Geo-information Science and Remote Sensing

Thesis Report GIRS-2025-49

# Is Mamba a Better Choice? A Comparison of Mamba, MHSA, and LSTM Neural Networks for Individual Next-Location Prediction

Changling Wang


September 23, 2025

WAGENINGEN
UNIVERSITY & RESEARCH

# Is Mamba a Better Choice? A Comparison of Mamba, MHSA, and LSTM Neural Networks for Individual Next-Location Prediction

Changling Wang

Registration number 1123556

Supervisors:

Jascha Grübel
Yanan Xin

A thesis submitted in partial fulfilment of the degree of Master of Science
at Wageningen University and Research Centre,
The Netherlands.

September 23, 2025
Wageningen, The Netherlands

Thesis code number: GRS-80436
Thesis Report: GIRS-2025-49
Wageningen University and Research Centre
Laboratory of Geo-Information Science and Remote Sensing

# Abstract

The next location prediction is a fundamental task in human mobility research and location-based services. However, existing deep learning models—such as Long Short-Term Memory (LSTM) networks and Transformer architectures—face significant challenges in modeling long-range dependencies and scaling efficiently with sequence length. The emerging Mamba architecture, grounded in State Space Models (SSMs), presents a promising alternative due to its linear time complexity and demonstrated capacity to capture long-sequence temporal dependencies effectively. This study presents a comprehensive comparative evaluation of Mamba, an encoder-only Transformer based on Multi-Head Self-Attention (MHSA), and LSTM for the next-location prediction task. Model performance is assessed across three key dimensions: prediction accuracy, computational efficiency, and robustness to domain shifts. To achieve systematic analysis, this study employs two synthetic benchmark datasets with distinct behavioral characteristics. The first dataset is generated based on the Exploration and Preferential Return (EPR) mechanism, simulating highly uncertain exploratory movement behavior; the second is generated based on the Density Transition-EPR (DT-EPR) mechanism, depicting highly regular daily travel patterns. Based on these two benchmark datasets, three next-location prediction networks(Mamba,MHSA, and LSTM) are trained and evaluated for performance. Subsequently, the trained models are tested on a structured causal interventional dataset, which achieves controlled domain transfer of group exploration tendencies by regulating the key parameters of the mechanistic generative simulator, to systematically assess the robustness of the models. The experimental results show that all models perform similarly on the regularity behavior dataset, while MHSA has a slight advantage in scenarios with higher uncertainty. Notably, both MHSA and Mamba significantly outperform LSTM, with Mamba demonstrating strong competitiveness under various conditions. Model robustness exhibits context-dependent characteristics: when the training data has higher uncertainty, MHSA shows stronger adaptability to changes in exploratory behavior. At the same time, LSTM is more robust when transitioning to regular patterns. Mamba maintains near-optimal prediction accuracy in both scenarios, consistently ranking second, with performance closely following the best network. Additionally, Mamba has a significant advantage in computational efficiency, with higher throughput and lower latency than MHSA and LSTM, especially as the sequence length increases. In conclusion, the model based on Mamba achieves an optimal overall trade-off in the next-location prediction task, and thus can be regarded as a highly competitive framework for future human mobility modeling research and scalable application deployment.



*Keywords: Human Mobility, Next Location Prediction, Mamba, Deep Learning, Robustness, Causal Inference.*

# Contents

# List of Figures



# List of Tables

# 1 Introduction

Human mobility, which encompasses the movement of people (individuals and groups) from one location to another, within a country or internationally, reflects the spatial-temporal characteristics of human behavior (Barbosa et al., 2018). Research on human mobility is crucial for advancing sustainable transportation, as it offers valuable insights into travel behavior, system design, and policy effectiveness (Cina et al., 2025). Understanding human mobility patterns enables the development of transportation systems that are environmentally friendly, economically viable, and socially equitable (Chakraborty et al., 2021). The integration of data-driven tools, such as GIS, simulation, and predictive models, supports urban planners in designing walkable, low-emission cities and making informed, evidence-based decisions (İnce, 2025).

Next location prediction is one of the fundamental tasks in mobility studies, aiming to forecast an individual's future location based on their historical trajectory data (Feng et al., 2018). This task plays a significant role in understanding and modeling individual mobility patterns, which are essential for many location-based applications, including urban planning, personalized recommendation systems, and traffic optimization (R. Wu et al., 2018). By accurately predicting a person's following location, Next location prediction models could improve resource allocation, enhance user experience through tailored services, and support proactive traffic management systems (Tian et al., 2019). Thus, it plays a significant role in the decarbonization of the transportation sector (Hong et al., 2022). The task involves capturing the intricate spatial-temporal dependencies in individual mobility data, which are often influenced by various factors such as behavior patterns, contextual constraints, and social interactions (Chekol & Fufa, 2022).

Recent advances in next location prediction have leveraged a variety of machine learning and deep learning techniques. Since next-location prediction can be formulated as a sequence prediction problem, similar to tasks in natural language processing and audio processing, models that have achieved success in these domains are frequently adapted for application (Hong et al., 2022). Thus, Methods such as recurrent neural networks (RNNs), long short-term memory networks (LSTMs), and transformer-based networks have been widely adopted to capture sequential and temporal dependencies in human mobility data, leading to huge improvements over traditional techniques like Markov Chain-based models (Feng et al., 2018; Franco et al., 2022; Hong et al., 2022; Wang et al., 2016). However, RNNs and their variants, such as LSTMs, are inherently sequential, which leads to difficulties in capturing long-term dependencies because of issues like vanishing or exploding gradients (Al-Selwi et al., 2024). These limitations can result in unstable training and reduced prediction accuracy for longer sequences (Dao & Gu, 2024). While Transformers overcome some of these issues through self-attention mechanisms and parallel computation, they introduce significant computational and memory burdens due to the quadratic complexity of attention for sequence length, making them resource-intensive and less practical for long or high-resolution sequences (Dao & Gu, 2024; Wozniak et al., 2023).

The recently proposed Mamba architecture, inspired by state space models (SSMs), addresses these limitations by achieving near-linear scalability with sequence length while maintaining strong modeling capabilities (Qu et al., 2025). Mamba demonstrates faster and higher inference throughput than Transformers, while delivering competitive or superior performance across various modalities, including language, vision, and remote sensing (Dao & Gu, 2024; Fan et al., 2024). This linear-time sequence

modeling makes Mamba a promising alternative for applications where computational efficiency and scalability are critical, effectively mitigating the resource consumption issues faced by Transformer-based models and the inherent defects of RNNs and LSTMs (Dao & Gu, 2024). These achievements reveal that mamba-based networks have the potential to have promising performance in the next position prediction task.

Despite considering the accuracy and computational efficiency of deep learning models, ensuring the robustness of predictive models under real-world conditions remains a critical challenge (Xin et al., 2022). Neural network optimization requires a training dataset. Therefore, it is widely acknowledged that its performance depends on the quality and representativeness of the training data (Yin et al., 2022). However, individual mobility behavior varies continuously across space and time (Xu et al., 2018a; Hong et al., 2023a). Consequently, the mobility data encountered by the prediction network during application often reflects different mobility patterns compared to the training data. This results in a domain shift, which refers to a discrepancy between the distributions of the training and testing data (Xin et al., 2022). Xin and Hong et al. (2025) proposed a causal intervention framework named IRMLMA(Interpretable and Robust Machine Learning for Mobility Analysis) to evaluate the impact of mobility behavior factors on the performance of neural networks in next-location prediction tasks. This framework simulates diverse mobility behavior patterns, generating trajectory data with specific characteristics, which are then input into well-trained prediction models to observe performance variations (Hong et al., 2025). Through this approach, they identify the influence of critical mobility behavior factors on prediction accuracy. Their study offers a novel perspective for improving the interpretability and evaluating the robustness of individual mobility prediction models.

The purpose of this study is to investigate the performance of the Mamba architecture in terms of prediction accuracy, robustness, and computational efficiency in the task of Next location prediction compared to the currently used encoder-only transformer (MHSA) and LSTM. I construct a Mamba-based model and then utilize the IRMLMA causal intervention framework to evaluate its robustness in an interpretable manner. At the same time, this study uses a Transformer-based model (Hong et al., 2022) and an LSTM-based model (Solomon et al., 2021) for the same operation. The results will be compared to answer the following research questions:

- RQ1: Can Mamba provide competitive or superior performance compared to the LSTM and MHSA models in individual next location prediction?
- RQ2: What are the differences in robustness among the trained Mamba, MHSA, and LSTM models when dealing with domain shift in human mobility patterns?
- RQ3: Does Mamba have an advantage over MHSA and LSTM in terms of computational efficiency?

# 2 Background and Related Work

## 2.1 *Next Location Prediction*

Generally, the Next location prediction task is defined through the following definitions.

$\mathcal{U} = \{u^{(1)}, u^{(2)}, \ldots, u^{(|\mathcal{U}|)}\}$ represent a set of users. For each user $u^{(i)}$, a location is identified when a user remains within a specific geographical area for a certain duration. The location sequence for the user $u^{(i)}$ is a time-ordered series $S^{(i)} = [L_1, L_2, \ldots, L_{w_{u^{(i)}}}]$. Each location $L = \langle l, t \rangle$ is characterized by a unique identifier $l$ from the set of all known locations $\mathcal{O}$ and the arrival time $t$.

The next location prediction task is to forecast a user's future movement. Given a historical location sequence $S^{(i)}_{\text{hist}} = [L_{n-m+1}, L_{n-m+2}, \ldots, L_n]$, where $n$ is the current timestep and $m$ is the number of past locations considered, the objective is to predict the identifier of the next location, $l_{n+1}$, that the user $u^{(i)}$ will visit, such that $l_{n+1} \in \mathcal{O}$.

## 2.2 Deep Learning Concepts

Deep learning is a subfield of machine learning. It draws inspiration from how the human brain processes information by building and training deep artificial neural networks (ANNs) with multiple processing layers to learn the intrinsic patterns and high-level representations of data(Schmidhuber, 2014). Unlike traditional machine learning methods that rely on manually engineered feature extraction, deep learning can automatically learn and extract features hierarchically from massive raw data, progressing from low-level simple features to high-level complex and abstract features(Elharrouss et al., 2022). This hierarchical feature learning capability has led to revolutionary breakthroughs in numerous fields such as computer vision, natural language processing, and recommender system tasks (e.g., next location prediction), making it a cornerstone of modern artificial intelligence technology.

### 2.2.1 Feedforward Neural Networks and Multi-Layer Perceptrons

Feedforward Neural Network is the most basic and common architecture type in deep learning. Its core characteristic is that information flows only in one direction within the network, starting from an input layer, passing through one or more hidden layers, and finally reaching an output layer(Baldi & Vershynin, 2019). There are no loops or feedback mechanisms in this data flow. The Multi-Layer Perceptron (MLP) is a typical example of a feedforward neural network, consisting of at least three layers of nodes: an input layer, one or more hidden layers, and an output layer. In an MLP, every neuron in a layer (except the input layer) is fully connected to all neurons in the preceding layer. These neurons process their weighted inputs through a non-linear activation function (e.g., ReLU or Sigmoid), which gives the network the ability to learn and model complex non-linear relationships(Pinkus, 1999).

### 2.2.2 Recurrent Neural Network (RNN)

A Recurrent Neural Network (RNN) is a neural network architecture designed for processing sequential data. Its core characteristic is the use of internal recurrent connections to maintain a **hidden state**, which encodes temporal information(Sherstinsky, 2020). At each time step $t$, an RNN cell receives the current input $x_t$ and the hidden state from the previous time step $h_{t-1}$, and then computes a new hidden state $h_t$. This process is typically defined by a non-linear transformation, with the mathematical expression being:

$$h_t = \tanh(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{1}$$

Here, $W_{xh}$ and $W_{hh}$ are the weight matrices for the input and recurrent connections, respectively; $b_h$ is the bias term; and $\tanh$ is the hyperbolic tangent activation function. The output $y_t$ at the current time step is typically generated from the hidden state $h_t$ through a linear transformation:

$$y_t = W_{hy}h_t + b_y \tag{2}$$

By updating the state in this recursive manner, RNNs can incorporate historical information into current computations, but they are often limited by the vanishing gradient problem when learning long-range dependencies.

Long Short-Term Memory (LSTM) is an advanced variant of RNN designed to overcome the vanishing gradient problem and effectively capture long-term dependencies through an intricate gating mechanism(Greff et al., 2017). The core of an LSTM cell is a dedicated cell state $C_t$, which acts as a road for information flow. This flow is controlled by three gates: the forget gate ($f_t$), the input gate ($i_t$), and the output gate ($o_t$).

At the time step $t$, the forget gate first decides what information to discard from the previous cell state $C_{t-1}$:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{3}$$

Next, the input gate decides what new information to store in the cell state, which consists of two parts:

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{4}$$

$$\tilde{C}_t = \tanh(W_C[h_{t-1}, x_t] + b_C) \tag{5}$$

The new cell state $C_t$ is updated by selectively forgetting old information and adding new information:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{6}$$

Finally, the output gate generates the hidden state $h_t$ based on the updated cell state $C_t$:

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{7}$$

$$h_t = o_t \odot \tanh(C_t) \tag{8}$$

Here, $\sigma$ is the Sigmoid function, and $\odot$ represents element-wise multiplication. This design allows LSTMs to effectively maintain and regulate the information flow across long sequences.

### 2.2.3 Multi-Head Self-Attention (MHSA) & Transformer

Multi-Head Self-Attention (MHSA) is the core building block of the Transformer model, designed to allow the model to simultaneously focus on information from different positions and different subspaces of the input sequence. MHSA is based on the Scaled Dot-Product Attention mechanism(Vaswani et al., 2017). For a set of input vectors, this mechanism first linearly maps them into three different vectors: Query (Q), Key (K), and Value (V). The attention output is obtained by computing the dot product of Q and K, scaling it by a factor of $\frac{1}{\sqrt{d_k}}$ (where $d_k$ is the dimension of the key vector), applying a Softmax function to get the weights, and finally performing a weighted sum of V. The mathematical expression is:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{9}$$

The MHSA mechanism performs $h$ independent linear transformations on Q, K, and V, and executes $h$ parallel attention calculations ("multiple heads"), allowing the model to learn relationships in different representation subspaces. The formula for each head's calculation is:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{10}$$

where $W_i^Q, W_i^K, W_i^V$ are the parameter matrices for the $i$-th head. Finally, the outputs from all heads are concatenated and passed through a final linear transformation to produce the ultimate output:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \tag{11}$$

This structure allows the model to jointly learn both global and local dependencies, significantly enhancing its expressive power.

### 2.2.4 State Space Models (SSM) & Mamba

Mamba is an emerging sequence modeling architecture designed to address the quadratic computational complexity and memory consumption issues of the Transformer when handling long sequences, while maintaining a strong ability to model long-range dependencies(Gu & Dao, 2023). The core of the model is a variant of the Structured State-Space Model (S4), which combines traditional

continuous-time state-space systems with modern deep learning methods. A State-Space Model (SSM) maps an input signal $x(t)$ to an output signal $y(t)$ via a hidden state $h(t)$, with its core dynamics described by the linear ordinary differential equations :

$$h'(t) = Ah(t) + Bx(t) \tag{12}$$

$$y(t) = Ch(t) \tag{13}$$

In Mamba, this continuous system is converted to a discrete form using fixed discretization rules (e.g., zero-order hold):

$$h_k = \bar{A}h_{k-1} + \bar{B}x_k \tag{14}$$

$$y_k = Ch_k \tag{15}$$

where $\bar{A}$ and $\bar{B}$ are discrete parameter matrices derived from a time step $\Delta$. A key innovation of Mamba is the introduction of a selection mechanism that makes the SSM parameters input-dependent. Specifically, the parameters $\bar{B}$, $\bar{C}$, and the time step $\Delta$ are functions of the current input $x_k$, for example, $\Delta_k = \text{softplus}(W_\Delta x_k + b_\Delta)$. This design allows the model to dynamically adjust its state transitions and output behavior based on the input content, enabling it to selectively focus on or forget information in the sequence(Gu & Dao, 2023). By combining an efficient parallel scan algorithm with this selective SSM, Mamba achieves linear time complexity with respect to sequence length and has demonstrated performance comparable to or even surpassing that of Transformers in various long-sequence tasks, such as language modeling and genomics(Qu et al., 2025).

## 2.3 Causal inference in Next-Location Prediction

Deep neural networks have demonstrated considerable efficacy in the domain of next-location prediction. However, their "black box" nature poses substantial challenges. These models frequently exhibit an absence of interpretability and demonstrate an inability to withstand dynamic shifts in mobility behavior, a phenomenon referred to as domain shift (Hong et al., 2025; Xin et al., 2022). This limitation arises because machine learning models typically operate on the level of association, which is the lowest level in the causal hierarchy. Causal inference provides a framework for addressing these issues by enabling higher-level reasoning, such as intervention and counterfactuals. The inherent interpretability of causal relationships, along with their invariance across different environments, renders them particularly well-suited for the development of robust and reliable predictive systems (Xin et al., 2022).

Hong et al., 2025 propose the utilisation of a causal intervention framework as a methodological approach to the systematic evaluation of the impact of behavioural factors on next-location prediction networks. The fundamental concept is to utilise controllable simulators to generate mobility data, implement causal interventions on key behavioural parameters to simulate real-world changes, and subsequently assess the performance impact on pre-trained networks. This process has been shown to reveal a network's robustness when confronted with specific domain shifts.

The framework relies on mechanistic generative models to synthesize individual location sequences, as these models contain interpretable parameters that can be treated as causal variables. The main models under this framework include the following(Hong et al., 2025):

**Exploration and Preferential Return (EPR) Model(Song et al., 2010):** The EPR model is based on two competing mechanisms: exploration and preferential return. The probability of exploring a new location is given by the formula:

$$p_{t+\Delta t}^{\text{new}} = \rho S_t^{-\gamma} \tag{16}$$

where $S_t$ is the number of distinct locations visited, while $\rho$ and $\gamma$ control the exploration tendency. With the complementary probability $(1 - p_{t+\Delta t}^{\text{new}})$, the individual returns to a previously visited location $j$ with a probability $\Pi_j$ proportional to its visit frequency $f_j$ ($\Pi_j \propto f_j$).

**Density-EPR (D-EPR) Model(Pappalardo et al., 2015):** The D-EPR model modifies the EPR exploration mechanism to account for population attractiveness. The probability $\Pi_j$ of selecting a new location $j$ depends on its attractiveness $n_j$ and the distance $r_{i,j}$ from the current location:

$$Pi_j \propto n_j r_{i,j}^{-2} \tag{17}$$

**Individual Preferential Transition (IPT) Model(C. Zhao et al., 2021):** The IPT model modifies the preferential return mechanism by conditioning it on the current location, forming a first-order Markov process. The probability $\Pi_j$ of returning to a location $j$ from the current location $i$ is proportional to the historical transition frequency $f_{i \to j}$:

$$Pi_j \propto f_{i \to j} \tag{18}$$

**Density Transition-EPR (DT-EPR) Model(Hong et al., 2025):** Proposed in the study by Hong et al., 2025, this model combines the exploration mechanism of d-EPR with the preferential return mechanism of IPT. The DT-EPR model can therefore capture both population attractiveness and individual preferences in location choices, generating more realistic mobility traces.

The key behavioral parameters in these models, which can serve as the targets for intervention, are the exploration tendency ($p^{\text{new}}$, determined by $\rho$ and $\gamma$), population attractiveness ($n$), and individual preference ($f$) . By altering the values or distributions of these parameters, researchers can simulate the movement trajectories of individuals under different mobility patterns. This lays a solid foundation for explaining the robustness of individual mobility prediction models under domain shift.

# 3 Data & Methodology

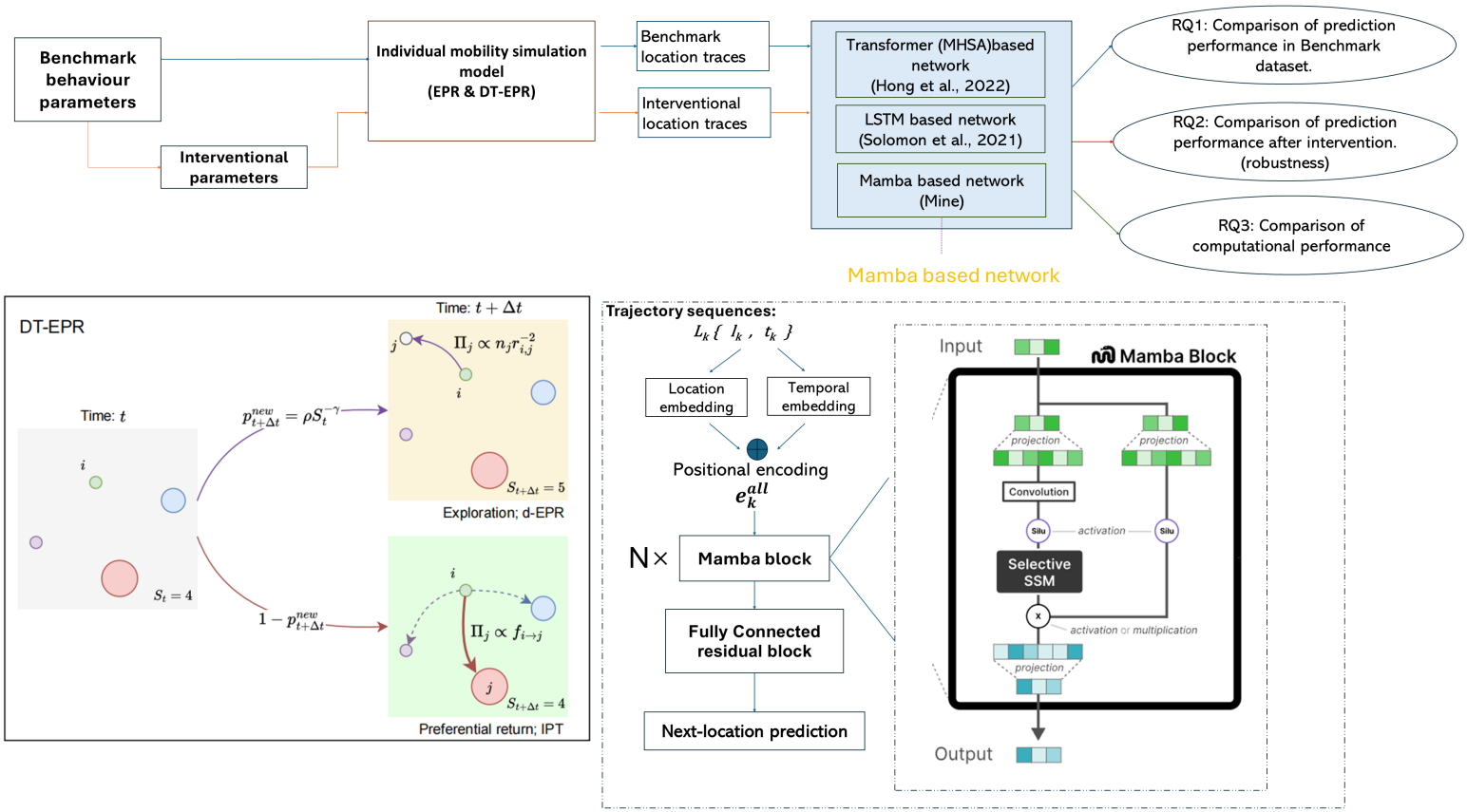


Figure 3.1: Overall Research Flowchart

This study conducts a comparative analysis of three deep learning architectures —LSTM, MHSA, and Mamba —for the next-location prediction task. As shown in Figure 3.1, these models are evaluated based on their prediction accuracy, robustness to behavioral shifts, and computational efficiency. Our methodology is built upon the causal intervention framework proposed by Hong et al., 2025, which utilizes synthetic data to create a controlled environment for rigorous model evaluation. Our evaluation is structured around three core research questions (RQs): prediction accuracy (RQ1), robustness against domain shift (RQ2), and computational efficiency (RQ3).

## 3.1 Synthetic Data

### 3.1.1 Benchmark Data Generation

Our benchmark datasets are generated using mechanistic simulation models to reflect human mobility patterns. The parameters for generating the benchmark datasets are calibrated and validated based on the findings of Hong et al., 2025, whose work was grounded in the large-scale SBB Green Class (GC) dataset. The SBB Green Class (GC) dataset, initiated by the Swiss Federal Railways (SBB), was designed to evaluate the influence of a Mobility-as-a-Service (MaaS) solution on travel behavior(Martin et al., 2019). The study engaged 139 participants residing in Switzerland over a period spanning from November 2016 to December 2017. It is important to note that due to the privacy-sensitive nature of individual mobility data, the original GC dataset is not publicly available. Therefore, the foundation user trajectories I used to generate our dataset were synthetic data generated by Hong using the DT-EPR model to simulate the GC dataset. This means our synthetic dataset was created by secondary synthesis of data based on the GC dataset.

The default parameter settings for the simulation, derived empirically from the GC data by Hong, are as follows:

**The Jump Length** ($P(\Delta r)$), which describes the probabilistic behavior of the distance traveled when moving from one location to an unvisited location, follows a log-normal distribution form with parameters: mean $\mu|_{\Delta r} = 7.72$ and standard deviation $\sigma|_{\Delta r} = 2.38$.

**The Waiting Time** ($P(\Delta t)$), which assumes that an individual changes its location after a waiting time, follows a log-normal distribution form with parameters: mean $\mu|_{\Delta t} = 0.75$ and standard deviation $\sigma|_{\Delta t} = 1.49$.

**The exploration tendency parameters**$P(\rho)$ and $P(\gamma)$ in Equation 16 , follow normal distributions. Specifically, $P(\rho)$ has a mean $\mu|_{\rho} = 0.18$ and standard deviation $\sigma|_{\rho} = 0.07$, while $P(\gamma)$ has a mean $\mu|_{\gamma} = 0.64$ and standard deviation $\sigma|_{\gamma} = 0.16$.

All generated datasets comprise mobility traces for 800 synthetic virtual users, with each user having a sequence of 2,000 location visits.

To begin with, this study generated four benchmark datasets using EPR, D-EPR, IPT, and DT-EPR models to compare the differences in datasets generated by different Individual mobility simulation models to decide which benchmark data will be selected to train our deep learning network. The datasets generated by the corresponding mechanisms will be referred to as the EPR dataset, D-EPR dataset, IPT dataset, and DT-EPR dataset, respectively, in the subsequent sections.

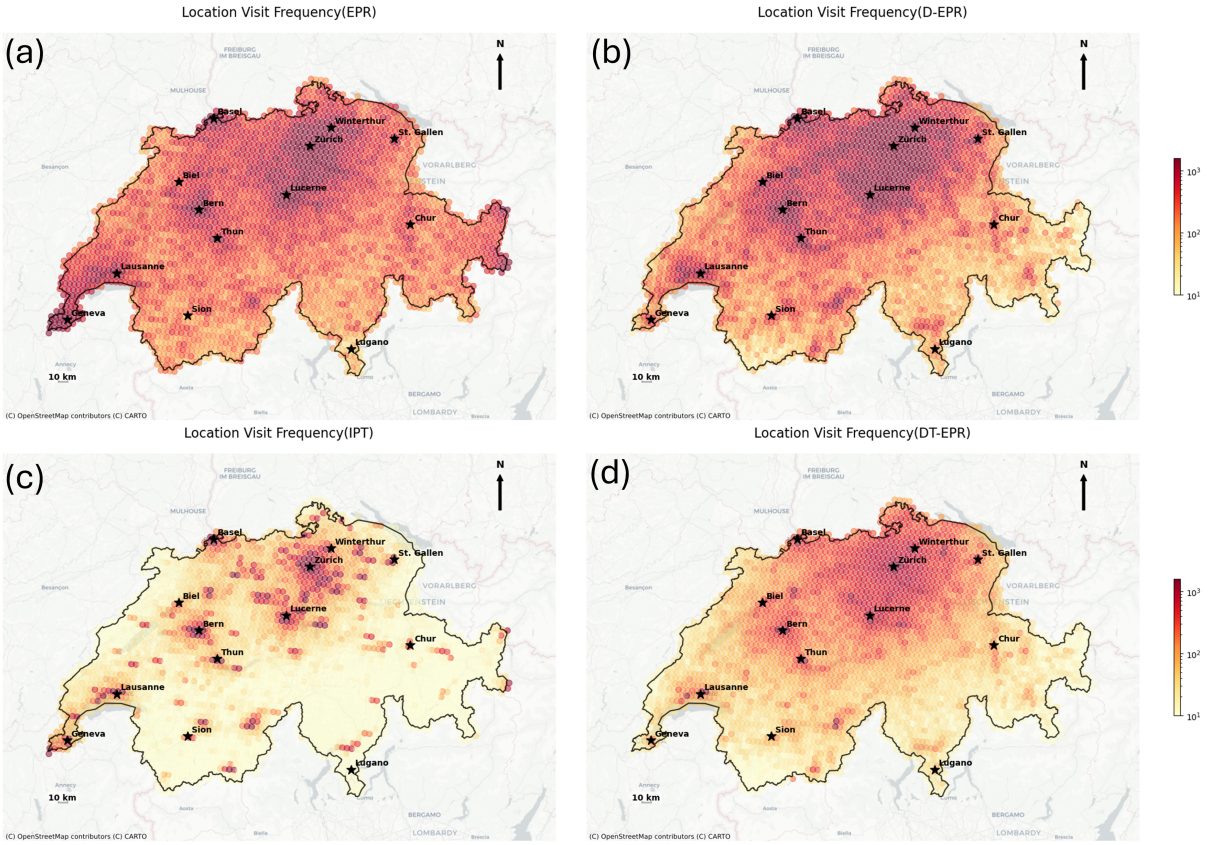


Figure 3.2: Figure 3.2: Locations visit frequency of the datasets generated by four different individual mobility simulation models. The panels show the results for datasets generated by: (a) the EPR model, (b) the d-EPR model, (c) the IPT model, and (d) the DT-EPR model. Darker colors indicate a higher frequency of visits.

Figure 3.2 displays the aggregated location visit frequency heatmaps for the simulated user population, and Figure 3.3 illustrates the trajectory of a randomly selected user from each dataset. These figures represent the distinct spatial characteristics of the four models. The EPR and D-EPR datasets' visit heatmap is broad and diffuse at the population level, corresponding to a high individual trajectory uncertainty. Individuals may have multiple frequently visited locations (as indicated by several green circles significantly larger than others in the diagram). In contrast, the population heatmaps of the IPT and DT-EPR datasets are highly concentrated in major urban centers, with individual trajectories that are more centered in two specific areas. However, compared to the IPT model, the DR-EPR model is more willing to visit more unknown locations.

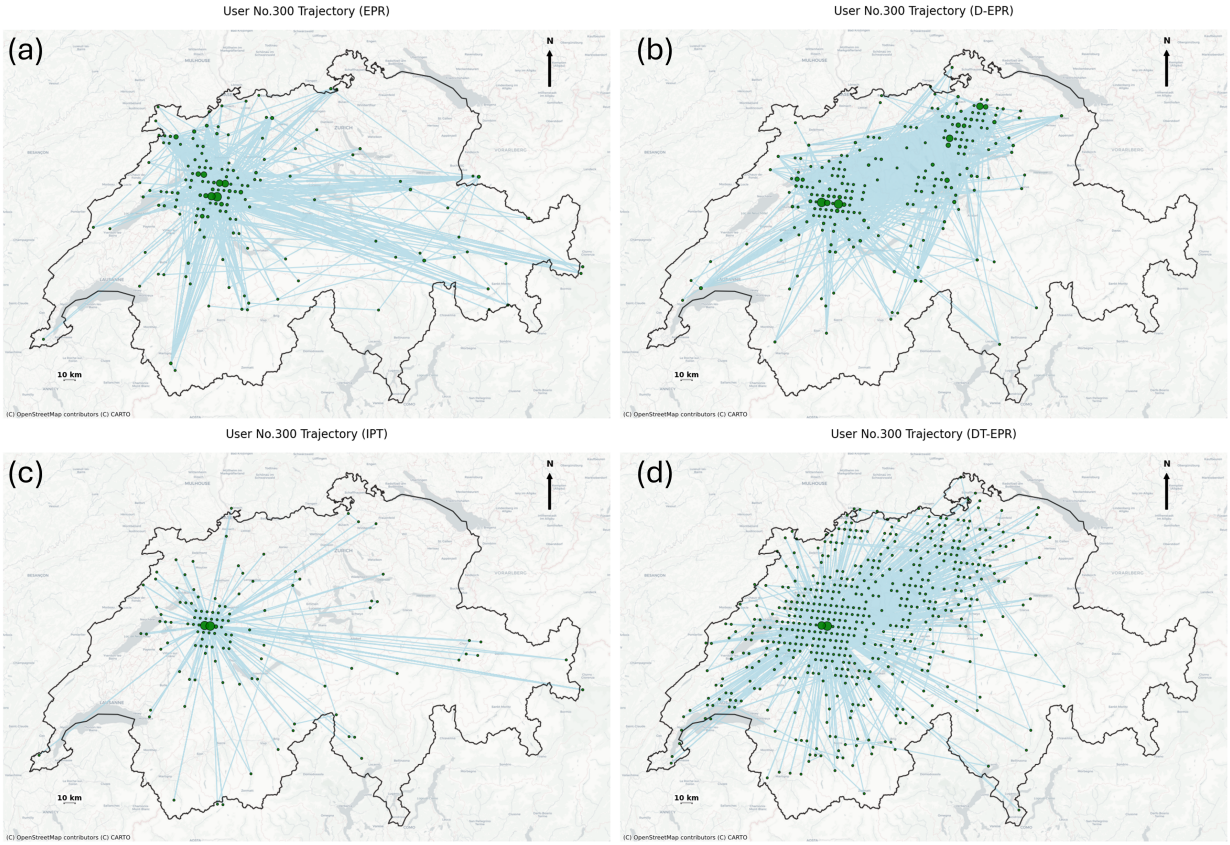


Figure 3.3: Comparison of an individual user trajectory generated by four different mobility simulation models. The panels display the trajectory for an exemplary user generated by: (a) the EPR model, (b) the d-EPR model, (c) the IPT model, and (d) the DT-EPR model. Green dots represent visited locations, with the size of each dot being proportional to the individual's visit frequency to that location.

To validate the generated data against the ground truth GC data, this study used two key metrics consistent with Hong et al. (2025). Real Entropy quantifies the regularity and theoretical predictability of location sequences by considering visit order and frequency. Mobility Motifs Proportion measures the prevalence of common, recurring daily travel patterns.

Figure 3.4 compares the distributions of mobility entropy and motifs proportion for Synthetic datasets generated by mobility generative models against the empirical GC dataset. For entropy (top row), the GC data show a unimodal distribution with a peak around 2.8. At the same time, the EPR dataset presents a distribution shape most similar to that of the GC data in this metric. For the motifs proportion (bottom row), the GC data has a broad distribution centered around 0.7. In contrast, the EPR and d-EPR models peak at a very low proportion, while the IPT model peaks at a higher one. The distribution of the DT-EPR data is again the most visually similar to the ground-truth GC data in both shape and location.

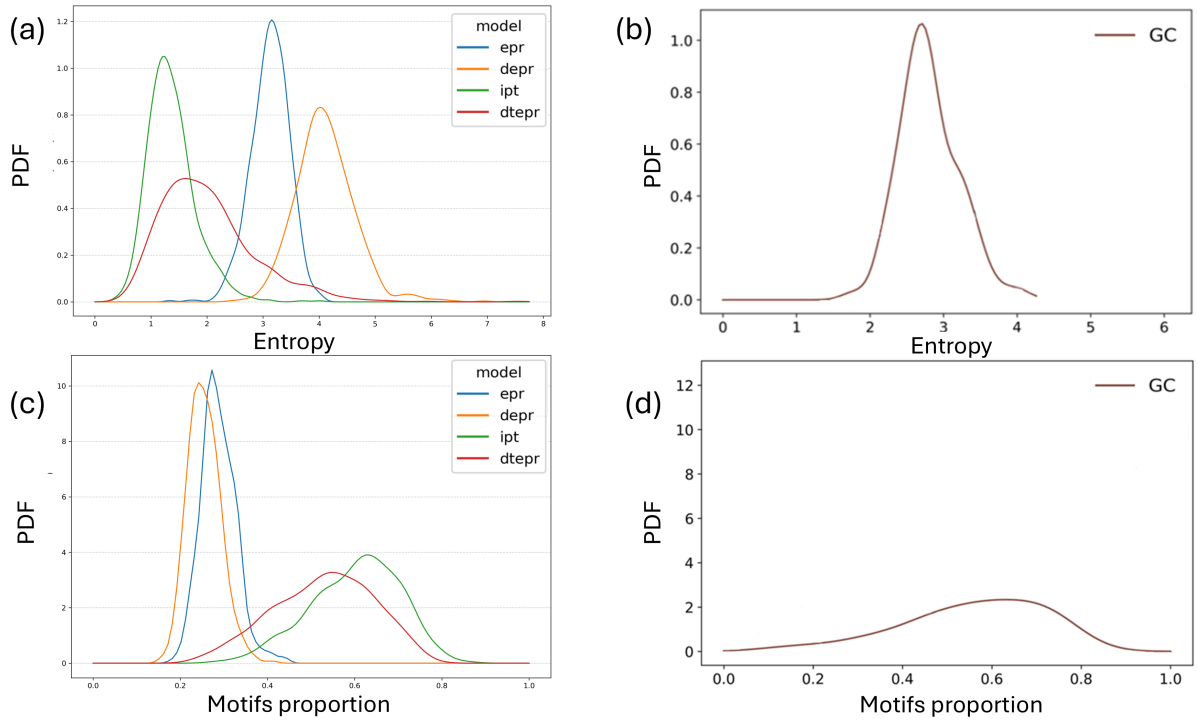


Figure 3.4: Comparison of mobility behavior metrics for synthetic datasets and the empirical GC dataset. (a) Probability density functions (PDFs) of mobility entropy for the four simulation datasets (EPR, d-EPR, IPT, DT-EPR). (b) PDF of mobility entropy for the ground-truth GC dataset. (c) PDFs of the motifs' proportions for the four simulation datasets. (d) PDF of motifs proportion for the ground-truth GC dataset.Note: The figures about GC data are adapted from Hong et al., 2025

.

Based on the analysis of mobility behavior metrics, this study selected the EPR and DT-EPR models to generate our two benchmark datasets, which will be used to train the Next location prediction deep learning network. The DT-EPR dataset was chosen as the high-regularity benchmark because its motif proportion distribution is the most visually similar to the ground-truth GC data, indicating it captures daily-routine-based travel patterns. In contrast, the EPR dataset was chosen as the high-uncertainty benchmark, as the figure shows it produces trajectories with the highest entropy and lowest motif proportion, representing a more chaotic and less predictable mobility pattern. By utilizing these two distinct datasets, this study can systematically evaluate the performance of the deep learning architectures under both realistic, structured conditions and challenging, less structured conditions.

### 3.1.2 Interventional Data Generation

To evaluate model robustness (RQ2), this study generated interventional datasets by applying causal interventions to the data generation process. It focused exclusively on interventions that manipulate the **exploration tendency** of the synthetic users. The specific intervention design involves altering the parameters that control the probability of exploring a new location ($p^{\text{new}}$) through two approaches: soft interventions, where the mean values for the distributions of parameters $\rho$ and $\gamma$ were systematically altered, and hard interventions, where the exploration probability $p^{\text{new}}$ itself was fixed to a constant value. We set the mean of the distributions for these parameters to values of 0.1, 0.25, 0.5, 0.75, 0.9 to simulate populations with varying degrees of exploration behavior. A different random seed was used for generating the interventional datasets compared to the benchmark datasets. Consequently, the users in the interventional datasets are treated as a distinct and unrelated population from the users in the benchmark dataset. This setup simulates a realistic scenario in which a pre-trained model encounters a new user group with different behavioral characteristics, thereby providing a robust test of the model's generalization capabilities.

Figure 3.5 and Figure 3.6 illustrate the outcomes of applying causal interventions to the EPR and DT-EPR benchmark datasets, respectively. The results show a consistent and systematic relationship between the exploration tendency and the resulting dataset's behavioral metrics across both models. Interventions that increase exploration lead to higher entropy and lower motif proportion, while interventions that suppress exploration have the opposite effect. The hard interventions on the exploration probability, $p^{\text{new}}$, demonstrate this most directly; as $p^{\text{new}}$ is increased, the entropy distributions for both datasets shift to higher values and the motifs proportion distributions shift to lower values (Figures 3.5a, 3.6a). Similarly, soft interventions that increase the mean of the $\rho$ parameter, which encourages exploration, also result in higher entropy and lower motifs (Figures 3.5b, 3.6b). Conversely, increasing the mean of the $\gamma$ parameter, which suppresses exploration, systematically decreases entropy and increases the motifs proportion, making the generated trajectories more regular for both models (Figures 3.5c, 3.6c). This confirms that the intervention design successfully created a spectrum of datasets with controlled and varying degrees of regularity. This lays a solid foundation for subsequent investigations into the robustness of different next location prediction deep learning networks when handling out-of-distribution data.

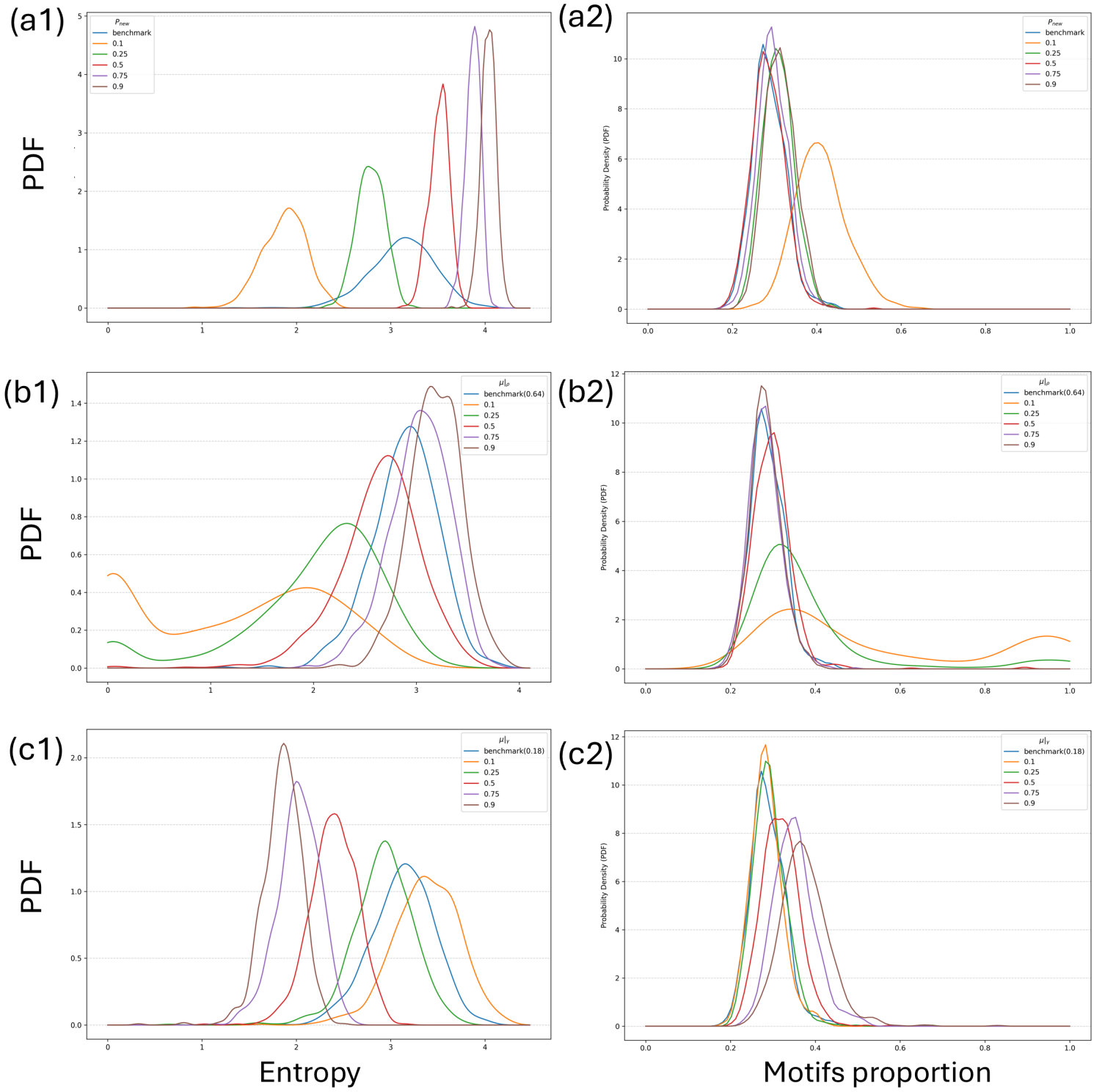


Figure 3.5: Distributions of mobility entropy and motifs proportion for the interventional EPR datasets. The left column (a1, b1, c1) shows the impact on entropy, while the right column (a2, b2, c2) shows the impact on motif proportion. The interventions applied are: (a) hard interventions fixing the value of $p^{\text{new}}$, (b) soft interventions altering the mean of the $\rho$ distribution, and (c) soft interventions altering the mean of the $\gamma$ distribution, each compared to the benchmark data.

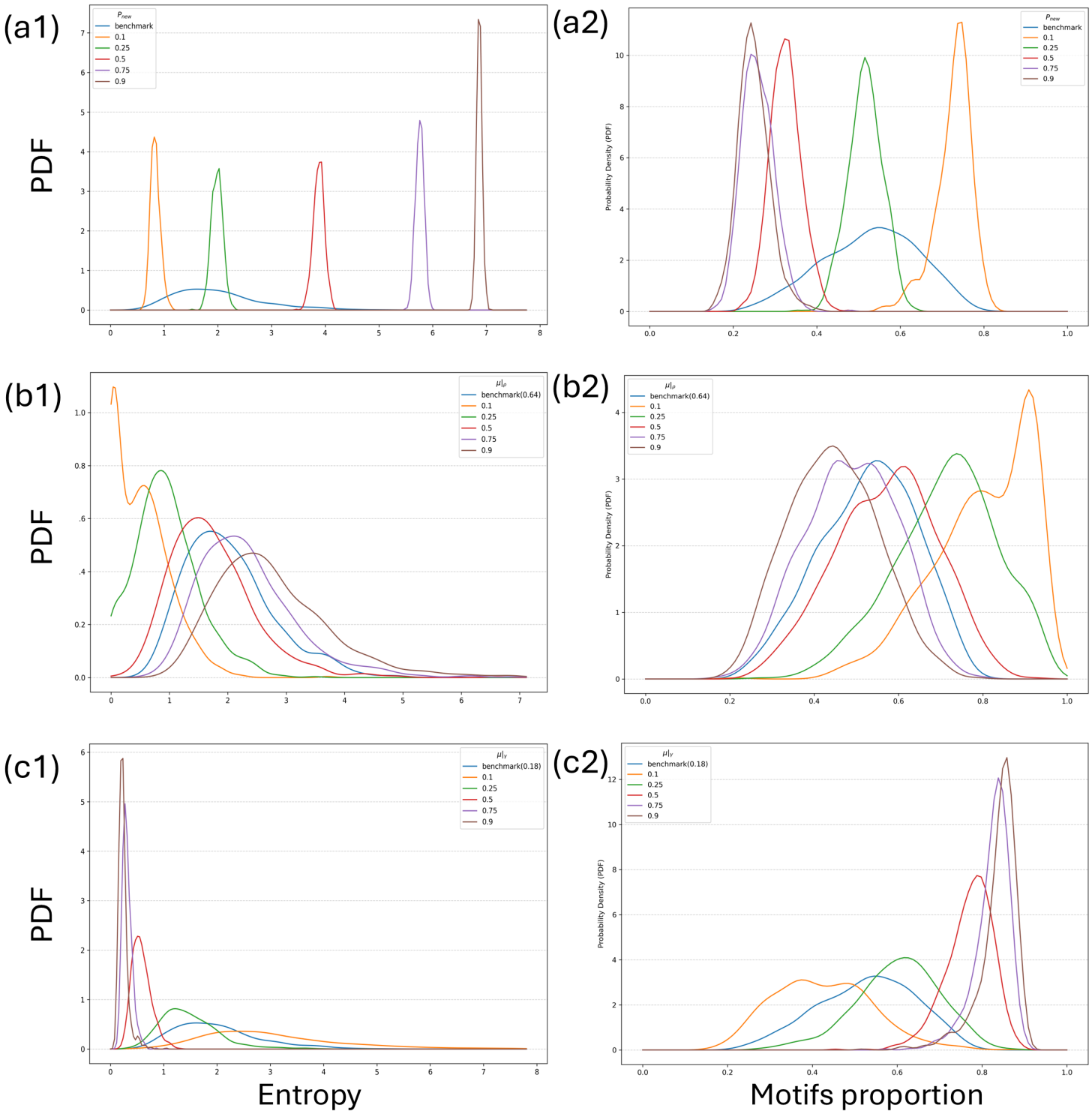


Figure 3.6: Distributions of mobility entropy and motifs proportion for the interventional DT-EPR datasets. The left column (a1, b1, c1) shows the impact on entropy, while the right column (a2, b2, c2) shows the impact on motif proportion. The interventions applied are: (a) hard interventions fixing the value of $p^{\text{new}}$, (b) interventions altering the mean of the $\rho$ distribution, and (c) interventions altering the mean of the $\gamma$ distribution, each compared to the benchmark data.

## 3.2 Next-Location Prediction Network

### 3.2.1 Model Architecture

In this study, the individuals' trajectory data from the past 7 days are used to predict their next location. The architecture for the next-location prediction networks, as shown in Figure 3.1, consists of three main components: an embedding block, a sequence encoder, and an output block. The sequence encoder is the component that varies using LSTM, MHSA, and Mamba, respectively.

First, a given trajectory sequence is converted into a series of high-dimensional vectors in the Embedding block. Following the method of Hong et al. (2025), the embedding block uses separate embedding layers for location identifiers and temporal features. The positional embedding directly uses the ID of each position, similar to treating it as a token in natural language processing tasks. The temporal features include the time of day (grouped into 15-minute bins), hour of the day, and day of the week. These embeddings are then combined to form the final input vector for each time step, $e_k^{\text{all}}$:

$$e_k^l = h^l(l_k; W^l) \quad \text{and} \quad e_k^t = h^t(t_k; W^t)$$
$$e_k^{\text{all}} = e_k^l + e_k^t$$

In this study, user embedding is not implemented. Because the users in the benchmark and interventional datasets are considered two different populations, the user IDs generated during simulation carry no transferable meaning.

Positional encoding is added to these vectors to inject information about the relative order of locations in the sequence. The reason is that the self-attention mechanism does not inherently process the order of the input sequence. To provide the model with information about the position of each token, we inject *positional encodings* into the input embeddings. These encodings are added to the input embeddings at the bottom of the encoder stacks. The positional encoding has the same dimension $d_{\text{model}}$ as the embeddings, so that the two can be summed. This study employed the most common positional encoding method introduced by Vaswani et al. (2017), which utilizes sine and cosine functions of varying frequencies.

The following equations define the positional encoding:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

Where $pos$ is the position of the token in the sequence, $i$ is the dimension index of the positional encoding vector. $d_{\text{model}}$ is the dimension of the model's embeddings.

Next, the Sequence Encoder processes the sequence of embedding vectors. This study implements and compares three different encoders: a standard LSTM network, a Transformer encoder based on MHSA, and a novel architecture utilizing the Mamba Block, whose core component is the Selective State Space Model (SSM) for modeling long-range dependencies.

Finally, the output from the sequence encoder is passed to an Output Layer, which is a Fully Connected (FC) layer with a residual connection. This block ultimately produces the visit probabilities of all locations at the next time step. This indicates that the task can be regarded as a multi-class classification problem.

### 3.2.2 Model Training

The models are trained to predict the next location based on a user's trajectory from the previous 7 days. The multi-class Cross-Entropy (CE) loss function is utilised, as it is commonly employed in classification tasks. The loss $\mathcal{L}$ is calculated as:

$$\mathcal{L} = -\sum_{k=1}^{|\mathcal{O}|} P(l_{n+1})^{(k)} \log(P(\hat{l}_{n+1})^{(k)})$$

where $|\mathcal{O}|$ is the total number of locations, $P(l_{n+1})$ is the one-hot encoded ground truth, and $P(\hat{l}_{n+1})$ is the model's predicted probability distribution.

The EPR and DT-EPR benchmark datasets were both partitioned into training (60%), validation (20%), and test (20%) sets. All models were trained for a maximum of 100 epochs using the AdamW optimiser, with an early stopping strategy implemented with a patience of 5 epochs in order to prevent overfitting. The optimal hyperparameters for each model architecture were determined through a comprehensive grid search, as detailed in Table B.1. All experiments were conducted on a consistent hardware platform consisting of a single NVIDIA RTX 4090 GPU (24GB), a 16 vCPU Intel(R) Xeon(R) Platinum 8352V CPU @ 2.10GHz, and 120GB of RAM. This process resulted in a total of six trained models: LSTM, MHSA and Mamba trained on the EPR dataset, and the same three architectures trained on the DT-EPR dataset. These models were subsequently used for evaluation.

### 3.2.3 Model Evaluation

To provide a comprehensive assessment of the LSTM, MHSA, and Mamba models, this evaluation is structured around three key aspects: Prediction Performance, robustness to domain shift, and computational efficiency.

In order to evaluate the predictiion accuracy of the models (RQ1), the standard metrics used in next-location prediction literature are adopted, in accordance with the work of Hong et al. (2025). The metrics employed include Accuracy@k (Acc@k), which quantifies the proportion of instances where the ground-truth next location is identified within the top-k predictions of the model (for k=1, 5, and 10).

The Weighted F1-Score is also utilised in this study. This metric is employed to calculate the F1-score, which is defined as the harmonic mean of precision and recall, for each location (class). Subsequently, a weighted average is calculated based on the number of true instances (support) for each location.

This underscores the significance of predicting frequently visited places. The formula is as follows:

$$\text{Weighted F1} = \sum_{i \in L} w_i \cdot \left( 2 \cdot \frac{P_i \cdot R_i}{P_i + R_i} \right)$$

where $L$ is the set of all locations, $P_i$ and $R_i$ are the precision and recall for location $i$, and $w_i$ is the proportion of instances of location $i$ in the dataset.

Finally, the Mean Reciprocal Rank (MRR) is utilised to evaluate the overall ranking quality of the predictions. The MRR is the average of the reciprocal ranks of the correct next location over the entire test set, calculated as follows:

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}$$

where $|Q|$ is the total number of predictions in the test set, and $\text{rank}_i$ is the rank of the ground-truth next location for the $i$-th prediction.

To evaluate the models' robustness against domain shift (RQ2), the interventional datasets are leveraged. The models trained exclusively on the benchmark dataset (either EPR or DT-EPR) are then subjected to direct evaluation on the various interventional test sets. This process occurs without the implementation of any fine-tuning or re-training. Robustness is quantified by measuring the performance degradation, defined as the drop in the Prediction Performance metrics (Acc@k, MRR, F1-Score) when the model transitions from the benchmark test set to an interventional test set. A model that exhibits a smaller performance drop when faced with these behavioural shifts is considered more robust and has better generalisation capabilities.

To evaluate computational efficiency (RQ3), I measure four key metrics that capture different aspects of a model's computational cost. The Number of Parameters indicates the size of the model and its memory footprint. FLOPs (Floating Point Operations) provide a hardware-agnostic measure of the theoretical computational complexity required for a single forward pass. To measure practical performance, I record the inference throughput as PPS (Predictions Per Second), which quantifies the number of sequences the model can process in one second. Finally, I measure inference latency in milliseconds per batch, which records the average time required to process a single batch of data. It is important to note that since the model input is based on the user's location history over the past 7 days, the sequence length is variable. Furthermore, the use of grid search for hyperparameter tuning results in optimal models with different total parameter counts. Therefore, to ensure a fair comparison of architectural speed, I conducted an additional set of experiments. In these controlled tests, I compared the training and inference speeds of the three models under conditions of a fixed, identical sequence length and a similar number of total parameters.

Towards the evaluation of computational efficiency (RQ3), several metrics were measured to capture different aspects of a model's computational cost. The assessment encompasses both theoretical complexity and practical speed measurements.

FLOPs (Floating Point Operations) provide a hardware-agnostic measure of the theoretical computational complexity for a single forward pass. Practical performance is measured by Throughput, which quantifies the number of sequences the model can process per second (samples/sec). We measure

both Training Throughput and Inference Throughput. Throughput is calculated as:

$$\text{Throughput} = \frac{\text{Number of Samples}}{\text{Total Time (s)}}$$

Inference Latency measures the average time in milliseconds required to process a single sample (ms/sample) and is calculated as:

$$\text{Latency} = \frac{\text{Total Time (s)} \times 1000}{\text{Number of Samples}}$$

It is important to note that since the model input in our main experiment has a variable sequence length and the models have different parameter counts after hyperparameter tuning, we conducted a dedicated set of controlled experiments for this evaluation. In these tests, the speeds of the three models were compared under conditions of fixed, identical sequence lengths and similar total parameter counts (LSTM: 2.57M, MHSA: 2.64M, Mamba: 2.51 M) to ensure a fair comparison. For these experiments, a relatively small batch size of 16 was used. This choice was made to enable the evaluation of model performance at longer sequence lengths without exceeding the available GPU memory.

# 4 Results

This section presents the empirical findings, comparing the models' Prediction Performance on two synthetic benchmark datasets (RQ1) and their robustness (RQ2) on out-of-distribution interventional datasets and computational efficiency in different sequence lengths (RQ3).

## 4.1 Prediction Performance on Benchmark Datasets

Initially, the optimal parameters were obtained through grid search within the hyperparameter combinations.(Table B.1). The optimal values were used to train the model that was ultimately employed for comparison. As outlined in Figure C.1, all models demonstrated effective convergence during training on both benchmark datasets. Training convergence for the three models on the DTEPR Dataset was faster than that of EPR, with early stopping achieved within 20 epochs. Conversely, training on the EPR Dataset necessitated between 50 and 70 epochs before the occurrence of early stopping.

The Prediction Performance of the LSTM, MHSA, and Mamba-based models, when trained and evaluated on the two distinct benchmark datasets, is summarized in Table 4.1. The results highlight the impact of data regularity on model performance and reveal the competitive capabilities of the different architectures.

On the high-uncertainty EPR dataset, which is characterized by its low regularity, the MHSA model demonstrated consistent superiority across all six evaluation metrics. The Mamba-based model was consistently ranked as the second-best performer on this dataset, with a significant outperformance of the LSTM model.

On the high-regularity DT-EPR dataset, which exhibits clear transition patterns, all three models achieved substantially higher accuracy due to the dataset's more structured and predictable nature. The models' performance on this dataset was much closer, with the MHSA and Mamba models showing highly competitive results. The Mamba model achieved the highest Acc@1, while the MHSA model secured the top performance in most other metrics.

Overall, the three networks demonstrate negligible differences in Prediction Performance on the high-regularity moving dataset. However, on the more random dataset, MHSA demonstrates slightly higher performance than Mamba, though both significantly outperform LSTM.

Table 4.1: Prediction Performance of LSTM, MHSA, and Mamba models on the EPR and DT-EPR benchmark datasets. The best performing model for each metric is highlighted in **bold**, and the second-best is underlined.

| Benchmark Dataset | Network | Acc@1 | Acc@3 | Acc@5 | Acc@10 | F1 | MRR |
|---|---|---|---|---|---|---|---|
| EPR | LSTM | 9.626 | 23.253 | 31.915 | 43.793 | 8.096 | 20.318 |
| | MHSA | **11.265** | **25.684** | **34.638** | **46.662** | **10.349** | **22.373** |
| | Mamba | 10.817 | 24.615 | 33.226 | 44.848 | 9.733 | 21.514 |
| DT-EPR | LSTM | 73.392 | 76.029 | 76.420 | 76.890 | 63.641 | 74.926 |
| | MHSA | 73.418 | **76.086** | **76.510** | **77.045** | **63.773** | **74.994** |
| | Mamba | **73.435** | 76.032 | 76.438 | 76.918 | 63.732 | 74.966 |

To provide a spatial perspective on these results, Figure 4.1 illustrates the per-location Top-5 accuracy for each model. On the high-regularity DT-EPR dataset, the figure shows that high prediction accuracy is heavily concentrated in the exact few high-frequency locations for all three models (a2, a3, a4), which correspond directly to the truth visit hotspots (a1). This visually confirms that all models excel at predicting routine, frequently visited destinations but struggle with infrequent ones. However, in the upper-middle regions of the map (a1, a2, a3), we can still observe that for infrequently visited points, the number of locations with high accuracy (shown in light green) follows the order MHSA > Mamba > LSTM.

In contrast, on the high-uncertainty EPR dataset, where truth visits are more diffuse (b1), a clear difference in spatial accuracy emerges. The LSTM accuracy map (b2) is visibly paler, meaning it is less accurate than Mamba (b3) and MHSA (b4). This provides evidence that Mamba and MHSA demonstrate significantly stronger capabilities than LSTM in predicting sparse categories within the training set. In comparison, MHSA is slightly stronger than Mamba.

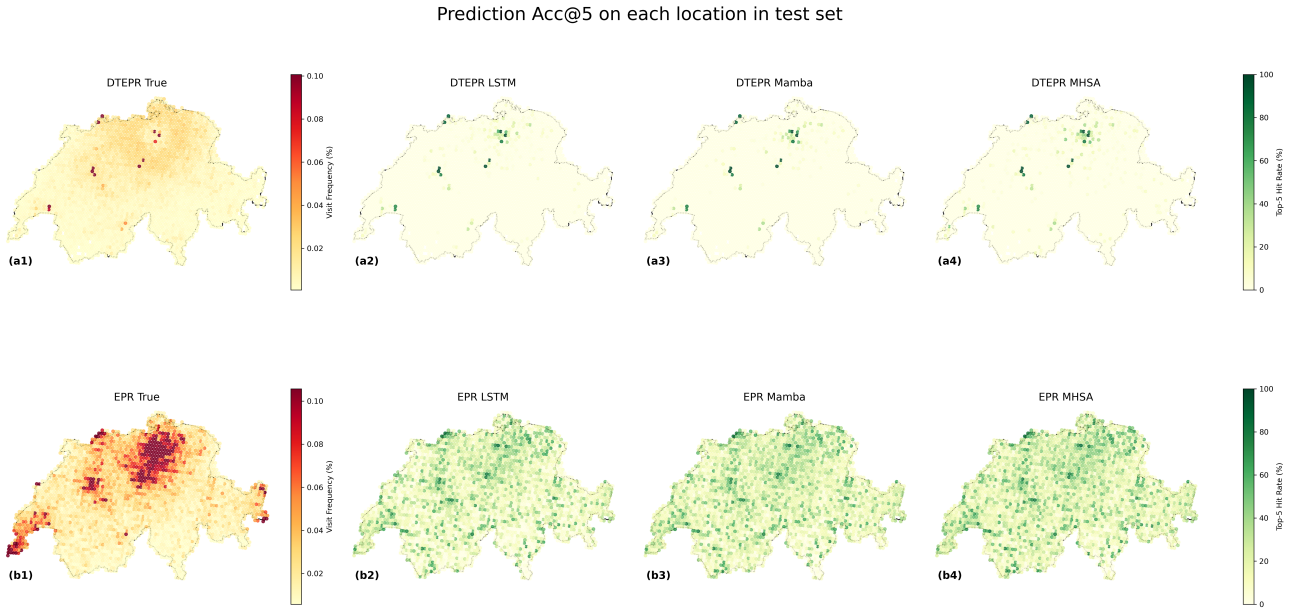


Figure 4.1: Per-location prediction Acc@5 in the test set for both benchmark datasets.

## 4.2 Robustness on Interventional Datasets

To answer the second research question regarding model robustness, this study tests the performance of the trained models on interventional datasets. The results, shown for the DT-EPR (Figure Figure 4.2) and EPR-based interventions (Figure Figure 4.3), reveal that the differences in model architecture and the mobility behavior patterns in the training data both impact the model's robustness when handling domain shifts.

On the interventional datasets generated by the high-regularity DT-EPR mechanism(Figure 4.2), all three models—LSTM, MHSA, and Mamba—exhibited nearly identical robustness profiles. Across most intervention types, the performance lines for the three models were virtually indistinguishable from one another. For instance, as the hard intervention on $p^{\text{new}}$ (Figure 4.2, a1-a4) increased the exploration tendency, all models showed a sharp, linear decline in performance across all metrics, with no model showing a clear advantage. While the performance trends were also largely identical in response to interventions on $\mu_\gamma$ of $P(\gamma)$(Figure 4.2,c1-c4), a subtle difference emerged during the intervention on $\mu_\rho$ of $P(\rho)$(Figure 4.2,b1-b4), where the MHSA models showed a significant performance advantage over Mamba and LSTM at a lower $\mu_\rho$ of $P(\rho)$ value. It is also worth noting that as $\mu$ the value of $P(\rho)$ increases from 0.1 to 0.9, leading to an increase in the exploration tendency, the accuracy of all models does not exhibit a monotonic decrease but instead undergoes a turning point at 0.25. Within the range of 0.1 to 0.25, the accuracy of all models increases. From 0.25 to 0.9, the accuracy decreases in a monotonic manner.

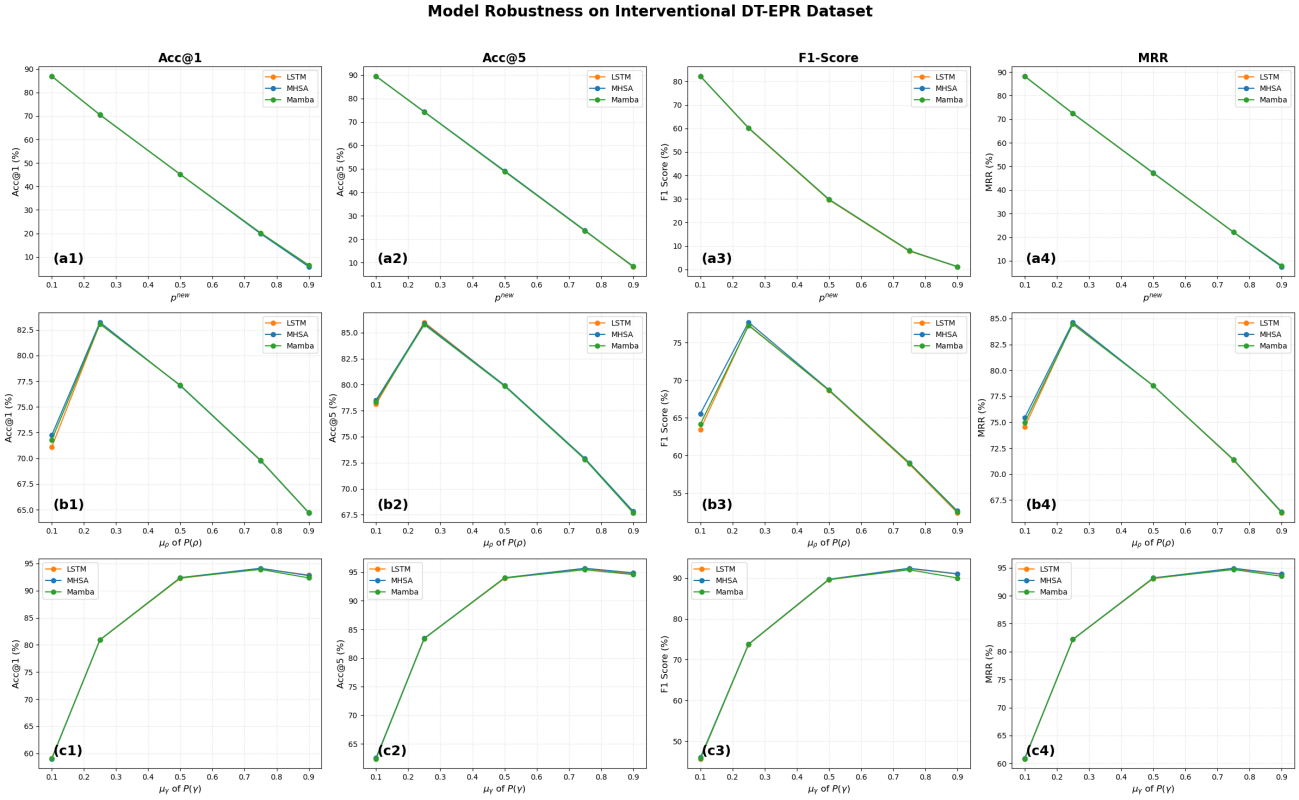


Figure 4.2: Model robustness on the interventional DT-EPR dataset. Rows show interventions on (a) $p^{\text{new}}$, (b) $\mu_\rho$ of $P(\rho)$, and (c) $\mu_\gamma$ of $P(\gamma)$. Columns show performance metrics: Acc@1, Acc@5, F1-Score, and MRR. All three models exhibit nearly identical performance degradation.

In stark contrast, robust performance exhibits significant variation on the high-uncertainty EPR datasets (Figure 4.3). The LSTM model outperforms others to become the most accurate model when $p^{\text{new}}$ is below 0.25, $\mu_\rho$ of $P(\rho)$ is below 0.25, and $\mu_\gamma$ of $P(\gamma)$ exceeds 0.5. Mamba surpasses MHSA when $p^{\text{new}}$ is below 0.25 or above 0.75, and $\mu_\gamma$ of $P(\gamma)$ exceeds 0.5. In all other scenarios, MHSA maintains its advantage. MHSA and Mamba demonstrate comparable trends in Prediction Performance across various interventions, though LSTM exhibits slight variations. Within the specified $\mu_\gamma$ of $P(\gamma)$ range of 0.1 to 0.25, all metrics of MHSA and Mamba demonstrate a decreasing trend, whereas LSTM's Acc@1, Acc@5, and MRR all show an upward trend.

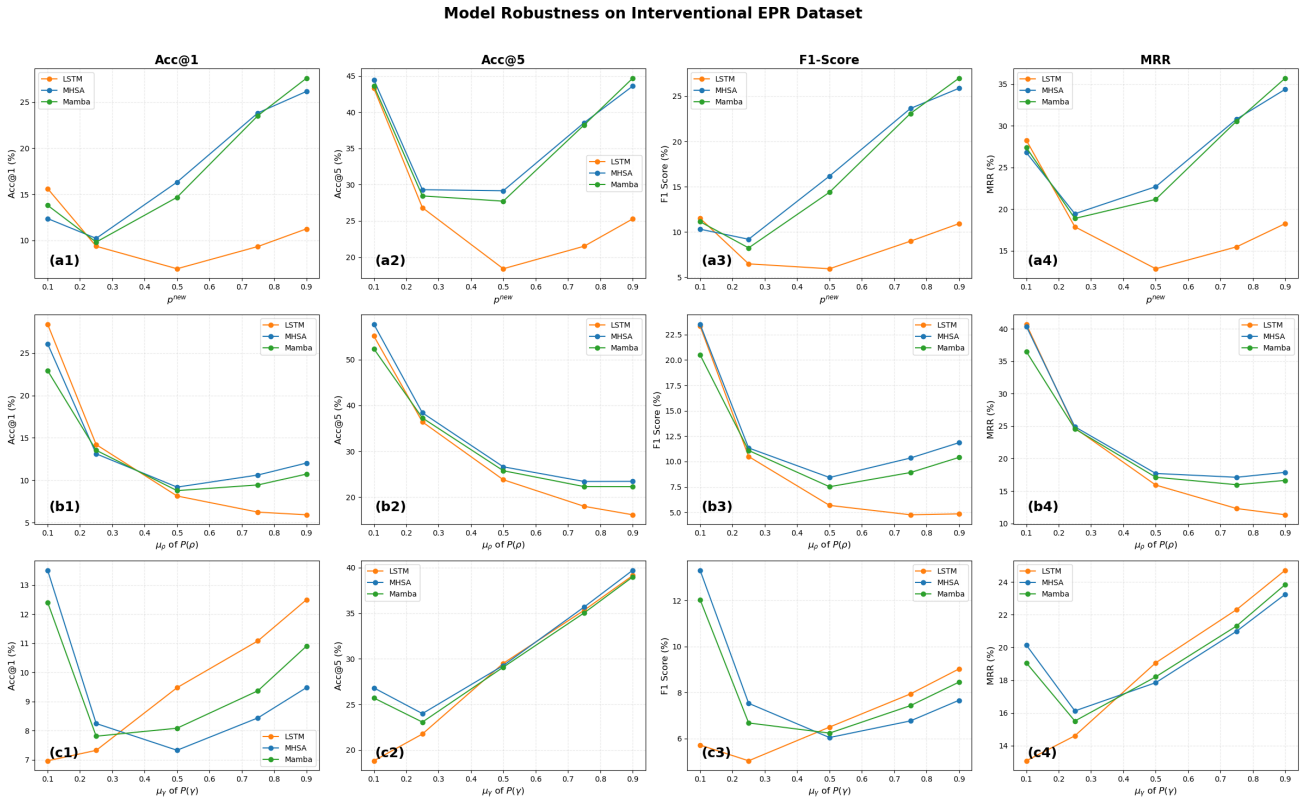


Figure 4.3: Model robustness on the interventional EPR dataset. Rows show interventions on (a) $p^{\text{new}}$, (b) $\mu_\rho$ of $P(\rho)$, and (c) $\mu_\gamma$ of $P(\gamma)$. Columns show performance metrics. Clear differences in robustness are visible, with Mamba and MHSA outperforming LSTM.

Overall, when trained on data generated by a high-regularity mobility data generation mechanism(DT-EPR), the LSTM, MHSA, and Mamba models exhibit nearly identical robustness when applied to test sets with varying exploration tendencies. However, when trained on data generated by a high-uncertainty mobility data generation mechanism(EPR), Mamba and MHSA perform comparably on high-exploration datasets and significantly outperform LSTM. Conversely, as the exploration propensity decreases, LSTM's Prediction Performance gradually approaches that of MHSA and Mamba, ultimately surpassing them. Under nearly all intervention conditions, Mamba demonstrates suboptimal Prediction Performance, lagging only slightly behind the best model. In certain specific scenarios (e.g., when the exploration probability $p^{\text{new}}$ is set above 0.75), this model proves to be the optimal solution.

## 4.3 Computational performance on different sequence lengths

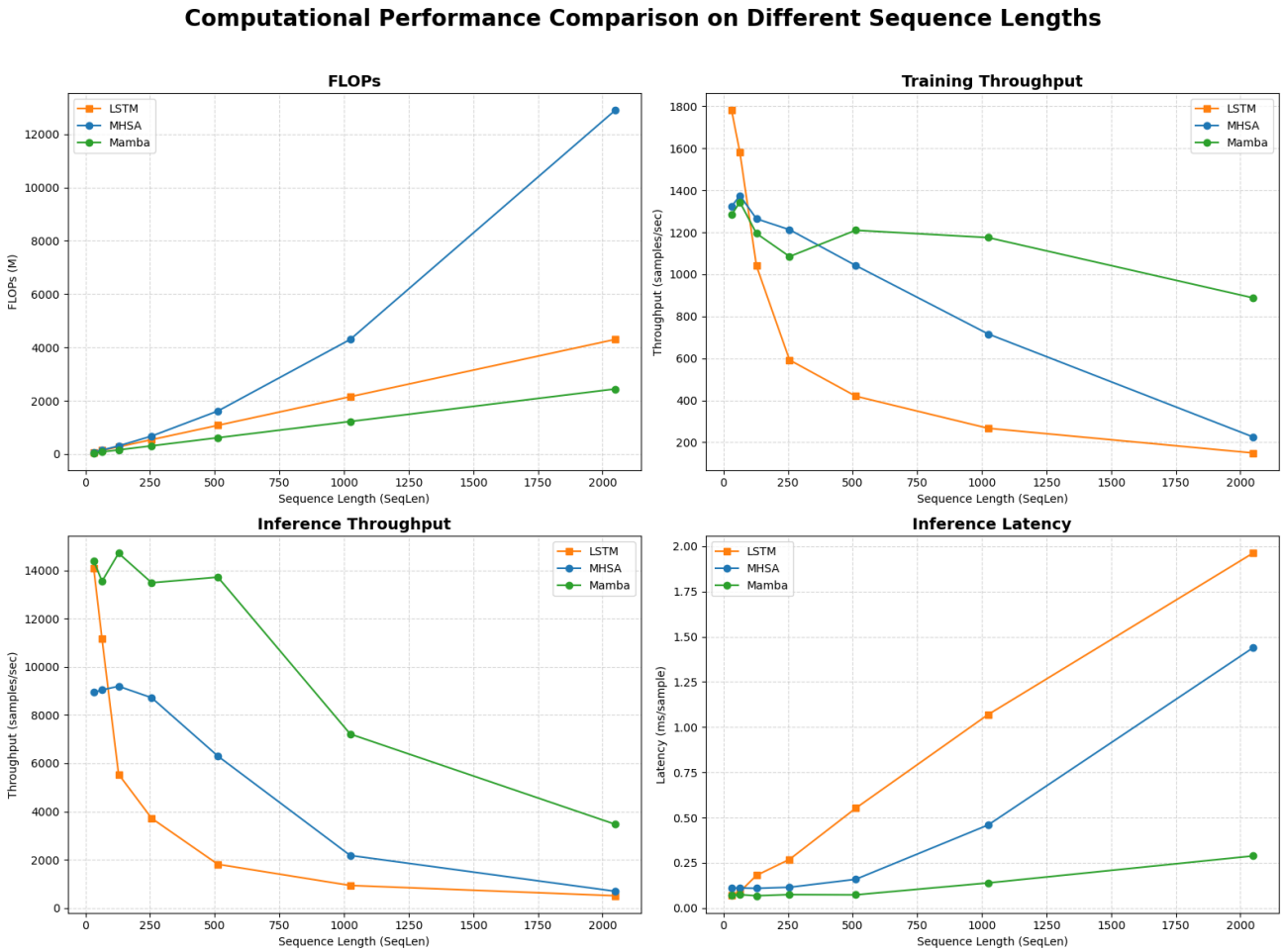


Figure 4.4: Computational performance comparison on different sequence lengths. The four plots show (Top-Left) theoretical cost in FLOPs, (Top-Right) training throughput, (Bottom-Left) inference throughput, and (Bottom-Right) inference latency. Mamba demonstrates superior efficiency and scalability.

The third research question was addressed by comparing the computational performance of the three models under a controlled total number of parameters level(2.5-2.7 million) with varying sequence lengths(Table E.1). The results, presented in Figure 4.4, highlight the distinct scalability and efficiency characteristics inherent to each architecture.

The theoretical computational cost, measured in FLOPs, revealed a clear distinction between the models. The MHSA model exhibited quadratic ($O(N^2)$) complexity, with its FLOPs count increasing exponentially as sequence length grew (Figure 4.4, Top-Left). In contrast, both the LSTM and Mamba models demonstrated linear ($O(N)$) complexity, with the Mamba model requiring the lowest number of FLOPs across all tested sequence lengths.

This theoretical advantage translated directly into practical performance. In terms of both inference throughput and latency(Figure 4.4, Bottom-Left), the Mamba network consistently outperformed

the other two architectures, especially at longer sequence lengths. Therefore, the Mamba network also achieved the lowest inference latency, and its latency increase is also the most gradual as the sequence length increased (Figure 4.4, Bottom-Right). When sequence lengths are very short (<100), LSTM demonstrates a significant advantage in training throughput(Figure 4.4, Top-Right). However, as sequences grow longer, LSTM becomes noticeably slower than MHSA and Mamba. Moreover, Mamba has a distinct advantage over the other two.

In summary, the results show that the Mamba architecture provides superior computational efficiency and scalability compared to both the LSTM and MHSA models.

# 5 Discussion

## 5.1 Prediction Performance

To understand the reasons behind the disparate performance of the models, it is necessary to discuss the fundamental differences in the generation of the benchmark datasets. The high-uncertainty EPR dataset and the high-regularity DT-EPR dataset are generated by different mechanisms. These mechanisms determine the data's underlying patterns, which in turn directly impact the capacity of each model architecture to learn.

The EPR model creates random trajectories with high entropy and few repeating patterns. This is because its preferential return rule is just the probability of returning to a known location '$j$', which depends only on its total visit count ($f_j$) (Song et al., 2010). This allows the model to make long-distance jumps to the unusually visited location. For example, a user might suddenly return to a location they have not visited in a long time, simply because it was visited some times in the past. This behavior is like a person spontaneously deciding to visit a distant, old favorite spot. Such exploratory travel increases the randomness of the trajectories and makes them harder to predict.

In contrast, the DT-EPR model employs a preferential transition mechanism, derived from the IPT model, which is a first-order Markov process (C. Zhao et al., 2021). This completely changes the logic of return behavior. The probability of moving from the current location '$i$' to the next location '$j$' is determined not by the total visits to '$j$', but by the number of times that specific transition from 'i' to 'j' has occurred in the past ($f_{i \to j}$) (Hong et al., 2025). This rule makes it almost impossible for a user to suddenly jump to a location that is unvisited for a long time, as that specific path would be extremely rare in their history. As a result, DT-EPR generates highly structured and regular sequences that tend to repeat fixed, common patterns. This is effective for simulating daily routines, like a home-to-work commute, but it fails to capture occasional, exploratory trips.

This difference in data generation is what causes the models' performance to diverge. On the highly regular DT-EPR dataset, all three models achieved high and similar performance scores(Table 4.1). The spatial accuracy maps confirmed that all architectures successfully identified the true high-frequency locations. Because DT-EPR trajectories are dominated by strong, short-range sequential patterns (like commutes from home to work), the main task for the models was simply to learn these common transitions. LSTM, with its recurrent structure, is naturally suited for this kind of sequential information flow (Greff et al., 2017; Solomon et al., 2021). Although MHSA and Mamba were designed for more complex tasks, they also easily learned these simple patterns. Consequently, the data's regularity obscured the architectural differences between the models, resulting in similar performance.

However, on the high-uncertainty EPR dataset, a clear performance hierarchy emerged. It was demonstrated that MHSA performed marginally better than Mamba, with both demonstrating significantly higher performance than LSTM. To achieve a high prediction performance with this benchmark dataset, it is essential that the model is capable of capturing weak, non-sequential, and long-distance dependencies. The MHSA's self-attention mechanism is a process that enables the system to acquire a global perspective, thereby facilitating direct computation of relationships between any two points in the sequence (Vaswani et al., 2017). This global context attention is key to finding the

chaotic, non-local patterns in the EPR dataset. Mamba also demonstrated a strong performance by utilizing its Selective State Space Model (SSM) core to efficiently capture long-range dependencies in a compressed hidden state (Gu & Dao, 2023). However, this compressed state provides a simplified representation compared to the MHSA's global attention, which may result in the oversight of some of the most subtle patterns. This discrepancy explains the second-place ranking of the latter. Meanwhile, the worst performance exhibited by LSTM highlights its primary limitation: a preference for capturing regular sequential patterns, which renders it ineffective when confronted with irregular sequential data due to its inability to capture the randomness inherent in EPR trajectories adequately. (Al-Selwi et al., 2024; Sherstinsky, 2020).

This comparative analysis reveals that for highly structured, regular movements, the powerful sequential design of LSTM is already sufficient. For chaotic exploratory movements, models employing more flexible architectures, such as MHSA, perform better due to their ability to learn more complex and hidden associations from the data. The Mamba architecture offers an efficient compromise, but in complex data scenarios where the primary goal is achieving maximum accuracy, MHSA's global context analysis remains the optimal choice.

## 5.2 Robustness to Behavioral Domain Shifts

The robustness of prediction models to domain shifts does not depend only on the inherent properties of their architecture. More critically, it also depends on the characteristics of the training data. Experiments show that training based on highly regular data may lead to overfitting of specific patterns, resulting in a significant decline in prediction performance for all models when facing distribution shifts. Conversely, training based on highly uncertain data forces models to learn more universal representations, thereby revealing differences in robustness among various architectures.

After training on the highly regular DT-EPR dataset, all three architectures exhibited nearly identical performance degradation when tested on intervention data(Figure 4.2). As the exploration tendency in the test data increased, the accuracy of all models declined sharply in tandem. Especially when strong interventions were directly applied to the exploration propensity $p^{\text{new}}$, the Acc@1, which was close to 90% when $p^{\text{new}} = 0.1$, dropped sharply to less than 10% when $p^{\text{new}} = 0.9$, this phenomenon stems from the DT-EPR training data being dominated by a few strong, predictable motion patterns. Regardless of the underlying architecture, all models learned to over-reliance on these specific patterns, leading to pattern overfitting. When test data generated via causal intervention deviates from these learned patterns, model prediction performance deteriorates significantly. This exemplifies the classic domain shift problem: models trained on one data distribution often perform poorly when deployed to another, as they have learned spurious correlations rather than true causal structures of motion. This conclusion aligns with findings from Hong and Xin's research(Hong et al., 2025; Xin et al., 2022).

When models are trained on highly uncertain EPR datasets, a starkly different picture emerges (Figure 4.3). In this scenario, differences in model robustness become markedly pronounced. When test data exhibit a high exploratory tendency, MHSA and Mamba typically display superior robustness. When $\mu_\gamma$<0.2, $p^{\text{new}}$>0.25,$\mu_\rho$ >0.75, they even achieve higher accuracy than on the benchmark dataset used for training. This occurs because training on chaotic EPR data forces these models to leverage their core strength: capturing long-range weak dependencies. The mobility characteristics they learn are less constrained by specific behavioral patterns and more exploratory, enabling better

adaptation when tested on other highly exploratory datasets.

Surprisingly, the LSTM model proved to be the most robust when the interventional test data shifted towards highly regular, low-exploration patterns(Figure 4.3 ). When the underlying mobility becomes strongly sequential (e.g., at low values of $p^{\text{new}}$ or high values of $\mu_\gamma$), LSTM's inherent sequential inductive preference, which was a liability on the chaotic training set, becomes a distinct advantage. It is better to recognize and exploit these emergent sequential patterns in the test data. In contrast, MHSA, having learned a more generalized representation from the diverse training data, is less adept at capitalizing on this sudden shift to strong regular sequentiality. Mamba demonstrates comparable accuracy to MHSA when exploration tendencies are high, while achieving prediction performance close to LSTM as exploration tendencies decrease and transition patterns stabilize. Therefore, Mamba can be considered a more all-around option.

## 5.3 Computational Efficiency

Although the models exhibit subtle differences in terms of accuracy and robustness, the Mamba architecture boasts a clear and decisive advantage in terms of computational efficiency and scalability. This superiority is particularly evident with long sequences and is a direct result of its foundation in state space models, which allow for linear-time complexity. This efficiency is arguably its most significant contribution to next-location prediction.

Analysis of the theoretical computational cost, measured in FLOPs, confirms the complexities of each architecture. The FLOPs count of the MHSA model grows quadratically ($O(N^2)$) with sequence length, which is a direct consequence of its self-attention mechanism. This mechanism must compute a pairwise similarity score between all $N$ tokens in the sequence (Vaswani et al., 2017; Wozniak et al., 2023). In contrast, both LSTM and Mamba exhibit linear ($O(N)$) complexity. However, Mamba requires the fewest FLOPs across all tested sequence lengths. The reason is that although the model parameters were controlled at the same level, the LSTM network (2.57M) still had 2% more parameters than Mamba (2.51M) during testing. This gap becomes wider as the FLOPs difference between the two increases with the sequence length.

This theoretical advantage is displayed directly in practical performance. In terms of both training and inference throughput, Mamba consistently outperforms the other architectures as the sequence length increases. It is worth noting that when sequence lengths are less than 100, both inference and training efficiency become highly unstable in MHSA and Mamba networks. This instability may be due to the input size—composed of the training batch size and sequence length—not fully utilizing the GPU, thereby failing to demonstrate the speed advantages offered by parallel computation. While LSTM shows high throughput for very short sequences, its inherently sequential processing—where each step depends on the previous one—creates a computational bottleneck that prevents parallelization across the time dimension, causing its performance to plummet on longer inputs (Solomon et al., 2021; Vaswani et al., 2017). MHSA's ability to process all tokens in parallel enables it to be faster than LSTM for longer sequences. However, its quadratic complexity still constrains its performance. Mamba's architecture, which leverages a hardware-aware parallel scan algorithm, uniquely combines the strengths of recurrent and parallel models. It can be computed recurrently for efficient inference and convolutionally for parallelized training, thereby achieving linear-time complexity without sacrificing parallelism (Dao & Gu, 2024; Gu & Dao, 2023; Qu et al., 2025). This results in the

highest throughput and, consequently, the lowest inference latency, which grows much more slowly with sequence length compared to its counterparts.

This computational advantage enables more practical and sustainable developments in mobility research. The high computational cost of Transformer models often forces researchers to compromise, such as using shorter historical trajectories or downsampling data. Mamba's efficiency eliminates this bottleneck, making it possible to analyze longer sequences—such as weeks or months of data instead of the 7 days used in this study. This facilitates the discovery of long-term seasonal or behavioral patterns—patterns previously difficult to model due to excessive computational complexity. Furthermore, this efficiency makes Mamba an ideal deployment solution for resource-constrained environments, such as on-device applications on smartphones or in-vehicle embedded systems, enabling real-time prediction and recommendation capabilities.(Barbosa et al., 2018; Dao & Gu, 2024).

## 5.4 Limitations and Recommendations

The primary limitation of this study is that it relies solely on synthetic data, as I was unable to use authentic GNSS trajectory datasets as the basis for directly generating synthetic trajectory data. Instead, I used a dataset synthesised by Hong et al. (2025) from an authentic GC dataset as the basis. This means that the study involved two rounds of data synthesis from real-world sources. My synthesized dataset was stacked on top of a DT-EPR synthetic dataset. This also explains why my DT-EPR exhibits fewer diverse movement patterns than Hong's DT-EPR, which was synthesized directly from GNSS data. Mechanistic models can control certain individual mobility characteristics, but they may fail to capture the complexity, noise, and irregularities present in real-world mobility data. (Barbosa et al., 2018; Pappalardo et al., 2015; C. Zhao et al., 2021). Therefore, the performance and robustness of these architectures must be validated on large-scale, real-world GPS datasets.

Secondly, the scope of the causal interventions was limited to the single dimension of "exploration tendency." A more comprehensive robustness evaluation would involve intervening on other causal factors, such as population attractiveness or individual preferences.

Thirdly, the current Mamba model only handles time series and unique location identifiers for each position, without explicitly modeling spatial relationships between locations. Integrating spatial processing mechanisms—such as graph convolutional layers that capture geographic network structures or encoding schemes that reflect spatial relationships (e.g., Google's Plus Code)—can enhance the spatial information in the data. Furthermore, incorporating rich semantic information, such as transportation modes and points of interest (POIs), can significantly expand the features the model can learn, thereby improving its predictive capabilities (Chen et al., 2024; M. Li et al., 2021; J. Zhao et al., 2024).

Finally, the models implemented were standard architectures; performance could potentially be enhanced by exploring more sophisticated designs, such as hybrid models that combine Mamba's efficiency with Transformer's global context or by integrating graph neural networks to explicitly model spatial structure (Z. Li et al., 2022, 2023; B. Wu et al., 2025).

# 6 Conclusion

This study provides a comprehensive comparison of Mamba, MHSA, and LSTM architectures for individual next-location prediction tasks, with a view to evaluating prediction accuracy, robustness to mobility behavior domain shifts, and computational efficiency. The results and findings of the study provide clear insights into the strengths and limitations of each model in terms of individual mobility prediction modeling.

The evaluation of prediction performance reveals that no single model dominates across all scenarios. The MHSA model demonstrates the highest level of accuracy in the context of high-uncertainty datasets, characterized by exploratory behavior. In contrast, the performance of all three models is comparable in the setting of high-regularity datasets, which are defined by the presence of routine patterns. Mamba has been shown to match or approach MHSA performance while significantly outperforming LSTM, particularly in the context of complex, low-regular mobility data.

The findings of robustness testing demonstrate that the model's resilience is contingent on the diversity of the training data. When trained on high-regularity data, all models demonstrate equivalent vulnerability to pattern changes. However, models trained on diverse, high-uncertainty data have been shown to exhibit different robustness profiles. The MHSA and Mamba models demonstrate superior capacity in handling shifts towards exploratory behavior, while the LSTM model exhibits notable resilience to transitions towards regular, sequential patterns.

The computational efficiency of Mamba is a key distinguishing advantage. The architecture demonstrates linear scalability with sequence length, delivering significantly higher training and inference throughput than the quadratically complex MHSA and the sequentially limited LSTM. This efficiency advantage becomes particularly pronounced with long input sequences, making Mamba the most practical choice for real-world deployment.

The findings establish Mamba as a promising architecture for mobility prediction applications. Although it may occasionally exhibit minor discrepancies in accuracy when compared with MHSA, its integration of advanced performance and exceptional computational efficiency makes it the optimal choice for both research advancement and practical implementation. The architecture successfully addresses the fundamental trade-off between model expressiveness and computational tractability.

This work raises several critical research directions. Firstly, hybrid architectures that combine Mamba's efficiency with MHSA's comprehensive context modeling have the potential to achieve both superior accuracy and practical scalability. Models integrating SSM blocks for global context with attention mechanisms for fine-grained patterns represent particularly promising avenues. Secondly, extending Mamba to model spatial relationships through graph-based mechanisms explicitly would result in the creation of more comprehensive spatio-temporal architectures. Thirdly, the validation of Mamba's effectiveness across a range of mobility tasks – including crowd flow prediction, trajectory generation, and travel mode inference – would serve to establish its role as a foundational model for mobility analysis.

The findings of this research demonstrate that efficient sequence modeling architectures, such as Mamba, have the capacity to match or exceed traditional approaches while significantly reducing computational costs. As the volume of mobility datasets continues to increase and real-time applica-

tions become more crucial, such efficiency gains become not merely advantageous but essential for advancing the field of human mobility modeling.

# 7 Bibliography


Al-Selwi, S. M., Hassan, M. F., Abdulkadir, S. J., Muneer, A., Sumiea, E. H., Alqushaibi, A., & Ragab, M. G. (2024). Rnn-lstm: From applications to modeling techniques and beyond—systematic review. *Journal of King Saud University-Computer and Information Sciences*, 102068. https://doi.org/10.1016/j.jksuci.2024.102068

Baldi, P., & Vershynin, R. (2019). The capacity of feedforward neural networks. *Neural networks : the official journal of the International Neural Network Society*, *116*, 288–311. https://doi.org/10.1016/j.neunet.2019.04.009

Barbosa, H., Barthelemy, M., Ghoshal, G., James, C. R., Lenormand, M., Louail, T., Menezes, R., Ramasco, J. J., Simini, F., & Tomasini, M. (2018). Human mobility: Models and applications [Human mobility: Models and applications]. *Physics Reports*, *734*, 1–74. https://doi.org/10.1016/j.physrep.2018.01.001

Chakraborty, S., Kumar, N. M., Jayakumar, A., Dash, S. K., & Elangovan, D. (2021). Selected aspects of sustainable mobility reveals implementable approaches and conceivable actions. *Sustainability*. https://doi.org/10.3390/su132212918

Chekol, A. G., & Fufa, M. S. (2022). A survey on next location prediction techniques, applications, and challenges. *EURASIP Journal on Wireless Communications and Networking*, *2022*(1), 29. https://doi.org/10.1186/s13638-022-02114-6

Chen, Z., Chen, D., Zhang, X., Yuan, Z., Chen, L., & Wang, S. (2024). Spatio-temporal pivotal graph neural networks for traffic flow forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, *38*(8), 8626–8634.

Cina, E., Elbasi, E., Elmazi, G., & AlArnaout, Z. (2025). The role of ai in predictive modelling for sustainable urban development: Challenges and opportunities. *Sustainability*. https://doi.org/10.3390/su17115148

Dao, T., & Gu, A. (2024). Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. https://arxiv.org/abs/2405.21060

Elharrouss, O., Akbari, Y., Almaadeed, N., & Al-Maadeed, S. (2022). Backbones-review: Feature extraction networks for deep learning and deep reinforcement learning approaches. *ArXiv*, *abs/2206.08016*. https://doi.org/10.1016/j.cosrev.2024.100645

Fan, W., Ning, L.-b., Liu, H., Xu, X., Qu, H., Derr, T., An, R., & Li, Q. (2024). A survey of mamba. *ArXiv*, *abs/2408.01129*. https://doi.org/10.48550/arXiv.2408.01129

Feng, J., Li, Y., Zhang, C., Sun, F., Meng, F., Guo, A., & Jin, D. (2018). Deepmove: Predicting human mobility with attentional recurrent networks. *Proceedings of the 2018 world wide web conference*, 1459–1468. https://doi.org/10.1145/3178876.318605

Franco, L., Galasso, F., Giuliari, F., Placidi, L., Hasan, I., & Cristani, M. (2022). Under the hood of transformer networks for trajectory forecasting. *Pattern Recognit.*, *138*, 109372. https://doi.org/10.48550/arXiv.2203.11878

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2017). Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(10), 2222–2232. https://doi.org/10.1109/TNNLS.2016.2582924

Gu, A., & Dao, T. (2023). Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*. https://arxiv.org/abs/2312.00752

Hong, Y., Martin, H., & Raubal, M. (2022). How do you go where? improving next location prediction by learning travel mode information using transformers. *Proceedings of the 30th International*

*Conference on Advances in Geographic Information Systems*, 1–10. https://doi.org/10.1145/3557915.3560996

Hong, Y., Xin, Y., Dirmeier, S., Perez-Cruz, F., & Raubal, M. (2025). A causal intervention framework for synthesizing mobility data and evaluating predictive neural networks. *Transportation Research Interdisciplinary Perspectives*, *31*, 101398. https://doi.org/10.1016/j.trip.2025.101398

İnce, E. (2025). Mapping the path to sustainable urban mobility: A bibliometric analysis of global trends and innovations in transportation research. *Sustainability*. https://doi.org/10.3390/su17041480

Li, M., Tong, P., Li, M., Jin, Z., Huang, J., & Hua, X.-S. (2021). Traffic flow prediction with vehicle trajectories. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*(1), 423–431.

Li, Z., Chen, B., Liu, B., & Zhou, P. (2022). Spade: A hybrid model of state space and attention for long sequence modeling. *arXiv preprint arXiv:2212.08136*.

Li, Z., Chen, B., Liu, B., & Zhou, P. (2023). Spade: State-space augmented transformer for efficient long-sequence modeling. *OpenReview*. https://openreview.net/forum?id=uUIFTjBREk

Martin, H., Becker, H., Bucher, D., Jonietz, D., Raubal, M., & Axhausen, K. W. (2019). Begleitstudie sbb green class-abschlussbericht. *Arbeitsberichte Verkehrs-und Raumplanung*, *1439*. https://doi.org/10.3929/ethz-b-000353337

Pappalardo, L., Simini, F., Rinzivillo, S., Pedreschi, D., Giannotti, F., & Barabási, A.-L. (2015). Returners and explorers dichotomy in human mobility. *Nature communications*, *6*(1), 8166. https://doi.org/10.1038/ncomms9166

Pinkus, A. (1999). Approximation theory of the mlp model in neural networks. *Acta Numerica*, *8*, 143–195. https://doi.org/10.1017/S0962492900002919

Qu, H., Ning, L., An, R., Fan, W., Derr, T., Liu, H., Xu, X., & Li, Q. (2025). A survey of mamba. https://arxiv.org/abs/2408.01129

Schmidhuber, J. (2014). Deep learning in neural networks: An overview. *Neural networks : the official journal of the International Neural Network Society*, *61*, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003

Sherstinsky, A. (2020). Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, *404*, 132306. https://doi.org/10.1016/j.physd.2019.132306

Solomon, A., Livne, A., Katz, G., Shapira, B., & Rokach, L. (2021). Analyzing movement predictability using human attributes and behavioral patterns. *Computers, Environment and Urban Systems*, *87*, 101596. Retrieved November 26, 2024, from https://www.sciencedirect.com/science/article/pii/S019897152100003X

Song, C., Koren, T., Wang, P., & Barabási, A.-L. (2010). Modelling the scaling properties of human mobility. *Nature physics*, *6*(10), 818–823. https://doi.org/10.1038/nphys1760

Tian, S., Li, X., Ji, H., & Zhang, H. (2019). Mobility prediction method to optimize resource allocation in heterogeneous networks. *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*, 1–6. https://doi.org/10.1109/ICCW.2019.8756817

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 30). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Wang, L., Tan, T., Liu, Q., & Wu, S. (2016). Predicting the next location: A recurrent model with spatial and temporal contexts. *Proceedings of the AAAI Conference on Artificial Intelligence, 30(1).*, 194–200. https://doi.org/10.1609/aaai.v30i1.9971

Wozniak, S., Zhao, Q., Peng, B., Koptyra, B., Zhu, R., Du, X., Wang, B., Kazienko, P., Wind, J. S., Anthony, Q. G., Arcadinho, S., Kong, J., He, X., Zhang, Z., Tang, X., Zhu, J., Albalak, A., Lau, H., Biderman, S., ... Zhou, P. (2023). Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*, 14048–14077. https://doi.org/10.18653/v1/2023.findings-emnlp.936

Wu, B., Shi, J., Wu, Y., Tang, N., & Luo, Y. (2025). Transxssm: A hybrid transformer state space model with unified rotary position embedding. *arXiv preprint arXiv:2506.09507*.

Wu, R., Luo, G., Shao, J., Tian, L., & Peng, C. (2018). Location prediction on trajectory data: A review. *Big Data Min. Anal.*, *1*, 108–127. https://doi.org/10.26599/BDMA.2018.9020010

Xin, Y., Tagasovska, N., Perez-Cruz, F., & Raubal, M. (2022). Vision paper: Causal inference for interpretable and robust machine learning in mobility analysis. *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*, 1–4. https://doi.org/10.1145/3557915.3561473

Zhao, C., Zeng, A., & Yeung, C. H. (2021). Characteristics of human mobility patterns revealed by high-frequency cell-phone position data. *EPJ Data Science*, *10*(1), 5. https://doi.org/10.1140/epjds/s13688-021-00261-2

Zhao, J., Li, Z., Liu, P., & Zhang, M. (2024). Spatial-temporal deep learning model based on similarity principle for dock shared bicycles ridership prediction. *Journal of Transport and Land Use*, *17*(1), 115–142.

# Appendices

## A Relevant code and files

All relevant code and process files for this article will be shared at the following link:

ChanglingWang609/Msc_Thesis_GRS-next-location-prediction.git

## B Hyperparameter Grid Search

Table B.1: Hyper-parameter search for next location prediction networks.

| Hyper-parameter | | Search range | Optimal value |
|---|---|---|---|
| LSTM | Embedding dim. | {32, 64, 128} | 128 |
| | Hidden dim. | {64, 128, 256, 512} | 256 |
| MHSA-based | Heads | {2, 4, 8} | 8 |
| | Layers | {2, 4, 6} | 2 |
| | Embedding dim. | {32, 64, 128} | 128 |
| | Feed-forward dim. | {64, 128, 256, 512} | 512 |
| Mamba-based | Layers | {2, 4, 6} | 2 |
| | Embedding dim. | {32, 64, 128} | 128 |
| | State dim. | {16, 32, 64} | 32 |
| | Conv. dim. | {2, 4, 6} | 4 |

# C Learning curves

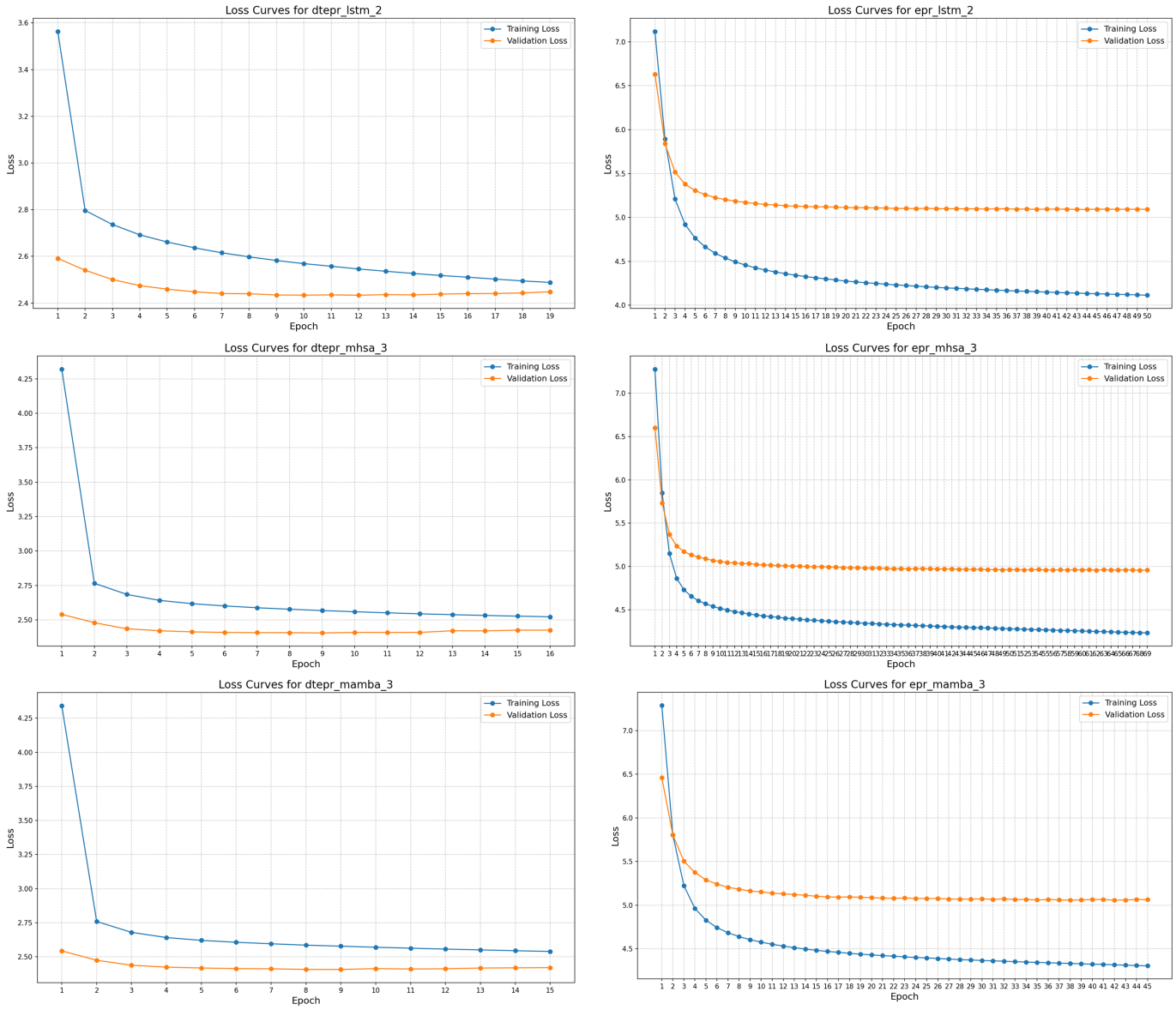


Figure C.1: Learning Curves

# D Robustness test result

Table D.1: Performance on the Intervened DT-EPR dataset.

| Model | Intervention | value | Acc@1 | Acc@3 | Acc@5 | Acc@10 | F1 | MRR |
|---|---|---|---|---|---|---|---|---|
| LSTM | $p^{\text{new}}$ | 0.10 | **86.941** | 89.156 | 89.485 | 89.827 | 82.214 | 88.192 |
| | | 0.25 | 70.449 | 73.785 | 74.310 | 74.947 | 60.063 | 72.392 |
| | | 0.50 | **45.149** | 48.325 | 48.906 | 49.794 | 29.610 | 47.161 |
| | | 0.75 | 20.201 | 22.978 | 23.574 | 24.448 | 7.802 | 22.094 |
| | | 0.90 | 6.215 | 7.739 | 8.247 | 9.093 | 1.131 | 7.522 |
| | $\mu_\rho$ of $P(\rho)$ | 0.10 | 71.088 | 76.393 | 78.139 | 80.690 | 63.472 | 74.529 |
| | | 0.25 | 83.082 | **85.497** | **86.001** | **86.780** | 77.363 | 84.564 |
| | | 0.50 | 77.094 | 79.522 | 79.907 | 80.365 | 68.634 | 78.522 |
| | | 0.75 | 69.778 | 72.419 | 72.830 | 73.363 | 58.876 | 71.347 |
| | | 0.90 | 64.699 | 67.301 | 67.734 | 68.288 | 52.383 | 66.270 |
| | $\mu_\gamma$ of $P(\gamma)$ | 0.10 | **59.071** | **61.978** | 62.477 | 63.108 | 45.675 | 60.835 |
| | | 0.25 | 80.889 | 83.117 | 83.427 | 83.808 | 73.722 | 82.175 |
| | | 0.50 | 92.258 | 93.653 | 93.920 | 94.214 | 89.581 | 93.074 |
| | | 0.75 | 94.027 | 95.290 | 95.625 | 96.019 | 92.319 | 94.797 |
| | | 0.90 | **92.871** | 94.301 | 94.741 | 95.518 | 91.010 | 93.837 |
| MHSA | $p^{\text{new}}$ | 0.10 | 86.937 | **89.188** | **89.499** | **89.852** | **82.224** | **88.202** |
| | | 0.25 | 70.519 | **73.811** | **74.356** | **75.016** | **60.246** | **72.463** |
| | | 0.50 | 45.127 | **48.415** | **49.176** | **50.241** | **29.892** | **47.306** |
| | | 0.75 | 19.874 | 22.919 | **23.769** | **25.147** | **8.005** | 22.130 |
| | | 0.90 | 5.779 | 7.527 | 8.334 | **9.795** | **1.209** | 7.484 |
| | $\mu_\rho$ of $P(\rho)$ | 0.10 | **72.238** | **77.129** | **78.514** | **80.920** | **65.555** | **75.431** |
| | | 0.25 | **83.240** | 85.433 | 85.891 | 86.750 | **77.717** | **84.643** |
| | | 0.50 | 77.065 | **79.541** | **79.939** | **80.410** | **68.736** | **78.525** |
| | | 0.75 | 69.784 | **72.452** | **72.914** | **73.506** | **59.043** | **71.403** |
| | | 0.90 | 64.698 | **67.334** | **67.838** | **68.495** | **52.617** | **66.339** |
| | $\mu_\gamma$ of $P(\gamma)$ | 0.10 | 59.010 | 61.966 | **62.554** | **63.345** | **45.981** | **60.874** |
| | | 0.25 | 80.923 | **83.131** | **83.449** | **83.854** | **73.814** | **82.210** |
| | | 0.50 | **92.383** | **93.760** | **94.026** | **94.311** | **89.715** | **93.185** |
| | | 0.75 | **94.117** | **95.400** | **95.691** | **96.196** | **92.422** | **94.921** |
| | | 0.90 | 92.758 | **94.469** | **94.886** | **95.664** | **91.061** | **93.864** |
| Mamba | $p^{\text{new}}$ | 0.10 | 86.881 | 89.115 | 89.463 | 89.821 | 82.222 | 88.151 |
| | | 0.25 | **70.564** | 73.746 | 74.253 | 74.858 | 60.243 | 72.445 |
| | | 0.50 | 45.126 | 48.322 | 48.892 | 49.775 | 29.801 | 47.191 |
| | | 0.75 | **20.232** | **23.016** | 23.584 | 24.530 | 7.967 | **22.213** |
| | | 0.90 | **6.496** | **8.055** | **8.534** | 9.429 | 1.160 | **7.896** |
| | $\mu_\rho$ of $P(\rho)$ | 0.10 | 71.792 | 76.810 | 78.345 | 80.710 | 64.169 | 74.974 |
| | | 0.25 | 83.072 | 85.317 | 85.768 | 86.393 | 77.283 | 84.451 |
| | | 0.50 | **77.110** | 79.486 | 79.866 | 80.303 | 68.693 | 78.521 |
| | | 0.75 | **69.819** | 72.427 | 72.824 | 73.353 | 58.984 | 71.383 |
| | | 0.90 | **64.719** | 67.277 | 67.686 | 68.241 | 52.499 | 66.287 |
| | $\mu_\gamma$ of $P(\gamma)$ | 0.10 | 59.050 | 61.932 | 62.406 | 63.056 | 45.737 | 60.835 |
| | | 0.25 | **80.927** | 83.085 | 83.396 | 83.775 | 73.775 | 82.188 |
| | | 0.50 | 92.351 | 93.716 | 94.000 | 94.272 | 89.652 | 93.144 |
| | | 0.75 | 93.902 | 95.155 | 95.429 | 95.957 | 92.042 | 94.686 |
| | | 0.90 | 92.335 | 94.140 | 94.596 | 95.331 | 90.026 | 93.484 |

Table D.2: Performance on the Intervened EPR dataset.

| Model | Intervention | value | Acc@1 | Acc@3 | Acc@5 | Acc@10 | F1 | MRR |
|---|---|---|---|---|---|---|---|---|
| LSTM | $p^{\text{new}}$ | 0.10 | **15.620** | **35.344** | 43.313 | 52.188 | **11.511** | **28.273** |
| | | 0.25 | 9.363 | 20.680 | 26.834 | 34.963 | 6.480 | 17.873 |
| | | 0.50 | 6.921 | 14.168 | 18.398 | 24.553 | 5.938 | 12.829 |
| | | 0.75 | 9.337 | 17.347 | 21.510 | 27.404 | 8.984 | 15.465 |
| | | 0.90 | 11.250 | 20.445 | 25.280 | 32.003 | 10.921 | 18.255 |
| | $\mu_\rho$ of $P(\rho)$ | 0.10 | **28.398** | 48.624 | 55.120 | 62.340 | 23.346 | **40.697** |
| | | 0.25 | **14.207** | 29.514 | 36.400 | 44.923 | 10.498 | 24.688 |
| | | 0.50 | 8.143 | 18.236 | 23.788 | 31.550 | 5.681 | 15.915 |
| | | 0.75 | 6.240 | 13.638 | 17.990 | 24.404 | 4.756 | 12.301 |
| | | 0.90 | 5.929 | 12.414 | 16.150 | 21.924 | 4.846 | 11.337 |
| | $\mu_\gamma$ of $P(\gamma)$ | 0.10 | 6.969 | 14.486 | 18.819 | 25.195 | 5.717 | 13.081 |
| | | 0.25 | 7.326 | 16.571 | 21.758 | 29.213 | 5.033 | 14.606 |
| | | 0.50 | **9.476** | **22.524** | **29.509** | 38.504 | **6.497** | **19.064** |
| | | 0.75 | **11.081** | **27.035** | 35.336 | 45.368 | **7.937** | **22.313** |
| | | 0.90 | **12.492** | **30.504** | 39.137 | 49.293 | **9.023** | **24.714** |
| MHSA | $p^{\text{new}}$ | 0.10 | 12.366 | 34.721 | **44.485** | **55.291** | 10.310 | 26.795 |
| | | 0.25 | **10.215** | **22.638** | **29.313** | **38.253** | **9.198** | **19.437** |
| | | 0.50 | **16.323** | **25.246** | **29.164** | **34.449** | **16.180** | **22.695** |
| | | 0.75 | **23.829** | **34.624** | **38.559** | **43.205** | **23.641** | **30.821** |
| | | 0.90 | 26.158 | 38.878 | 43.610 | 49.376 | 25.875 | 34.395 |
| | $\mu_\rho$ of $P(\rho)$ | 0.10 | 26.087 | **49.595** | **57.699** | **65.897** | **23.527** | 40.379 |
| | | 0.25 | 13.119 | **30.416** | **38.382** | **48.027** | **11.337** | **24.898** |
| | | 0.50 | **9.190** | **20.451** | **26.607** | **34.952** | **8.428** | **17.700** |
| | | 0.75 | **10.617** | **19.000** | **23.409** | **29.747** | **10.347** | **17.115** |
| | | 0.90 | **12.030** | **19.710** | **23.438** | **28.824** | **11.855** | **17.875** |
| | $\mu_\gamma$ of $P(\gamma)$ | 0.10 | **13.502** | **22.429** | **26.818** | **32.890** | **13.311** | **20.150** |
| | | 0.25 | **8.247** | **18.263** | **23.995** | **32.099** | **7.530** | **16.126** |
| | | 0.50 | 7.330 | 21.182 | 29.272 | **40.131** | 6.043 | 17.851 |
| | | 0.75 | 8.440 | 25.886 | **35.690** | **47.547** | 6.765 | 20.992 |
| | | 0.90 | 9.478 | 29.162 | **39.721** | **52.186** | 7.660 | 23.247 |
| Mamba | $p^{\text{new}}$ | 0.10 | 13.804 | 34.764 | 43.634 | 53.761 | 11.136 | 27.390 |
| | | 0.25 | 9.815 | 21.915 | 28.453 | 37.220 | 8.244 | 18.875 |
| | | 0.50 | 14.674 | 23.593 | 27.729 | 33.364 | 14.392 | 21.171 |
| | | 0.75 | 23.533 | 34.273 | 38.254 | **43.205** | 23.089 | 30.575 |
| | | 0.90 | **27.589** | **40.102** | **44.698** | **50.403** | **26.988** | **35.730** |
| | $\mu_\rho$ of $P(\rho)$ | 0.10 | 22.927 | 44.930 | 52.342 | 60.907 | 20.510 | 36.487 |
| | | 0.25 | 13.534 | 29.485 | 37.262 | 46.675 | 11.091 | 24.584 |
| | | 0.50 | 8.777 | 19.727 | 25.757 | 34.025 | 7.519 | 17.124 |
| | | 0.75 | 9.448 | 17.721 | 22.297 | 28.811 | 8.915 | 15.970 |
| | | 0.90 | 10.725 | 18.413 | 22.283 | 27.854 | 10.406 | 16.633 |
| | $\mu_\gamma$ of $P(\gamma)$ | 0.10 | 12.402 | 21.250 | 25.708 | 31.938 | 12.034 | 19.062 |
| | | 0.25 | 7.812 | 17.542 | 23.064 | 31.075 | 6.677 | 15.509 |
| | | 0.50 | 8.086 | 21.379 | 29.089 | 39.383 | 6.235 | 18.208 |
| | | 0.75 | 9.370 | 25.797 | 35.042 | 46.312 | 7.428 | 21.314 |
| | | 0.90 | 10.900 | 29.313 | 38.986 | 50.700 | 8.450 | 23.839 |

# E Computational Performance Comparison

Table E.1: Computational Performance Comparison

| SeqLen | Model | Parameters (M) | FLOPs (M) | Inference | | Training |
|---|---|---|---|---|---|---|
| | | | | Throughput (samples/sec) | Latency (ms/batch) | Throughput (samples/sec) |
| 32 | LSTM | 2.57 | 68.61 | 14072.1 | 1.137 | 1780.39 |
| | MHSA | 2.64 | 70.73 | 8939.05 | 1.7899 | 1323.16 |
| | Mamba | 2.51 | 39.58 | 14383.3 | 1.1124 | 1283.8 |
| 64 | LSTM | 2.57 | 135.72 | 11149.8 | 1.435 | 1580.17 |
| | MHSA | 2.64 | 144.14 | 9040.06 | 1.7699 | 1375.47 |
| | Mamba | 2.51 | 77.66 | 13545.5 | 1.1812 | 1343.94 |
| 128 | LSTM | 2.57 | 269.94 | 5544.77 | 2.8856 | 1040.43 |
| | MHSA | 2.64 | 303.56 | 9191.7 | 1.7407 | 1264.49 |
| | Mamba | 2.51 | 153.81 | 14705.9 | 1.088 | 1194.65 |
| 256 | LSTM | 2.57 | 538.37 | 3717.39 | 4.3041 | 591.898 |
| | MHSA | 2.64 | 672.72 | 8716.5 | 1.8356 | 1212.6 |
| | Mamba | 2.51 | 306.11 | 13482.8 | 1.1867 | 1084.8 |
| 512 | LSTM | 2.57 | 1075.24 | 1812.97 | 8.8253 | 419.656 |
| | MHSA | 2.64 | 1612.38 | 6298.47 | 2.5403 | 1042.01 |
| | Mamba | 2.51 | 610.73 | 13715.1 | 1.1666 | 1209.81 |
| 1024 | LSTM | 2.57 | 2148.99 | 935.251 | 17.1077 | 266.749 |
| | MHSA | 2.64 | 4296.99 | 2179.57 | 7.3409 | 716.082 |
| | Mamba | 2.51 | 1219.95 | 7212.41 | 2.2184 | 1175.1 |
| 2048 | LSTM | 2.57 | 4296.47 | 509.957 | 31.3752 | 149.193 |
| | MHSA | 2.64 | 12887.45 | 695.604 | 23.0016 | 225.892 |
| | Mamba | 2.51 | 2438.39 | 3475.16 | 4.6041 | 888.139 |

# F Use of generative AI statement

Throughout my thesis research and writing process, AI primarily served as a coding assistant and writing coach. During code development, Gemini 2.5 Pro was utilized for debugging and generating non-core code components, such as simple file imports, data extraction, and exploratory data analysis. However, I thoroughly double-checked to ensure that I avoided potential logical errors in data processing (e.g., code that runs but contains false logic). The core code of computational logic and deep learning architectures was manually written to ensure perfect alignment with my methodological approach. AI was only employed for debugging. For writing tasks, DeepL assisted with translations, while Grammarly handled sentence refinement, grammar checks, and spelling verification.