The Story of Mobility: Combining State Space Models and Transformers for Multi-Step Trajectory Prediction

Jonas Gunkel*
jonas.gunkel@dlr.de
Institute for the Protection of Terrestrial Infrastructures, German Aerospace Center
Sankt Augustin, Germany

Andrea Tundis
andrea.tundis@dlr.de
Institute for the Protection of Terrestrial Infrastructures, German Aerospace Center
Sankt Augustin, Germany

Max Mühlhäuser
max@tk.tu-darmstadt.de
Technical University of Darmstadt
Darmstadt, Germany

Abstract

Machine learning models for predicting human mobility often require large datasets for training, which are not always available. As a result, methods capable of learning from limited data are essential. The Human Mobility Challenge 2024 was designed to evaluate the effectiveness of various approaches in such constrained scenarios. In this paper, we present a deep learning model that integrates state space models with transformers for multi-city trajectory prediction. Specifically, the model employs the state space model Mamba as an encoder to process long-range trajectories, while a transformer decoder predicts future locations by querying past trajectories with future timestamps. Our results demonstrate the model's effectiveness and suggest strong generalizability across cities. The approach ranked in the top 10 of the challenge, highlighting its competitiveness in limited-data settings.

CCS Concepts

• **Computing methodologies** → **Machine learning**; • **Information systems** → **Spatial-temporal systems**; • **Applied computing** → **Forecasting**.

Keywords

Human Mobility, Deep Learning, Transformer, State Space Models

ACM Reference Format:

Jonas Gunkel, Andrea Tundis, and Max Mühlhäuser. 2024. The Story of Mobility: Combining State Space Models and Transformers for Multi-Step Trajectory Prediction. In *2nd ACM SIGSPATIAL International Workshop on the Human Mobility Prediction Challenge (HuMob'24), October 29-November 1, 2024, Atlanta, GA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3681771.3699912

1 Introduction

Predicting the future locations of citizens in urban areas is a critical task with various applications, including traffic management and disaster response. In recent years, machine learning models have been used excessively and with great success to reproduce the individual mobility patterns [7]. However, such models require extensive training data to generate realistic future trajectories. This massive amount of data may not exist for some regions of interest, rendering training a corresponding model difficult. The Human Mobility Challenge 2024, held at SIGSPATIAL 2024, is designed to test such a setting. Given mobility data of individuals from four different cities, each city with a different amount of available data, the task is to predict the trajectories of a defined set of individuals from different cities over 15 days.

Predicting future trajectories has been approached with a variety of machine learning approaches [7]. As trajectories form a sequence of locations, similar to a sentence being a sequence of words, several works adopted methods from natural language processing to predict an individual's future locations [4, 11, 13]. Recent methods often base on transformer models, which have been proven powerful for modeling sequential dependencies [8, 10]. Recently, advances in state space models (e.g., Mamba [3]) have shown success comparably to transformers. The recurrent structure with efficient pre-computation of states alleviates the quadratic-time complexity of attention without reducing performance. Consequently, state space models have been proposed for various sequence-to-sequence tasks [5, 6].

Following the success of language models for trajectory prediction, we propose to combine recent advances in state space models with earlier approaches to trajectory prediction using transformers. Inspired by [14], we employ a Mamba encoder to encode a past trajectory as context. Subsequently, a transformer decoder queries the computed encodings to infer the locations at future time stamps. Our analysis confirms that our model not only competes with the performance of pure transformer-based models, but also significantly reduces the computational effort, making it an efficient solution for trajectory prediction.

1.1 Scenario Description

The Human Mobility Challenge 2024 focuses explicitly on multi-city prediction. That is, predicting the mobility for a specific city based on data from potentially different cities. The challenge data contains trajectories from individuals in four cities, for each in different quantity: The amount of provided trajectories ranges between 100k (city A), 25k (city B), 20k (city C), and 6k (city D). The task consists of predicting the last 15 days of 3000 individuals from city B, C, and D, respectively, as depicted in fig. 1.

*Corresponding Author

HuMob'24, October 29-November 1, 2024, Atlanta, GA, USA

ACM ISBN 979-8-4007-1150-3/24/10
https://doi.org/10.1145/3681771.3699912

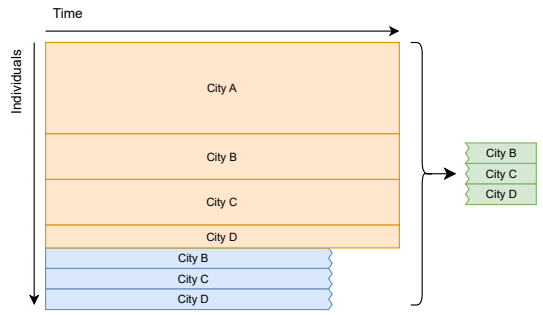


Figure 1: Representation of the task. The goal is to predict the trajectories of 3000 individuals from cities B, C, and D for the last 15 days (marked in green). As training data and prediction context, the trajectories of these individuals for the first 60 days of the period (marked in blue), as well as trajectories for the entire period of a large set of individuals from City A, B, C, and D (marked in yellow) are provided.

2 Data

The data set provided for the Human Mobility Challenge 2024 consists of reports of the individuals' locations and their corresponding time stamps over 75 days, as described in [12]. The locations are represented as discrete coordinates of cells on a grid, into which each city is divided. The grids have a cell size of $500m \times 500m$, partitioning the cities' areas into a total of 200×200 cells, respectively. The location reports for a single individual form a trajectory ordered by their temporal information. Each trajectory consists of tuples (l_x, l_y, d, t) representing the x and y coordinates (l_x and l_y), and the corresponding time stamp given by the day d and half-hour interval t.

The resulting trajectories are neither complete nor exclusively report location changes. This means, on the one hand, that there may be periods without reported locations for a given individual, leading to trajectories of varying lengths. Moreover, most trajectories report locations for only a fraction of time stamps, as seen in fig. 2(a). On the other hand, the trajectories contain stationary sub-trajectories that consist only of one location, as illustrated in fig. 2(b).

For the task at hand, we define a trajectory as a sequence of tuples $\tau = (x, y, d, t, \omega, \delta)$. Here, the elements $x, y \in \{1, \ldots, 200\}$ specify the x- and y-coordinates on the grid, $d \in \{0, \ldots, 6\}$ represents the day of the week, and $t \in \{0, \ldots, 47\}$ constitutes the time of the day as the corresponding half-hour interval. Considering the periodicity of the mobility volumes, as depicted in fig. 3, we introduce a binary variable ω, describing whether it is a working day or a weekend day. Moreover, we follow [11] and include δ, representing the difference in time to the previously reported location of the trajectory. Furthermore, we denote the elements of a trajectory with masked locations by $\hat{\tau}$ and a tuple of coordinates by $l = (x, y)$.

3 Method

Our model is designed to perform a multi-step prediction for future trajectory locations. By predicting several locations in parallel, the model must learn more complex dependencies within the provided trajectories. Thereby, we aimed to reduce the risk of predicting the last given location as the following location. We deemed this property critical, as the provided data includes many trajectories that contain repeating sub-sequences over a longer period (fig. 2(b)).

For inferring the future trajectory locations, the model is fed the entire past trajectory as context. This context trajectory is processed by an encoder module that adopts the task of computing representations of the respective sequence's elements. The encoder's output is the basis for a subsequent decoder to compute representations for the future locations. A final linear layer generates the location probabilities for each step of the future trajectory. An overview of the model's architecture can be seen in fig. 4. In the following, the different design choices of our method are portrayed in detail.

3.1 Embedding Layer

The model's embedding layer embeds the input trajectory in the initial phase. Following previous work [10, 11], each component of a trajectory's element $(x, y, d, t, \omega, \delta)$ is embedded individually. The embeddings are then concatenated, serving as input to the encoder and decoder.

3.2 Encoder

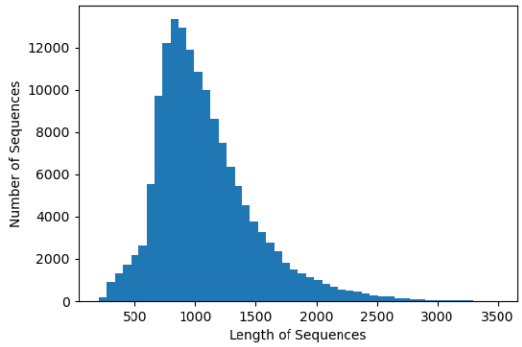
The encoder was chosen to build on the success of past work that adopted methods from NLP, specifically transformer models [10, 11]. However, transformers, having quadratic complexity in sequence length, may lead to slow training and inference times. We employ Mamba [3], a recently proposed state space model, as the encoder to reduce these times. In comparison to a transformer encoder, Mamba is a strictly causal model. As such, the representation of a sequence element at position i only depends on the sequence's previous elements. By stacking multiple Mamba layers equipped with residual connections in between, the encoder is expected to learn trajectory features similarly to a transformer encoder while processing long inputs significantly faster.

During training, the encoder's output is subject to a next-location prediction loss. The idea behind this loss is to force the model to learn representations of the trajectory's elements that focus on the most relevant features for predicting the future trajectory.

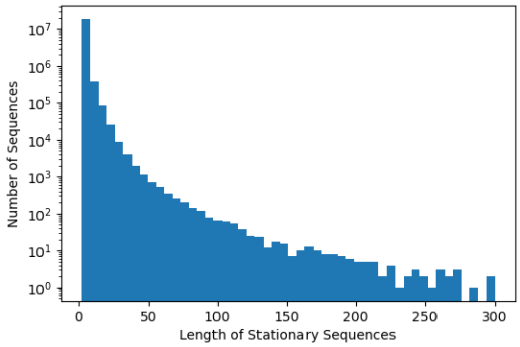
3.3 Decoder

The sequence of processed tokens, as obtained from the encoder, is the basis for the decoder's inference of the future trajectory. Similarly to query-based methods in object detection [1] or next action anticipation [2, 14], the decoder is trained to create candidates for future locations by querying the encoder's output. Via cross-attention, the decoder combines the time stamps of future locations (representing the query vectors) with the processed context (representing the key and value vectors).

By stacking several transformer decoders, each featuring a self-attention layer, the decoder can update the location candidate for each time stamp based on the whole sequence of future location candidates.



(a) Lengths of the trajectories.



(b) Lengths of stationary sub-trajectories on logarithmic scale.

Figure 2: Histograms of trajectory characteristics. fig. 2(a) portrays the sparsity of trajectories: over 90% of trajectories have a length that is less than half of the maximum possible length. The histogram in fig. 2(b) shows the distribution of stationary sub-trajectories. These stationary sub-trajectories represent a significant share of the whole data. In average, more than every fifth location is identical to its preceding location.

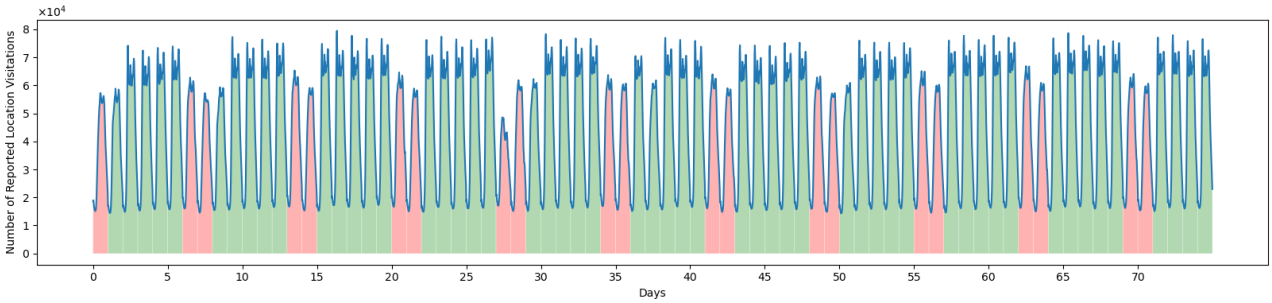


Figure 3: Number of location visitations in all cities over the entire period. Clearly, the data exhibits strong periodicity. More specifically, the weekends as days of decreased mobility volumes (marked as red) can be reasonably distinguished from the workdays (marked as green).

The first decoder layer plays a crucial role, computing the initial representations for the future locations and laying the foundation for the subsequent layers. Starting from the second decoder layer, the model features residual connections between them. Consequently, each consecutive layer subsequently updates the initially computed representations.

3.4 Prediction Head

Two linear layers compute the final output of the model, each generating the probabilities for a single coordinate. Both layers project the decoder's output, representing the future locations, to a vector of dimension 200, representing the possible x and y coordinates.

3.5 Positional Encodings

Unlike transformers, Mamba, which can be interpreted as a recurrent model, does not require positional encodings representing the input order. However, the transformer decoder critically depends on such additional information in order to comprehend the relative distances between the sequence's elements. Instead of using standard sinusoidal positional encodings, which encode the absolute position of an element in a sequence, we compute the positional encodings based on the time stamp of the reported location. Consequently, the positional encodings represent not only a strict order of elements, but also their relative time difference.

3.6 Training Procedure

For training the model, we combine all available trajectory data from the four cities as training data. Each individual's trajectory is split into a context sequence and a prediction sequence. The context sequence serves as input to the encoder, and the time stamps of the prediction sequence as input to the decoder. The training objective is to infer the locations of the prediction sequence.

To increase the diversity of training samples, the trajectories are randomly split into context and prediction sequence: for each batch, $3\% - 10\%$ of the trajectories serve as prediction sequence. Moreover, depending on their length, the sequences are split randomly for each epoch into sub-sequences, further augmenting the training data.

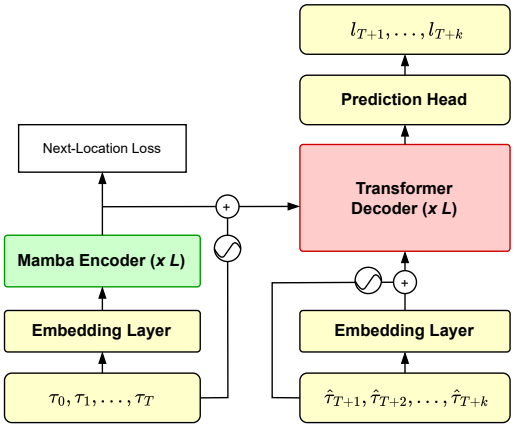


Figure 4: Architecture of the used model. The encoder is trained to compute representations for all previous tokens, respectively, via a next-location loss. The decoder takes future time stamps and infers the corresponding locations via cross-attention, which uses the encoder's output to compute the key and value.

3.7 Inference Procedure

Multiple multi-step predictions are performed autoregressively to predict the future steps of a given individual over multiple days. After predicting k future steps, these steps are appended to the previous context, representing the context for the next prediction. Consequently, the number of such multi-step predictions is intricately dependent on the length of the total prediction window and the number of steps for each prediction.

For the Human Mobility Challenge 2024, we used 5% of the context length as the prediction window length for each prediction step. This choice resulted in an average of 5.081 steps to be performed, being similar across the different cities, as depicted in table 1.

Moreover, we adopted the strategy for suppressing long sequences of identical locations proposed in [11]. The authors multiplied the probability of consecutively predicting the same location by a decay factor to increase the heterogeneity in sequences. For this work, we set the decay factor to 0.8.

Table 1: Average number of inference steps to be performed for the last 3000 individuals for each city.

City	Inference Steps
B	5
C	4.969
D	5.275
total	5.081

4 Experiments and Results

To evaluate the effectiveness of our model and justify central design decisions, we conducted multiple experiments resembling the Human Mobility Challenge scenario. For each city, we restricted to the individuals for whom data over the entire period is provided, ignoring the individuals who were to be predicted for the challenge. For cities B, C, and D, we chose the last 1000 individuals to serve as test individuals. We tested the model by evaluating the prediction for these individuals for the last 15 days of the period. Consequently, the entire trajectories for the remaining individuals, along with the trajectories of the first 60 days for the test individuals, constituted the training data for the experiments. From this set of trajectories, 5% was held back as a validation set. Moreover, we set the embedding dimension to 32 and used four layers for the encoder and decoder, respectively.

We evaluated the results by calculating the GEO-BLEU [9] and the dynamic time warping score (DTW) as similarity measures of trajectories. GEO-BLEU has been developed to assess the similarity of two trajectories by comparing their sub-trajectories. Hence, it focuses on local features instead of assessing the trajectories' global similarity. DTW, on the other hand, aligns two given sequences and subsequently computes their distance. Therefore, it evaluates the similarity of the trajectories' global pattern, supplementing GEO-BLEU. For GEO-BLEU, a higher score indicates a greater similarity, while for DTW, a lower score indicates a greater similarity of trajectories.

4.1 Model

For analyzing the model's architecture, we ablated its central components. These encompass the Mamba encoder, the choice of positional encodings, and the auxiliary next-location loss, as introduced in section 3.

To evaluate the performance of the Mamba encoder, we compared it to a causal transformer encoder and a regular transformer encoder. The results show not only that the Mamba encoder is competitive to both transformer encoders but that it surpasses them in terms of performance (table 2). Moreover, the results show that employing a Mamba encoder significantly reduces training time.

Table 2: Comparison of different encoders.

Encoder	GEO-BLEU	DTW	Training Time
Causal Transformer	0.285	32.39	1574s
Transformer	0.289	30.91	1616s
Mamba	0.301	28.98	1022s

Additionally, we compare our method of computing positional encodings, which contain the relative time difference between sequence elements, to positional encodings that restrict themselves to representing only the order of these elements. Our results indicate that our positional encodings successfully introduced a relative distance between the trajectory elements, improving the model's performance significantly, as seen in table 3.

Finally, we evaluate the role of the auxiliary next-location loss applied to the encoder's output by comparing the average losses

Table 3: Comparison of different positional encodings.

Positional Encodings	GEO-BLEU	DTW
Standard	0.295	31.89
Relative (ours)	0.301	28.99

per epoch of training with the auxiliary loss to training without the auxiliary loss. As illustrated in fig. 5, instructing the encoder to compute optimized representations for predicting the sequence's respective consecutive locations enhances the model's performance in an initial training phase. However, during the majority of the training period, this loss appears to restrict the model, preventing it from attaining its full potential. Nevertheless, this restriction diminishes with the progressing number of epochs, and the training losses converge.

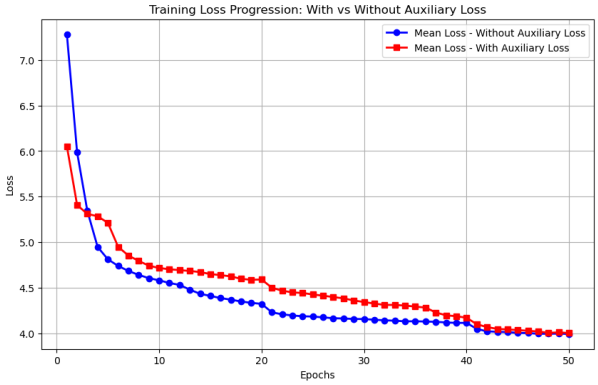


Figure 5: Effect of auxiliary next-location loss.

4.2 Training

Furthermore, we evaluated the training procedure, specifically the inclusion of random trajectory split (section 3.6) during training and the effect of the prediction window length. Splitting trajectories randomly aimed to increase the training set heterogeneity to improve the model's performance on unseen test data. While the splitting did not influence GEO-BLEU significantly, it improved DTW, as evidenced in table 4.

Table 4: Effect of randomly splitting trajectories during training.

Training Data	GEO-BLEU	DTW
Without Split	0.301	28.99
With Split	0.302	27.58

The effect of the prediction window length during training is depicted in table 5. A random relative prediction length of 3-10% of the context sequence length during training outperforms the other tested variations, partially by a large margin. Note that in the *absolute* length case, the same fixed prediction window length was used during inference. In the *relative* case, a prediction window length of 5% was chosen during inference, independent of the used length during training.

Table 5: Effect of different prediction window lengths during training. *Absolute* refers to a fixed prediction window length, *relative* refers to a prediction length as a percentage of the context length.

Length	GEO-BLEU	DTW
20 (absolute)	0.260	35.20
50 (absolute)	0.283	30.97
5% (relative)	0.292	32.25
3-10% (relative)	0.301	28.99

4.2.1 Fine-tuning. Dividing the experiment into the three cities, B, C, and D, we found significant differences in performance (see table 6). In particular for city D, being underrepresented in the training data, the model suffers from a notably worse performance compared to cities B and C. Therefore, we evaluated the effect of fine-tuning the model on city D data before inferring future trajectories for the corresponding test individuals of city D. Our results indicate that fine-tuning does not improve the model's accuracy for city D, as evidenced in table 7. Contrarily, training the model with data from all cities resulted in better predictions, even compared to a model trained from scratch only with data from city D.

Table 6: Performance split by city.

City	GEO-BLEU	DTW
B	0.307	24.93
C	0.297	18.75
D	0.297	43.28

Table 7: Effect of fine-tuning on city D.

Model	GEO-BLEU	DTW
Fine-tuned on City D	0.289	49.39
Trained on City D	0.230	46.04
Trained on All Cities	0.297	43.28

4.3 Inference

For the inference procedure, we tested different prediction window lengths and decay factors. The results are presented in table 8 and table 9.

We found that predicted sequences benefit from a long prediction length. Notably, predicting the entire future trajectory at once did not diminish the performance, although the model was initially trained to predict the next 3-10%.

Table 8: Comparison of different prediction window lengths during inference.

Length	GEO-BLEU	DTW
3%	0.297	29.35
5%	0.301	28.99
7%	0.303	28.59
10%	0.305	28.23
all	0.308	28.38

The effect of different decay factors (section 3.7) can be assessed as marginal, as evidenced in table 9. Lower decay factors appear to improve a trajectory's local features, while higher decay factors seem superior regarding global features, as evidenced by the improving GEO-BLEU and degrading DTW score for decreasing decay factors.

Table 9: Comparison of different decay factors during inference.

Decay Factor	GEO-BLEU	DTW
1	0.296	28.53
0.9	0.298	28.79
0.8	0.301	28.99
0.7	0.303	29.24

4.4 Zero-Shot Application

Motivated by the challenge's context of multi-city mobility prediction, we evaluated the zero-shot performance of our model. To do this, we trained our model with the trajectories over the entire period of only cities A, B, and C and evaluated on the first 3000 individuals of city D. Our results underscore the potential of our model, as it has learned general trajectory characteristics that are transferable across cities. We achieved a GEO-BLEU score of 0.298 and a DTW score of 42.86, showcasing its promising performance. This result closely matches the results for city D, as presented in table 6. Consequently, one may assume that the model did not only learn trajectory characteristics that are *transferable* across cities but characteristics that are *independent* of the underlying cities. However, further experiments are needed to validate this assumption.

5 Conclusion and Limitations

This paper presented a sequence-to-sequence model specifically designed to capture long-term dependencies in non-equidistant trajectories. The proposed model is based on past success of transformer and state space models in sequence modeling. The model's encoder-decoder architecture is capable of creating realistic trajectories over a long period: our results ranked in the top ten of the Human Mobility Challenge 2024. However, there exist several limitations that future work can elaborate on. For example, the proposed method does not include any data on the city structures. Future work may study how to incorporate data such as the locations of point of interest as additional context. Moreover, further analyzing the proposed model's zero-shot capabilities and testing strategies for further improvement is an exciting field for future work, directly related to the presented experiments.

Acknowledgement

The research was conducted in the context of the project "urbanModel" funded by the *German Aerospace Center (DLR e.V.)*. Furthermore, this activity was carried out in cooperation with the *LOEWE-Zentrum emergenCITY*.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-End Object Detection with Transformers. In *Computer Vision – ECCV 2020: 16th Europ. Conf., August 23–28, 2020, Proceedings, Part I* (Glasgow, United Kingdom). Springer-Verlag, Berlin, Heidelberg, 213–229. https://doi.org/10.1007/978-3-030-58452-8_13

[2] D. Gong, J. Lee, M. Kim, S. Ha, and M. Cho. 2022. Future Transformer for Long-term Action Anticipation. In *2022 IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 3042–3051. https://doi.org/10.1109/CVPR52688.2022.00306

[3] Albert Gu and Tri Dao. 2023. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. https://doi.org/10.48550/arXiv.2312.00752 arXiv:2312.00752

[4] Renhe Jiang, Xuan Song, Zipei Fan, Tianqi Xia, Quanjun Chen, Satoshi Miyazawa, and Ryosuke Shibasaki. 2018. DeepUrbanMomentum: An Online Deep-Learning System for Short-Term Urban Mobility Prediction. *Proc. of the AAAI Conference on Artificial Intelligence* 32, 1 (Apr. 2018). https://doi.org/10.1609/aaai.v32i1.11338

[5] Kai Li, Guo Chen, Runxuan Yang, and Xiaolin Hu. 2024. SPMamba: State-space model is all you need in speech separation. arXiv:2404.02063 http://arxiv.org/abs/2404.02063

[6] Kunchang Li, Xinhao Li, Yi Wang, Yinan He, Yali Wang, Limin Wang, and Yu Qiao. 2024. VideoMamba: State Space Model for Efficient Video Understanding. arXiv:2403.06977 http://arxiv.org/abs/2403.06977

[7] Massimiliano Luca, Gianni Barlacchi, Bruno Lepri, and Luca Pappalardo. 2021. A Survey on Deep Learning for Human Mobility. *ACM Comput. Surv.* 55, 1, Article 7 (Nov. 2021), 44 pages. https://doi.org/10.1145/3485125

[8] Liushuai Shi, Le Wang, Sanping Zhou, and Gang Hua. 2023. Trajectory Unified Transformer for Pedestrian Trajectory Prediction. In *2023 IEEE/CVF Int. Conf. on Computer Vision (ICCV)*. 9641–9650. https://doi.org/10.1109/ICCV51070.2023.00887

[9] Toru Shimizu, Kota Tsubouchi, and Takahiro Yabe. 2022. GEO-BLEU: similarity measure for geospatial sequences. In *Proc. of the 30th Int. Conf. on Advances in Geographic Information Systems* (Seattle, Washington) *(SIGSPATIAL '22)*. ACM, New York, NY, USA, Article 17, 4 pages. https://doi.org/10.1145/3557915.3560951

[10] Aivin V. Solatorio. 2023. GeoFormer: Predicting Human Mobility using Generative Pre-trained Transformer (GPT). In *Proc. of the 1st Int. Workshop on the Human Mobility Prediction Challenge* (Hamburg, Germany) *(HuMob-Challenge '23)*. ACM, New York, NY, USA, 11–15. https://doi.org/10.1145/3615894.3628499

[11] Haru Terashima, Naoki Tamura, Kazuyuki Shoji, Shin Katayama, Kenta Urano, Takuro Yonezawa, and Nobuo Kawaguchi. 2023. Human Mobility Prediction Challenge: Next Location Prediction using Spatiotemporal BERT. In *Proc. of the 1st Int. Workshop on the Human Mobility Prediction Challenge* (Hamburg, Germany) *(HuMob-Challenge '23)*. ACM, New York, NY, USA, 1–6. https://doi.org/10.1145/3615894.3628498

[12] Takahiro Yabe, Kota Tsubouchi, Toru Shimizu, Yoshihide Sekimoto, Kaoru Sezaki, Esteban Moro, and Alex Pentland. 2024. YJMob100K: City-scale and longitudinal dataset of anonymized human mobility trajectories. *Scientific Data* 11, 1 (April 2024). https://doi.org/10.1038/s41597-024-03237-9

[13] Cunjun Yu, Xiao Ma, Jiawei Ren, Haiyu Zhao, and Shuai Yi. 2020. Spatio-Temporal Graph Transformer Networks for Pedestrian Trajectory Prediction. In *Computer Vision – ECCV 2020: 16th Europ. Conf., August 23–28, 2020, Proceedings, Part XII* (Glasgow, United Kingdom). Springer-Verlag, Berlin, Heidelberg, 507–523. https://doi.org/10.1007/978-3-030-58610-2_30

[14] Zeyun Zhong, Manuel Martin, Frederik Diederichs, and Juergen Beyerer. 2024. QueryMamba: A Mamba-Based Encoder-Decoder Architecture with a Statistical Verb-Noun Interaction Module for Video Action Forecasting @ Ego4D Long-Term Action Anticipation Challenge 2024. arXiv:2407.04184 http://arxiv.org/abs/2407.04184