DASS: A Dual-Branch Attention-based Framework for Trajectory Similarity Learning with Spatial and Semantic Fusion

Jiayi Li, **Junhua Fang***, **Pingfu Chao**, **Jiajie Xu**, **Pengpeng Zhao**
Department of Computer Science and Technology, Soochow University
jylisuzyli14@stu.suda.edu.cn,{jhfang,pfchao,xujj,ppzhao}@suda.edu.cn

Abstract

Trajectory similarity aims to identify pairs of similar trajectories, serving as a crucial operation in spatial-temporal data mining. Although several approaches have been proposed, they encounter the following two issues: 1) An overemphasis on spatial similarity in road networks while the rich semantic information embedded in trajectories is not fully exploited; 2) Dependence on Recurrent Neural Network (RNN) architectures would struggle to capture long-term dependencies. To address these limitations, we propose a Dual-branch Attention-based framework with Spatial and Semantic information (DASS) based on self-supervised learning. Specifically, DASS comprises two core components: 1) A trajectory representation module that models spatial-temporal adjacent relationships in the form of graph and converts semantics into numerical embeddings. 2) A backbone encoder with a co-attention module to independently process two features before they are integrated. Extensive experiments on real-world datasets demonstrate that DASS outperforms state-of-the-art methods, establishing itself as a novel paradigm.

1 Introduction

Trajectory similarity search plays an important role in spatial-temporal data analysis, which quantifies the correlation between trajectories and serves as a fundamental operation for identifying both individual movement patterns and collective operational trends [Shang *et al.*, 2017; Yu *et al.*, 2019]. It holds immense value across various domains, including personalized recommendation [Bao *et al.*, 2017; He *et al.*, 2020], navigation [Yao *et al.*, 2022b; Zhang *et al.*, 2020], and intelligent transportation [Feng *et al.*, 2023; Ruan *et al.*, 2022; Yi *et al.*, 2023].

The landscape of trajectory similarity methods is rich with proposals, encompassing both heuristic-based [Koide *et al.*, 2020; Yi *et al.*, 1998; Vlachos *et al.*, 2002; Yuan and Li, 2019; Wang *et al.*, 2018], which suffer from high computational cost of $O(n^2)$; and learning-based approaches [Yao *et al.*, 2019; Zhang *et al.*, 2020; Zhao *et al.*, 2024; Tao *et al.*, 2021], which extract generalizable and concise representations from sequential trajectory [Liu *et al.*, 2024]. Our study falls within the latter category, where existing methods have increasingly exploited road networks or scene-related information offered by various location-based applications. Specifically, one category [Fang *et al.*, 2021; Zhang *et al.*, 2023; Mao *et al.*, 2022] incorporates static road network structure into representation to primarily capture spatial proximity and path similarity. Another set of methods [Jiang *et al.*, 2023; Yuan *et al.*, 2022; Liu *et al.*, 2020] gradually leverages semantic aspects, such as travel or time regularity, for a more nuanced representation. While both perspectives provide valuable insights, current proposals fail to adequately capture the interaction and complementarity between two views. Therefore, we argue that an effective similarity measure should integrate both spatial and semantic similarities to achieve a more comprehensive, multidimensional representation.

The motivation for our work can be described from two perspectives: the practical application and the technical viewpoint. ***(i) For the application:*** Focusing solely on spatial proximity or semantic closeness usually fails to yield accurate results. Taking Figure 1 as an example, where illustrates three trajectories $\mathcal{T}_1$, $\mathcal{T}_2$ and $\mathcal{T}_3$, each comprising four points, with the spatial coordinates and semantic label annotated next to each point. When only considering spatial proximity, $\mathcal{T}_1$ and $\mathcal{T}_2$ are the most similar, calculated by three distance metrics: ED [Clarke, 1976], $L1_{dis}$ and MD [Cassels, 2012]. However, when semantic information is taken into account, and computed via a hybrid type attribute dissimilarity formula [Kaufman and Rousseeuw, 2009], $\mathcal{T}_1$ and $\mathcal{T}_3$ are determined to be the most similar. From the above observation, we can conclude that including or excluding semantic information would lead to different outcomes in similarity search tasks. ***(ii) For the implementation technology:*** The deep neural network framework offers a promising solution to combine various information in trajectory representation to enhance the generalizability. Broadly speaking, converting the connectivity and shape of temporal trajectory into vectors ensures the correctness of representation, while embedding semantic information through a dedicated encoding network enhances the model's ability to capture activity-related similarities.

To integrate both spatial and semantic information into the learning model for trajectory similarity calculation, three

* Corresponding author.

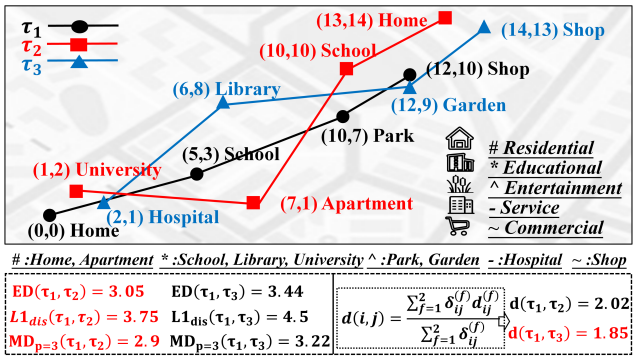


Figure 1: An example illustrating that using more measurement features will lead to varying trajectory similarity results. Three spatial distance functions are listed in the bottom left, while the spatial-semantic oriented similarity is shown in the bottom right. In the equation $d(i,j)$, $\delta_{ij}^{(f)}$ equals to 0 if two points have the same nominal semantic f and 1 otherwise, and $d_{ij}^{(f)}$ is the dissimilarity for f. Higher similarity between $\mathcal{T}_1$ and $\mathcal{T}_2, \mathcal{T}_3$ is marked in red.

non-trivial challenges have to be tackled:

C1: Challenge of Data Representation. The key of trajectory modeling lies in constructing a unified representation that captures both the complex spatial configurations of road networks and the temporal dynamics of trajectories. Existing methods often model spatial information by learning shared representations of equally-sized grids [Li *et al.*, 2018; Yao *et al.*, 2019], resulting in weak connections for records located far apart in sequence modeling. In light of this, we design a Graph-based Spatial-Temporal Modeling module (GSTM), to obtain a multidimensional information fusion representation. This module is flexible and generic, in that it can be integrated with any existing trajectory representation proposal for spatio-temporal similarity learning.

C2: Challenge of Model Framework. To achieve viable unified similarity learning, a fundamental challenge lies in designing a model framework that facilitates the seamless and effective integration of various components. A straightforward idea is to adopt a simple weighted combination. However, such a design struggles with accurate weights set and may not fully capture the nuanced relationships between spatial and semantic features. To overcome this limitation, we develop a Dual feature Co-Attention Fusion module, termed DualCAF, which enables the effective integration of diverse information within trajectories over road networks.

C3: Challenge of Effectiveness Improvement. To achieve both high accuracy and efficiency in similarity queries, the sophistication processes and the optimization parameters must be carefully managed. Current approaches, particularly those leveraging recurrent neural networks, often have difficulty with scalability and capturing long-term dependencies, limiting their utility in large-scale, real-world datasets. In response, we develop a dual-branch Transformer-based network coupled with an SST-Aware loss function to improve effectiveness. It enables our model to balance computational efficiency with the accurate representation of the inherent characteristics across diverse trajectory patterns.

In conclusion, we propose a novel framework DASS to counter the above challenges in trajectory similarity learning. Our major contributions are summarized as follows:

- We propose an attention-based framework, which not only captures the structural information reflected by spatial-temporal graph, but considers semantic activity information to enrich trajectory representation.
- We present a dual-branch trajectory backbone encoder, which incorporates a DualCAF module to simultaneously process spatial and semantic information via distinct attention mechanisms tailored to each feature type.
- We design a weight-adaptive fusion loss function that allows our model to dynamically adjust the importance of each component during training, promoting balanced optimization and enhanced performance.
- We conduct extensive experiments on real-world datasets, demonstrating that DASS outperforms state-of-the-art methods, establishing itself as a novel paradigm for trajectory similarity learning on road networks.

2 Related Work

Trajectory Similarity Learning. Learning-based methods have gained considerable attention for their ability to automatically derive low-dimensional representations from trajectories data [Li *et al.*, 2018; Yao *et al.*, 2017; Ma *et al.*, 2024; Messaoud *et al.*, 2021; Jiang *et al.*, 2023], which enables fast trajectory similarity computation in a linear time. NEUTRAJ [Yao *et al.*, 2019] is the first learning-based trajectory similarity measure, which samples trajectories as seeds and uses pair-wise comparisons as guidance. To enhance training efficiency, Traj2SimVec [Zhang *et al.*, 2020] simplifies the process by converting training trajectories into triplet samples. ST2vec [Fang *et al.*, 2022] focuses on both spatial characteristics and temporal regularities within trajectories, addressing the inherent complexity of spatiotemporal data. [Chen *et al.*, 2020] introduces a novel metric to evaluate spatial and textual domain trajectory semantic similarity comprehensively. To further improve deep representation for long trajectories, TrajGAT [Yao *et al.*, 2022a] introduces a quad-tree index to model long-term dependencies.

Graph Neural Networks. Recent studies have explored integrating GNNs [Kipf and Welling, 2016] for trajectory similarity on road networks, demonstrating their potential to capture the spatial information of trajectories [Chen *et al.*, 2019]. Han et al. [Han *et al.*, 2021] first apply GNNs into this domain and utilize unique POI information. Their subsequent work [Zhou *et al.*, 2023] further extends this approach, leveraging LSTM and GAT to model sequential dependencies and spatial interactions within trajectories. Most recently, Start [Jiang *et al.*, 2023] presents a spatial networks-based representation method on the basis of GAT, and achieves innovative results by considering travel semantics hidden in trajectories.

3 Preliminaries

3.1 Basic Concepts

DEFINITION 1 ***(Trajectory).*** *A trajectory, denoted by $\mathcal{T}$, is a time-ordered sequence of GPS points of a moving object, i.e., $\mathcal{T} = \langle p_1, p_2, ..., p_n \rangle$, where each point $p_i = (x_i, y_i, t_i)$ is a longitude, latitude, and timestamp triplet.*

DEFINITION 2 ***(Road Network).*** *A road network is represented as an undirected graph $G = (V, E)$, where each node $v \in V$ is a Point of Interest (POI); and each edge $e = \langle u, v \rangle \in E$ is a road segment connecting points u and v.*

Given a trajectory $\mathcal{T}$, we align each trajectory point p_i with vertex v_i in road network through a commonly used map-matching procedure [Brakatsoulas *et al.*, 2005], to obtain the corresponding vertex trajectory $\mathcal{T}^{road} = \langle v_1, v_2, ..., v_n \rangle$.

DEFINITION 3 ***(Top-K Similarity Query).*** *Given a query trajectory $\mathcal{QT}$, a set of trajectories $\mathbf{T} = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_m\}$, and a similarity measure $f(\cdot)$, the Top-K similarity query returns K trajectories in $\mathbf{T}$ that are most similar to $\mathcal{QT}$ under $f(\cdot)$.*

3.2 Problem Definition

Given a set of trajectories $\mathbf{T}$ and corresponding road network graph G, DASS aims to learn a trajectory encoder $\mathbb{E} : \mathcal{T} \rightarrow \mathcal{E}$, mapping each trajectory $\mathcal{T}$ to a d-dimensional embedding vector $\mathcal{E}$ that jointly captures the trajectory characteristics and road network information. For $\forall \mathcal{T}_i \in \mathbf{T}$, the objective is to find $\mathcal{T}_j \in \mathbf{T}$ that is most similar to $\mathcal{T}_i$ under the similarity function $f(\cdot)$. Such problem can be formalized as below:

$$\underset{\mathcal{T}_j \in \mathbf{T},\, i \neq j}{\arg\min}\ f\left(\mathcal{E}(\mathcal{T}_i), \mathcal{E}(\mathcal{T}_j)\right). \tag{1}$$

$\mathbb{E}$ aims to accurately reflect the similarity between trajectories through their embeddings, i.e., $\mathcal{E}(\mathcal{T}_i)$ and $\mathcal{E}(\mathcal{T}_j)$ are close (distant) if $\mathcal{T}_i$ and $\mathcal{T}_j$ are similar (dissimilar), demonstrating the effectiveness of $\mathbb{E}$ in capturing inherent characteristics.

4 Methodology

Figure 2 shows the framework of DASS, which consists of three following major modules:

- **Graph-based Spatial-temporal Modeling**, which models spatial and temporal adjacency relationships between points with knowledge graph embedding, and then constructs a similarity-based graph to capture the underlying structural patterns (Section 4.1).
- **Semantic Information Embedding**, which comprises three specific components to capture contextual information within the trajectory and enrich representation with deeper semantic context (Section 4.2).
- **Dual-Branch Fusion Encoding**, which adaptively processes the features from both perspectives, ensuring that the unique characteristics are effectively captured before they are integrated (Section 4.3).

The above three modules of DASS work collaboratively to obtain a unified representation, and the whole framework is trained by the self-supervised paradigm.

4.1 Graph-based Spatial-temporal Modeling

Road networks inherently possess rich topological structures, while trajectories are time-ordered sequences. However, due to varying sampling rates and data sparsity, adjacent points in trajectory may not always correspond to adjacent points in the road network. Based on this, we classify adjacency into two types: *1) temporal adjacency*, where two points appear sequentially over time within the trajectory; *2) spatial adjacency,* where adjacent points are directly connected by edges on road networks, indicating spatial continuity. From an intuitive view, a trajectory knowledge graph can be constructed to model these adjacent relationships, expressed as (h, r, t), where h and t refer to POIs within the road network, r is the relationship between two vertices and are defined as follows: ① **temporal adjacency** R_t**,** ② **spatial adjacency** R_s**, and** ③ **spatio-temporal adjacency** R_{st}.

To accurately model various relationships, DASS first utilizes TransD [Ji *et al.*, 2015] to obtain the knowledge graph representation. Then it introduces a point-relation similarity function to consolidate the multidimensional similarities between points into a unified similarity computation framework. Drawing inspiration from KGE techniques [Wang *et al.*, 2014], the distance between two points u and v under a specific relationship r is defined as:

$$d_r\left(e_u, e_v\right) = \left\| e_u + e_r - e_v \right\|, \tag{2}$$

where e_u, e_v, e_r are the embedding through TransD. Following Equation (2), we design a bilateral neighbor similarity function to calculate the similarity between two nodes, which not only accounts for the direct similarity between points, but incorporates contextual information from neighbors to capture broader spatial distribution within the local structure. The similarity function is defined as follows:

$$s(e_u, e_v) = \frac{1}{|\mathcal{N}(u)|\,|\mathcal{N}(v)|} \sum_{i \in \mathcal{N}(u)} \sum_{j \in \mathcal{N}(v)} e^{-d_r(e_i, e_j)}, \tag{3}$$

where $|\mathcal{N}(u)|$ and $|\mathcal{N}(v)|$ denote the neighbor sets of u and v, respectively. Based on the similarity calculation in Equation (3), we construct a graph where each point connects its Top-K most similar neighbors. High-similarity relationships between points are reflected by neighbor connection, ensuring that adjacent nodes maintain closely related representations later. The graph construction is formulated as below:

$$G_{st}(i, j) = \begin{cases} 1 & \text{if } v_j \in N_s(v_i) \\ 0 & \text{otherwise} \end{cases}. \tag{4}$$

The resulting graph G_{st} will be used as the spatial branch input for the subsequent network (detailed in Section 4.3).

4.2 Semantic Keywords Embedding

Each point in the trajectory $\mathcal{T}$ has an associated keyword to represent the context of an activity. To efficiently convert such nominal attributes into numerical embeddings, similar to BERT's implementation process, the activity keyword embeddings are composed of three components: ***1) Positional encoding.*** Drawing inspiration from the position embedding in Transformer [Vaswani, 2017], each keyword k_i in

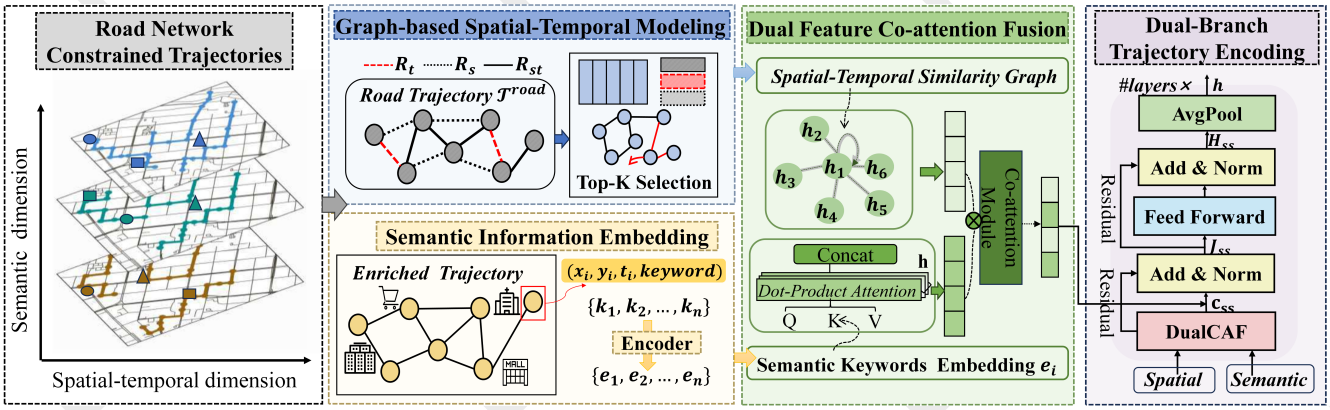


Figure 2: The framework of DASS.

the sequence of trajectory keywords $K = \{k_1, k_2, ..., k_n\}$ is assigned a fixed sinusoidal positional encoding embedding, $E_P(i)$, calculated as follows:

$$E_P(i) = \begin{cases} E_P(i, 2k) = \sin\left(\frac{i}{10000^{2k/d}}\right) \\ E_P(i, 2k+1) = \cos\left(\frac{i}{10000^{2k/d}}\right) \end{cases}. \quad (5)$$

Here, d is the embedding dimension, and $sin(\cdot)$, $cos(\cdot)$ serve as periodic activation functions that capture sequential behaviors. This component preserves sequential information, enhancing the model's sensitivity to sequence order. ***2) Token encoding.*** For each activity keyword k_i, we use the pre-trained GloVe model to generate a token embedding $E_T(k_i)$, capturing the intrinsic semantic properties of the word. ***3) Segment encoding.*** Each activity category (e.g., entertainment) is associated with a unique segment embedding $E_S(k_i)$ through GloVe. By mapping each keyword to its corresponding category, this component reinforces the model's ability to distinguish and categorize similar activities within the trajectory. The final representation of k_i is expressed as:

$$E(k_i) = E_T(k_i) + E_P(i) + E_S(k_i). \quad (6)$$

Next, we feed $E(k_i)$ in Equation (6) into BERT's transformer architecture step by step to obtain a smoothed representation $e_i \in R^d$ for each keyword. These embeddings $E = \{e_1, e_2, ..., e_n\}$ have integrated both semantic content and sequential order, offering a comprehensive foundation for semantic similarity analysis.

4.3 Dual-Branch Fusion Encoding

Spatial and semantic information of trajectories can be treated as two observations of the same concept, then a co-attention fusion module (DualCAF) is designed for our DASS model to generate spatio-semantic oriented representation.

DualCAF. It takes both graph G_{st} and semantic embeddings E as input and outputs the joint representations of trajectory points. To capture the topological structure within the graph, the multi-head GAT is employed to assign attention coefficients to neighboring nodes. Formally, the hidden state of node v_i is updated by aggregating information from its neighbors, weighted by learned attention coefficients from multiple attention heads, represented as follows:

$$h_i = ||_{k=1}^{K} \sigma(\sum_{v_j \in N(v_i)} \alpha_{ij}^{(k)} W^{(k)} h_j), \quad (7)$$

where $\alpha_{ij}^{(k)}$ represents the attention coefficient from the k-th attention head between node v_i and its neighboring node v_j, $W^{(k)}$ is a learnable weight matrix, and h_j is the hidden state of node v_j. The attention coefficient can be computed as:

$$e_{ij} = a^T \left[W^{(k)} h_i \| W^{(k)} h_j\right], j \in \mathcal{N}(i) \quad (8)$$

$$\alpha_{ij}^{(k)} = \frac{\exp\left(\text{LeakyReLU}\left(e_{ij}\right)\right)}{\sum_{m \in \mathcal{N}(i)} \exp\left(\text{LeakyReLU}\left(e_{ik}\right)\right)} \quad (9)$$

where $||$ denotes concatenation. So far, spatial representation S_g have been learned by GAT layers, which integrates topological relations derived from the input graph G_{st}.

For semantic branch, the semantic feature matrix E is first projected into value V_i, key K_i, and query Q_i matrices, where i denotes the i-th head. The introduction of muti-head attention enables to focus on different parts of the semantic feature space when learning co-attention. Formally, the attention coefficient matrix of the i-th head A_i is computed as follows:

$$A_i = Softmax(\frac{Q_i {K_i}^T}{\sqrt{d_k}}). \quad (10)$$

Based on Equation (10), the attention coefficient matrix A_i is multiplied with value matrix V_i to obtain the hidden output h_s^i, and the outputs h_s^i from each self-attention head are concatenated and passed through a linear transformation to generate the final output h_s as follows:

$$h_s^i = A_i \times V_i, \quad (11)$$

$$h_s = Concat(h_s^1, h_s^2, ..., h_s^H) W_o, \quad (12)$$

where W_o is the learnable weight matrix.

With both spatial and semantic features embedded, they are fused using a concatenation strategy, denoted as F_{ss}. F_{ss} is then passed through another self-attention mechanism to capture complex feature interactions. Subsequently, a layer normalization, an MLP module with residual connection, and another layer normalization are applied to ensure optimization stability and model effectiveness, as described in Equations (13) and (14). Finally, DualCAF outputs the dual feature fusion representation C_{ss}.

DASS. As illustrated in Figure 2, DASS follows the overall structure of a multi-layer Transformer encoder and possesses two branches with complementary responsibilities. It takes G_{st} and E as the input, which are processed through the DualCAF module to jointly model spatial and semantic features, learning their complex interactions. The output of DualCAF C_{ss}, is further refined through a series of residual connections, dropout layers, and layer normalization to stabilize gradient flow and mitigate issues like gradient vanishing or explosion. The updated representation J_{ss} is then passed through an MLP, followed by another residual connection and layer normalization. The process can be summarized as:

$$J_{ss} = LayerNorm(S_g + Dropout(C_{ss})), \tag{13}$$

$$H_{ss} = LayerNorm(J_{ss} + Dropout(Feedforward(J_{ss}))), \tag{14}$$

where S_g is the original representation for the similarity-based graph, and H_{ss} refers to the updated representation matrix, incorporating both the topological and semantic features. The model is implemented with stacked layers (two layers in the experiments) of the DualCAF and MLP blocks. After the final layer, average pooling is applied to H_{ss} to obtain the final trajectory representation $h \in R^d$ as follows:

$$h = AvgPool(H_{ss}). \tag{15}$$

Based on Equation (15), the final representation incorporates both structural and semantic information, providing a comprehensive foundation for downstream similarity analysis.

4.4 SST-Aware Loss Function

To enhance the accuracy and robustness of trajectory representation, we introduce a novel SST (Spatial-Semantic-Trajectory)-Aware loss function to optimize model training.

Graph-Based Spatial-temporal Contrastive Loss. It encourages each node in the similarity-based graph G_{st} to be close to high-similarity nodes (positive samples) while remaining distant from unrelated nodes (negative samples). The objective is to minimize the distance between node v and its positive samples v_{pos}, while maximizing the distance between v and its negative samples v_{neg}, thus enhancing the model's understanding of underlying graph structure. The loss function is defined as follows:

$$L_{st} = -\log \frac{\exp\left(\mathrm{s}\left(v, v_{pos}\right)/\tau\right)}{\exp\left(\mathrm{s}\left(v, v_{pos}\right)/\tau\right) + \sum_{v_{neg}} \exp\left(\mathrm{s}\left(v, v_{neg}\right)/\tau\right)}, \tag{16}$$

where $\mathrm{s}(\cdot)$ represents the vector dot-product function.

Semantic Pair-wise Loss. The semantics of each point is extracted in the form of a *(category, subcategory)* structure. For each $v_i \in \mathcal{T}_i$, two points with the same category are considered positive samples; others are negative. Then, the semantic pair-wise loss encourages semantic consistency and is formulated as follows:

$$L_{se} = -\sum_{i=1}^{m}\sum_{j=1}^{|\tau_i|} \log\sigma\left(s\left(v_j^i, v_{\mathrm{pos}}^i\right) - s\left(v_j^i, v_{\mathrm{neg}}^i\right)\right), \tag{17}$$

where σ is sigmoid function, and v_j^i is a point in $\mathcal{T}_i$.

SST-Aware Loss. For each trajectory $\mathcal{T}_i$, the most similar trajectory (ground truth) is considered as $\mathcal{T}_{pos}$, and $\mathcal{T}_{neg}$ is a negative sample selected from $\mathbf{T}\backslash\{\mathcal{T}_i, \mathcal{T}_{pos}\}$. The trajectory loss function is formulated as follows, from the perspective of the entire trajectory similarity:

$$L_t = -\sum_{i=1}^{m} \log\sigma\left(s\left(\mathcal{T}_i, \mathcal{T}_{pos}\right) - s\left(\mathcal{T}_i, \mathcal{T}_{neg}\right)\right). \tag{18}$$

Finally, SST-Aware loss function is composed of Equations (16) to (18), expressed as:

$$L = W_\lambda L_{st} + W_\mu L_{se} + W_\eta L_t, \tag{19}$$

where each weight W_λ, W_μ and W_η are learned adaptively through a softmax function within an attention mechanism.

5 Experiments

In this section, we conduct multiple experiments to verify the effectiveness of DASS and answer the following questions:

- **RQ1:** How does DASS perform compared with existing trajectory similarity computation methods?
- **RQ2:** How does every module that we design contribute to the model performance?
- **RQ3:** How do variations in parameter settings impact the performance of DASS?

5.1 Experimental Setup

Data Description. We use two road networks from OpenStreetMap: Beijing Road Network (BJRN), containing $28,342$ nodes and $27,690$ edges; and New York Road Network (NYRN), with $95,581$ nodes and $260,855$ edges. For BJRN, the public real-life trajectory dataset T-drive is adopted, which includes trajectories from $10,357$ taxis in Beijing over a week. For NYRN, we randomly sample a subset of taxi trip data from New York to generate the dataset. After preprocessing, we collect 5231,420 trajectories for T-drive and 9541,270 for New York. Both datasets are divided into training, evaluation, and testing sets, with $20\%, 10\%$, and 70% allocated to each set, respectively.

Baselines. To evaluate the effectiveness of the proposed DASS, we implement in total eight baselines. **i) Traj2vec** [Yao *et al.*, 2018] represents trajectory features via the moving behavior and reconstruction loss. **ii) Siamese** [Pei *et al.*, 2016] employs a Siamese Network with a cross-entropy loss function. **iii) NeuTraj** [Yao *et al.*, 2019] combines metric learning with spatial attention memory and a distance-weighted ranking loss. **iv) At2vec** [Liu *et al.*, 2020] addresses uneven sampling issues with LSTM. **v) Traj2SimVec** [Zhang *et al.*, 2020] designs a custom strategy to extract distance information from sub-trajectories. **vi) GTS** [Han *et al.*, 2021] is the first graph-learning based approach for trajectory similarity on road networks. **vii) ST2Vec** [Fang *et al.*, 2021] first considers spatio-temporal similarity in trajectory representation. **viii) GRLSTM** [Zhou *et al.*, 2023] designs a residual-LSTM to model the trajectory data.

https://opendata.cityofnewyork.us

Method	Beijing					New York				
	HR@1	HR@5	HR@10	HR@20	HR@50	HR@1	HR@5	HR@10	HR@20	HR@50
Traj2vec	0.0582	0.1057	0.1864	0.2883	0.4007	0.0495	0.0933	0.1613	0.2457	0.3724
Siamese	0.0633	0.1325	0.2017	0.3261	0.4558	0.0523	0.1111	0.1875	0.2774	0.4216
At2vec	0.0728	0.1487	0.2348	0.3759	0.4973	0.0593	0.1329	0.2173	0.2682	0.4337
NeuTraj	0.0772	0.1978	0.2754	0.3963	0.5357	0.0615	0.1557	0.2328	0.3018	0.4843
Traj2SimVec	0.0781	0.2042	0.2917	0.4014	0.5675	0.0631	0.1703	0.2646	0.3252	0.5055
GTS	0.0921	0.2500	0.3548	0.4807	0.6612	0.0843	0.2164	0.3253	0.4169	0.5817
ST2Vec	0.1036	0.3372	0.4235	0.5492	0.7523	0.0983	0.2514	0.4128	0.4637	0.6328
GRLSTM	0.1296	0.3271	0.4438	0.5734	0.7402	0.1133	0.2785	0.3936	0.4812	0.6247
DASS	**0.1953**	**0.4701**	**0.5473**	**0.6185**	**0.7687**	**0.1617**	**0.3770**	**0.4728**	**0.5284**	**0.6429**
Improvement	**50.69%**	**39.41%**	**23.32%**	**7.81%**	**3.85%**	**42.70%**	**35.37%**	**14.53%**	**9.81%**	**1.59%**

Table 1: Top-K similarity performance comparison on Beijing and New York datasets.

Evaluation Metrics. Following existing works, we adopt the Top-K hitting ratio (HR@K) as the major performance metric, which captures the degree of overlap between a Top-K result (Definition 3) and the corresponding ground-truth result. The ground-truth are the exact Top-K similarity search results obtained through NetERP [Koide *et al.*, 2020].

Parameter Settings. All experiments are conducted with PyTorch 1.8.1 on a LINUX server (Intel Xeon 5118 6-Core CPU, 32 GB of RAM, and a GeForce GTX 3090 Ti GPU). We train DASS using the Adam optimizer, the learning rates and Top-K selection in Equation (4) are set as 5e-4, 15 in the Beijing dataset and 1e-3, 30 in the NewYork dataset respectively. The embedding dimensionality d is set to 128 for all learned methods. For DualCAF, the number head of K in Equation (7) and s in Equation (12) are set as 8 and 4. For DASS, The number of encoder #layers is 2.

5.2 Overall Performance (RQ1)

Effectiveness Evaluation. We conduct Top-K similarity queries and compare DASS with all 8 baselines. Table 1 lists the results on two datasets. From these results, we provide observations and analyses as follows.

The first observation is that DASS achieves the best performance in terms of HR@K on both two datasets, significantly outperforming all the baseline methods. It confirms the superior performance of our framework in learning trajectory representations by introducing spatial-temporal features and semantics in the pre-training phase. Specifically, DASS achieves a consistent hitting ratio exceeding 35% (except HR@1), with an average improvement of 25.88% over the best baseline GRLSTM on the Beijing dataset and 22.19% on the New York dataset. Moreover, we can observe that methods like Traj2vec and NeuTraj perform relatively poorly, likely because they lack explicit modeling of road network constraints and primarily rely on trajectory-level features without considering the underlying topological information.

The third observation is that although ST2Vec and GRLSTM account for road network topology, DASS offers two key advantages contributing to its superior performance. First, our approach incorporates activity semantic – a feature absent in the other two methods, which captures critical information from multiple dimensions to enrich representation. Second, its Transformer-based network, unlike LSTM-based methods, can better capture long-term dependencies. It is also worth noting that DASS's performance advantage is less pronounced on the New York dataset compared to Beijing. One possible reason is the significantly larger scale and complexity of the New York road network, which may introduce more noise and less consistent trajectory patterns.

Model	*Training time*	*Inference time*
GTS	89.35s	3.14 ms
ST2Vec	127.93s	2.82 ms
GRLSTM	148.25s	7.73 ms
DASS	117.46s	5.27 ms

Table 2: Training and inference efficiency on BJ dataset.

Efficiency Evaluation. We evaluate the efficiency of each measure in both model training time (per epoch) and inference time (the process of converting trajectories into representation). The experimental results are shown in Table 2. For simplicity, we only present the four models with relatively better efficiency. From Table 2, it is evident that GTS distinguishes itself with the shortest training time, attributed to its relatively simple structure and less dependence on trajectory features. In contrast, the other three models consider additional trajectory features, such as temporal and semantic similarity, which need to be computed during the training process, resulting in longer training times. Our model, DASS, although it includes complex components such as graph modeling and text representation, still maintains high efficiency through optimized design. Its training time is only slightly longer than GTS but achieves higher accuracy in trajectory similarity queries. It indicates that DASS successfully balances computational complexity and model effectiveness.

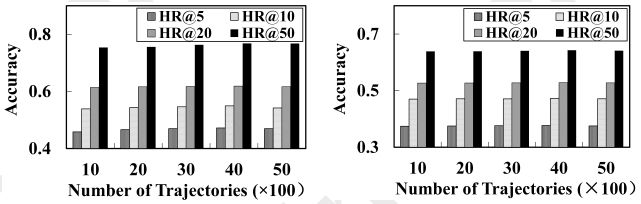
5.3 Ablation Study (RQ2)

In the ablation study, we quantify the contribution of each DASS component. The results on Beijing dataset are shown in Table 3, confirming the necessity of each component in the DASS framework. A similar trend is observed on the New York dataset; therefore, we omit it for brevity.

Experimental results of seven variants illustrate that the most significant decrease occurs when the DualCAF component (w/o DualCAF) is excluded due to its adaptive fusion of both input features. The removal of both graph modeling (w/o GM) and the semantic branch (w/o Se branch)

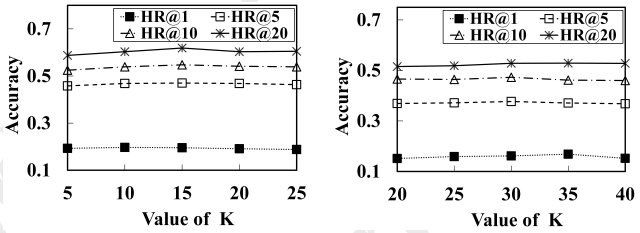
	HR@5	HR@10	HR@20	HR@50
w/o GSM	0.4292	0.5073	0.5798	0.7413
w/o Se Branch	0.4318	0.5198	0.5811	0.7462
w/o DualCAF	0.4159	0.5098	0.5774	0.7397
w/o G-Loss	0.4652	0.5289	0.6018	0.7517
w/o S-Loss	0.4687	0.5328	0.6093	0.7559
w/o Transformer	0.4528	0.5217	0.5917	0.7435
w/o ResNet	0.4634	0.5289	0.6031	0.7521
DASS	**0.4701**	**0.5473**	**0.6185**	**0.7683**

Table 3: Ablation study results on Beijing dataset.



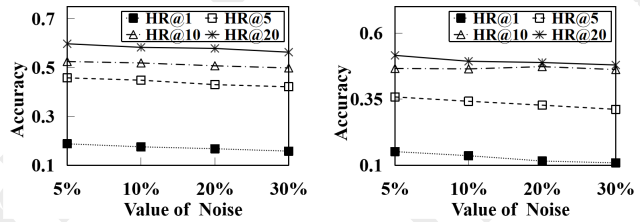
(a) Beijing Dataset (b) NewYork Dataset

Figure 3: Performance of DASS varying trajectory length.



(a) Beijing Dataset (b) NewYork Dataset

Figure 4: Performance of DASS varying parameter K.



(a) Beijing Dataset (b) NewYork Dataset

Figure 5: Performance of DASS under semantic noise.

also brings a performance decline of approximately 3.79%, highlighting the importance of richer characteristics for better model representation. Similarly, excluding the ResNet layer (w/o ResNet) leads to performance degradation, as ResNet addresses the gradient vanishing problem without increasing training time. The inclusion of the Transformer module (w/o Transformer) offers additional advantages that better capturing contextual and sequential information for representation learning and achieving improvement from 0.4528 to 0.4701 on HR@5. Moreover, the efficacy of two novel loss functions is also verified, resulting in a 1.93% improvement.

5.4 Parameter Sensitivity Study (RQ3)

Sensitive to datasize. We investigate the effect of trajectory length on the performance of DASS. As shown in Figure 3, the similarity learning performance of DASS is examined as the trajectory length varies from $1,000$ to $5,000$. It can be observed that the performance of DASS slightly decreases with the increasing trajectory length. However, it generally exhibits stable performance on both the Beijing and New York datasets. This suggests that DASS is capable of effectively capturing the essential spatial and semantic information embedded in longer trajectories, demonstrating its robustness across varying trajectory lengths.

Varying parameter K. We analyze the impact of the graph construction parameter K in Equation (4), which determines the number of neighbors considered for each trajectory point in the GSTM module. Figure 4 shows the results, where we vary K across different values. We observe that the choice of K has a noticeable impact on the model's performance. Specifically, the best results are achieved when K is set to 15 for Beijing dataset and 30 for New York dataset. For example, in term of HR@10 metric, the results are 0.5473 and 0.4728 on Beijing, NewYork dataset, respectively. This also suggests that a smaller K in Beijing works better due to the denser road network and more localized movement patterns, while New York's more complex urban landscape benefits from a larger neighborhood size.

Semantic Noise. DASS incorporates semantics from trajectories, making semantics a crucial factor in model performance. We test various levels of semantic noise, and the results are shown in Figure 5. It is evident that the model's performance gradually declines with increasing noise levels in both two datasets. However, the impact of noise on DASS is relatively limited, with performance degradation controlled within 5.3%. As shown in Table 3 under the "w/o Se Branch" condition, even in the extreme case where semantic information is removed, DASS still outperforms other baselines. This is attributed to DASS's unique dual-branch structure, where the spatiotemporal features extracted from the other branch can effectively complement the semantic branch, mitigating the negative impact of noise on overall performance.

6 Conclusion

In this paper, we propose DASS, a self-supervised trajectory similarity learning model with a dual-branch backbone encoder. DASS can effectively capture both the spatial and semantic information embedded in trajectory, and utilizes the DualCAF module to ensure that the unique characteristics of each view are captured before being integrated into a unified representation. We have conducted extensive experiments on real-world trajectory datasets, demonstrating that DASS achieves significant performance improvements on similarity search and outperforms all comparing baselines. The universality of DASS makes it a promising candidate with ample potential for various location-based applications.

Acknowledgments

This work was supported by the Natural Science Foundation of the National Natural Science Foundation of China under grant (61802273), Jiangsu Higher Education Institutions of China (23KJA520011), the National Key Research and Development Program of China(2023YFF0725002), China Science and Technology Plan Project of Suzhou (SYG202139).

References

[Bao *et al.*, 2017] Jie Bao, Tianfu He, Sijie Ruan, Yanhua Li, and Yu Zheng. Planning bike lanes based on sharing-bikes' trajectories. In *SIGKDD*, pages 1377–1386, 2017.

[Brakatsoulas *et al.*, 2005] Sotiris Brakatsoulas, Dieter Pfoser, Randall Salas, and Carola Wenk. On map-matching vehicle tracking data. In *VLDB*, page 853–864, 2005.

[Cassels, 2012] John William Scott Cassels. *An introduction to the geometry of numbers*. Springer Science & Business Media, 2012.

[Chen *et al.*, 2019] Yu Chen, Lingfei Wu, and Mohammed J Zaki. Reinforcement learning based graph-to-sequence model for natural question generation. *arXiv*, 2019.

[Chen *et al.*, 2020] Lisi Chen, Shuo Shang, Christian S Jensen, Bin Yao, and Panos Kalnis. Parallel semantic trajectory similarity join. In *ICDE*, pages 997–1008, 2020.

[Clarke, 1976] Frank H Clarke. Optimal solutions to differential inclusions. *J Optim Theory Appl*, 19(3):469–478, 1976.

[Fang *et al.*, 2021] Ziquan Fang, Yuntao Du, Xinjun Zhu, Lu Chen, Yunjun Gao, and Christian S Jensen. St2vec-spatio-temporal trajectory similarity learning in road networks. In *SIGKDD*, pages 347–356, 2021.

[Fang *et al.*, 2022] Ziquan Fang, Yuntao Du, Xinjun Zhu, Danlei Hu, Lu Chen, Yunjun Gao, and Christian S Jensen. Spatio-temporal trajectory similarity learning in road networks. In *SIGKDD*, pages 347–356, 2022.

[Feng *et al.*, 2023] Tao Feng, Huan Yan, Huandong Wang, Wenzhen Huang, Yuyang Han, Hongsen Liao, Jinghua Hao, and Yong Li. Ilroute: A graph-based imitation learning method to unveil riders' routing strategies in food delivery service. In *SIGKDD*, pages 4024–4034, 2023.

[Han *et al.*, 2021] Peng Han, Jin Wang, Di Yao, Shuo Shang, and Xiangliang Zhang. A graph-based approach for trajectory similarity computation in spatial networks. In *SIGKDD*, pages 556–564, 2021.

[He *et al.*, 2020] Tianfu He, Jie Bao, Ruiyuan Li, Sijie Ruan, Yanhua Li, Li Song, Hui He, and Yu Zheng. What is the human mobility in a new city: Transfer mobility knowledge across cities. In *WWW*, pages 1355–1365, 2020.

[Ji *et al.*, 2015] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. Knowledge graph embedding via dynamic mapping matrix. In *ACL-IJCNLP*, pages 687–696, 2015.

[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.

[Kaufman and Rousseeuw, 2009] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv*, 2016.

[Koide *et al.*, 2020] Satoshi Koide, Chuan Xiao, and Yoshiharu Ishikawa. Fast subtrajectory similarity search in road networks under weighted edit distance constraints. *arXiv*, 2020.

[Li *et al.*, 2018] Xiucheng Li, Kaiqi Zhao, Gao Cong, Christian S Jensen, and Wei Wei. Deep representation learning for trajectory similarity computation. In *ICDE*, pages 617–628, 2018.

[Liu *et al.*, 2020] An Liu, Yifan Zhang, Xiangliang Zhang, Guanfeng Liu, Yanan Zhang, Zhixu Li, Lei Zhao, Qing Li, and Xiaofang Zhou. Representation learning with multi-level attention for activity trajectory similarity computation. *TKDE*, 34(5):2387–2400, 2020.

[Liu *et al.*, 2024] Zihan Liu, Bowen Du, Junchen Ye, Xianqing Wen, and Leilei Sun. An ncde-based framework for universal representation learning of time series. In *IJCAI*, pages 4623–4633, 2024.

[Ma *et al.*, 2024] Zhipeng Ma, Zheyan Tu, Xinhai Chen, Yan Zhang, Deguo Xia, Guyue Zhou, Yilun Chen, Yu Zheng, and Jiangtao Gong. More than routing: Joint gps and route modeling for refine trajectory representation learning. In *WWW*, pages 3064–3075, 2024.

[Mao *et al.*, 2022] Zhenyu Mao, Ziyue Li, Dedong Li, Lei Bai, and Rui Zhao. Jointly contrastive representation learning on road network and trajectory. In *CIKM*, pages 1501–1510, 2022.

[Messaoud *et al.*, 2021] Kaouther Messaoud, Nachiket Deo, Mohan M Trivedi, and Fawzi Nashashibi. Trajectory prediction for autonomous driving based on multi-head attention with joint agent-map representation. In *IV*, pages 165–170, 2021.

[Pei *et al.*, 2016] Wenjie Pei, David MJ Tax, and Laurens van der Maaten. Modeling time series similarity with siamese recurrent networks. *arXiv*, 2016.

[Ruan *et al.*, 2022] Sijie Ruan, Cheng Long, Zhipeng Ma, Jie Bao, Tianfu He, Ruiyuan Li, Yiheng Chen, Shengnan Wu, and Yu Zheng. Service time prediction for delivery tasks via spatial meta-learning. In *SIGKDD*, pages 3829–3837, 2022.

[Shang *et al.*, 2017] Shuo Shang, Lisi Chen, Zhewei Wei, Christian Søndergaard Jensen, Kai Zheng, and Panos Kalnis. Trajectory similarity join in spatial networks. *VLDB*, 10(11), 2017.

[Tao *et al.*, 2021] Yaguang Tao, Alan Both, Rodrigo I Silveira, Kevin Buchin, Stef Sijben, Ross S Purves, Patrick Laube, Dongliang Peng, Kevin Toohey, and Matt Duckham. A comparative analysis of trajectory similarity measures. *GIScience & Remote Sensing*, 58(5):643–669, 2021.

[Vaswani, 2017] A Vaswani. Attention is all you need. *NeurIPS*, 2017.

[Vlachos *et al.*, 2002] Michail Vlachos, George Kollios, and Dimitrios Gunopulos. Discovering similar multidimensional trajectories. In *ICDE*, pages 673–684, 2002.

[Wang *et al.*, 2014] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, volume 28, 2014.

[Wang *et al.*, 2018] Sheng Wang, Zhifeng Bao, J Shane Culpepper, Zizhe Xie, Qizhi Liu, and Xiaolin Qin. Torch: A search engine for trajectory data. In *SIGIR*, pages 535–544, 2018.

[Yao *et al.*, 2017] Di Yao, Chao Zhang, Zhihua Zhu, Jianhui Huang, and Jingping Bi. Trajectory clustering via deep representation learning. In *IJCNN*, pages 3880–3887, 2017.

[Yao *et al.*, 2018] Di Yao, Chao Zhang, Zhihua Zhu, Qin Hu, Zheng Wang, Jianhui Huang, and Jingping Bi. Learning deep representation for trajectory clustering. *Expert Systems*, 35(2):e12252, 2018.

[Yao *et al.*, 2019] Di Yao, Gao Cong, Chao Zhang, and Jingping Bi. Computing trajectory similarity in linear time: A generic seed-guided neural metric learning approach. In *ICDE*, pages 1358–1369, 2019.

[Yao *et al.*, 2022a] Di Yao, Haonan Hu, Lun Du, Gao Cong, Shi Han, and Jingping Bi. Trajgat: A graph-based long-term dependency modeling approach for trajectory similarity computation. In *SIGKDD*, pages 2275–2285, 2022.

[Yao *et al.*, 2022b] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *NeurIPS*, 35:20744–20757, 2022.

[Yi *et al.*, 1998] Byoung-Kee Yi, Hosagrahar V Jagadish, and Christos Faloutsos. Efficient retrieval of similar time sequences under time warping. In *ICDE*, pages 201–208, 1998.

[Yi *et al.*, 2023] Liping Yi, Han Yu, Zhuan Shi, Gang Wang, Xiaoguang Liu, Lizhen Cui, and Xiaoxiao Li. Fedssa: Semantic similarity-based aggregation for efficient model-heterogeneous personalized federated learning. *arXiv*, 2023.

[Yu *et al.*, 2019] Qingying Yu, Yonglong Luo, Chuanming Chen, and Shigang Chen. Trajectory similarity clustering based on multi-feature distance measurement. *Applied Intelligence*, 49:2315–2338, 2019.

[Yuan and Li, 2019] Haitao Yuan and Guoliang Li. Distributed in-memory trajectory similarity search and join on road network. In *ICDE*, pages 1262–1273, 2019.

[Yuan *et al.*, 2022] Yuan Yuan, Jingtao Ding, Huandong Wang, Depeng Jin, and Yong Li. Activity trajectory generation via modeling spatiotemporal dynamics. In *SIGKDD*, pages 4752–4762, 2022.

[Zhang *et al.*, 2020] Hanyuan Zhang, Xingyu Zhang, Qize Jiang, Baihua Zheng, Zhenbang Sun, Weiwei Sun, and Changhu Wang. Trajectory similarity learning with auxiliary supervision and optimal matching.(2020). In *IJCAI*, pages 11–17, 2020.

[Zhang *et al.*, 2023] Qianru Zhang, Zheng Wang, Cheng Long, Chao Huang, Siu-Ming Yiu, Yiding Liu, Gao Cong, and Jieming Shi. Online anomalous subtrajectory detection on road networks with deep reinforcement learning. In *ICDE*, pages 246–258, 2023.

[Zhao *et al.*, 2024] Zeyun Zhao, Changjian Wang, Zhen Huang, Yongheng Li, and Gaojin He. G2vec: Spatial-temporal trajectory generation network based on multi-resolution feature correlation. In *IJCNN*, pages 1–6, 2024.

[Zhou *et al.*, 2023] Silin Zhou, Jing Li, Hao Wang, Shuo Shang, and Peng Han. Grlstm: trajectory similarity computation with graph-based residual lstm. In *AAAI*, volume 37, pages 4972–4980, 2023.