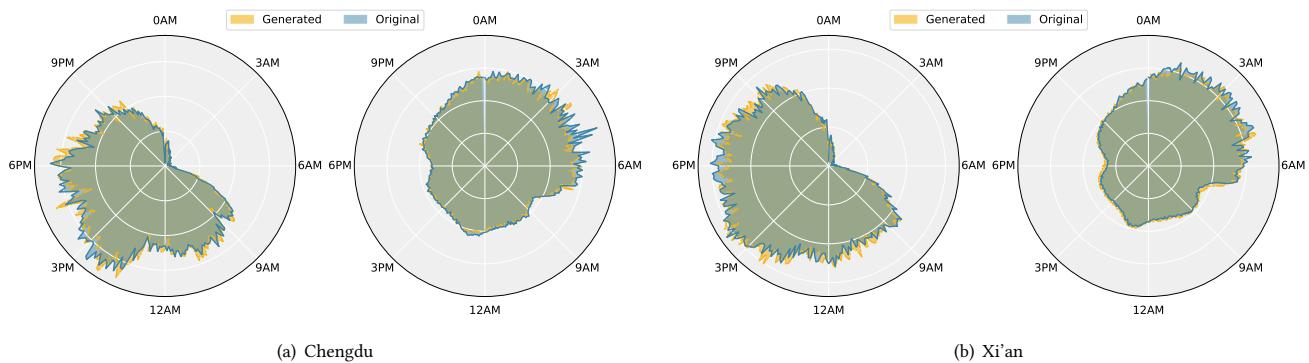
Figure 13: Generated trajectories comparison with Visualization.(Porto)



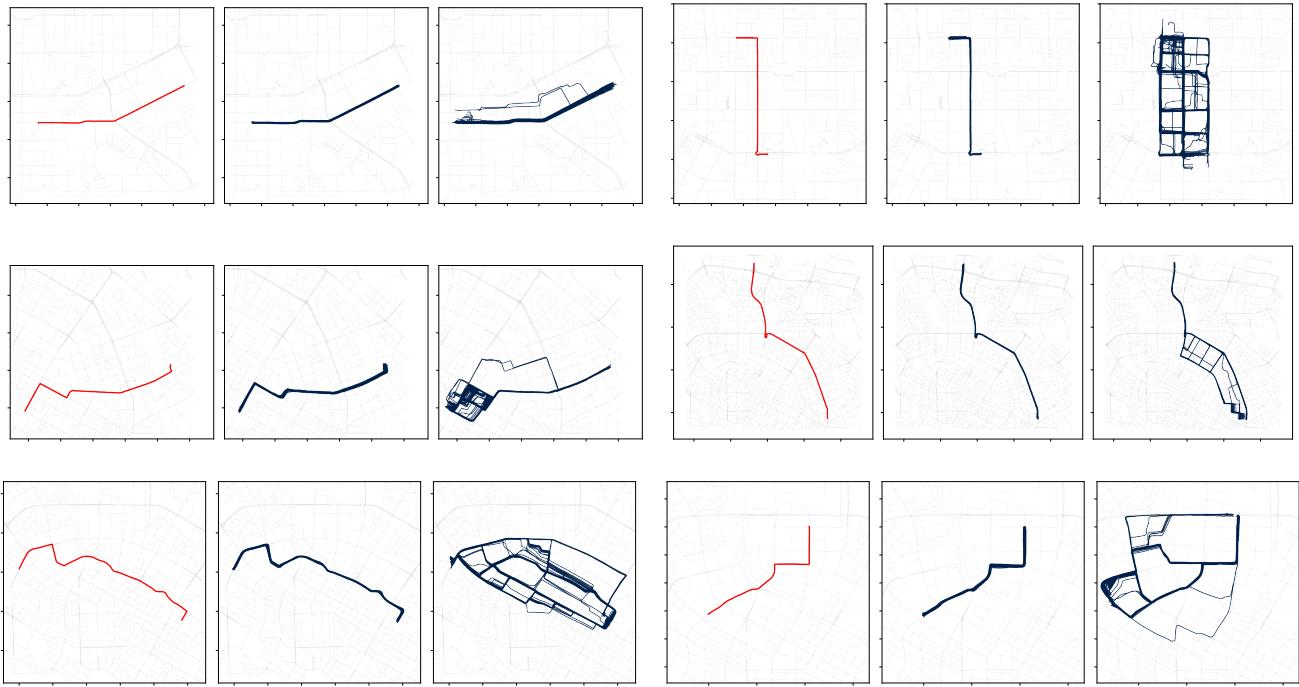
**Figure 14: Comparison of heatmaps with generated trajectories (Xi'an).**



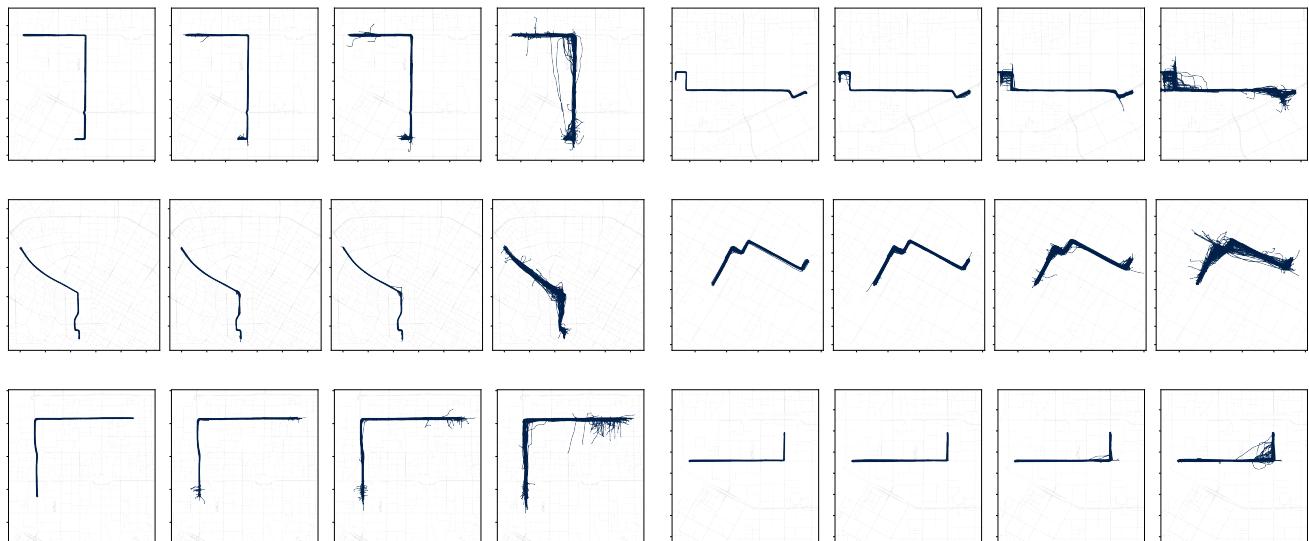
**Figure 15: Comparison of heatmaps with generated trajectories (Chengdu).**



**Figure 16: Comparison of volume of trips and average speed with generated trajectories**

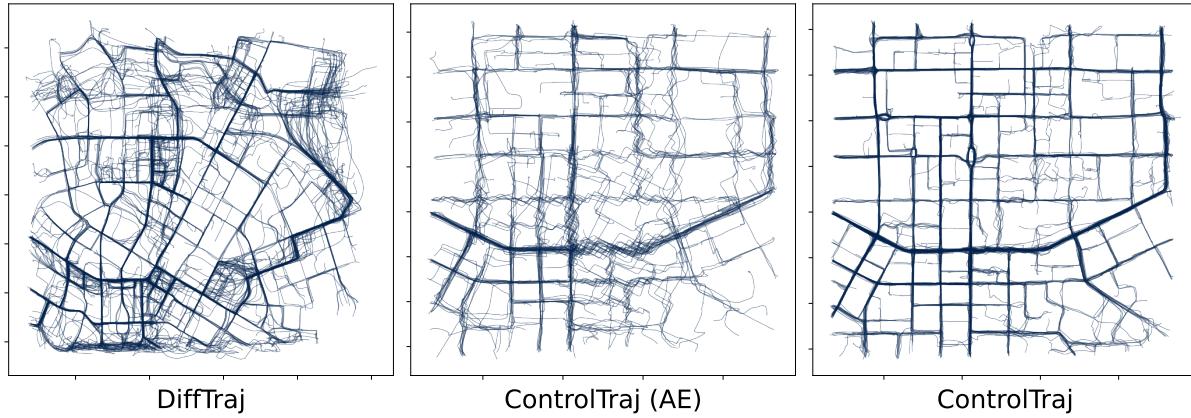


**Figure 17: Comparison of controlled trajectory generation in road segment topology constraints.**

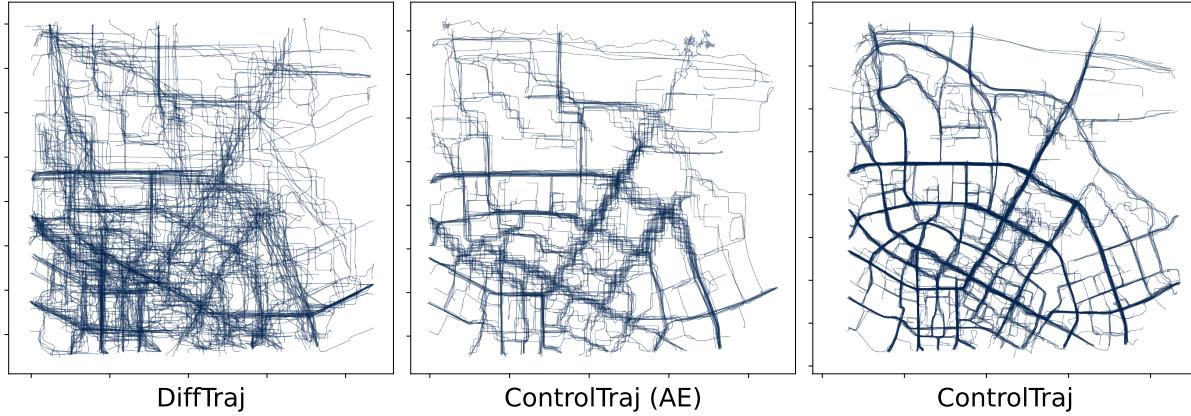


**Figure 18: Controllable generation with different mask ratio  $\{0, 0.25, 0.5, 0.75\}$  from left to right.**

ControlTraj: Controllable Trajectory Generation with Topology-Constrained Diffusion Model



**Figure 19: Zero-shot learning for Generalizability (Chengdu → Xi'an).**



**Figure 20: Zero-shot learning for Generalizability (Xi'an → Chengdu).**