J. Highw. Transp. Res. Dev. 2026, 20(1): 28−34

Journal of Highway and Transportation Research and Development

Home page: www.sciopen.com/journal/2095-6215

GNTI: Gaussian noise-based trajectory imputation via self-supervised learning

Siqi Liu[1*], Penghao Zhao[1**], Lei Dong[1], Na Dong[1], Liyuan Ding[2], Guangshi Pei[2]

1. *Beijing Transportation Comprehensive Law Enforcement Corps, Beijing 100044, China*
2. *RIOH High Science and Technology Group, Beijing 100088, China*

Abstract: Trajectory imputation aims to reconstruct complete movement sequences from noisy or incomplete GPS data, crucial for intelligent transportation systems (ITS). This study proposes GNTI (Gaussian Noise-based Trajectory Imputation), a self-supervised learning framework that introduces Gaussian noise during model training to simulate real-world GPS errors. By perturbing the input trajectories with probabilistic Gaussian noise, GNTI enables the model to learn robust trajectory representations without relying on labeled datasets. A transformer-based BERT encoder is employed to capture complex spatial-temporal dependencies, while a simple multilayer perceptron (MLP) decoder predicts corrected trajectory points based on contextualized embeddings. Extensive experiments were conducted on two large-scale real-world datasets, Chengdu and Porto. Comparative results show that GNTI outperforms traditional Seq2Seq-based models (gated recurrent unit (GRU), long short-term memory (LSTM)) and recent transformer-based models (Transformer, ST-BerImp), achieving the highest Micro-F1 scores across all settings. Specifically, GNTI improves Micro-F1 scores by 3%–5% over ST-BerImp. Ablation studies demonstrate that Gaussian noise augmentation improves model robustness by approximately 5% compared to models trained without augmentation. GNTI offers a practical and scalable solution for trajectory imputation tasks, enhancing robustness to GPS inaccuracies and reducing the need for complex multi-task objectives. Future work may explore extending the method to denser urban environments and optimizing it for real-time deployment.

Keywords: Transportation engineering; ITS; trajectory imputation; Gaussian noise; self-supervised learning; GPS noise; transformer

1 Introduction

With the rapid advancement of intelligent transportation systems, trajectory data has become an essential resource for applications such as traffic monitoring, route planning, and urban mobility analysis. However, real-world GPS trajectory data often suffers from measurement errors due to signal interference, environmental factors, and device limitations. These errors lead to inaccurate positioning, affecting downstream tasks such as map matching, travel behavior analysis, and anomaly detection [1−2].

Traditional trajectory imputation methods can be broadly classified into statistical interpolation techniques and deep learning-based approaches. Early statistical methods, such as Kalman filtering and spline interpolation, attempt to correct trajectory errors based on predefined motion models. However, these methods struggle with complex spatial-temporal dependencies and dynamic road conditions [3−4]—limitations that become more critical when the imputed trajectories serve as input for dynamic traffic applications (e.g., the real-time cooperative merging in Ref. [2] or adaptive signal control in Ref. [5], which demand high-precision temporal-spatial data). More recently, deep learning-based models, including recurrent neural networks (RNNs) and transformer-based architectures [5−7], have shown promising results in trajectory recovery by learning latent patterns from large datasets [8−9]. Despite their effectiveness, these models typically rely on large amounts of labeled training data and may not generalize well [10] to unseen trajectory distortions—an issue that hinders their deployment in scenarios where labeled data is scarce, such as the calibration of regional traffic flow prediction models [1] or the testing of connected and autonomous vehicles (CAV) cooperative algorithms [2] in new urban areas.

To address these challenges, we propose GNTI (Gaussian Noise-based Trajectory Imputation), a self-supervised learning framework that enhances trajectory modeling by introducing Gaussian noise during training. Unlike conventional data augmentation techniques that rely on random dropout or contrastive learning, GNTI explicitly simulates real-world GPS errors, allowing the model to learn robust trajectory representations without requiring extensive labeled data. By leveraging a transformer-based encoder, GNTI effectively captures spatial-temporal dependencies and improves trajectory recovery accuracy under various noise conditions. The main contributions of this work are as follows.

We introduce an augmentation strategy Gaussian noise method to enhance trajectory robustness, effectively simulating real-world GPS errors.

We propose a self-supervised learning framework that eliminates the need for manually labeled trajectory data while maintaining high imputation accuracy.

We conduct extensive experiments on two large-scale real-world datasets (Chengdu and Porto), demonstrating that GNTI outperforms existing trajectory imputation methods in terms of precision, recall, and generalization.

The remainder of this paper is structured as follows: Section 2 reviews related work in trajectory imputation and self-supervised

Received: 9 May 2025; **Revised:** 15 July 2025; **Accepted:** 29 August 2025
* *E-mail address*: liusiqi647@jtw.beijing.gov.cn; ** *Corresponding author*: 16286876@qq.com
DOI: 10.26599/HTRD.2026.9480088

learning. Section 3 presents the methodology of GNTI, detailing the noise-based augmentation strategy and model architecture. Section 4 describes the experimental setup and evaluation results. Finally, Section 5 concludes the paper and discusses potential future research directions.

2 Related work

2.1 Trajectory imputation and recovery

Trajectory imputation aims to reconstruct complete and accurate movement sequences from partial, noisy, or corrupted GPS data. In real-world applications such as traffic monitoring and urban planning, the quality of GPS data often suffers due to signal loss, urban canyons, and device limitations. Traditional methods rely on statistical techniques like Kalman filters, linear interpolation, and spline smoothing [11−12], which assume regular motion models but often fail in complex spatial-temporal scenarios.

With the rise of deep learning, many recent approaches have modeled mobility patterns using sequence models like RNNs (GRU, LSTM) [11−12], or attention-based architectures such as Transformers [13−14]. These models can learn from large-scale trajectory datasets but often require extensive supervision and lack robustness under noisy input conditions.

2.2 Self-supervised learning and BERT-backbone for trajectories

Inspired by the success of self-supervised pretraining in natural language processing, bidirectional encoder representations from transformers (BERT) based models have been adopted for spatial-temporal sequence modeling, including trajectory prediction and mobility understanding (Fig. 1). These models leverage masked modeling or contrastive objectives to learn contextual representations without labeled data.

However, existing BERT-style trajectory models [15−16] often involve multi-task learning or contrastive pretraining [17−20], which increases model complexity and training overhead. Moreover, few works explicitly simulate real-world GPS noise during training.

In contrast, our proposed approach, GNTI, introduces a Gaussian Noise-based augmentation strategy within a BERT encoder framework to improve the model's robustness through self-supervised training. This design achieves competitive performance while maintaining a lightweight and easy-to-implement architecture.

3 Methodology

3.1 Problem definition

Let the road network be represented as a directed graph $G=(V,E)$, where V is the set of nodes (e.g., intersections) and E is the set of directed edges (e.g., road segments). A trajectory is defined as:

$$T=\{e_1, e_2, \cdots, e_n\},\ e_i \in E \tag{1}$$

Due to GPS noise, the observed trajectory $\tilde{T}$ may deviate from the ground truth trajectory. The objective is to learn a mapping function $f: \tilde{T} \to T$ that accurately reconstructs the original trajectory from its noisy counterpart.

3.2 Overview of GNTI framework

The GNTI framework offers a robust approach to trajectory imputation by leveraging the power of BERT encoders and incorporating a self-supervised training strategy. This framework addresses the common challenge of incomplete trajectory data due to GPS errors and other real-world irregularities. It focuses on learning robust representations of trajectories that can effectively handle noisy and sparse data.

The GNTI architecture is built upon three key components:

(1) Gaussian noise augmentation on GPS inputs: This crucial step simulates realistic GPS errors by adding Gaussian noise to the raw GPS data during the training process. This allows the model to learn to cope with noisy inputs and improve the robustness of its imputations.

(2) Trajectory embedding and positional encoding: The raw GPS data is transformed into a suitable representation for the BERT encoder. Positional encoding is incorporated to capture the sequential nature of trajectory data, allowing the model to understand the temporal order of locations within a trajectory.

(3) BERT encoder to model contextual dependencies: A BERT encoder, known for its ability to capture long-range dependencies in sequential data, is employed to model the contextual relationships between different points within a trajectory. This allows the model to learn complex spatial-temporal patterns and make more accurate imputations.

3.3 Gaussian noise-based augmentation

As shown in Fig. 2, Gaussian noise-based augmentation serves as a crucial component of the GNTI framework, enhancing the model's robustness to the inherent noise and inaccuracies present in real-world GPS data.

By simulating these imperfections during training, the model learns to effectively filter noise and produce more reliable trajectory imputations. The augmentation process involves injecting Gaussian noise into the trajectory coordinates. Specifically, for each GPS point in the original trajectory T, Gaussian noise sampled from a normal distribution with zero mean (μ=0) and a specified variance (σ^2) is added to the coordinates. This perturbation pro-

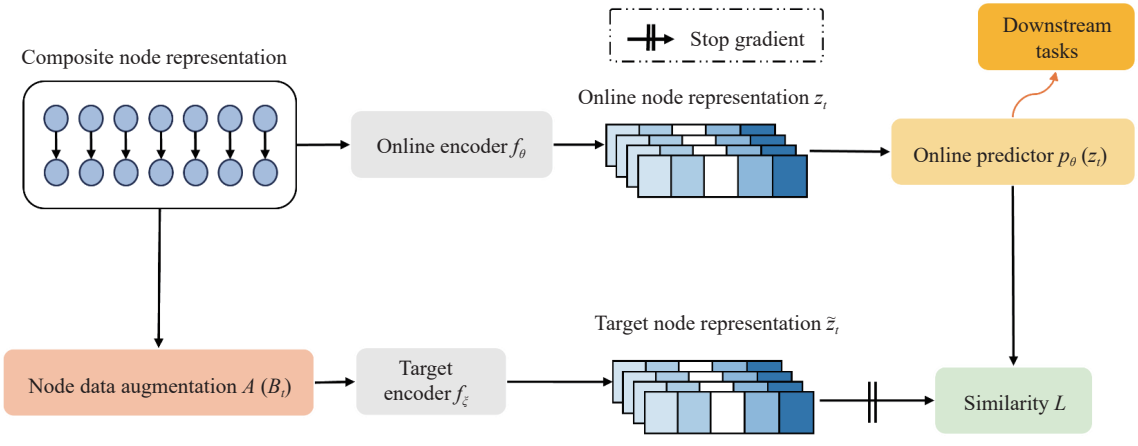

Fig. 1 Overview of self-supervised framework.

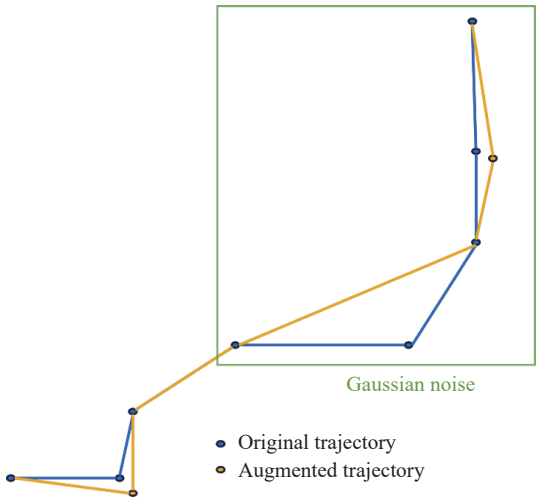

Fig. 2 Gaussian-noised node augmentation.

cess can be formalized as:

$$\begin{aligned}\tilde{x}_t &= x_t + N(0,\sigma^2)\\ \tilde{y}_t &= y_t + N(0,\sigma^2)\end{aligned} \tag{2}$$

where, $\tilde{x}_t,\tilde{y}_t$ are the perturbed coordinates; x_t,y_t which from the origin map, are the original coordinates; $N(0,\sigma^2)$ denotes the Gaussian noise with zero mean and variance σ^2.

The parameter σ controls the magnitude of the noise, effectively simulating varying levels of GPS accuracy. A larger σ introduces more significant perturbations, simulating scenarios with higher GPS error. The perturbation is applied with probability p_{noise} to simulate varying noise rates, To further refine the noise injection process and mirror the intermittent nature of real-world GPS errors, the noise is applied probabilistically. A probability parameter p_{noise} is introduced, determining the likelihood of a given GPS point being perturbed. The noise injection process then becomes:

$$(\tilde{x}_t,\tilde{y}_t) = \begin{cases}(x_t,y_t)+N(0,\sigma^2) & \text{if rand}(0,1) \leqslant p_{\text{noise}}\\ (x_t,y_t) & \text{if rand}(0,1) > p_{\text{noise}}\end{cases} \tag{3}$$

This probabilistic approach allows the model to learn to handle both noisy and relatively clean data points within the same trajectory, improving its adaptability to varying real-world conditions.

This augmented trajectory $\tilde{T}$, consisting of the perturbed coordinates, is then fed as input to the GNTI model. The model is trained to reconstruct the original trajectory T from this noisy input using a self-supervised loss function. This training strategy encourages the model to learn robust representations that capture the underlying trajectory patterns despite the presence of noise. By learning to filter out the injected Gaussian noise, the model becomes more resilient to real-world GPS inaccuracies and produces more accurate and reliable trajectory imputations.

This augmented trajectory, combined with feature extraction and BERT encoding as described below, allows the model to learn robust representations and perform accurate trajectory prediction, even with noisy or missing data. The BERT encoder captures contextual dependencies within the trajectory, and multi-task learning with position and order prediction further enhances the model's ability to reconstruct complete and accurate trajectories. The weighted MSE loss function balances the importance of both position and order accuracy, leading to improved overall imputation performance.

3.4 Multi-faceted feature extraction

The core objective of this module is to construct a comprehensive and informative representation of each individual point within a trajectory. This is achieved by integrating a diverse range of features that capture the spatial, temporal, and edge-related context of each point. The richer the feature representation, the better equipped the model will be to understand the nuances of movement patterns and make accurate imputations. This method is illustrated in Fig. 3.

(1) Spatial features: These features form the foundation of the representation by encoding the precise geographic location of each point. The raw GPS coordinates (x_t,y_t) provide the basis for this encoding. While the direct use of coordinates is possible, further enhancements could involve incorporating map-matching techniques to project the raw GPS data onto the road network, improving accuracy and providing a more structured spatial context. Additionally, features like the bearing or heading of movement at each point could be included to capture directional information.

(2) Temporal features: Capturing the temporal context is crucial for understanding trajectory dynamics. Several key temporal features are incorporated:

Time of day: The time of day is a critical factor influencing movement patterns. Cyclical encoding is employed to represent the time of day, effectively capturing its periodic nature. This allows the model to differentiate between various periods like rush hour, midday, and nighttime, each exhibiting distinct traffic characteristics.

Day of the week: Movement patterns often vary significantly between weekdays and weekends. One-hot encoding is used to represent the day of the week, allowing the model to learn specific behaviors associated with each day.

Other temporal information: The flexibility of the framework allows for the inclusion of additional temporal features relevant to the specific application. For instance, holidays, special events, or even weather conditions could be incorporated to further refine the temporal context.

(3) Edge-type features: These features provide valuable context about the environment surrounding each trajectory point, specifically focusing on the road network and points of interest (POIs).

Road category: The type of road on which a point lies significantly influences movement characteristics. Distinguishing between highways, arterial roads, residential streets, and other road types allows the model to learn appropriate speed profiles and movement constraints.

POI information: The presence of or proximity to POIs can of-

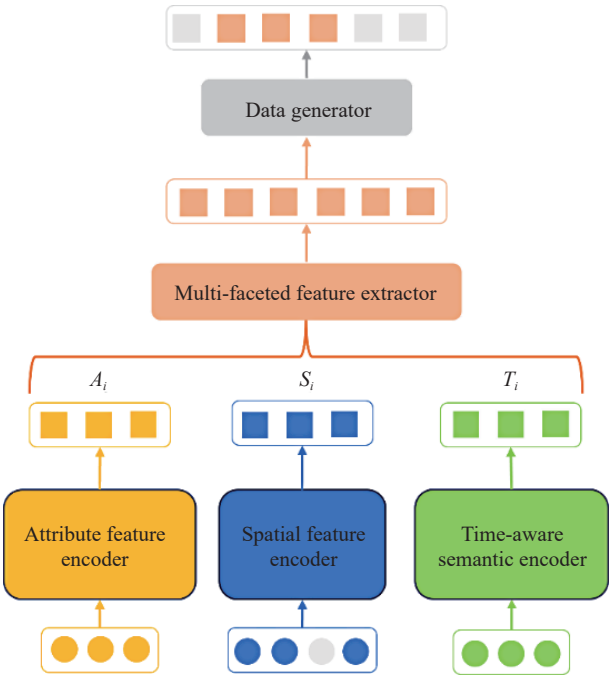

Fig. 3 Multi-faceted feature extractor

fer insights into the purpose of movement. Features could represent the distance to various POI categories (e.g., restaurants, shopping malls, gas stations) or indicate whether a point falls within a specific POI's area of influence.

These diverse spatial, temporal, and edge-type features are concatenated into a single feature vector F_t for each trajectory point t. This concatenation effectively combines the multi-faceted information into a single representation. A linear projection, represented by P_{L}, then transforms this concatenated vector into a dense, lower-dimensional embedding $F_t^{'}$:

$$F_t^{'} = P_{\mathrm{L}}(F_t) \tag{4}$$

This embedding process serves to reduce dimensionality, improve computational efficiency, and potentially capture non-linear relationships between the features.

3.5 Trajectory encoding with BERT

This module leverages the strengths of the BERT [8] architecture, a powerful transformer-based model pre-trained on a massive text corpus. Its ability to capture long-range dependencies and complex relationships is adapted here to understand the contextual dependencies between trajectory points. This contextual awareness is paramount for discerning the overall flow and coherence of movement within a trajectory, especially in tasks like imputation where understanding the broader context helps infer missing information. The trajectory imputation module is illustrated in Fig. 4.

(1) Positional encoding: While the multi-faceted feature extraction module provides a rich representation of individual points, capturing their spatial, temporal, and edge-related attributes, it doesn't inherently encode the sequential nature of the trajectory. This sequential information, the order in which the points occur, is crucial for understanding the movement. Therefore, learnable positional encodings are added to the feature embeddings $F_{\mathrm{t}}^{'}$ (output from the feature extraction module after the linear projection) to explicitly represent the order of the points within the trajectory:

$$E_t = F_t^{'} + P(t) \tag{5}$$

where, $P(t)$ is the positional encoding for the t-th point. These encodings are learned during the training process and provide a way for the BERT model to understand the temporal relationships between points. This is crucial because BERT, by design, treats input sequences as unordered sets, and without positional encodings, it would lose the vital information about the order of the points in the trajectory.

(2) BERT encoding: The sequence of positionally encoded feature embeddings $E_1, E_2, \cdots, E_n(t)$ is then fed as input to a multi-layer BERT encoder. The BERT architecture, renowned for its ability to capture long-range dependencies and complex relationships in sequential data, processes the entire trajectory sequence. Unlike recurrent neural networks that process sequences step-by-step, BERT processes the entire sequence simultaneously, allowing it to understand the relationships between distant points effectively. This allows the model to learn intricate patterns of movement and understand how each point relates to the others within the overall trajectory:

$$H = \Phi(E_1, E_2, \cdots, E_n) \tag{6}$$

where, H is the sequence of contextualized embeddings produced by the BERT encoder. These embeddings are not just representations of individual points; they are now enriched with contextual information from the entire trajectory. This means each embedding in H now reflects not only the spatial, temporal, and edge-type information of its corresponding point but also the influence of the preceding and succeeding points in the trajectory. This contextual awareness, provided by BERT, is what makes this representation particularly powerful for downstream tasks like trajectory imputation, prediction, and analysis. The BERT encoder's ability to capture long-range dependencies is especially valuable for imputation, as it allows the model to infer missing information based on the broader context of the trajectory, even if there are large gaps in the observed data.

3.6 Prediction and loss function

The final hidden representations $\boldsymbol{H} = \{\boldsymbol{h}_1, \cdots, \boldsymbol{h}_n\}$ are passed through a simple MLP decoder to predict the corrected position of each point:

$$\hat{x}_t = \psi(\boldsymbol{h}_t) \tag{7}$$

We use a mean squared error loss between the predicted and true GPS coordinates:

$$L_{\mathrm{impute}} = \frac{1}{N}\sum_{t=1}^{N} \| \hat{x}_t - x_t \|^2 \tag{8}$$

This single-task objective allows the model to learn accurate spatial representations while avoiding the complexity of multi-task or contrastive learning.

4 Experimental

4.1 Datasets

To evaluate the effectiveness of the proposed GNTI framework, we conduct experiments on two large-scale real-world trajectory datasets: Chengdu Taxi Trajectories and Porto Taxi Trajectories.

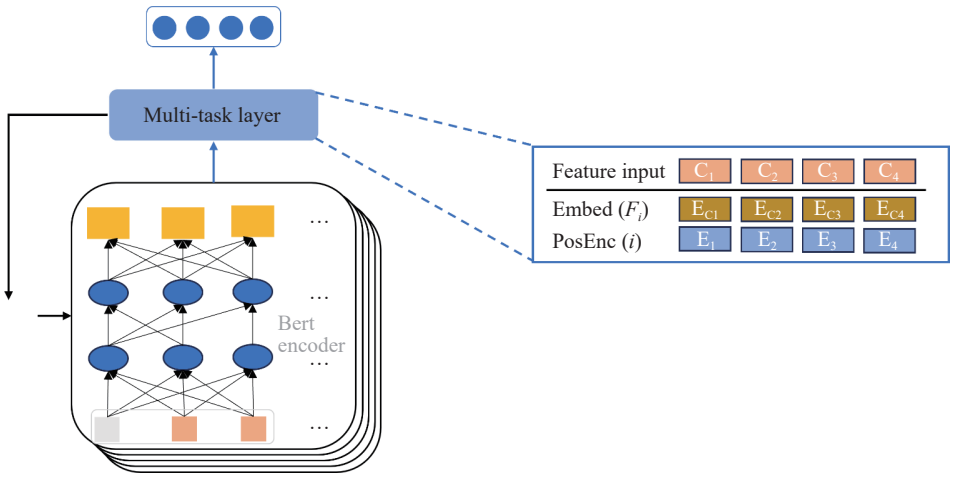

Fig. 4 Trajectory imputation module with BERT

These datasets contain GPS records from taxis operating in urban environments, providing a rich source of real-world trajectory data with diverse spatial and temporal characteristics.

● Chengdu Dataset: This dataset consists of approximately 1.4 billion GPS records collected from over 1,400 taxis in Chengdu, China, spanning from August 3 to 30, 2014. The dataset includes latitude, longitude, timestamp, and road segment information.

● Porto Dataset: This dataset contains one year of GPS trajectories from 442 taxis operating in Porto, Portugal, collected between July 1, 2013, and June 30, 2014. The dataset provides detailed trajectory information, including GPS coordinates, timestamps, and taxi trip identifiers.

Both datasets are publicly available and can be accessed through their respective repositories. The datasets will be shared upon request and will be made available prior to publication.

4.2 Data preprocessing

To ensure the quality and reliability of the trajectory data used for evaluation, a rigorous preprocessing pipeline is employed. This pipeline consists of several key steps designed to clean, refine, and augment the data, ultimately leading to a more robust and realistic evaluation of the GNTI model's performance.:

(1) Trajectory filtering: Raw GPS data can often contain inaccuracies and inconsistencies. This step focuses on removing or correcting such issues to create a cleaner dataset.

Missing timestamps: Trajectory points lacking timestamps are removed as they disrupt the temporal continuity and make it impossible to accurately assess time-dependent aspects of movement.

Extreme coordinate values: GPS readings can sometimes be erroneous due to various factors like signal interference or multipath effects. Points with extreme coordinate values that fall outside the valid geographic region of interest are identified and removed. This prevents these outliers from skewing the analysis and ensures that the data reflects realistic movement patterns within the defined area.

(2) Map matching: Raw GPS data often exhibits inaccuracies due to factors like GPS drift. Map matching addresses this by aligning the trajectory points with the underlying road network. A map-matching algorithm is used to project each GPS point onto the nearest road segment. This process effectively "snaps" the trajectory points to the road network, correcting for GPS errors and providing a more accurate representation of movement along roads. This is crucial for applications that rely on road network information, such as traffic analysis or route planning.

(3) Trajectory segmentation: Extremely long trajectories can present challenges for both computational efficiency and model training. Segmentation addresses this by dividing long trajectories into shorter, more manageable subsequences.

Time gap-based segmentation: Trajectories are segmented based on significant time gaps between consecutive points. A predefined time threshold is used to identify these gaps, indicating potential breaks in the trajectory, such as stops or changes in mode of transportation. This segmentation ensures that the model is trained and evaluated on consistent and coherent movement segments.

4.3 Baselines

To validate the performance of GNTI, we compare it with several state-of-the-art trajectory imputation methods:

● Top-K: A statistical method that predicts missing trajectory points based on the most frequently occurring road segments.

● Seq2Seq-GRU: A sequence-to-sequence model using GRUs for trajectory recovery.

● Seq2Seq-LSTM: A similar model utilizing LSTM networks instead of GRUs.

●Transformer: A transformer-based model that captures long-range dependencies between trajectory points.

DeepMove [21]: A deep learning-based model designed for human mobility prediction using recurrent neural networks.

AttnMove [22]: An attention-based trajectory recovery model that refines missing trajectory points using historical mobility patterns.

ST-BerImp [16]: A BERT-based trajectory imputation method that incorporates spatial-temporal dependencies.

4.4 Model implementation

The proposed GNTI framework is implemented using PyTorch, with the following architecture and hyperparameter settings:

Feature embedding: Each trajectory point is represented using a 128-dimensional feature vector, incorporating spatial and temporal attributes.

● Transformer encoder: The model consists of 4 transformer layers with 8 attention heads per layer.

● Optimization: We use the Adam optimizer with an initial learning rate of 0.000 1 and batch size of 64.

● Training procedure: The model is trained for 20 epochs with early stopping based on validation performance.

All experiments are conducted on a server with an Intel Xeon CPU, 128 GB RAM, and an NVIDIA Titan X GPU (12 GB VRAM).

4.5 Evaluation metrics

To assess the trajectory imputation performance, we use the following standard evaluation metrics:

● Precision (PPP): Measures the proportion of correctly imputed trajectory points among all predicted points.

● Recall (RRR): Measures the proportion of correctly imputed trajectory points among all ground truth points.

● $F_{1\text{micro}}$: A balanced metric that combines Precision (P) and Recall (R), calculated as:

$$F_{1\text{micro}} = 2 \times \frac{R \times P}{R + P} \tag{9}$$

These metrics provide a comprehensive evaluation of the model' s ability to restore missing trajectory data accurately.

4.6 Ethical considerations

Since this study is based on publicly available trajectory datasets, no personally identifiable information (PII) is used. The datasets have been anonymized to remove any potential privacy concerns. This study does not involve human or animal subjects, and therefore, no ethical approval is required.

4.7 Data and code availability

The datasets used in this study are publicly accessible. The preprocessed versions of the datasets, along with the code for the GNTI framework, will be made available on GitHub upon publication. Readers can replicate and build upon our results using the provided scripts and documentation.

5 Results and discussion

5.1 Performance comparison

To evaluate the effectiveness of GNTI, we compare it against state-of-the-art trajectory imputation methods on the Chengdu and Porto datasets. Table 1 presents the results in terms of Precision, Recall, and $F_{1\text{micro}}$ under different missing data rates.

5.1.1 Quantitative Results

Table 1 shows the trajectory imputation performance of GNTI compared to other models.

Key observations as follows:

● GNTI outperforms all baseline models in both datasets,

Table 1 Performance comparison of GNTI and baseline methods

Dataset	Method	P	R	F_{1micro}
Chengdu	Top-k	76.53	76.49	76.51
	Seq2Seq+GRU	22.91	18.11	20.56
	Seq2Seq+LSTM	24.47	19.23	21.72
	Transformer	87.45	82.42	84.92
	DeepMove	70.10	69.84	69.97
	AttnMove	72.30	72.12	72.20
	MTrajRec	88.48	82.51	85.54
	ST-BerImp	88.92	83.05	85.97
	GNTI	**92.10**	**90.14**	**91.12**
Porto	Top-k	82.66	82.55	82.61
	Seq2Seq+GRU	33.12	27.54	29.78
	Seq2Seq+LSTM	34.52	28.63	31.18
	Transformer	87.48	89.97	90.51
	DeepMove	75.21	74.92	75.04
	AttnMove	77.33	77.13	77.21
	MTrajRec	88.49	89.99	90.48
	ST-BerImp	88.48	89.97	90.51
	GNTI	**93.64**	**93.08**	**93.68**

achieving the highest F_{1micro}.

● Traditional Seq2Seq-based models (GRU, LSTM) perform poorly, indicating their limitations in handling noisy trajectories.

● Transformer-based models (Transformer, ST-BerImp) perform well, but their reliance on additional training objectives makes them less efficient than GNTI.

● GNTI achieves a 3%-5% improvement over ST-BerImp, demonstrating the effectiveness of Gaussian Noise augmentation in improving trajectory imputation accuracy.

5.2 Impact of Gaussian noise augmentation

To further analyze the effect of Gaussian noise augmentation, we conduct an ablation study comparing GNTI (with noise) vs. No Augmentation. The result is illustrated in Table 2.

Table 2 Performance of different augmentation type

Dataset	Augmentation type	F_{1micro} (%)
Chengdu	None	84.07
	Dropout	84.07
	Gaussian	88.63
	Gaussian + Dropout	93.14
Porto	None	90.52
	Dropout	92.86
	Gaussian	93.40
	Gaussian + Dropout	94.50

5.2.1 Ablation study on noise augmentation

Key findings as follows:

● Adding Gaussian noise improves performance by 5% in F_{1micro}, showing that training with noise-perturbed trajectories enhances robustness.

● Without augmentation, the model struggles to generalize when handling real-world GPS errors.

● Gaussian noise augmentation effectively simulates GPS errors, allowing the model to learn more robust trajectory representations.

5.3 Robustness under different noise levels

We further evaluate how GNTI performs under different levels of Gaussian noise. Fig. 1 illustrates the F_{1micro} as the noise standard deviation σ increases.

● Performance initially increases as noise is introduced, indicating that noise augmentation helps the model generalize better.

● Excessive noise (σ>0.05) degrades performance, suggesting that extreme perturbations can introduce unrealistic trajectory distortions.

● Optimal noise level is around σ=0.03, balancing augmentation benefits and trajectory realism.

● GNTI-generated trajectories closely follow the ground truth, demonstrating its ability to handle GPS noise effectively.

5.4 Discussion and practical implications

Based on the experimental findings, we highlight several key takeaways:

5.4.1 Advantages of GNTI

● Improved accuracy: GNTI consistently outperforms baseline models across multiple datasets and noise levels.

● Robustness to GPS errors: The Gaussian noise based augmentation strategy enhances the model's ability to recover missing trajectory points.

● Efficient training: Unlike contrastive learning approaches [7, 23] that require negative samples, GNTI achieves high performance with a simple self-supervised training objective.

5.4.2 Limitations and future work

Despite its advantages, GNTI has some limitations:

● Performance degradation under extreme noise: While moderate noise improves generalization, excessive noise can degrade trajectory recovery.

● Scalability to dense urban environments: Future work should explore extensions of GNTI for more complex road networks.

● Integration with real-time applications: Deploying GNTI in live navigation systems requires further optimization for low-latency inference.

6 Conclusions

In this paper, we proposed GNTI, a self-supervised learning framework designed to improve trajectory imputation accuracy by introducing Gaussian noise augmentation. Unlike traditional trajectory recovery methods that rely on large labeled datasets or complex multi-task learning strategies, GNTI leverages simple yet effective noise-based augmentation to enhance the model's robustness against real-world GPS errors.

Through extensive experiments on two large-scale real-world datasets (Chengdu and Porto), we demonstrated that GNTI outperforms state-of-the-art trajectory imputation methods in terms of P, R, and F_{1micro}. The key findings of our study include:

● Gaussian noise augmentation significantly improves trajectory recovery, with a 5% increase in F_{1micro} compared to models trained without augmentation.

● GNTI achieves state-of-the-art performance, surpassing transformer-based baselines while maintaining a simpler training framework.

● The model remains robust under different noise levels, with an optimal augmentation setting of σ=0.03, effectively balancing noise injection and trajectory realism.

● Beyond improving trajectory imputation accuracy, this work highlights the importance of noise-based augmentation in spatial-temporal learning tasks. By explicitly simulating real-world GPS errors during training, GNTI enables better generalization and adaptability to noisy trajectory data.

Acknowledgements

The authors would like to thank the anonymous reviewers for their constructive comments and suggestions that helped improve the quality of this manuscript.

Funding

This work was supported by the Science and Technology Project of Beijing Municipal Transportation Industry (Grant No. BJJTW-2023-ZHJT-11).

Declaration of competing interest

The author has no competing interests to declare that are relevant to the content of this article.

Author contribution statement

Siqi Liu: Conceptualization, methodology, software, writing original draft. Penghao Zhao: Data curation, validation, formal analysis. Lei Dong & Na Dong: Investigation, resources. Liyuan Ding & Guangshi Pei: Supervision, project administration, writing review & editing.

Data availability statement

The data are not publicly available due to privacy or ethical restrictions.

Use of AI statement

During the preparation of this work, the authors used ChatGPT in order to improve language and readability. After using this tool/service, the authors reviewed and edited the content as needed and takes full responsibility for the content of the publication.

References

[1] Li WH, Chen YY, Pan YY, et al. An improved spatio-temporal network traffic flow prediction method based on impedance matrix. Journal of Highway and Transportation Research and Development (English Edition), 2024, 18(2): 67–75.

[2] Huang W, Lu SF, Li HC, et al. The study of cooperative merging algorithm based on the assignment model for connected and automated vehicles. Journal of Highway and Transportation Research and Development (English Edition), 2024, 18(3): 61–67.

[3] Wang JY, Wu, N, Lu, XX, et al. Deep trajectory recovery with fine-grained calibration using Kalman filter. IEEE Transactions on Knowledge and Data Engineering, 2021, 33(3): 921–934.

[4] Wu H, Chen ZY, Sun WW, et al. Modeling trajectories with recurrent neural networks. Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence. Melbourne: International Joint Conferences on Artificial Intelligence Organization, 2017. https://doi.org/10.24963/ijcai.2017/430.

[5] Zhang HJ. Novel algorithm of traffic signals adaptive control on urban roads intersections. Journal of Highway and Transportation Research and Development (English Edition), 2025, 19(1): 25–28.

[6] Chen YL, Cong G, Anda C. TERI: An effective framework for trajectory recovery with irregular time intervals. Proceedings of the VLDB Endowment, 2023, 17(3): 414–426.

[7] Yang L, Guo RJ. Short term traffic flow prediction and timing optimization at signalized intersections based on SG-LSTM and particle swarm optimization. Journal of Highway and Transportation Research and Development (English Edition), 2024, 18(4): 34–41.

[8] Zheng K, Zheng Y, Xie X, et al. Reducing uncertainty of low-sampling-rate trajectories. 2012 IEEE 28th International Conference on Data Engineering. Arlington: IEEE, 2012: 1144–1155. https://doi.org/10.1109/icde.2012.42.

[9] Ren DF, Li YP, Ma ZL. Test and analysis on the errors of GPS observation in mining field. Procedia Earth and Planetary Science, 2009, 1(1): 1233–1236.

[10] Shang Q, Yang ZS, Gao S, et al. An imputation method for missing traffic data based on FCM optimized by PSO-SVR. Journal of Advanced Transportation, 2018, 2018: 2935248.

[11] Shi ZY, Xu M, Pan Q, et al. LSTM-based flight trajectory prediction. 2018 International Joint Conference on Neural Networks (IJCNN). Rio de Janeiro: IEEE, 2018. https://doi.org/10.1109/ijcnn.2018.8489734.

[12] Giuliari F, Hasan I, Cristani M, et al. Transformer networks for trajectory forecasting. 2020 25th International Conference on Pattern Recognition (ICPR). Milan: IEEE, 2021: 10335–10342.

[13] Musleh M, Mokbel MF. Kamel: A scalable BERT-based system for trajectory imputation. Proceedings of the VLDB Endowment, 2023, 17(3): 525–538.

[14] Chen YQ, Zhang HY, Sun WW, et al. RNTRajRec: Road network enhanced trajectory recovery with spatial-temporal transformer. 2023 IEEE 39th International Conference on Data Engineering (ICDE). Anaheim: IEEE, 2023: 829–842.

[15] Ta XX, Liao TX, Zhang QM, et al. Context aware trajectory imputation via spatio-temporal representation learning. In 2023 IEEE Smart World Congress (SWC). Portsmouth: IEEE, 2023. https://doi.org/10.1109/swc57546.2023.10449332.

[16] Xia T, Qi YH, Feng J, et al. AttnMove: History enhanced trajectory recovery via attentional network. Proceedings of the AAAI Conference on Artificial Intelligence, 2021, 35(5): 4494–4502.

[17] Zhang Z, Yang Q. A survey on multi-task learning. IEEE Transactions on Knowledge and Data Engineering, 2022, 34(12): 5586–5609.

[18] Ren HM, Ruan SJ, Li YH, et al. MTrajRec: Map-constrained trajectory recovery via Seq2Seq multi-task learning. Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. Virtual Event Singapore. ACM, 2021: 1410–1419.

[19] Grill JB, Strub F, Altché F, et al. Bootstrap your own latent: A new approach to self-supervised learning. Proceedings of the 34th International Conference on Neural Information Processing Systems. New York: Curran Associates Inc., 2020: 21271–21284.

[20] Yan BQ, Zhao G, Song L, et al. PreCLN: Pretrained-based contrastive learning network for vehicle trajectory prediction. World Wide Web, 2023, 26(4): 1853–1875.

[21] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. Proceedings of the 31st International Conference on Neural Information Processing Systems. New York: Curran Associates Inc., 2017: 6000−6010.

[22] Devlin J, Chang MW, Lee, K, et al. BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis: Association for Computational Linguistics. 2019: 4171−4186.

[23] Feng J, Li Y, Zhang C, et al. DeepMove: Predicting human mobility with attentional recurrent networks. Proceedings of the 2018 World Wide Web Conference. Lyon: ACM, 2018: 1459–1468. https://doi.org/10.1145/3178876.3186058.