DGCPATH: Distribution-Aware Generative Contrastive Framework for Self-supervised Path Representation Learning

Sean Bin Yang[1], **Hao Miao**[2*], **Zongyi Xu**[3*], **Jilin Hu**[4],
Xiangmeng Wang[2], **Hua Lu**[1], **Bin Yang**[4], **Christian S. Jensen**[1]
[1]Aalborg University, Denmark
[2]The Hong Kong Polytechnic University, Hong Kong
[3]Chongqing Univeristy of Posts and Telecommunications, China
[4]East China Normal University, China
{seany, luhua, csj}@cs.aau.dk, {hao.miao, xiangmengpoly.wang}@polyu.edu.hk,
xuzy@cqupt.edu.cn, {jlhu, byang}@dase.ecnu.edu.cn

Abstract

Due to proliferation of vehicle trajectory data from advanced sensing technologies, path representation learning has become a pivotal task in intelligent transportation systems. Although existing self-supervised approaches work to some extent, their dependence on deterministic contrastive learning paradigms and handcrafted view augmentation strategies inherently restricts cross-scenario generalization capabilities. To address these limitations, we present DGCPATH – an innovative **D**istribution-aware **G**enerative **C**ontrastive learning framework for **P**ath representation. This architecture establishes a synergistic connection between generative modeling and distributional contrastive learning, enabling the acquisition of robust and transferable feature embeddings. Specifically, our framework incorporates: (1) a diffusion-based view generator that autonomously produces semantically coherent yet diversified trajectory views from Gaussian noise distributions; (2) a variational contrastive mechanism enforcing latent feature alignment at the distribution level, transcending conventional instance-wise consistency; and (3) a novel generative cross-supervision module that reinforces view-level consistency through cross-view reconstruction learning. Comprehensive evaluations on three real-world trajectory datasets demonstrate DGCPATH's superior performance over state-of-the-art baselines in two distinct downstream tasks, validating its enhanced generalization capacity and representation effectiveness.

1 Introduction

Increasingly large volumes of vehicle path data are being accumulated that hold the potential to enable an expanding range of analyses in intelligent transportation systems [Jensen *et al.*, 2024; Zhao *et al.*, 2025; Yang *et al.*, 2023; Yang *et al.*, 2021; Han *et al.*, 2025; Li *et al.*, 2015; Li *et al.*, 2018; Li *et al.*, 2016; Li *et al.*, 2017]. Representation learning for such data is a key enabler of such analyses, as it represents raw paths by compact and informative low-dimensional vectors [Yang *et al.*, 2023; Yang *et al.*, 2021; Yang *et al.*, 2022a; Yang *et al.*, 2025; Yang *et al.*, 2026]. These representations facilitate intelligent transportation applications, such as traffic analysis [Zhao *et al.*, 2025; Zhou *et al.*, 2025; Jiang *et al.*, 2023; Wu *et al.*, 2024; Lin *et al.*, 2023; Mao *et al.*, 2025; Yu *et al.*, 2024; Pan *et al.*, 2023], path recommendation [Yang *et al.*, 2022b; Xu *et al.*, 2025], and traffic prediction [Wei *et al.*, 2025; Yang *et al.*, 2023; Fang *et al.*, 2021].

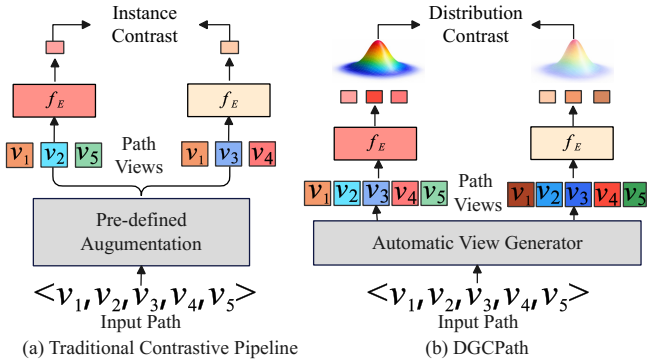

Figure 1: Traditional Contrastive Pipeline vs. DGCPATH.

Thus, substantial efforts have been devoted to learning generic path representations capable of supporting a wide range of downstream applications [Yang *et al.*, 2023; Wei *et al.*, 2025; Yang *et al.*, 2021; Xu *et al.*, 2025; Yang *et al.*, 2022a]. However, the reliance of existing proposals on deterministic contrastive loss functions is a core limitation. As illustrated in Figure 1(a), most existing proposals adopt a traditional contrastive learning pipeline that contrasts two deterministic path samples (i.e., Instance Contrast) to learn generic path representations. For example, PIM [Yang *et al.*, 2021] employs an InfoNCE-based loss to align deterministic positive samples and separate negative ones. In contrast, as shown in Figure 1(b), the contrastive formulation in Figure

*Corresponding Author: Hao Miao and Zongyi Xu

1(a) is a special case of the more general distribution-based contrastive training. The former's rigid, deterministic regime limits the expressiveness of learned representations and reduces their ability to generalize in dynamic, real-world scenarios.

A further key limitation of existing proposals is their use of pre-defined view augmentation strategies. For instance, methods like START [Jiang *et al.*, 2023] generate different path views through node dropping (e.g., as shown in Figure 1(a)) to obtain augmented path views and optimize them using deterministic contrastive objectives. Although this general approach enables strong empirical performance, it relies heavily on the use of carefully chosen augmentations, often requiring extensive trial-and-error tuning. This static and inflexible design constrains the adaptability of learned representations, particularly when transferred across difference tasks.

To overcome these limitations, we introduce DGCPATH, a novel **D**istribution-aware **G**enerative **C**ontrastive learning framework for **P**ath representation learning. Rather than enforcing consistency on deterministic feature vectors, DGCPATH models each path representation as a latent probability distribution, enabling the learning of continuous, smooth, and uncertainty-aware embeddings. By aligning distributions rather than fixed instances, DGCPATH relaxes the rigid constraints inherent to deterministic proposals and mitigates overfitting to specific samples [Kingma and Welling, 2014]. The probabilistic formulation and the resulting enhanced flexibility of the encoder improves the generalization capabilities of the learned representations markedly across different tasks.

DGCPATH consists of three main components: a path encoder, a path view generator, and a generative variational–contrastive component. First, the path encoder maps each path into a latent representation by jointly capturing its spatial structure and temporal dynamics. Second, we introduce a diffusion-based path view generator that automatically synthesizes diverse yet semantically consistent path views from random Gaussian noise. This generative mechanism facilitates the construction of informative contrastive pairs without relying on manually designed augmentation heuristics.

Next, we leverage contrastive learning to constrain the distribution of latent features such that samples originating from the same path are encouraged to share similar distributions, while those from different paths are separated in the latent space. Specifically, we unify generative modeling with contrastive learning by proposing a generative variational contrastive model that integrates the strengths of both paradigms: capturing path-invariant characteristics via generative learning, and emphasizing inter-path discriminability through contrastive objectives. We also propose a generative cross-supervision module that replaces traditional self-supervision with a cross-view alignment mechanism. This design promotes consistency in the learned feature distributions across different views of the same path, thereby enhancing the robustness and semantic fidelity of the representations.

The main contributions are as follows:

- We propose DGCPATH that integrates generative modeling and distribution-based contrastive learning to learn robust and generalizable path representations.
- We propose a diffusion-based path view generator that automatically produces semantically consistent yet diverse path views from noise, eliminating the need for handcrafted augmentation.
- We propose a generative cross-supervision module and distributional contrastive loss to enforce consistency among views of the same path while promoting separation across different paths at the distribution level.
- We report on extensive experiments on three real-world datasets and considering two downstream tasks, showing that DGCPATH can consistently outperform state-of-the-art methods in terms of both representation quality and downstream performance.

2 Related Work

Path Representation Learning Path representation learning has been studied widely due to its key role in facilitating intelligent transportation system analyses [Xu *et al.*, 2025; Wei *et al.*, 2025; Yang *et al.*, 2021; Yang *et al.*, 2022a; Yang *et al.*, 2023; Zhao *et al.*, 2025; Zhou *et al.*, 2024; Zhou *et al.*, 2025; Jiang *et al.*, 2023]. Most existing methods focus on single-modality data (i.e., path sequences). For instance, PIM [Yang *et al.*, 2021] utilizes curriculum-based negative sampling for contrastive learning on graph-based path embedding, and TPR [Yang *et al.*, 2022a] enhances temporal path representations through temporally-informed view construction, while LIGHTPATH [Yang *et al.*, 2023] introduces a lightweight and scalable framework using mask-ratio-based path view generation. From a multimodal perspective, MM-PATH [Xu *et al.*, 2025] incorporates path–image pairs and learns general representations via path–image contrastive learning, whereas PATH-LLM [Wei *et al.*, 2025] leverages path textual descriptions and large language models (LLMs) to enrich path semantics. We do not include these methods as baselines, as our study focuses exclusively on the path modality. However, two key limitations remain. First, except for multimodal approaches, existing methods generally rely on manually designed or fixed path view generation strategies. Second, both generative and contrastive objectives are typically deterministic, which can limit the diversity and generalizability of learned path representations.

3 Preliminaries

Definition 1. ***(Road Network)*** *A road network is modeled as a directed graph $G = (V, E)$, where V denotes a set of N nodes representing road intersections, and E denotes a set of directed M edges representing road segments.*

Definition 2. ***(Path)*** *A path $p = \langle v_1, v_2, v_3, \ldots, v_{|p|} \rangle$ is an ordered sequence of connected nodes. Consecutive nodes in the sequence are connected by directed edges, i.e., $(v_i, v_{i+1}) \in E$ for all $1 \leq i < |p|$, ensuring the topological continuity of the path.*

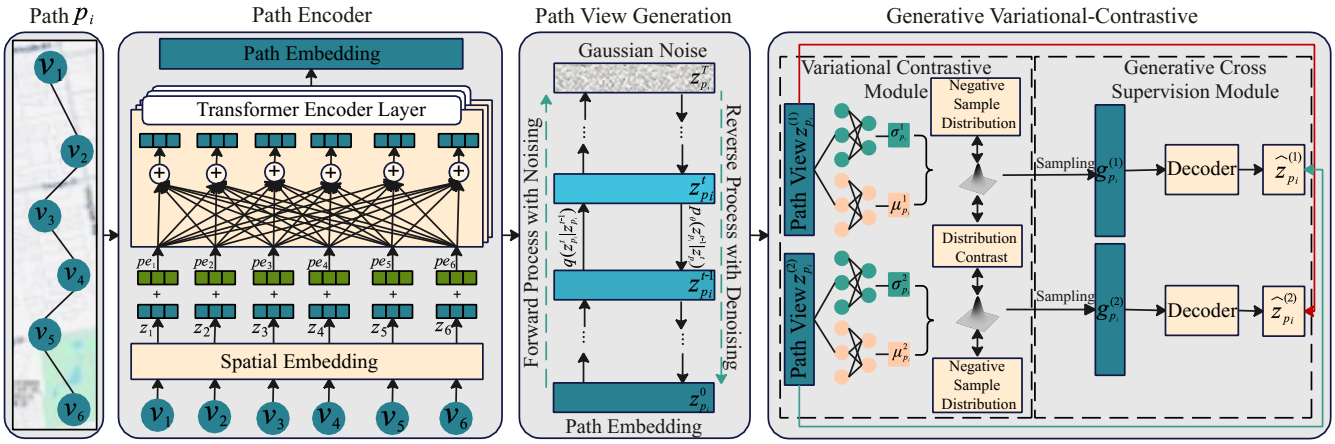

Figure 2: The DGCPATH Framework.

Problem Statement Given a road network G and a set of paths $P = \{p_i\}_{i=1}^{\mathbb{N}}$, the objective is to learn a mapping function $f_\theta : P \rightarrow \mathbb{R}^d$, such that f_θ maps each path $p_i \in P$ to a d-dimensional representation vector in a latent space, with θ denoting the learnable parameters of the function.

4 Methodology

The DGCPATH framework is illustrated in Figure 2. The core idea is to integrate the path encoder, the diffusion-based path view generator, and the generative variational contrastive component to enable more generic path represenation learning.

4.1 Path Encoder

Spatial Embedding The encoder aims to learn segment embeddings that preserve the topological structure of the road network, as such representations are fundamental to path modeling. Given a road network graph $G = (V, E)$, we employ the NODE2VEC algorithm [Grover and Leskovec, 2016] to initialize spatial embeddings for each node. These embeddings capture the structural context of the network and serve as the basis for the subsequent path representation learning, which is formulated as follows:

$$z_G = Node2vec(G(V, E)), \tag{1}$$

where $z_G \in \mathbb{R}^{d_n}$ is a set of node embeddings for road network G and d_n is the feature dimensionality.

Path Embedding The path embedding module aims to learn path representations by aggregating consecutive road segment embeddings through spatial embedding. As shown in Figure 2, for a given path $p_i = \langle v_1, v_2, v_3, v_4, v_5, v_6 \rangle$, the spatial embedding module first transforms the sequence of nodes in a path to a sequence of node embeddings for the path, i.e., $\langle z_1, z_2, \ldots, z_6 \rangle = z_G(\langle v_1, v_2, \ldots, v_t \rangle)$. Next, we employ the Transformer architecture to process the spatial embeddings (i.e., the z_i) and the positional embedding of the road segments within the path, computing the path embedding as follows:

$$\begin{aligned} h_{p_i} &= \langle z_G(v_1) + pe_1, \ldots, z_G(v_i) + pe_i \rangle, \\ z_{p_i} &= \sigma\left(\sum \frac{Transformer(h_{p_i})}{|p_i|} \right), \end{aligned} \tag{2}$$

where the v_i are the nodes in the path, z_G is the spatial embedding function, the h_{p_i} captures the node embeddings after incorporating the position embedding pe_i, $|p_i|$ is the number of nodes in a path, and z_{p_i} is the resulting path embedding. The function $Transformer(\cdot)$ refers to a Transformer architecture based on the self-attention mechanism.

4.2 Path View Generator

To facilitate path encoder learning, we propose an automatic path view generation strategy based on a diffusion and denoising process applied to the initial path embedding.

Forward Process Conditioned on the path embedding z_{p_i} generated by the path encoder, we aim to learn the underlying distribution of path embeddings to enable automatic path view generation. This distribution is modeled using a generative diffusion process, where the values of the path embedding vector are diffused progressively and then denoised over discrete time steps. Specifically, given the initial path embedding z_{p_i}, we formulate the diffusion process as a Markov chain $z_{p_i} = (z^0_{p_i}, z^1_{p_i}, \ldots, z^T_{p_i})$, where T denotes the total number of diffusion steps. Going from $z^0_{p_i}$ to $z^T_{p_i}$ (i.e., the forward process with noising), we iteratively add small amounts of Gaussian noise, gradually corrupting the original path embedding until it becomes pure Gaussian noise. This process has an analogy in the image domain, where noise is typically applied independently across dimensions [Yang *et al.*, 2024]. The corruption of path embeddings is governed by the following transition probabilities:

$$q(z^t_{p_i} | z^{t-1}_{p_i}) := \mathcal{N}\left(z^t_{p_i}; \sqrt{1-\beta_t} z^t_{p_i}, \beta_t I \right), \tag{3}$$

where $\beta_t = 1 - \alpha_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_t$.

Reverse Process The view generation process for the path embedding z_{p_i} is formulated as a reverse denoising procedure, iteratively refining the embedding from $z^T_{p_i}$ back to $z^0_{p_i}$. Specifically, starting from the fully noised embedding $z^T_{p_i}$, we progressively remove the noise step by step to ultimately recover a sample from the original data distribution. This process is defined as follows:

$$p_\theta(z^{t-1}_{p_i} | z^t_{p_i}) := \mathcal{N}\left(z^{t-1}_{p_i}; \mu_\theta(z^t_{p_i}, t), \Sigma_\theta(z^t_{p_i}, t) \right), \tag{4}$$

where Σ_θ is set to fixed, time-dependent constants to simplify the computation. To complete the reverse process, we incorporate a trainable rounding step $p_\theta(z_{p_i} \mid z^0_{p_i})$ that maps the denoised hidden states back to the final path view embedding. Specifically, we employ a learnable model $f_\theta(z^t_{p_i}, t)$ to parameterize this reverse mapping, effectively modeling $p_\theta(z_{p_i} \mid z^0_{p_i})$.

To train the diffusion model, we adopt a Kullback–Leibler (KL) divergence-based objective to enable more efficient optimization. This approach compares the reverse transition distribution $p_\theta(z^{t-1}_{p_i} \mid z^t_{p_i})$ with the corresponding posterior of the forward process, which leads to a simplified mean squared error (MSE) training loss, denoted as $\mathcal{L}_{diff}$:

$$\mathcal{L}_{diff}\left(z^0_{p_i}\right) = \sum_{t=1}^{T} \mathop{\mathbb{E}}_{q\left(z^t_{p_i} | z^0_{p_i}\right)} \left\| \mu_\theta\left(z^t_{p_i}, t\right) - \hat{\mu}\left(z^t_{p_i}, z^0_{p_i}\right) \right\|^2, \tag{5}$$

where $\mu_\theta\left(z^t_{p_i}, t\right)$ is the predicted mean of $p_\theta(z^{t-1}_{p_i} \mid z^t_{p_i})$ computed by a neural network, and $\hat{\mu}\left(z^t_{p_i}, z^0_{p_i}\right)$ denotes the mean of the posterior $q(z^{t-1}_{q_i} | z^t_{q_i}, z^0_{q_i})$, which is tractable when conditioned on $z^0_{q_i}$.

Automatic View Generation In contrastive learning, the diffusion model is employed as a path view generator to obtain multiple diverse yet semantically consistent representations of a given path embedding (i.e., $z^{(1)}_{p_i}$ and $z^{(2)}_{p_i}$). These generated views serve to improve the quality and robustness of contrastive representation learning. Given a path embedding z_{p_i}, we begin by sampling a noise vector from a standard Gaussian distribution, denoted as $\tilde{z}^T_{p_i} \sim \mathcal{N}(0, \mathrm{I})$, where I is the identity matrix. Starting from this noisy input, we iteratively apply the reverse diffusion process to progressively denoise the representation and recover a refined embedding $z^0_{p_i}$. In each reverse step, the process involves two key operations: 1) A rounding step that maps the intermediate state $z^t_{p_i}$ back into the path embedding space; 2) a replacement step in which certain components of the recovered embedding $z^{t-1}_{p_i}$ are substituted with corresponding segments from the original path embedding z_{p_i} to retain critical semantic information. This iterative procedure yields a single generated path view. To obtain a pair of views for contrastive learning, the process is performed twice with independently sampled Gaussian noise, resulting in $z^{(1)}_{p_i}$ and $z^{(2)}_{p_i}$ (see Figure 2). While originating from the same input path embedding, the stochastic nature of the initial noise introduces meaningful variation, which is essential for learning robust and discriminative representations through contrastive objectives.

4.3 Generative Variational Contrastive Module

We further detail the design of our distribution-aware generative contrastive module, which explicitly aligns distributions of positive samples while maximizing divergence from negative ones in a generative latent space [Wang *et al.*, 2024].

Distributional Contrastive Learning For each pair of path views, $z^{(1)}_{p_i}$ and $z^{(2)}_{p_i}$, generated by the automatic view generator, two independent fully connected layers, denoted as $f_\mu(\cdot)$ and $f_\sigma(\cdot)$, are employed to project the features into a high-dimensional invariant space. As a result, each sample yields two sets of parameters: $(\mu^{(1)}_{p_i}, \sigma^{(1)}_{p_i})$, $(\mu^{(2)}_{p_i}, \sigma^{(2)}_{p_i})$, corresponding to the mean and standard deviation of each view. We assume that the latent features of the two path views follow Gaussian distributions parameterized by the respective mean and standard deviation vectors.

$$z^{(1)}_{p_i}(x) \sim \mathcal{N}(x; \mu^{(1)}_{p_i}, \sigma^{(1)}_{p_i}), z^{(2)}_{p_i}(x) \sim \mathcal{N}(x; \mu^{(2)}_{p_i}, \sigma^{(2)}_{p_i}) \tag{6}$$

The KL divergence can be employed as a regularization term to enforce that the latent representations of all samples approximate a standard normal distribution, with mean 0 and standard deviation 1.

$$D_{KL}(\mathcal{N}(\mu, \sigma), \mathcal{N}(0, 1)) \tag{7}$$

In generative tasks, encouraging the model to produce samples similar to the training set is often beneficial. However, in tasks such as path recommendation and traffic analysis, forcing all samples to follow a uniform distribution can weaken the discriminative capabilities of the feature extractor. To address this issue, our method aims to constrain samples from the same path (i.e., positive samples) to follow similar Gaussian distributions, while ensuring that samples from different paths (i.e., negative samples) are mapped to distributions that are as dissimilar as possible. Achieving this objective involves computing the distance between two distributions.

A natural choice is to use the KL divergence, commonly used in Variational Autoencoders (VAEs) and defined as follows:

$$D_{KL}(z^{(1)}_{p_i}(x) \| z^{(2)}_{p_i}(x)) = \int z^{(1)}_{p_i}(x) \ln \frac{z^{(1)}_{p_i}(x)}{z^{(2)}_{p_i}(x)} dx. \tag{8}$$

However, the KL divergence is inherently asymmetric, meaning that it yields different values depending on the order of the input $z^{(1)}_{p_i}(x)$ and $z^{(2)}_{p_i}(x)$. In VAEs [Kingma and Welling, 2014], this asymmetry is not problematic, as all latent representations are explicitly regularized toward a fixed standard normal distribution with zero mean and unit variance, yielding a unidirectional mapping. However, in contrastive learning, the goal is to estimate the similarity between distributions of two augmented path views derived from the same input path. Thus, the similarity measure must be order-invariant to ensure that the positive pairs are treated symmetrically, regardless of the order in which they occur as arguments.To achieve this, we adopt the Jensen–Shannon (JS) divergence, a symmetric variant of the KL divergence. The JS divergence computes the similarity between two distributions in an order-invariant manner, making it well-suited for contrastive learning, where symmetric comparison of augmented views is essential. Formally, the JS divergence between two distributions $z^{(1)}_{p_i}(x)$ and $z^{(2)}_{p_i}(x)$ is defined as follows:

$$\begin{aligned} D_{JS}(z^{(1)}_{p_i}(x), z^{(2)}_{p_i}(x)) = &\frac{1}{2} D_{KL}\left(z^{(1)}_{p_i}(x) \| \frac{z^{(1)}_{p_i}(x) + z^{(2)}_{p_i}(x)}{2}\right) \\ &+ \frac{1}{2} D_{KL}\left(z^{(2)}_{p_i}(x) \| \frac{z^{(1)}_{p_i}(x) + z^{(2)}_{p_i}(x)}{2}\right). \end{aligned} \tag{9}$$

Here, the distance between two distributions is normalized to be in the interval $[0, 1]$, with a smaller value indicating higher similarity. Consequently, the objective $\mathcal{L}_{vc}$ of variational contrastive learning is to minimize this divergence for

positive pairs, encouraging consistency between their latent representations.

$$\mathcal{L}_{vc} = \Big\{ \min \left(JS\left(z_{p_i}^{(1)}(x), z_{p_i}^{(2)}(x)\right)\right), \max \sum_{k=1,k\neq i}^{B} JS\left(z_{p_i}^{(1)}(x), z_{p_k}^{(1)}(x)\right) + JS\left(z_{p_i}^{(1)}(x), z_{p_k}^{(2)}(x)\right) \Big\}, \tag{10}$$

where B denotes the number of samples in a batch.

In addition, from the definition of KL divergence, we have the following:

$$D_{KL}\left(z_{p_i}^{(1)}(x) \,\Big\|\, \frac{z_{p_i}^{(1)}(x) + z_{p_i}^{(2)}(x)}{2}\right) = \int z_{p_i}^{(1)}(x) \ln z_{p_i}^{(1)}(x)\,\mathrm{d}x - \int z_{p_i}^{(1)}(x) \ln \frac{z_{p_i}^{(1)}(x) + z_{p_i}^{(2)}(x)}{2}\,\mathrm{d}x. \tag{11}$$

The JS divergence between two multivariate Gaussian distributions can now be formulated as follows:

$$D_{JS}\left(z_{p_i}^{(1)}(x), z_{p_i}^{(2)}(x)\right) = -\frac{1}{4}\sum_{i=1}^{N} \mathrm{In}\frac{\left(\sigma_{p_i}^1 \sigma_{p_i}^2\right)^2}{\left(\frac{(\sigma_{p_i}^1)^2 + (\sigma_{p_i}^2)^2}{2}\right)^2} + \frac{1}{4}\sum_{i=1}^{N} \frac{(\sigma_{p_i}^1)^2 + (\sigma_{p_i}^2)^2 + 2\left(\frac{\mu_{p_i}^1 - \mu_{p_i}^2}{2}\right)^2}{\left(\frac{(\sigma_{p_i}^1)^2 + (\sigma_{p_i}^2)^2}{2}\right)^2} + 2. \tag{12}$$

The JS divergence provides a convenient metric for quantifying the distance between two probability distributions. Inspired by the SimCLR framework [Chen and He, 2021], we reformulate the contrastive distribution loss in Eq. 10 as follows:

$$\mathcal{L}_{vc} = -\ln \frac{e^{\frac{-D_{JS}\left(z_{p_i}^{(1)}(x), z_{p_i}^{(2)}(x)\right)}{t}}}{\sum_{k=1,k\neq i}^{N} e^{\frac{-D_{JS}\left(z_{p_i}^{(1)}(x), z_{p_k}^{(1)}(x)\right)}{t}} + e^{\frac{-D_{JS}\left(z_{p_i}^{(1)}(x), z_{p_k}^{(2)}(x)\right)}{t}}}, \tag{13}$$

where N is the batch size and t is the temperature parameter.

Generative Cross-Supervision Learning As illustrated in Figure 2, to encourage the encoder to capture invariant features, we incorporate a generative supervision signal into the framework. Specifically, latent embeddings $g_{p_i}^{(1)}$ and $g_{p_i}^{(2)}$ are sampled from the respective latent distributions $z_{p_i}^{(1)}(x)$ and $z_{p_i}^{(2)}(x)$. However, direct sampling from these distributions is non-differentiable, which hinders end-to-end training via backpropagation. To address this limitation, we adopt the reparameterization trick [Kingma and Welling, 2014] that enables differentiable sampling by expressing a stochastic variable as a deterministic function of the distribution parameters and a noise term:

$$g = \mu + \sigma\epsilon, \tag{14}$$

where $\epsilon \sim \mathcal{N}(0,1)$, so that $g \sim \mathcal{N}(\mu, \sigma)$. The resulting samples $g_{p_i}^{(1)}$ and $g_{p_i}^{(2)}$ are then passed to a decoder with shared parameters. The decoder, implemented as a three-layer fully connected network with batch normalization, transforms the latent samples into reconstructed path embeddings $\hat{z}_{p_i}^{(1)}$ and $\hat{z}_{p_i}^{(2)}$. To guide this reconstruction, we employ a self-supervised objective. Conventional approaches typically use a direct supervision strategy, where the original embeddings $z_{p_i}^{(1)}$ and $z_{p_i}^{(2)}$ are used to supervise their respective reconstructions $\hat{z}_{p_i}^{(1)}$ and $\hat{z}_{p_i}^{(2)}$. Instead, we introduce a cross-supervision mechanism, where $z_{p_i}^{(1)}$ is used to supervise $\hat{z}_{p_i}^{(2)}$, and $z_{p_i}^{(2)}$ is used to supervise $\hat{z}_{p_i}^{(1)}$. This design offers two key advantages: (1) it forces the model to focus on semantically invariant features rather than spatially anchored or coordinate-dependent cues; and (2) as $g_{p_i}^{(1)}$ and $g_{p_i}^{(2)}$ are sampled from two distinct distributions, the cross-supervision strategy facilitates alignment between them, thereby reinforcing the core objective of variational contrastive learning. The optimization objective of this module is formally defined as follows:

$$\mathcal{L}_{cs} = \max \left(\mathrm{sim}\left(\hat{z}_{p_i}^{(1)}, z_{p_i}^{(2)}\right) + \mathrm{sim}\left(\hat{z}_{p_i}^{(2)}, z_{p_i}^{(1)}\right)\right), \tag{15}$$

where $\mathrm{sim}(\cdot)$ denotes the similarity between two path embeddings and is computed via MSE.

As a result, the final objective is formulated as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{diff} + \lambda_2 \mathcal{L}_{vc} + \lambda_3 \mathcal{L}_{cs}, \tag{16}$$

where λ_1, λ_2, and λ_3 are weight hyper-parameters that are used for balancing the contributions of the individual losses. Specifically, we adopt an uncertainty-based weighting strategy to dynamically adjust each loss component during training [Kendall *et al.*, 2018].

5 Experiments

Datasets We conduct experiments on three real-world datasets: Aalborg (Denmark) [Yang *et al.*, 2022b], Chengdu (China) [Yuan *et al.*, 2022], and Harbin (China) [Li *et al.*, 2019]. Each dataset includes a road network extracted from OpenStreetMap and GPS trajectories. The Aalborg road network consists of 10,017 nodes and 11,597 edges; Chengdu comprises 6,632 nodes and 17,038 edges; and Harbin contains 8,497 nodes and 14,497 edges. After map-matching [Yang and Gidófalvi, 2018; Han *et al.*, 2026], we obtain 33,264 paths for Aalborg, 54,491 for Chengdu, and 259,615 for Harbin.

Downstream Tasks *Path Travel Time Estimation:* Each path is associated with a travel time (seconds) obtained from trajectories. We aim to build a regression model to estimate the travel times of paths. We evaluate the accuracy of the estimations using Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and Root Mean Square Error (RMSE). Smaller values indicate higher estimation accuracy. *Path Ranking:* Given a set of candidate paths—typically sharing the same source and destination—each path is assigned a ranking score within the range $[0, 1]$, reflecting its relative preference based on historical trajectory data. Follow previous studies [Yang *et al.*, 2022b; Yang and Yang, 2020; Yang and Yang, 2019], we compute these scores using similarity measurements derived from real-world trajectories. To

	Method	Aalborg			Chengdu			Harbin		
		MAE↓	MAPE(%)↓	RMSE↓	MAE↓	MAPE(%)↓	RMSE↓	MAE↓	MAPE(%)↓	RMSE↓
Supervised	PATHRANK	2.913	18.713	4.832	5.802	34.014	8.894	4.127	14.339	5.301
	HIERETA	3.097	18.620	5.058	5.060	28.820	7.921	4.897	16.506	5.961
	COMPACTETA	3.030	19.726	4.791	4.832	28.678	7.447	4.372	14.974	5.362
	HMTRL	3.515	19.955	5.792	4.861	28.529	7.473	4.704	15.543	5.803
Unsupervised	NODE2VEC	5.701	35.485	8.165	5.997	35.919	8.653	5.399	18.709	6.773
	TOAST	3.439	21.311	5.555	4.918	28.567	7.513	4.627	16.101	5.803
	PIM	3.471	21.314	5.640	4.924	28.553	7.493	4.538	15.713	5.682
	START	2.965	18.432	4.979	5.174	30.769	7.726	4.402	15.262	5.503
	LIGHTPATH	2.920	18.254	4.932	4.850	28.387	7.406	4.221	14.962	5.310
	RED	2.457	14.907	3.809	4.845	28.380	7.296	4.097	14.214	5.153
	DGCPATH	**2.131**	**13.654**	**3.725**	**4.659**	**27.488**	**7.064**	**4.016**	**13.905**	**5.042**

Table 1: Overall Accuracy at Travel Time Estimation.

	Method	Aalborg			Chengdu			Harbin		
		MAE↓	TAU(τ)↑	RHO(ρ)↑	MAE↓	TAU(τ)↑	RHO(ρ)↑	MAE↓	TAU(τ)↑	RHO(ρ)↑
Supervised	PATHRANK	0.159	0.727	0.751	0.123	0.703	0.742	0.145	0.821	0.832
	HIERETA	0.152	0.757	0.767	0.178	0.735	0.774	0.133	0.846	0.879
	COMPACTETA	0.169	0.698	0.709	0.195	0.661	0.705	0.160	0.796	0.833
	HMTRL	0.163	0.729	0.741	0.149	0.779	0.789	0.133	0.838	0.874
Unsupervised	NODE2VEC	0.253	0.632	0.637	0.246	0.464	0.505	0.211	0.468	0.512
	TOAST	0.138	0.768	0.779	0.131	0.734	0.752	0.161	0.840	0.878
	PIM	0.141	0.764	0.777	0.125	0.725	0.745	0.119	0.879	0.911
	START	0.178	0.839	0.859	0.228	0.638	0.679	0.151	0.859	0.890
	LIGHTPATH	0.118	0.781	0.791	0.195	0.711	0.748	0.122	0.881	0.913
	RED	0.109	0.819	0.823	0.129	0.797	0.833	0.127	0.858	0.865
	DGCPATH	**0.107**	**0.846**	**0.870**	**0.114**	**0.819**	**0.859**	**0.112**	**0.898**	**0.925**

Table 2: Overall Accuracy at Path Ranking.

evaluate the performance of path ranking, we use three widely used metrics: Mean Absolute Error (MAE), the Kendall rank correlation coefficient (τ), and the Sperman rank correlation coefficient (ρ). These metrics collectively assess both the accuracy of the predicted ranking scores and the consistency of the predicted orderings relative to the ground truth.

Baselines We consider 10 baseline methods, including 6 unsupervised and 4 supervised methods. The unsupervised methods include NODE2VEC [Grover and Leskovec, 2016], TOAST [Chen *et al.*, 2021], PIM [Yang *et al.*, 2021], START [Jiang *et al.*, 2023], LIGHTPATH [Yang *et al.*, 2023], and RED [Zhou *et al.*, 2024]. Among them, PIM, START, and LIGHTPATH rely on pre-defined view augmentation strategies, such as random masking or node dropping. The supervised methods include PathRank [Yang *et al.*, 2022b], HIERETA [Chen *et al.*, 2022], COMPACTETA [Fu *et al.*, 2020], and HMTRL [Liu *et al.*, 2023]. These methods leverage task-specific labels to learn path representations.

For all unsupervised methods, we first pre-train the path encoder on unlabeled trajectory data (i.e., 30/50/200K samples from Aalborg/Chengdu/Harbin) to obtain path embeddings. These embeddings are subsequently fed to a regression model trained on a smaller labeled subset (e.g., 12K labeled paths from Aalborg) to predict travel time and path ranking scores. We adopt the Gradient Boosting Regressor (GBR) [Peter *et al.*, 2017] as the predictive model due to the regression nature of our tasks. In contrast, supervised methods train the path encoder end-to-end using only the labeled data (e.g., 12K samples per city) without leveraging unlabeled paths.

Impementation Details We employ a Transformer architecture and randomly initialize all learnable parameters with uniform distributions. We employ node2vec to embed each node to 128-dimensional vectors and set the dimension for path representation to 128. For a fair comparison, we set the path representation dimensionality of all baseline methods as 128. We use the AdamW optimizer with a cosine decay learning rate schedule over 600 epochs, with a warm-up period of 50 epochs. The source code is available at https://github.com/Sean-Bin-Yang/DGCPath.

5.1 Performance Findings

We first evaluate the accuracy of all methods on the downstream tasks. Tables 1 and 2 show the performance of all methods at travel time estimation and path ranking tasks on the three datasets. The overall best performance is marked in bold, and the second-best performance is underlined. We make three main observations: 1) DGCPATH achieves the best performance across all baselines at both tasks on all three datasets. Specifically, at the travel time estimation task on the Aalborg dataset, DGCPATH reduces MAE and RMSE by up to 13.27% and 2.21%, respectively. At the path ranking task on the Harbin dataset, it increases Kendall's τ and Spearman's ρ by up to 4.66% and 6.94%, respectively. These im-

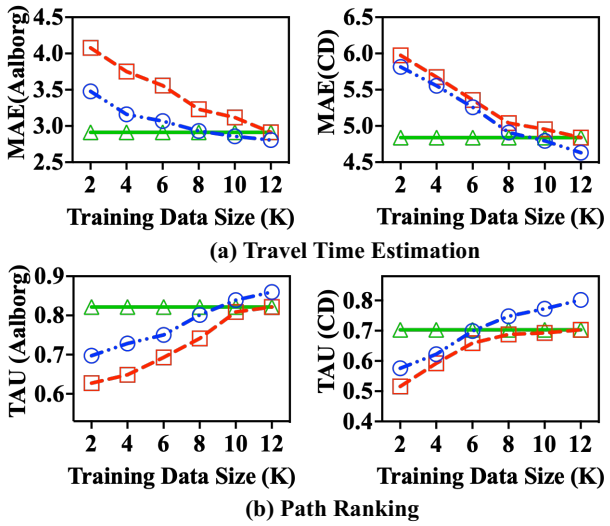

Figure 3: Effects of Pre-training.

provements stem from the improved feature learning capabilities of the proposed diffusion-based view generator. Furthermore, the use of a distribution-aware generative contrastive loss enables the model to learn more generalizable path representations, leading to enhanced overall performance. 2) The supervised methods, including PATHRANK, HIERETA, COMPACTETA, and HMTRL, show relatively poor performance, primarily due to the limited size of the labeled training data. Collecting task-specific labels is often expensive and labor-intensive. The experiments are thus performed in a realistic setting where labeled data is scarce. 3) All unsupervised methods are capable of learning generic path representations. Methods like TOAST, PIM, START, LIGHTPATH, and RED consistently outperform NODE2VEC across all three datasets. This is because NODE2VEC, originally designed for node-level representation, fails to capture the sequential dependencies between nodes in a path. In contrast, the other unsupervised methods are designed specifically for path representation learning and recognize the importance of effectively modeling spatial dependencies. Among them, DGCPATH achieves superior performance due to its integration of a diffusion-based view generator and a distribution-aware contrastive learning framework, enabling it to better capture both spatial structure and representation diversity.

5.2 Ablation Study

We study three additional variations of our method to evaluate the contribution of each key component: (1) *w/o DVG* excludes the diffusion-based view generator, (2) *w/o VCM* excludes the variational contrastive module, and (3) *w/o GCSM* excludes the generative cross-supervision module. The results, in Table 3, show that all components contribute to the performance of DGCPATH at both tasks, as removing any of the components yields a performance drop. However, the extent of each component's impact varies by task. Notably, the *VCM* plays a more critical role than *DVG* and *GCSM*, indicating its importance in learning generalizable path representations. Removing *DVG* or *GCSM* also degrades performance, highlighting the complementary strengths of all modules.

Methods	Aalborg	
	Travel Time Estimation	Path Ranking
	MAE/MAPE/RMSE	MAE/τ/ρ
w/o DVG	2.33/14.23/3.89	0.12/0.80/0.81
w/o VCM	2.64/16.05/4.75	0.13/0.77/0.80
w/o GCSM	2.31/14.74/4.01	0.14/0.83/0.84
DGCPATH	**2.13/13.65/3.73**	**0.11/0.85/0.87**
Methods	**Chengdu**	
	Travel Time Estimation	Path Ranking
	MAE/MAPE/RMSE	MAE/τ/ρ
w/o DVG	4.66/27.49/7.06	0.13/0.79/0.83
w/o VCM	4.84/28.44/7.28	0.15/0.77/0.80
w/o GCSM	4.74/27.85/7.21	0.14/0.81/0/84
DGCPath	**4.66/27.49/7.06**	**0.11/0.82/0.86**
Methods	**Harbin**	
	Travel Time Estimation	Path Ranking
	MAE/MAPE/RMSE	MAE/τ/ρ
w/o DVG	4.12/14.36/5.14	0.12/0.89/0.91
w/o VCM	4.43/15.92/5.52	0.13/0.86/0.89
w/o GCSM	4.24/14.94/5.29	0.13/0.88/0.90
DGCPATH	**4.02/13.91/5.04**	**0.11/0.90/0.93**

Table 3: Effects of DVG, VCM Loss, and GCSM Loss

5.3 Effects of Pre-training

To further assess the benefits of the proposed method, we use DGCPATH to pre-train the Transformer-based path encoder for the supervised model PATHRANK. Specifically, DGCPATH is first trained in an unsupervised manner, after which the learned parameters are transferred to initialize the Transformer encoder in PATHRANK. Figure 3 presents the performance of PATHRANK with and without pre-training at travel time estimation and path ranking, under varying labeled data sizes: 2K, 4K, 6K, 8K, 10K, and 12K. We observe that with pre-training, PATHRANK achieves comparable performance to its non-pretrained counterpart using significantly fewer labeled training samples. For instance, at the travel time estimation task, PATHRANK with pre-training requires only 8K and 9K labeled samples on Aalborg and Chengdu, respectively, to match the performance of PATHRANK trained from scratch with 12K labeled samples. When PATHRANK is trained using the full 12K labeled samples with pre-training, it outperforms the non-pretrained counterpart.

6 Conclusion

We propose DGCPATH, a distribution-aware generative contrastive framework for path representation learning. Rather than relying on hand-crafted augmentations, DGCPATH employs a learnable diffusion-based generator to produce two path views. It then uses a distribution-aware contrastive objective to align positive-pair distributions while separating positive and negative pairs. A generative cross-supervision module further enhances encoder training, yielding more generalizable path representations and improved performance across downstream tasks.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (Nos. 62472174 and 62402082), the Independent Research Fund Denmark (No. 1032-00481B), the ECNU Multifunctional Platform for Innovation (No. 001), the SCRI of The Hong Kong Polytechnic University (No. Q-CDDG), the New Chongqing Youth Innovation Talent Program (No. CSTB2025YITP-QCRCX0034), the Natural Science Foundation Project of Chongqing, Chongqing Science and Technology Commission (Nos. CSTB2025NSCQ-GPX1276 and 202502ZYDF029), and the Chongqing Municipal Education Commission (No. KJQN202400637).

References

[Chen and He, 2021] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, pages 15750–15758, 2021.

[Chen *et al.*, 2021] Yile Chen, Xiucheng Li, Gao Cong, Zhifeng Bao, Cheng Long, Yiding Liu, Arun Kumar Chandran, and Richard Ellison. Robust road network representation learning: When traffic patterns meet traveling semantics. In *CIKM*, pages 211–220, 2021.

[Chen *et al.*, 2022] Zebin Chen, Xiaolin Xiao, Yue-Jiao Gong, Jun Fang, Nan Ma, Hua Chai, and Zhiguang Cao. Interpreting trajectories from multiple views: A hierarchical self-attention network for estimating the time of arrival. In *KDD*, pages 2771–2779, 2022.

[Fang *et al.*, 2021] Ziquan Fang, Lu Pan, Lu Chen, Yuntao Du, and Yunjun Gao. MDTP: A multi-source deep traffic prediction framework over spatio-temporal trajectory data. *PVLDB*, 14(8):1289–1297, 2021.

[Fu *et al.*, 2020] Kun Fu, Fanlin Meng, Jieping Ye, and Zheng Wang. Compacteta: A fast inference system for travel time prediction. In *KDD*, pages 3337–3345, 2020.

[Grover and Leskovec, 2016] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *KDD*, pages 855–864, 2016.

[Han *et al.*, 2025] Chengkai Han, Jingyuan Wang, Yongyao Wang, Xie Yu, Hao Lin, Chao Li, and Junjie Wu. Bridging traffic state and trajectory for dynamic road network and trajectory representation learning. In *AAAI*, pages 11763–11771, 2025.

[Han *et al.*, 2026] Chenxu Han, Sean Bin Yang, and Jilin Hu. Diffmm: Efficient method for accurate noisy and sparse trajectory map matching via one step diffusion. In *AAAI*, pages 14783–14791, 2026.

[Jensen *et al.*, 2024] Christian S. Jensen, Bin Yang, Chenjuan Guo, Jilin Hu, and Kristian Torp. Routing with massive trajectory data. In *ICDE*, pages 5542–5547, 2024.

[Jiang *et al.*, 2023] Jiawei Jiang, Dayan Pan, Houxing Ren, Xiaohan Jiang, Chao Li, and Jingyuan Wang. Self-supervised trajectory representation learning with temporal regularities and travel semantics. In *ICDE*, pages 843–855, 2023.

[Kendall *et al.*, 2018] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, pages 7482–7491, 2018.

[Kingma and Welling, 2014] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[Li *et al.*, 2015] Yongfu Li, Bin Yang, Taixiong Zheng, Yinguo Li, Mingyue Cui, and Srinivas Peeta. Extended-state-observer-based double-loop integral sliding-mode control of electronic throttle valve. *IEEE Trans. Intell. Transp. Syst.*, 16(5):2501–2510, 2015.

[Li *et al.*, 2016] Yongfu Li, Yuhao Kang, Bin Yang, Srinivas Peeta, Li Zhang, Taixong Zheng, and Yinguo Li. A sliding mode controller for vehicular traffic flow. *Phys. A: Stat. Mech. Appl.*, 462:38–47, 2016.

[Li *et al.*, 2017] Yongfu Li, Yu Song, Bin Yang, Taixiong Zheng, Huizong Feng, and Yinguo Li. A new lattice hydrodynamic model considering the effects of bilateral gaps on vehicular traffic flow. *Nonlinear Dyn.*, 87(1):1–11, 2017.

[Li *et al.*, 2018] Yongfu Li, Huan Yang, Bin Yang, Taixiong Zheng, and Chao Zhang. An extended continuum model incorporating the electronic throttle dynamics for traffic flow. *Nonlinear Dyn.*, 93(4):1923–1931, 2018.

[Li *et al.*, 2019] Xiucheng Li, Gao Cong, Aixin Sun, and Yun Cheng. Learning travel time distributions with deep generative model. In *WWW*, pages 1017–1027, 2019.

[Lin *et al.*, 2023] Yan Lin, Huaiyu Wan, Jilin Hu, Shengnan Guo, Bin Yang, Youfang Lin, and Christian S. Jensen. Origin-destination travel time oracle for map-based services. *Proc. ACM Manag. Data*, 1(3):217:1–217:27, 2023.

[Liu *et al.*, 2023] Hao Liu, Jindong Han, Yanjie Fu, Yanyan Li, Kai Chen, and Hui Xiong. Unified route representation learning for multi-modal transportation recommendation with spatiotemporal pre-training. *VLDB J.*, 32(2):325–342, 2023.

[Mao *et al.*, 2025] Xiaowei Mao, Yan Lin, Shengnan Guo, Yubin Chen, Xingyu Xian, Haomin Wen, Qisen Xu, Youfang Lin, and Huaiyu Wan. Dutytte: Deciphering uncertainty in origin-destination travel time estimation. In *AAAI*, pages 12390–12398, 2025.

[Pan *et al.*, 2023] Zhicheng Pan, Yihang Wang, Yingying Zhang, Sean Bin Yang, Yunyao Cheng, Peng Chen, Chenjuan Guo, Qingsong Wen, Xiduo Tian, Yunliang Dou, Zhiqiang Zhou, Chengcheng Yang, Aoying Zhou, and Bin Yang. Magicscaler: Uncertainty-aware, predictive autoscaling. *Proc. VLDB Endow.*, 16(12):3808–3821, 2023.

[Peter *et al.*, 2017] Sven Peter, Ferran Diego, Fred A. Hamprecht, and Boaz Nadler. Cost efficient gradient boosting. In *NeurIPS*, pages 1551–1561, 2017.

[Wang *et al.*, 2024] Bo-Hua Wang, Zhiqiang Tian, Aixue Ye, Feng Wen, Shaoyi Du, and Yue Gao. Generative variational-contrastive learning for self-supervised point cloud representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(9):6154–6166, 2024.

[Wei *et al.*, 2025] Yongfu Wei, Yan Lin, Hongfan Gao, Ronghui Xu, Sean Bin Yang, and Jilin Hu. Path-llm: A multi-modal path representation learning by aligning and fusing with large language models. In *WWW*, pages 2289–2298, 2025.

[Wu *et al.*, 2024] Xinle Wu, Xingjian Wu, Bin Yang, Lekui Zhou, Chenjuan Guo, Xiangfei Qiu, Jilin Hu, Zhenli Sheng, and Christian S. Jensen. Autocts++: zero-shot joint neural architecture and hyperparameter search for correlated time series forecasting. *VLDB J.*, 33(5):1743–1770, 2024.

[Xu *et al.*, 2025] Ronghui Xu, Hanyin Cheng, Chenjuan Guo, Hongfan Gao, Jilin Hu, Sean Bin Yang, and Bin Yang. Mm-path: Multi-modal, multi-granularity path representation learning. In *KDD*, pages 1703–1714, 2025.

[Yang and Gidófalvi, 2018] Can Yang and Gyözö Gidófalvi. Fast map matching, an algorithm integrating hidden markov model with precomputation. *Int. J. Geogr. Inf. Sci.*, 32(3):547–570, 2018.

[Yang and Yang, 2019] Sean Bin Yang and Bin Yang. Pathrank: A multi-task learning framework to rank paths in spatial networks. *CoRR*, abs/1907.04028, 2019.

[Yang and Yang, 2020] Sean Bin Yang and Bin Yang. Learning to rank paths in spatial networks. In *ICDE*, pages 2006–2009, 2020.

[Yang *et al.*, 2021] Sean Bin Yang, Chenjuan Guo, Jilin Hu, Jian Tang, and Bin Yang. Unsupervised path representation learning with curriculum negative sampling. In *IJCAI*, pages 3286–3292, 2021.

[Yang *et al.*, 2022a] Sean Bin Yang, Chenjuan Guo, Jilin Hu, Bin Yang, Jian Tang, and Christian S. Jensen. Weakly-supervised temporal path representation learning with contrastive curriculum learning. In *ICDE*, pages 2873–2885, 2022.

[Yang *et al.*, 2022b] Sean Bin Yang, Chenjuan Guo, and Bin Yang. Context-aware path ranking in road networks. *TKDE*, 34(7):3153–3168, 2022.

[Yang *et al.*, 2023] Sean Bin Yang, Jilin Hu, Chenjuan Guo, Bin Yang, and Christian S. Jensen. Lightpath: Lightweight and scalable path representation learning. In *KDD*, pages 2999–3010, 2023.

[Yang *et al.*, 2024] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. *ACM Comput. Surv.*, 56(4):105:1–105:39, 2024.

[Yang *et al.*, 2025] Sean Bin Yang, Ying Sun, Yunyao Cheng, Yan Lin, Torp Kristian, and Jilin Hu. Spatio-temporal trajectory foundation model-recent advances and future directions. *arXiv preprint arXiv:2511.20729*, 2025.

[Yang *et al.*, 2026] Sean Bin Yang, Ying Sun, Jilin Hu, Zongyi Xu, Kristian Torp, Hua Lu, Bin Yang, and Christian S. Jensen. Refine: Trajectory representation learning via closed-looptranscription. In *KDD*, 2026.

[Yu *et al.*, 2024] Chengqing Yu, Fei Wang, Zezhi Shao, Tangwen Qian, Zhao Zhang, Wei Wei, and Yongjun Xu. Ginar: An end-to-end multivariate time series forecasting model suitable for variable missing. In *KDD*, pages 3989–4000, 2024.

[Yuan *et al.*, 2022] Haitao Yuan, Guoliang Li, and Zhifeng Bao. Route travel time estimation on A road network revisited: Heterogeneity, proximity, periodicity and dynamicity. *PVLDB*, 16(3):393–405, 2022.

[Zhao *et al.*, 2025] Jie Zhao, Chao Chen, Yuanshao Zhu, Mingyu Deng, and Yuxuan Liang. Unitr: A unified framework for joint representation learning of trajectories and road networks. In *AAAI*, pages 13348–13356, 2025.

[Zhou *et al.*, 2024] Silin Zhou, Shuo Shang, Lisi Chen, Christian S. Jensen, and Panos Kalnis. RED: effective trajectory representation learning with comprehensive information. *PVLDB*, 18(2):80–92, 2024.

[Zhou *et al.*, 2025] Silin Zhou, Shuo Shang, Lisi Chen, Peng Han, and Christian S. Jensen. Grid and road expressions are complementary for trajectory representation learning. In *KDD*, 2025.