SST: A Simplified Swin Transformer-based Model for Taxi Destination Prediction based on Existing Trajectory

Zepu Wang[1*] zepu@seas.upenn.edu
Yifei Sun[2*] yifeisun@upenn.edu
Zhiyu Lei[1] zlei6@seas.upenn.edu
Xincheng Zhu[1] kyriezxc@seas.upenn.edu

***Abstract*— Precise destination prediction of taxi trajectories can benefit many intelligent location-based services such as accurate advertisements for passengers. To predict the final destination, a possible way is to turn the taxi trajectory into a two-dimensional image and rely on computer vision techniques. Swin Transformer is an innovative architecture in computer vision and has been demonstrated to have a good performance in many vision downstream tasks. However, it is not widely applied to solve real-world trajectory problems. In this paper, we propose a simplified Swin transformer (SST) structure to not use the idea of the shifted window in traditional Swin transformer because of the consecutive nature of trajectory data. Comprehensive experiments based on real trajectory data show that the SST can achieve higher accuracy than the state-of-the-art methods.**

I. Introduction

Efficient and stable transportation systems are critical to the smooth functioning of modern society, as they facilitate the movement of people and goods [1]. Taxis are a popular mode of transportation and play an important role in the overall traffic system. However, with the rise of online ride-hailing services, traditional taxi companies are facing challenges in terms of efficient scheduling and security monitoring of their vehicles, particularly because taxi drivers cannot know their destinations in advance.

Fortunately, most taxis are equipped with mobile GPS devices that record and report their trajectories. Analyzing this trajectory data can provide insights into the destination of a taxi, which can yield several benefits such as providing location-based services and applications, alleviating traffic congestion, and optimizing taxi dispatch. At the same time, analysis of the destinations of taxi trajectories can yield several benefits such as providing alleviating traffic congestion, optimizing taxi dispatch, and location-based services and applications such as recommending sightseeing places, accurate ads based on destinations, etc.

Destination prediction based on vehicle trajectories involves using machine learning algorithms to analyze the trajectory data collected from the GPS devices installed in taxis. These algorithms can identify patterns in the data and use these patterns to predict the destination of a taxi with a high degree of accuracy. This can help taxi companies to efficiently schedule their vehicles and ensure that they are being utilized effectively.

Moreover, destination prediction can also help improve the security monitoring of taxis. By analyzing the trajectory data, it is possible to detect anomalies such as sudden deviations from a usual route or unexpected stops, which could be indicative of criminal activity [1].

Vehicle destination prediction is typically based on analyzing previous GPS records along with the surrounding environment, which includes factors such as the road structure and other nearby vehicles [2]. A variety of models have been developed to address this issue, including conventional approaches and deep learning methods. Conventional methods such as physics-based, maneuver-based, and interaction-aware models [3], [4] are limited in their ability to capture the complex spatiotemporal dependencies in the data, resulting in suboptimal prediction accuracy. With the emergence of deep learning, researchers have explored the use of convolutional neural networks (CNNs) and recurrent neural networks (RNNs) for trajectory prediction [5], [6]. These methods leverage the power of deep learning to capture non-linear relationships and long-range dependencies in the data, resulting in significant improvements in prediction accuracy. More recently, graph-based techniques such as graph convolutional networks (GCNs) [7] have been incorporated to model the spatial structure and interactions between taxis within the road network. GCNs can effectively model the underlying structure of road networks and capture the interactions between different taxis, leading to improved prediction accuracy.

In order to fully capture the spatial information of a trajectory, researchers often convert it into a two-dimensional map since it is highly related to the structure of road networks. This allows for the utilization of more advanced computer vision techniques to solve prediction problems. In recent years, with the breakthroughs in computer vision using vision transformers, many scholars have been inspired to use them for trajectory prediction and have achieved good results [8], [9]. Furthermore, Swin-transformer, a variant of vision transformers, has become a general-purpose backbone for computer vision tasks [10]. However, to the best of our knowledge, the Swin architecture has not been widely used in trajectory analysis or destination prediction before.

The main contributions of this paper are as follows:

- Firstly, an SST is proposed that is better suited to the trajectory image problem. This model is shown to be competitive for spatiotemporal prediction of taxi destinations, providing a new perspective for researchers seeking to apply state-of-the-art computer vision tech-

* Zepu Wang and Yifei Sun contribute to this paper equally.
[1] Department of Computer and Information Science, University of Pennsylvania
[2] Stuart Weitzman School of Design, University of Pennsylvania

niques to trajectory prediction problems.
- Secondly, the study compares three image-based modeling approaches for trajectory prediction and evaluates their effectiveness in fitting traditional trajectory data into a trajectory image. The results of this comparison can provide insights into the most effective ways to convert traditional trajectory data into trajectory images for further analysis.

The remainder of this paper is structured as follows. Section II provides a comprehensive review and summary of previous studies related to trajectory prediction. Section III defines the problem statement and presents the challenges associated with predicting taxi destinations based on trajectory data. The proposed methodology, including data processing and model structure, is described in Section IV. Section V presents experimental results that compare several models and their performance against our proposed approach. The results demonstrate the effectiveness of our SST in predicting taxi destinations based on trajectory data. In Section VI, we summarize the contributions of this study and highlight its potential impact on the transportation system and society as a whole. Section VII outlines two limitations of our proposed model and suggests avenues for future research. Finally, ethical concerns related to the use of deep learning for predicting vehicle trajectory are discussed in Section VIII.

II. Literature Review

Trajectory analysis is widely studied in the literature using traditional and machine learning (deep learning) approaches. Early studies in trajectory prediction employed physics-based models such as dynamic models [11], [3] and kinematic models [12], [13], which predict future vehicle motion based on vehicle attributes, control inputs, and external factors such as the vehicle's position, heading, and speed. While physics-based models are widely used in trajectory prediction and collision risk estimation, their ability to predict trajectories over a long time is limited by their reliance on low-level motion properties.

In contrast, some researchers propose maneuver-based models that consider prior knowledge, making them more reliable than physics-based models. These models are based on prototype trajectories [14] or maneuver intention estimation [15], [16]. However, they do not consider external objects such as surrounding vehicles, which can cause misjudgments. To address this limitation, interaction-aware models were developed that treat vehicles as maneuvering entities that can be affected by other vehicles in a scene. These models are based on either prototype trajectories [17] or Dynamic Bayesian Networks [18], and show better results than traditional maneuver-based models. Nonetheless, these models suffer from expensive computation problems, as they need to compute all possible vehicle trajectories.

In recent years, with the advancement of deep learning, learning-based techniques are increasingly applied to solve vehicle trajectory prediction problems. As trajectories possess sequential attributes, the problem can be addressed as a time-series prediction task. Therefore, many scholars utilize typical recurrent neural networks (RNNs) [19], long short-term memory (LSTM) neural networks[20], and gated recurrent unit (GRU) networks[21] as their basic structures to design the model. For instance, Kim et al.[22] propose an LSTM-based framework to learn various behaviors of vehicles from massive trajectory records. Deo et al.[23] propose an LSTM model for trajectory prediction under the scene of the freeway, which not only includes track histories but also takes into account surrounding vehicles and road structures as input. Lee et al. use the RNN Encoder-decoder framework to build the DESIRE model[24], which accurately predicts the future locations of objects across various scenes.

Most existing methods for trajectory prediction focus on modeling trajectories as a one-dimensional time series, which may not fully capture the complex nonlinear spatial-temporal correlations inherent in trajectory data. This limitation becomes particularly evident when predicting trajectories that are highly related to road structures, such as those involving corners or winding paths. To overcome this limitation, some recent works propose to transform trajectory data into a two-dimensional image-like format, where each pixel corresponds to a specific location and encodes information about the presence and movement of vehicles in that location over time. Therefore, more computer vision architectures such as convolutional neural networks (CNNs) can be used to extract more spatial information and build relationships with surrounding objects. For instance, Lv et al. propose a CNN-based model [5] that takes vehicle trajectory prediction as an image prediction task and combines multi-scaled trajectory patterns. The model shows high accuracy in trajectory prediction tasks. Similarly, Guo et al. combine CNN with LSTM [25] to predict the trajectory of surrounding vehicles by merging the spatial expansion properties of CNN and the temporal expansion capabilities of LSTM. The model shows better performance than using time-series models such as LSTM or GRU alone.

It is worth noting that since 2017, attention-based/transformer-related models demonstrate impressive performance in various application scenarios. In the field of computer vision, the Vision Transformer (ViT) proposed by Dosovitskiy et al. [26] has shown remarkable results in many computer vision tasks. In our research, we employ a more specific type of transformer, namely the Swin Transformer [27]. The Swin Transformer generates hierarchical feature maps by merging image patches in deeper layers, and its linear computation complexity to input image size is due to the computation of self-attention only within each local window. As a result, it can function as a general-purpose backbone for both image classification and dense recognition tasks. A comprehensive description of our modified Swin structure is provided in Section IV.

III. Problem Statement

When taxis start carrying passengers, it is able to begin collecting the travel trajectory of occupied taxis. Specifically, given a taxi X_i, its j-th trajectory Y_{ij} is recorded as a sequence of GPS locations in a fixed period: $Y_{ij} =$

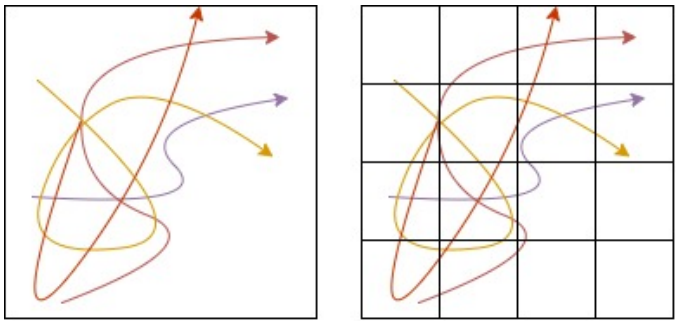

Fig. 1. An illustration of two-dimensional image representation

$\langle l_{ij1}, l_{ij2}, l_{ij3}, \ldots, l_{ijN_{ij}} \rangle$. Each l_{ijk} in the sequence represents a GPS location of longitude and latitude pair (A_{ijk}, B_{ijk}), collected instantly. Here, N_{ij} is the total length of the taxi's current trip path Y_{ij}. It is worth noting that the total length of different trajectories can vary.

To clarify, l_{ij1} is the start of the trip where the taxi takes on the passenger(s). The destination of the taxi is represented by the last location in the sequence, $l_{ij,N_{ij}}$. We define the destination of any trajectory Y_{ij} as $\zeta_{Y_{ij}}$. Our prediction problem can be defined as predicting the final destination $\zeta_{Y_{ij}}$ of a taxi X_i in a trip Y_{ij}, given its historical trajectory set.

IV. Proposed Method

A. Image-based modeling of trajectory

To extract spatial patterns from the taxi trajectories, a two-dimensional image representation is adopted. Then, divide the map into an $M \times M$ grid, where M is a constant resolution of the map. Each GPS location l_{ijk} is mapped onto a corresponding grid cell G_{mn} based on its latitude and longitude, where m and n represent the row and column indices of the grid cell, respectively. This mapping relationship is denoted as $l_{ijk} \rightarrow G_{mn}$. By applying binary, linear, or quadratic transformation methods, the pixel values $I_{ij}(m,n)$ (where $1 \leq m, n \leq M$) of the resulting $M \times M$ matrix I_{ij} can be defined. These matrices serve as the two-dimensional image representation of the taxi trajectories. Here, we present three methods to make the taxi trajectories. In Figure 1, different arrows represent 4 different trajectories and it is displayed in a 4×4 matrix.

1) Binary Method:

$$I_{ij}^{\text{bin}}(m,n) = \begin{cases} 1, \text{ if } \exists l_{ijk} \in Y_{ij} \wedge l_{ijk} \rightarrow G_{mn} \\ 0, \text{ else} \end{cases} \quad (1)$$

In the binary method, a value of 1 is assigned to a grid cell if the taxi passes through the area where the grid cell represents at any given time, and a value of 0 is assigned otherwise. However, this method only captures the track of the vehicle and does not account for temporal information.

2) Linear Method:

$$I_{ij}^{\text{lin}}(m,n) = \begin{cases} N_{ij}/m, \text{ if } \exists l_{ijk} \in Y_{ij} \wedge l_{ijN_{ij}} \rightarrow G_{mn} \\ 0, \text{ else} \end{cases} \quad (2)$$

In the linear method, the temporal dimension of the trajectory is taken into account by equally dividing the time into $N_{ij} - 1$ parts, where N_{ij} is the total length of the trajectory. As time passes during the taxi trip, the pixel value representing the location of the taxi gradually increases from 0 to 1, providing an intuitive way of encoding the temporal information in the trajectory image.

3) Quadratic Method:

$$I_{ij}^{\text{qua}}(m,n) = \begin{cases} (N_{ij}/m)^2, \text{ if } \exists l_{ijk} \in Y_{ij} \wedge l_{ijN_{ij}} \rightarrow G_{mn} \\ 0, \text{ else} \end{cases} \quad (3)$$

In the quadratic method, the pixel values of the linear method are quadratically transformed at each time step. This approach allows for a more nuanced representation of the temporal information in the trajectory images. Specifically, if the final destination is more strongly correlated with the later portions of the trajectory and less with the initial locations, the quadratic method assigns relatively higher values to the later portion of the trajectory. As a result, the quadratic processed images exhibit more prominent features associated with the later portion of the trajectory compared to those processed using the linear method.

B. Simplified Swin Transformer (SST)

Inspired by the successful application of the Swin transformer [10] in many computer vision tasks, we design an SST for the destination prediction task.

1) General Structure: An overview of the Swin Transformer architecture, which has been adopted in this paper for trajectory prediction, is presented in Figure 2. The architecture first splits the input trajectory image into non-overlapping patches using a patch-splitting module. However, unlike traditional image inputs, the trajectory image only has one channel. In this implementation, we use a patch size of 10 x 10 in the patch partition stage.

After processing by the first stage, in order to produce a hierarchical representation, the network merges small patches into bigger ones as it goes deeper. As illustrated in the figure, the first patch merging layer concatenates the features of each group of 2 × 2 neighboring patches. Since the trajectory image size in our experiment is 40, after the second merging, the network calculates the global attention to the big matrix. In Figure 2, we illustrate how the architecture calculates attention scores in different stages. The picture is the mean of all the trajectories that we use in Section V. In this way, we can capture the global relationships between different pixels without having to calculate the global attention multiple times, which could waste computational resources.

The Swin Transformer architecture has several key advantages to our trajectory prediction task. It allows for efficient computation and scalability to handle larger images by using a hierarchical patch-based approach, which can be applied to the trajectory image data. Additionally, the use of attention mechanisms in the architecture allows for the model to better capture long-range dependencies between pixels, which is important for accurately predicting the final destination of a taxi trajectory.

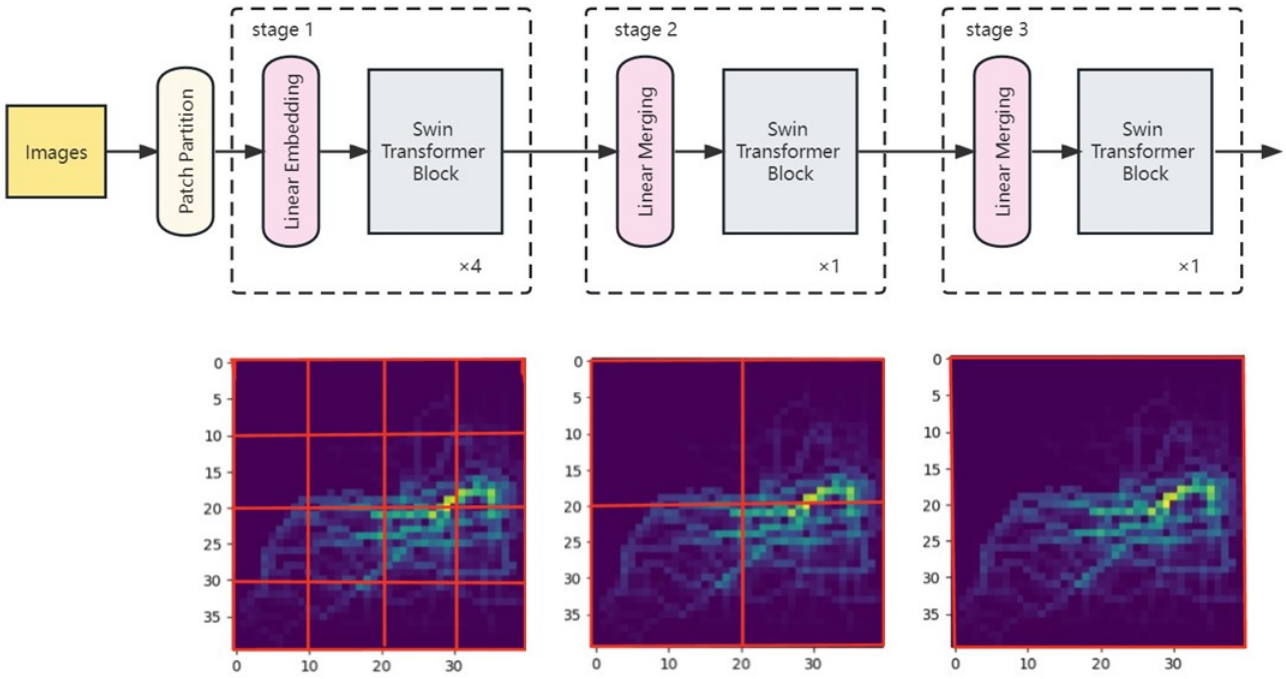

Fig. 2. The architecture of the SST

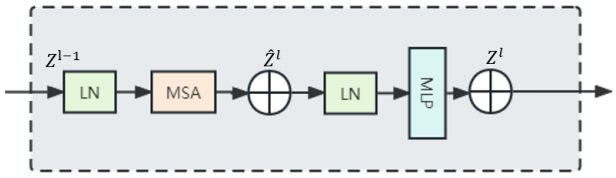

Fig. 3. The architecture of the SST block

2) SST block: Figure 3 provides an illustration of the SST block. The input is first processed by a Layernorm (LN) layer, followed by a conventional multi-head self-attention (MSA) module. MSA is an extension of self-attention, which is a technique used to calculate the importance of different parts of the input when predicting an output. In MSA, we run k self-attention operations, in parallel, and combine their outputs together. The output then undergoes processing by an LN layer and MLP layer, before being subject to another residual connection.

In contrast to the traditional Swin Transformer, which proposes computing self-attention within local windows and applying a shifted window partitioning approach to enhance connections between non-overlapping windows, this paper argues against using shifted window-based MSA. This decision is based on the nature of trajectory images, which represent the continuous change in a vehicle's state within the real world, with the velocity of vehicles limited to a specific range.

Using a trajectory image as an input implies that a specific pixel representing the vehicle's location at a given time should be more related to its few previous states and next several locations than pixels further away. Hence, applying the shifted window technique in this work would result in the network finding relationships between two patches located far from each other, even though such a relationship should not exist. Therefore, our SST block adopts a more straightforward approach to self-attention, avoiding the shifted window technique and focusing on capturing local and global dependencies in a way that is better suited to the characteristics of trajectory images.

V. EXPERIMENT

A. Data Preparation

In our experiment, we evaluate the performance of the SST model along with other baseline models on a real trajectory dataset from the ECML-PKDD competition [28]. The dataset comprises 1.7 million complete trajectories collected from 442 taxis that operated in the city of Porto for a year, from 2013-07-01 to 2014-06-30. Each trajectory consists of a list of longitude and latitude pairs that represent the recorded positions of a taxi during the trip. To reduce the dataset's

size, we randomly sample 100,000 trajectories. As the city of Porto is vast, we only retain trajectories within a certain longitude and latitude range ($[-8.7, -8.6]$ and $[41.1, 41.2]$ respectively) since most of the trajectories fall within this range.

Next, we apply MIN-MAX normalization to map all the longitude and latitude values to the range of $[0, 1]$. We then convert these values to their corresponding pixel coordinates on an $M \times M$ grid, where M is chosen to be 40 after several trials and errors. Finally, we use the last coordinate in the list as the target for the prediction task and the remaining coordinates as input to construct the trajectory image.

After these preprocessing steps, we randomly split the dataset into three sets: 60% for training, 20% for validation, and 20% for testing.

B. Evaluation Metrics

In this experiment, the problem is approached as a regression task, in which the final destination is predicted based on the trajectory image. The mapping relationship also processes the predicted coordinates of the destination. To evaluate the performance of the different models, the mean square error (MSE) is used as the primary evaluation metric.

The MSE is defined as the average of the squared differences between the predicted values and the ground truth values, and is expressed as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2, \tag{4}$$

where y_i and $\hat{y_i}$ denote the ground truth and predicted values, respectively, and n is the total number of samples.

In this experiment, the mean absolute error (MAE), which is another commonly used evaluation metric, is not utilized. This decision is based on the observation that the dataset is preprocessed and cleaned, and therefore, there are no significant outliers in the data. As such, the MSE is preferred over the MAE, as it tends to penalize larger differences between the predicted and actual values more severely, which is desirable in this context.

C. Result Evaluation

We adopt three baseline models in this experiment:

- Multilayer perceptron (MLP): A simple multi-layer perceptron with four hidden layers of 150 neurons followed by a dropout layer.
- Convolutional neural networks (CNNs): A simple convolutional neural network with a 7×7 convolution kernel with 128 channels followed by two fully connected layers.
- Long short-term memory neural networks (LSTMs): Process the sequential input directly without the trajectory image. The list of pixel coordinates is first converted to a 200×3 tensor as the model input. The 200 rows represent a sequence with fixed length 200: for a list with length L greater than 200, only take the last 200 coordinates; for L less than 200, the last L rows of the tensor are filled with coordinates while the rest are padded with zeros. The three columns indicate three input features: the first two are the pixel coordinates from the list while the third is the constant one for non-zero-padded rows. The LSTM model has one recurrent layer and hidden states with four features. The final output layer is used to map the last hidden state in the sequence to the target coordinate.

TABLE I
MEAN SQUARE ERRORS ON TEST SET

Model \ Method	Binary	Linear	Quadratic
MLP	6.8748	3.2009	2.6359
CNN	5.1156	1.7879	1.5543
SST	5.2952	1.7722	**1.4865**
LSTM	1.6562		

Table I presents the experimental results obtained by employing three types of image-based modeling of trajectory definition. Our findings suggest that the SST transformer model with the quadratic trajectory preprocessing method achieves the lowest MSE error of 1.4865.

The quadratic method outperforms the binary and linear methods in all MLP, CNN, and SST models. This result aligns with our expectations since the quadratic method captures more prominent features associated with the later portion of the trajectory, resulting in better performance of prediction methods. In contrast, the binary method loses sequential information during the trajectory image conversion, thereby preventing the CNN and SST models from capturing vehicle direction in the trajectory. While the performance of linear and quadratic methods is similar, the slightly better performance of the quadratic method indicates that it can more accurately predict the final destination. Hence, we use experiments to prove that the quadratic processed images exhibit more prominent features associated with the later portion of the trajectory compared to those processed using the linear method.

Additionally, we also compare the performance of the traditional Swin transformer and our modified simple version in Figure 4. For all binary, linear, and quadratic methods, SST has a higher accuracy. Therefore, to some extent, we demonstrate that the shifted window technique is not appropriate for this destination prediction problem.

Interestingly, our experiment also find that the LSTM model shows relatively good performance compared to most other experiments. This finding suggests that LSTM is an ideal trajectory analysis technique, despite the fact that it

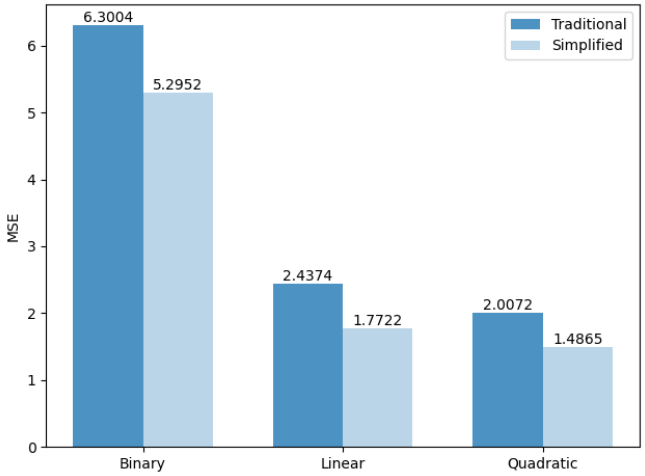

Fig. 4. Comparison between traditional and SST

requires the length of the trajectory to be the same in order to train the LSTM model. However, since real-world trajectories often have different lengths, embedding the various trajectory sequences into a common embedding space may be a promising research direction for building more complex LSTM-related models.

VI. Conclusion

This paper proposes a novel approach to predicting the destinations of taxi trajectories, which involves three different trajectory image formation methods and the use of a simplified Swin transformer (SST) model. Our results show that the quadratic method is the most effective technique for this task, while also demonstrating that the SST outperforms the traditional version, indicating that the shifting window technique is not necessary for trajectory analysis.

In the future, further experiments with additional datasets could be conducted to evaluate the performance of the SST more comprehensively. Moreover, as highlighted in Lv et al. [5], different portions of a trajectory may have varying contributions to the final prediction. Thus, exploring the use of trajectory processing methods that can leverage these differences could be a fruitful avenue for future research.

VII. Limitations

In this section, we will explain two limitations of this experiment.

A. Computational Efficiency

Although Swin Transformer is an improvement over the Vision Transformer and is faster, it still incurs a higher computational cost than conventional deep learning algorithms such as RNNs and CNNs.

B. Short Term Trajectory

When the length of a trajectory is short, the resulting trajectory image may contain too few informative pixels, leading to a sparse data issue, as shown in Figure 5. This can pose a challenge in accurately capturing important features and patterns from the trajectory data.

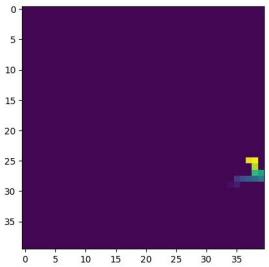

Fig. 5. An illustration about the sparse trajectory image

VIII. Ethical Concern

As the use of deep learning continues to grow, ethical concerns surrounding its application have garnered significant attention. In this study, there are potential negative implications of our model that must be addressed. One of the most prominent concerns is privacy. Deep learning models rely on a vast amount of data to make accurate predictions, and if this data contains sensitive information about individuals such as their travel patterns or location history, there is a risk of privacy violations. Taxi riders may not be aware that their data is being collected and used in this way, which could lead to a breach of their privacy.

Another concern is the potential misuse of the model. If the model falls into the wrong hands, it could be used for purposes other than predicting taxi trajectories, such as tracking the movements of individuals for surveillance or other nefarious purposes. This might pose a serious threat to personal privacy and security.

Appendix: Hyperparameters Tuning

For tuning the hyperparameters in our three baseline models, we randomly sample 10,000 trajectories with 80% for training and 20% for validation and use the linear image-based modeling method to evaluate the hyperparameters.

A. MLP

Table II shows the performance of MLPs with different hidden layer numbers and sizes. We decide to use 4 hidden layers of 150 neurons

B. CNN

Table III shows the performance of CNNs with different convolution channels and kernel sizes. We decide to use a 7×7 convolution kernel with 128 channels.

C. LSTM

Table IV shows the performance of LSTMs with different hidden state sizes. We decide to use 4 features for hidden states.

IX. Acknowledgement

Thanks to Professor Ungar, Professor Korald, TA Xinyue, and all the other TAs for their contributions this semester.

TABLE II

MEAN SQUARE ERRORS FOR MLPS WITH VARIOUS HIDDEN LAYER NUMBERS AND SIZES

Num \ Size	50	100	150	200
1	12.08	11.21	9.37	9.36
2	11.51	9.27	8.79	8.70
3	13.53	9.98	7.72	7.30
4	13.71	7.84	6.85	7.30

TABLE III

MEAN SQUARE ERRORS FOR CNNS WITH VARIOUS CONVOLUTION CHANNELS AND KERNEL SIZES

Chann \ Size	3	5	7	9
16	6.63	6.00	5.48	5.63
32	6.78	4.94	4.37	4.49
64	5.70	4.63	4.26	4.51
128	4.91	4.36	4.08	5.84

TABLE IV

MEAN SQUARE ERRORS FOR LSTMS WITH VARIOUS HIDDEN STATE SIZES

Hidden State	2	3	4	5
MSE	51.53	48.78	28.66	29.38

REFERENCES

[1] Z. Wang, P. Sun, and A. Boukerche, "A novel time efficient machine learning-based traffic flow prediction method for large scale road network," in *ICC 2022-IEEE International Conference on Communications*. IEEE, 2022, pp. 3532–3537.

[2] S. Mozaffari, O. Y. Al-Jarrah, M. Dianati, P. Jennings, and A. Mouzakitis, "Deep learning-based vehicle behavior prediction for autonomous driving applications: A review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 1, pp. 33–47, 2022.

[3] R. Pepy, A. Lambert, and H. Mounier, "Reducing navigation errors by planning with realistic vehicle model," in *2006 IEEE Intelligent Vehicles Symposium*. IEEE, 2006, pp. 300–307.

[4] T. Gindele, S. Brechtel, and R. Dillmann, "A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments," in *13th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2010, pp. 1625–1631.

[5] J. Lv, Q. Li, Q. Sun, and X. Wang, "T-conv: A convolutional neural network for multi-scale taxi trajectory prediction," in *2018 IEEE international conference on big data and smart computing (bigcomp)*. IEEE, 2018, pp. 82–89.

[6] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 1468–1476.

[7] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[8] J. Zhao, X. Li, Q. Xue, and W. Zhang, "Spatial-channel transformer network for trajectory prediction on the traffic scenes," *arXiv preprint arXiv:2101.11472*, 2021.

[9] Y. Liu, J. Zhang, L. Fang, Q. Jiang, and B. Zhou, "Multimodal motion prediction with stacked transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7577–7586.

[10] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[11] C.-F. Lin, A. G. Ulsoy, and D. J. LeBlanc, "Vehicle dynamics and external disturbance estimation for vehicle path prediction," *IEEE Transactions on Control Systems Technology*, vol. 8, no. 3, pp. 508–518, 2000.

[12] S. Ammoun and F. Nashashibi, "Real time trajectory prediction for collision risk estimation between vehicles," in *2009 IEEE 5th International Conference on Intelligent Computer Communication and Processing*. IEEE, 2009, pp. 417–422.

[13] N. Kaempchen, K. Weiss, M. Schaefer, and K. C. Dietmayer, "Imm object tracking for high dynamic driving maneuvers," in *IEEE Intelligent Vehicles Symposium, 2004*. IEEE, 2004, pp. 825–830.

[14] D. Vasquez and T. Fraichard, "Motion prediction for moving objects: a statistical approach," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 4. IEEE, 2004, pp. 3931–3936.

[15] S. Klingelschmitt, M. Platho, H.-M. Groß, V. Willert, and J. Eggert, "Combining behavior and situation information for reliably estimating multiple intentions," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE, 2014, pp. 388–393.

[16] H. Berndt, J. Emmert, and K. Dietmayer, "Continuous driver intention recognition with hidden markov models," in *2008 11th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2008, pp. 1189–1194.

[17] E. Käfer, C. Hermes, C. Wöhler, H. Ritter, and F. Kummert, "Recognition of situation classes at road intersections," in *2010 ieee international conference on robotics and automation*. IEEE, 2010, pp. 3960–3965.

[18] G. Agamennoni, J. I. Nieto, and E. M. Nebot, "Estimation of multivehicle dynamics by considering contextual information," *IEEE Transactions on robotics*, vol. 28, no. 4, pp. 855–870, 2012.

[19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[22] B. Kim, C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi, "Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 399–404.

[23] N. Deo and M. M. Trivedi, "Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms," in *2018 IEEE intelligent vehicles symposium (IV)*. IEEE, 2018, pp. 1179–1184.

[24] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, "Desire: Distant future prediction in dynamic scenes with in-

teracting agents," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 336–345.

[25] G. Xie, A. Shangguan, R. Fei, W. Ji, W. Ma, and X. Hei, "Motion trajectory prediction based on a cnn-lstm sequential model," *Science China Information Sciences*, vol. 63, pp. 1–21, 2020.

[26] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu, Z. Yang, Y. Zhang, and D. Tao, "A survey on vision transformer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 87–110, 2023.

[27] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 10 012–10 022.

[28] Kaggle, "Kaggle competiation." [Online]. Available: https://www.kaggle.com/c/pkdd-15-predict-taxi- service-trajectory-i, 2015