Only Attending What Matter within Trajectories
– *Memory-Efficient Trajectory Attention*

Mingzhi Hu[1], Xin Zhang[2], Yanhua Li[1], Yiqun Xie[3], Xiaowei Jia[4],
Xun Zhou[5], Jun Luo[6]
Worcester Polytechnic Institute[1], San Diego State University[2], University of Maryland[3],
University of Pittsburgh[4], University of Iowa[5],
Logistics and Supply Chain MultiTech R&D Centre Limited[6]
mhu3@wpi.edu

1 Abstract

Human-generated Spatial-Temporal Data (HSTD), represented as trajectory sequences, has undergone a data revolution, thanks to advances in mobile sensing, data mining, and AI. Previous studies have revealed the effectiveness of employing attention mechanisms to analyze massive HSTD. However, traditional attention models face challenges when managing lengthy and noisy trajectories as their computation comes with large memory overheads. Furthermore, attention scores within HSTD trajectories are sparse (*i.e.*, most of the scores are zeros), and clustered with varying lengths (*i.e.*, consecutive tokens clustered with similar scores). To address these challenges, we introduce an innovative strategy named Memory-efficient Trajectory Attention (MeTA). We leverage complicated spatial-temporal features (*e.g.*, traffic speed, proximity to PoIs) and design an innovative feature-based trajectory partition technique to shrink trajectory length. Additionally, we present a learnable dynamic sorting mechanism, with which attention is only computed between sub-trajectories that have prominent correlations. Empirical validations using real-world HSTD demonstrate that our approach not only yields competitive results but also significantly lowers memory usage compared with state-of-the-art methods. Our approach presents innovative solutions for memory-efficient trajectory attention, offering valuable insights for handling HSTD efficiently.

Keywords: Human-generated Spatial-Temporal Data Mining, Sparse Attention

2 Introduction

Recent advancements in mobile sensing, data mining, and AI have ushered in a paradigm shift in handling Human-generated spatial-temporal data (HSTD). Notably, companies like Uber [20] and Lyft [14] have harnessed human mobility models, utilizing GPS data from pedestrians, drivers, and gig-workers, to enhance safety and authorization [6]. HSTD plays a pivotal role in urban status monitoring, traffic management [27], and deciphering mobility patterns [25]. Moreover, attention mechanisms have made significant inroads into HSTD research. Transformer-based models have demonstrated their prowess in encoding spatial and temporal information for location generation and successful pre-training and fine-tuning on large-scale real-world datasets, yielding remarkable performances [7, 9, 12].

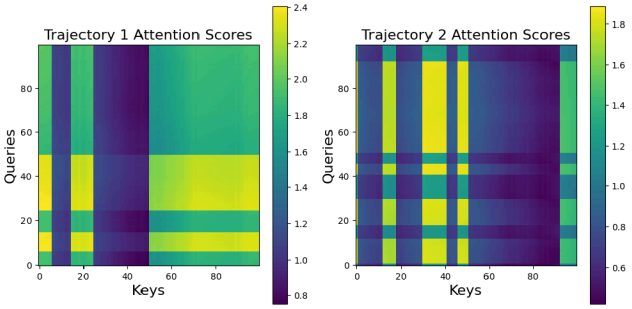

Figure 1: Attention scores heatmaps. Each heatmap corresponds to a distinct trajectory, highlighting block-wise sparsity patterns within each trajectory.

Limitations of State-of-the-art (SOTA). Attention mechanisms [21] have demonstrated impressive performances for enabling deep learning models to handle long-range dependencies in data more effectively. While powerful, the memory cost of the traditional full attention mechanisms [21] scales quadratically with sequence length, becoming infeasible for long HSTD sequences, like vehicle trajectories, and air quality maps in a big city. This leads to an increasing demand for

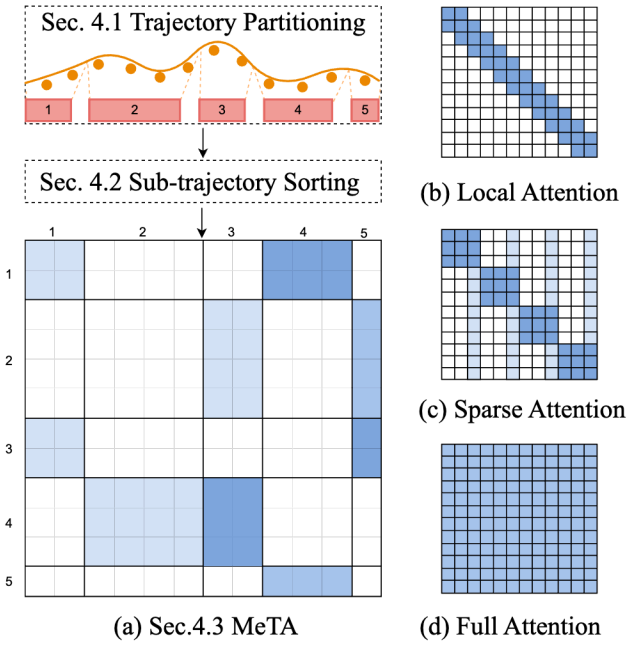

Figure 2: Our MeTA (a), and a comparison with SOTA attention maps (b) local attention [13], (c) sparse attention [18] and (d) full attention [21].

the development of sparse and efficient attention mechanisms [2–4, 10, 18, 23], and is particularly pronounced in the context of efficient spatial-temporal pre-training models. These works have explored sparse, local attention mechanisms for memory efficiency, where each token attends to a small subset of key-value pairs rather than the full set [21]. However, these methods use predefined sparse patterns or sliding windows over local contexts. As a result, they cannot capture complicated spatial-temporal correlations. *e.g.*, in a city, short segments might be used in dense urban areas with frequent turns and stops, whereas longer segments could be used on highways where movement is more uniform. This allows for a more detailed analysis of traffic flow and congestion in varying city zones.

Our preliminary investigations in Fig. 1 show that using a full attention mechanism is not necessary for spatial-temporal data. Fig. 1 shows the heat maps of the attention scores of two trajectories. They show that when using full attention on a driver identification task trained on taxi trajectory data, the resulting attention scores demonstrate (1) sparse attention patterns (*i.e.*, most of the attention scores are zeros, indicated by a darker color), and (2) strong varying-length clusters (*i.e.*, tokens on the same road grouped in sub-trajectories of a similar color). This sparsity in attention scores is attributed to the variable nature of spatial-temporal data in taxi trajectories, where segments of high activity, such as busy city streets, alternate with more uniform segments like highways. Consequently, the model focuses more on complex, frequently changing areas. Additionally, unique driving behaviors, influenced by varied urban environments, lead to distinct attention clusters, exemplified by concentrated attention in stop-and-go urban traffic versus less focus on consistent highway driving. These observations underscore the importance of intelligently grouping GPS records to form sub-trajectories based on geographical features, allowing for a deeper exploration of attention patterns within each trajectory. Additionally, applying full attention across an extended sequence introduces redundancy and noise when extracting spatial-temporal representations. **Our approach** tackles the above limitations with a memory-efficient attention mechanism for trajectory data. It involves a feature-based trajectory partitioning step, and a learnable sorting mechanism to focus attention exclusively on essential sub-trajectories. All of these design choices contribute to our Memory-efficient Trajectory Attention (MeTA) mechanism, ultimately enhancing the memory efficiency of spatial-temporal tasks. *Our contributions* can be summarized as follows:

- We introduce an advanced partition strategy to transform lengthy trajectories into shorter ones. This mechanism strategically segments trajectories into groups of consecutive spatial-temporal data point groups (or sub-trajectories) based on the strong spatial correlation and key geographical attributes. (Section 4.1)
- We further develop a learnable sorting mechanism so that sub-trajectories with stronger correlations are ranked higher. With this, we filter out unnecessary query-key computation and create a sparse attention matrix. This achievement sets a groundbreaking precedent for achieving high memory efficiency within the context of trajectory attention in the spatial-temporal domain. (Section 4.2 and Section 4.3).
- We validate our framework using real-world HSTD and demonstrate competitive results compared to baselines. Our approach plays a pivotal role in enhancing the memory efficiency of our spatial-temporal attention mechanism and refining the memory efficiency of HSTD analysis(See Section 5). *We made our code and unique dataset available to contribute to the research community via GitHub link.*[1].

3 Overview

In this section, we introduce the memory-efficient trajectory attention problem and outline associated challenges in research. For brevity, we provide a summary

[1]MeTA page: `https://github.com/mhu3/MeTA`

Table 1: Notations.

Notations	Descriptions
$p = \langle lat, lng, t, sta, spd \rangle$	GPS record.
$\tau = \{a, (p_1, p_2, \cdots, p_n)\}$	Trajectory.
$\tau' = \{a, (p_i, \cdots, p_{i+k})\}$	Sub-trajectory.
$\mathcal{T}$	Trajectory set.
$\boldsymbol{Z}$	Trajectory embedding.
$\boldsymbol{Z'}$	Sub-trajectory embedding.
L	Attention layer number.
H	Hidden size.
A	Attention matrix.
W_q, W_k, W_v	Query, key, value matrices.
d_q, d_k, d_v	Dimension of query, key, value.
q_j, k_j, v_j	Query, key, value of j-th head.
Q, K, V	query, key and value of $\boldsymbol{Z}$.
n_{head}	Attention head number.

of the notations used in this paper in Table 1.

3.1 Human-Generated Spatial-Temporal Data as Trajectories.

Human-generated spatial-temporal data (HSTD) encapsulates sequential human decisions during mobility. For instance, freight tracking, represented as GPS traces, and automatic fare collection data, recorded as transaction records, offer insights into choices made in delivery routes and daily commutes. Hence, HSTD can be interpreted as a series of trajectories, with humans navigating through spatial-temporal regions. We formally define these concepts subsequently.

Definition 1. (A trajectory τ). With the wide use of GPS devices on vehicles, smartphones, smartwatches, *etc.*, people can generate substantial spatial-temporal data at any location and time. Each GPS point p consists of latitude lat, longitude lng, a timestamp t, driving status sta, and speed spd, denoted as $p = \langle lat, lng, t, sta, spd \rangle$. Driving status illuminates the mobility pattern of a trajectory. For example, taxis might be marked based on whether they have a passenger on board. Driving status and speed are both important features for spatial-temporal data mining problems. A trajectory τ is a sequence of n GPS points generated by the human agent a, denoted as $\tau = \{a, (p_1, p_2, \cdots, p_n)\}$ and we denote the set of trajectories as $\mathcal{T}$.

Definition 2. (A sub-trajectory τ'). A sub-trajectory τ' is a contiguous segment of a trajectory τ. It is represented as $\tau' = \{a, (p_i, p_{i+1}, \cdots, p_{i+k})\}$, where a is the human agent generating the trajectory, and $(p_i, p_{i+1}, \cdots, p_{i+k})$ is a consecutive sequence of k GPS points from the original trajectory τ. The sub-trajectory τ' captures the spatial-temporal movement of the agent within a specific time and location range. Sub-trajectories can be partitioned based on the speed or location, *i.e.*, crucial features for spatial-temporal data analysis.

3.2 Limitations of Attention in Human Trajectory Analysis.

Though attention mechanisms have proven to be a powerful tool for capturing intricate spatial-temporal patterns in sequence data, its time and space complexity is quadratic in sequence length [7, 9, 12, 26]. There are two fundamental components of attention mechanisms: i) *single-head attention* and ii) *multi-head attention*. These components, while crucial in enhancing model performance, also contribute to the quadratic complexity.

Given a trajectory $\tau = \{a, (p_1, p_2, \cdots, p_n)\}$ of n GPS records, an attention module first transforms it into an embedding $\boldsymbol{Z} = [\boldsymbol{z}_1\ \boldsymbol{z}_2\ \cdots\ \boldsymbol{z}_n]$, where each $\boldsymbol{z}_i$ is an embedding of a GPS point p_i with embedding dimension H.

Single-head attention is the standard attention used in a transformer [21]. For a trajectory embedding $\boldsymbol{Z}$, it is linearly transformed using a query matrix $W_q \in \mathbb{R}^{H \times H}$, a key matrix $W_k \in \mathbb{R}^{H \times H}$, and a value matrix $W_v \in \mathbb{R}^{H \times H}$ respectively. Each input trajectory embedding is transformed into queries $Q = [\boldsymbol{q}_1\ \boldsymbol{q}_2\ \cdots\ \boldsymbol{q}_n]$ with $\boldsymbol{q}_i = \boldsymbol{z}_i W_q$, keys $K = [\boldsymbol{k}_1\ \boldsymbol{k}_2\ \cdots\ \boldsymbol{k}_n]$ with $\boldsymbol{k}_i = \boldsymbol{z}_i W_k$, and values $V = [\boldsymbol{v}_1\ \boldsymbol{v}_2\ \cdots\ \boldsymbol{v}_n]$ with $\boldsymbol{v}_i = \boldsymbol{z}_i W_v$. Single-head attention attends each query $\boldsymbol{q}_i \in Q$ to every key $\boldsymbol{k}_i \in K$, which are further used to compute a weighted sum of all values $\boldsymbol{v}_i \in V$, leading to the output,

$$\mathsf{Attn}(Q, K, V) = \mathsf{Softmax}\left(\frac{QK^\intercal}{\sqrt{H}}\right) V.$$

Multi-head attention runs through the single-head attention n_{head} times in parallel. Each embedding trajectory $\boldsymbol{Z}$ is chopped into n_{head} pieces, with H/n_{head} dimensions, and gets processed by n_{head} single-head attention modules in parallel. These output values from all heads are then concatenated and linearly transformed by a matrix W_{MSA} to produce the final outputs. The attention computation for each head can be formulated as:

$$\mathsf{SA}_j(\boldsymbol{Z}) = \mathsf{Attn}(\boldsymbol{Z}W_{q_j}, \boldsymbol{Z}W_{k_j}, \boldsymbol{Z}W_{v_j}), j = 1, \cdots, n_{\text{head}}.$$

Finally, the outputs of all n_{head} attention heads are concatenated and projected using a learnable matrix $W_{\text{MSA}} \in \mathbb{R}^{n_{head}H \times H}$,

$$\mathsf{MSA}(\boldsymbol{Z}) = \mathsf{Concat}\left(\mathsf{SA}_1(\boldsymbol{Z}), \cdots, \mathsf{SA}_{n_{\text{head}}}(\boldsymbol{Z})\right) W_{\text{MSA}}.$$

In the above computation, the matrix sizes of Q, K, and V increase with the trajectory length, and the element-wise computation of the attention maps (*i.e.*, $QK^\intercal$) is quadratic in time and space. This represents a major bottleneck for efficiently applying attention mechanisms to long HSTD trajectories.

3.3 Problem Definition: Memory-efficient Trajectory Attention.

To deal with the limitations in the SOTA works, we formally define our Memory-efficient Trajectory Attention (MeTA) problem below:

MeTA Problem Definition. Given a set of trajectories $\mathcal{T}$, our goal is to design a trajectory attention mechanism that can deal with lengthy data with various geographical features and compute the attention map efficiently in terms of memory usage.

Challenges. As illustrated in the introduction, the MeTA problem is challenging from two perspectives: (**C1**) Considering the vastness of HSTD, how can we leverage the strong spatial-temporal correlation within each trajectory to group tokens with similar geographical relationships, thereby reducing attention computation overhead? (See Section 4.1) (**C2**) With the grouped trajectory representation, how can we develop a memory-efficient attention mechanism that filters out unnecessary attention calculations? (See Section 4.2 and 4.3)

4 Methodology

To tackle the above challenges, we introduce our Memory-efficient Trajectory Attention, *i.e.*, MeTA, and illustrate its implementation. To cluster similar tokens together for challenge **C1**, we divide trajectories into distinct sub-trajectories based on their geographical features. With a learnable sorting and ranking technique, we then identify correlations between these sub-trajectories. Sub-trajectories with stronger connections receive higher ranking scores. This sorting system enables us to filter out unnecessary attention computations, yielding a sparse attention map, thereby addressing challenge **C2**. Our MeTA mechanism is depicted in Fig. 2, comparing it with both the state-of-the-art memory-efficient attention mechanism and the full attention approach.

4.1 MeTA: Trajectory Partitioning.

To enhance memory efficiency while preserving the valuable insights contained within the HSTD trajectory data, we introduce a novel approach by partitioning a complete HSTD trajectory into variable-length sub-trajectories for further sorting.

Conventional methods applied in transformers [2, 3, 10, 18, 23] often segment sequences into equal lengths, which overlook critical information embedded in specific features such as the speed and driving patterns of taxi drivers. These features are valuable for understanding traffic patterns and driver decision-making.

To address this limitation, we adopt a dynamic approach where sub-trajectories are of varying lengths based on their geographic features. This enables us to capture the inherent information present in the original features. By breaking down trajectories in this manner, we facilitate more comprehensive segmentation that takes into account the diversity of real-world HSTD trajectories (*e.g.*, vehicle trajectories, air quality dynamics), thereby enhancing its ability to efficiently process and analyze HSTD trajectory data preserving the richness of valuable features. The transformation can be expressed as follows:

$$[\tau'_1, \cdots, \tau'_{N_B}] = \psi_P(\tau). \tag{4.1}$$

In this equation, the function $\psi_P(\cdot)$ represents the segmentation operation that maps a trajectory τ into a list of N_B sub-trajectories. Each sub-trajectory τ'_b with $b = 1, \cdots, N_B$ corresponds to a specific length l_b. It is important to note that the lengths of these sub-trajectories, *i.e.*, l_b are not uniform; they can vary. This variability is influenced by factors such as the underlying characteristics of the trajectories, the inherent diversity in the data, and the need to capture detailed information for different portions of a trajectory. As a result, $[\tau'_1, \cdots, \tau'_{N_B}]$ represents a transformed representation of the original variable-length trajectory τ to adapt to the complexity and richness of the underlying data.

4.2 MeTA: Sub-trajectory Sorting.

This step is inspired by Sinkhorn ranking operation [17, 18], as one can learn a differentiable ranking that can be optimized while training the attention model. Therefore, we aim to learn the relationships between sub-trajectories using Sinkhorn ranking. It includes learning an adaptive relation vector for each sub-trajectory, followed by normalizing and assembling them into a sorting matrix.

Adaptive relation learning. When calculating the relationships between sub-trajectories, we first follow the traditional transformer and transform each trajectory into an embedding $\boldsymbol{Z}$. The sub-trajectories in the embedding form can then be denoted as $[\boldsymbol{Z}'_1, \cdots, \boldsymbol{Z}'_{N_B}]$. We represent each sub-trajectory by the sum of its tokens, *i.e.*, $\tilde{\boldsymbol{Z}}'_b = \sum_{j=1}^{l_b}(\boldsymbol{Z}'_{b,j})$ with $b = 1, \cdots, N_B$. The trainable sorting network is defined as,

$$R_b = P(\tilde{\boldsymbol{Z}}'_b),$$

where b denotes the sub-trajectory index, and $P(\cdot)$ represents an arbitrary parameterized function that takes an input sub-trajectory representation $\tilde{\boldsymbol{Z}}'_b$ and returns a relation vector of N_B dimensions. Each output dimension indicates the correspondence of the input sub-trajectory to one of the other sub-trajectories. One possible parameterization of $P(\tilde{\boldsymbol{Z}}_b)$ involves using a two-layered feed-forward network with ReLU activations:

$$P(\tilde{\boldsymbol{Z}}_b) = \sigma(W_B\sigma(W_P\tilde{\boldsymbol{Z}}_b + b_P) + b_B),$$

where W_P and W_B are weight matrices for the two linear layers. Essentially, each sub-trajectory undergoes a learning process to establish a connection with up to N_B other sub-trajectories, effectively determining the relationship to other sub-trajectories.

Sinkhorn sub-trajectory normalization. Stacking the relation vectors R_b for the N_B sub-trajectories leads to the relation matrix R. The matrix R can be viewed as a form of a permutation matrix when it becomes doubly stochastic, signifying that the matrix is non-negative, and both its rows and columns sum to 1. To provide further clarity, a relation matrix R represents a specialized instance of a doubly stochastic matrix, where all elements are exclusively 0 or 1. It is important to note that any permutation matrix can be considered a convex combination of doubly stochastic matrices. Therefore, it involves the training of matrices that approximate the characteristics of permutation matrices while allowing for a degree of relaxation, which we refer to as relaxed permutation matrices.

The Sinkhorn normalization approach refines the sorting matrix R through iterative row and column operations to approximate a doubly stochastic matrix [1]. This procedure unfolds as follows,

$$\begin{aligned} S_0(R) &= \exp(R), \\ S^k(R) &= F_c(F_r(S^{k-1}(R))), \\ S(R) &= \lim_{k\to\infty} S_k(R). \end{aligned}$$

Here, F_r and F_c are matrices that affect row and column normalization on R, and k represent the iteration number. Specifically,

$$\begin{aligned} F_r^k(\boldsymbol{Z}) &= F_r^{k-1}(\boldsymbol{Z}) \oslash \left(\boldsymbol{Z} 1_\ell 1_N^T\right), \\ F_c^k(\boldsymbol{Z}) &= F_c^{k-1}(\boldsymbol{Z}) \oslash \left(1_\ell 1_N^T \boldsymbol{Z}\right). \end{aligned}$$

In this context, $\oslash$ denotes the element-wise division operator, and N corresponds to the length of the input matrix. For numerical precision, these transformations can be applied in the log domain. It has been observed that after k iterations, the resulting $S(R)$ comes close to being doubly stochastic.

4.3 MeTA: Memory-efficient Trajectory Attention.

Given the partitioned sub-trajectories and the learned Sinkhorn sorting matrix, we can coordinate them in one attention mechanism and introduce our MeTA in this section. MeTA plays a crucial role in addressing the memory demands associated with processing HSTD trajectories. It is an innovative approach that focuses on operating with variable-length sub-trajectories, departing from the conventional method of applying full attention across all steps within each trajectory.

4.3.1 MeTA Attention on Sub-trajectories.

After partitioning the trajectory into sub-trajectories using Eq. (4.1), we obtain the query matrix for each sub-trajectory Q_i and calculate attention scores concerning selected sub-trajectory keys K_j and corresponding value matrices V_j. Then the revised computation for the attention mechanism can be expressed as follows:

$$A_{ij} = \frac{Q_i K_j^\intercal}{\sqrt{d_k}}.$$

Here, A_{ij} represents the attention scores for query vector Q_i to the block keys K_j. Subsequently, the computation of the attention matrix for block i entails the utilization of the attention scores A_{ij} in conjunction with the value matrix V_j. The attention scores are intelligently weighted by the values of R_{ij}, which serve as indicators of the extent to which a given block should allocate its attention to other blocks within the trajectory,

$$Y_{ij} = R_{ij} \cdot \mathsf{Softmax}(A_{ij}) \cdot V_j.$$

Here, Y_{ij} represents the attention-weighted values for block Q_i with the block keys K_j and V_j. The significance of R_{ij} lies in its capacity to determine the extent to which each block j contributes its attention to block i. This, in turn, exerts a notable influence on the overarching attention dynamics and the overall sparsity of the attention mechanism. To better understand the localized context, we also add local attention for each block Q_i, $Y_{ii} = \mathsf{softmax}(A_{ii}) \cdot V_i$ which makes the attention mechanism more adaptive to both global and nearby context. So the final output attention matrix for the block Q_i is:

$$Y_i = \sum_{j=1}^{N_B} Y_{ij} + Y_{ii}.$$

So the final attention matrix for the original input is:

$$Y = \mathsf{Concat}(Y_1, Y_2, ..., Y_{N_B}).$$

which provides a flexible and adaptive attention mechanism for variable-length sequences.

4.3.2 Diverse Multi-head Sinkhorn Attention.

To further enhance the capacity of our model, we also incorporate a multi-head attention mechanism. Unlike traditional multi-head attention where all heads share the same parameters, in our framework, each head employs a unique sorting network. The multi-head Sparse Sinkhorn Attention computes multiple sets of attention values, one for each attention head, using the custom sorting networks. These attention values are then collectively used to form the final output,

0	1	0	0
0	0	0	1
1	0	0	0
0	0	1	0

(a) T = 0.01

0.22	0.24	0.11	0.43
0.09	0.40	0.43	0.08
0.30	0.01	0.40	0.29
0.39	0.35	0.06	0.20

(b) T = 1

Figure 3: Variation of the R matrix w.r.t. temperature.

enhancing the model's capability to capture various facets of the input trajectories. For each of the n_{head} heads, we have dedicated query, key, and value matrices. The distinctions for head h are represented with the superscript $^{(h)}$:

$$A_{ij}^{(h)} = \frac{Q_i^{(h)}(K_j^{(h)})^\intercal}{\sqrt{d_k}},$$
$$Y_{ij}^{(h)} = R_{ij} \cdot \mathsf{Softmax}(A_{ij}^{(h)}) \cdot V_j^{(h)}.$$

Upon incorporating the outputs from multiple heads, the aggregated output for each sub-trajectory becomes:

$$Y_i^{\text{multi}} = \mathsf{Concat}\left(\sum_{h=1}^{n_{\text{head}}} Y_{ij}^{(h)} + Y_{ii}^{(h)}\right).$$

This multi-head formulation ensures that our model captures a more comprehensive representation of the input trajectories.

4.3.3 Memory-Efficient Attention Using Gumbel Softmax.

To realize the sparsity of the trajectory attention mechanism and accomplish the memory-efficiency trajectory attention, we utilize the Gumbel noise [8] as below:

$$S(\tau) = S\left(\frac{\tau + \epsilon}{T}\right),$$

where ϵ denotes the injected Gumbel Softmax, and T is the temperature. As the temperature T is lowered, the function $S(\tau)$ increasingly approximates a permutation matrix, characterized by distinct 1s and 0s. This process transforms the continuous values in $S(\tau)$ into discrete binary values, reflecting a more distinct, sparser selection in the attention mechanism. As a result, within the MeTA framework, memory efficiency is significantly enhanced by eliminating the necessity for each block Q_i to attend to every key and query block. Conversely, as the temperature T approaches 1, the sorting matrix R becomes increasingly dense. This indicates that each block Q_i retains attention across all key and query blocks. Fig. 3 illustrates these variations in the matrix R as a function of the changing temperature parameter T.

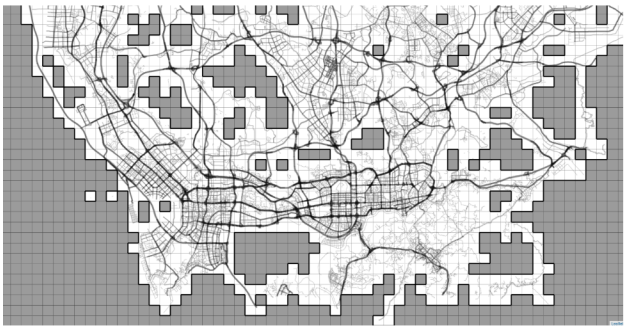

Figure 4: Map gridding demonstration.

5 Experiments

In this section, we evaluate the performances of our approach MeTA using the taxi GPS dataset collected in Shenzhen, China in July 2016. We juxtapose our model against various baselines to underscore its proficiency as a memory-efficient attention mechanism.

5.1 Data and Experiment Description. Our work takes two urban data sources as input, including (1) taxi GPS trajectory data and (2) road map data. Both datasets were collected in Shenzhen, China in 2016. **Taxi trajectory data** is gathered from 17,877 taxis in Shenzhen, China, spanning from July 1 to September 30, 2016. Each of these trajectories comprises multiple GPS records associated with a single taxi. A given GPS record encompasses six attributes: the taxi's plate ID, longitude, latitude, timestamp, driving speed during the trip, and a driving status mode, which is a binary value indicating whether a passenger is currently on board.

Road map data provides the layout of Shenzhen, covering the area between 22.44° to 22.87° latitude and 113.75° to 114.63° longitude. The data is sourced from OpenStreetMap [15], comprising approximately 21,000 roads across six levels.

Map gridding and time quantization. In our effort to protect data anonymity and minimize re-identification risks, we utilize data anonymization techniques to discretize trajectories. This involves dividing the Shenzhen area into grid cells, each with uniform side lengths of 0.01° for latitude and longitude, as previously detailed in studies [11]. After eliminating ocean-based and irrelevant cells, we have 1,934 valid cells as shown in Fig. 4. We further partition each day into 288 five-minute intervals denoted as $I = \{\tilde{t}_k\}$, where $1 \leqslant k \leqslant 288$. A spatial-temporal region r comprises a grid cell g, a time interval $\tilde{t}_k$, status sta, and speed v. Each GPS record, represented as $p = \langle lat, lng, t, sta, v\rangle$, can be mapped to an aggregated state $S = \langle g, \tilde{t}_k, sta, v\rangle$. This transformation results in agent trajectories represented as

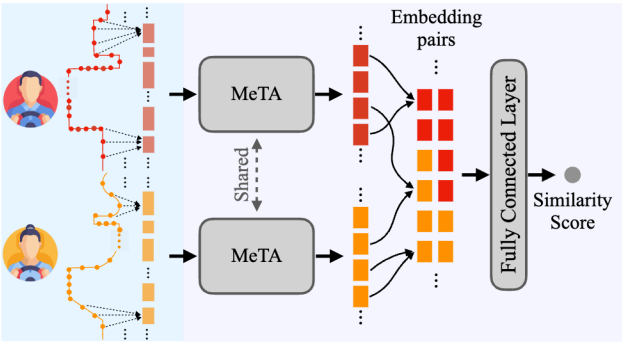

Figure 5: Self-supervised sub-trajectory similarity learning for generic spatial-temporal trajectory representations.

$\tau = \{a, \langle r_1, r_2, ..., r_n \rangle\}$.

Experiment Setups. We implement our work using Python 3.10.9 and PyTorch version 1.13.1. Our experiments are conducted on a virtual machine running Linux Ubuntu 20.04-x86_64 with NVIDIA A100-SXM4-80GB GPU. We use standard backpropagation with the Adam optimization method using hyperparameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We use a mini-batch size of 8, a fixed learning rate of 0.0001, two-layer Transformer for all models for comparison.

Experiment Description. We align our approach with the methodology employed in recent research by Hu et al. [7] on self-supervised pre-training for sub-trajectory similarity learning in the context of generic spatial-temporal trajectory representations, as illustrated in Fig. 5). In our study, we randomly select 500 drivers from the dataset, covering a 3-month period from July 1 to September 30, 2016. Within this chosen subset, we generate a total of 50,000 pairs of unique driver trajectories for training. For validation and testing, we independently select two additional subsets, each comprising 100 different drivers, and create 10,000 distinct driver trajectory pairs for each phase. To obtain the meaningful sub-trajectories illustrated in Eq. 4.1, we extract sub-trajectories based on speed. We set a threshold to segment the trajectories at 30 km/h, which efficiently differentiates between stop-and-go traffic and free-flowing conditions, ensuring the representation of diverse driving behaviors in our data.

5.2 Baseline Methods

- **Transformer [21]** introduces an architecture leveraging self-attention mechanisms, which effectively captures dependencies among elements of input sequences. Transformers excel at modeling long-range dependencies and have consistently achieved state-of-the-art performance in sequential data tasks. Notably, it employs full attention, which can be computationally demanding.
- **Sinkhorn Transformer [18]** is a attention mechanism that focuses on reducing the memory complexity of dot-product attention, it lies a Sinkhorn ranking operation [1] that is used for learning differentiable rankings over internal representations. It is based on differentiable Sinkhorn balancing and is successfully applied to differentiable sorting on large-scale tasks.
- **Local Attention [13]** is a memory-efficient attention mechanism suitable for sequential data. It selects a small subset of source positions to focus on for each target element, *i.e.*, a single element in the sequence only attends to a neighboring window of source position, which offers the advantage of avoiding computationally expensive operations seen in soft attention.
- **Fixed Sparse Transformer [4]** introduces a fixed attention pattern that summarizes information from previous positions and shares it with future positions, allowing distant elements to communicate efficiently. When using a stride of 128 and a chunk size of 8, positions greater than 128 can attend to positions 120-128, and so on. This fixed attention pattern ensures that even though attention is sparse, all sequence positions still access a global context.

5.3 Evaluation Results

In this section, we present the outcomes of our experimental study, where we evaluated the proposed MeTA model against several baseline models in the context of generic spatio-temporal trajectory representations. In particular, we evaluated MeTA and baseline models using test data to assess its memory usage with a batch size of 16, which highlights its practical applicability for real-world, trajectory-focused applications. We employ a threshold of 0.001 for the R_{ij} values. This strategic threshold step is applied after the Sinkhorn normalization iteration, a step necessary because extremely small values may emerge even with a small temperature T, as shown in Fig. 6. This decision aims to exclude these insignificant values from the sub-trajectory segmented attention mechanism, thereby enhancing memory efficiency.

We partitioned our trajectory into four sub-trajectories, *i.e.*, $N_B = 4$, based on the speed and set temperature $T = 0.01$. Table 2 presents a performance evaluation, model parameter comparison, and memory usage between MeTA and various baselines. Notably, our MeTA model achieved the lowest memory usage among the compared methods. This highlights the sparsity of the R_{ij} matrix achieved with a low temperature and underscores its efficiency in resource utilization. Although there may be a minor trade-off in accuracy, it effectively demonstrates a memory-efficient attention mechanism.

Table 2: Performance evaluation, model parameter, and memory usage comparison between MeTA *vs*. baselines.

Methods	# Params	Accuracy	Memory(GB)
MeTA	1.22M	0.8342	**1.60**
Transformer	1.22M	0.8433	3.99
Sinkhorn	1.22M	0.8444	2.28
Local	1.18M	0.8390	2.69
Fixed Sparse	1.22M	0.8356	2.96

5.0000e-01	0.0000e+00	5.0000e-01	0.0000e+00
0.0000e+00	9.8623e-26	0.0000e+00	1.0000e+00
6.3640e-21	1.0000e+00	9.1112e-27	0.0000e+00
0.0000e+00	5.0000e-01	0.0000e+00	5.0000e-01

0.0000e+00	1.0370e-43	1.0000e+00	0.0000e+00
3.7859e-38	5.0224e-21	0.0000e+00	1.0000e+00
0.0000e+00	1.0000e+00	7.3759e-08	2.4787e-09
9.0000e-01	1.0000e-01	0.0000e+00	1.0988e-35

Figure 6: Real R-value matrix of MeTA sorting.

6 Related Work

6.1 Memory-Efficient Attention Mechanisms. Transformers-based [21], exemplified by BERT [5], have had a profound impact on the Natural Language Processing (NLP) domain due to their exceptional performance. However, their reliance on full attention mechanisms, leading to a quadratic dependency on sequence length, poses computational challenges and memory inefficiencies. To address computational challenges posed by full attention, researchers have explored the concept of employing a fixed window size, known as local attention [13]. While this approach intuitively handles longer sequences, its limited window restricts tokens from accessing broader context, hindering the capture of long-term dependencies. Nevertheless, research on block-based local attention has thrived with significant contributions in recent literature [4, 10, 13, 16, 19]. Building upon the foundation of local attention windows, the Sparse Transformer [4] introduced an innovative approach that factorizes attention computation into both local and strided operations. It empowers different attention heads to focus on various sparse patterns, which has demonstrated promising results. Additionally, models like Longformer [3] and Big Bird [23] have embraced sparse attention mechanisms, effectively mitigating the quadratic dependency issue. Furthermore, sparse Sinkhorn Attention [18] introduces a learning-based sorting network for efficient sequence permutations, enabling quasi-global attention in localized windows and significantly boosting memory efficiency.

6.2 Transformer-Based Spatial-Temporal Data Mining. Transformer-based spatial-temporal data mining represents a vital intersection of transformative deep learning techniques and the exploration of spatial-temporal patterns within HSTD. The prowess of transformer-based architectures becomes evident when applied to prevalent challenges within the spatial-temporal domain, such as traffic forecasting, next-location prediction, and driver identification. Notable works, including those by Zhang *et al*. [24] and Xu *et al*. [22], focus on passenger and traffic flow forecasting, harnessing spatial-temporal transformer-based architectures to exploit dynamic spatial and temporal dependencies, consequently enhancing the accuracy of traffic forecasting with remarkable efficacy. Furthermore, transformer-based pre-training models have demonstrated remarkable effectiveness in this realm. Lin *et al*. [12] introduce a pre-training model meticulously crafted for learning representations tailored to individual locations, with a primary focus on location-related tasks like next location prediction. Meanwhile, Hu *et al*. [7] devise a self-supervised learning task that utilizes transformer-based models to bolster the performance of various downstream tasks. These advancements highlight the transformative impact that transformer-based methods are having in revolutionizing the field of spatio-temporal data mining.

7 Conclusion

In this work, we address the challenges of handling extensive trajectories within the spatial-temporal domain, specifically focusing on enhancing memory efficiency. Our primary achievement lies in the introduction of an innovative partition strategy, transforming elongated trajectories into more manageable sub-trajectories. This streamlines the trajectory analysis process and ensures data integrity by leveraging the intrinsic spatial correlation and significant geographical attributes. Our learnable sorting mechanism efficiently prioritizes correlated sub-trajectories, reducing unnecessary query-key computations and improving memory efficiency. The effectiveness of our approach is further validated by its performance on real-world data.

Future Direction. To enhance our model's performance, future research will focus on optimizing algorithmic efficiency and expanding the model's adaptability to various spatial-temporal datasets. However, our MeTA model initiates an exciting area in spatial-temporal trajectory attention.

8 Acknowledgements

Mingzhi Hu and Yanhua Li were supported in part by NSF grants IIS-1942680 (CAREER), CNS-1952085, and DGE-2021871. Yiqun Xie and Xiaowei Jia were supported by NSF award IIS-2147195.

References

[1] R. P. Adams and R. S. Zemel, *Ranking via sinkhorn propagation*, arXiv preprint arXiv:1106.1925, (2011).

[2] J. Ainslie, S. Ontanon, C. Alberti, V. Cvicek, Z. Fisher, P. Pham, A. Ravula, S. Sanghai, Q. Wang, and L. Yang, *Etc: Encoding long and structured inputs in transformers*, arXiv preprint arXiv:2004.08483, (2020).

[3] I. Beltagy, M. E. Peters, and A. Cohan, *Longformer: The long-document transformer*, arXiv preprint arXiv:2004.05150, (2020).

[4] R. Child, S. Gray, A. Radford, and I. Sutskever, *Generating long sequences with sparse transformers*, arXiv preprint arXiv:1904.10509, (2019).

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, arXiv preprint arXiv:1810.04805, (2018).

[6] D. Hallac, A. Sharang, R. Stahlmann, A. Lamprecht, M. Huber, M. Roehder, J. Leskovec, et al., *Driver identification using automobile sensor data from a single turn*, in 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2016, pp. 953–958.

[7] M. Hu, Z. Zhong, X. Zhang, Y. Li, Y. Xie, X. Jia, X. Zhou, and J. Luo, *Self-supervised pre-training for robust and generic spatial-temporal representations*, in 2023 IEEE International Conference on Data Mining (ICDM), IEEE, 2023.

[8] E. Jang, S. Gu, and B. Poole, *Categorical reparameterization with gumbel-softmax*, arXiv preprint arXiv:1611.01144, (2016).

[9] J. Jiang, D. Pan, H. Ren, X. Jiang, C. Li, and J. Wang, *Self-supervised trajectory representation learning with temporal regularities and travel semantics*, in 2023 IEEE 39th International Conference on Data Engineering (ICDE), IEEE, 2023, pp. 843–855.

[10] N. Kitaev, Ł. Kaiser, and A. Levskaya, *Reformer: The efficient transformer*, arXiv preprint arXiv:2001.04451, (2020).

[11] Y. Li, M. Steiner, J. Bao, L. Wang, and T. Zhu, *Region sampling and estimation of geosocial data with dynamic range calibration*, in 2014 IEEE 30th International Conference on Data Engineering, IEEE, 2014, pp. 1096–1107.

[12] Y. Lin, H. Wan, S. Guo, and Y. Lin, *Pre-training context and time aware location embeddings from spatial-temporal trajectories for user next location prediction*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 4241–4248.

[13] M.-T. Luong, H. Pham, and C. D. Manning, *Effective approaches to attention-based neural machine translation*, arXiv preprint arXiv:1508.04025, (2015).

[14] Lyft, *Lyft services*, 2023.

[15] OpenStreetMap contributors, *Planet dump retrieved from https://planet.osm.org* . `https://www.openstreetmap.org`, 2017.

[16] J. Qiu, H. Ma, O. Levy, S. W.-t. Yih, S. Wang, and J. Tang, *Blockwise self-attention for long document understanding*, arXiv preprint arXiv:1911.02972, (2019).

[17] R. Sinkhorn, *A relationship between arbitrary positive matrices and doubly stochastic matrices*, The annals of mathematical statistics, 35 (1964), pp. 876–879.

[18] Y. Tay, D. Bahri, L. Yang, D. Metzler, and D.-C. Juan, *Sparse sinkhorn attention*, in International Conference on Machine Learning, PMLR, 2020, pp. 9438–9447.

[19] Y. Tay, S. Wang, L. A. Tuan, J. Fu, M. C. Phan, X. Yuan, J. Rao, S. C. Hui, and A. Zhang, *Simple and effective curriculum pointer-generator networks for reading comprehension over long narratives*, arXiv preprint arXiv:1905.10847, (2019).

[20] Uber, *Uber services*, 2023.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, *Attention is all you need*, Advances in neural information processing systems, 30 (2017).

[22] M. Xu, W. Dai, C. Liu, X. Gao, W. Lin, G.-J. Qi, and H. Xiong, *Spatial-temporal transformer networks for traffic flow forecasting*, arXiv preprint arXiv:2001.02908, (2020).

[23] M. Zaheer, G. Guruganesh, K. A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, et al., *Big bird: Transformers for longer sequences*, Advances in neural information processing systems, 33 (2020), pp. 17283–17297.

[24] W. Zhang, C. Zhang, and F. Tsung, *Transformer based spatial-temporal fusion network for metro passenger flow forecasting*, in 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE), IEEE, 2021, pp. 1515–1520.

[25] X. Zhang, Y. Li, X. Zhou, and J. Luo, *Unveiling taxi drivers' strategies via cgail: Conditional generative adversarial imitation learning*, in 2019 IEEE International Conference on Data Mining (ICDM), IEEE, 2019, pp. 1480–1485.

[26] X. Zhang, Y. Li, X. Zhou, Z. Zhang, and J. Luo, *Trajgail: Trajectory generative adversarial imitation learning for long-term decision analysis*, in 2020 IEEE International Conference on Data Mining (ICDM), IEEE, 2020, pp. 801–810.

[27] Y. Zhang, Y. Li, X. Zhou, X. Kong, and J. Luo, *Strans-gan: Spatially-transferable generative adversarial networks for urban traffic estimation*, in 2022 IEEE International Conference on Data Mining (ICDM), IEEE, 2022, pp. 743–752.