RESEARCH ARTICLE

A Multi-Scale Traffic-State Gated Attention Network for Bus Arrival Time Prediction

Ruping Liu[1], **Xiaoqi Yin**[1,*], **Jing Zhou**[1], **Chenxi Ma**[1] **and Chen Li**[1]

[1] School of Electronic Science and Technology, Huai'an University, Huai'an 223003, China

Abstract

Accurate bus arrival time prediction supports passenger information services and transit dispatching, but irregular sampling and time-varying traffic conditions make the task strongly nonlinear. This paper proposes a Spatial Attention Multi-Scale Traffic-State Gated Long Short-Term Memory (SA-MS-TG-LSTM) network. Seven trajectory features are constructed from Global Positioning System (GPS) records, including acceleration, congestion index, and speed volatility. Scale-preserving Spatial Feature Attention (SFA) first reweights the feature dimensions, and three Traffic-State Gated Long Short-Term Memory (TG-LSTM) branches subsequently encode the latest 5, 10, and 15 time steps. Their outputs are concatenated and mapped to the remaining travel time. Experiments are conducted on 8,834 GPS records from 293 trips on a single bus route (Route 86). Under a controlled setting in which all recurrent models use a hidden size of 16, one recurrent layer, and at most 50 training epochs, SA-MS-TG-LSTM achieves the best aggregate performance, with a root mean square error (RMSE) of 1.056 min, mean absolute error (MAE) of 0.762 min, mean absolute percentage error (MAPE) of 12.06%, and coefficient of determination (R^2) of 0.9631. Compared with the Convolutional Neural Network–Gated Recurrent Unit (CNN-GRU), it reduces RMSE, MAE, and MAPE by 6.30%, 5.57%, and 11.26%, respectively. Trip-level analysis further confirms statistically significant MAE improvements over a standard Long Short-Term Memory (LSTM) model and TG-LSTM. These results demonstrate the effectiveness of the proposed method on the evaluated route, while broader multi-route validation remains an important direction for future work.

Keywords: bus arrival time prediction, intelligent transportation, LSTM, traffic-state gating, multi-scale temporal modeling, feature attention.

1 Introduction

Bus arrival time prediction estimates the remaining travel time from a vehicle's current location to a target stop. It supports passenger information systems, operational dispatching, and service reliability analysis. Automatic vehicle location and passenger-counting data have therefore been widely used for real-time transit arrival and departure

Academic Editor:
Muhammad Awais

Submitted: 15 July 2026
Revised: 25 August 2026
Accepted: 26 August 2026
Published: 29 September 2026

Vol. 3, **No.** 3, 2026.
10.62762/TIS.2026.906270

***Corresponding author:**
✉ Xiaoqi Yin
hy_xuebao2009@126.com

Citation
Liu, R., Yin, X., Zhou, J., Ma, C., & Li, C. (2026). A Multi-Scale Traffic-State Gated Attention Network for Bus Arrival Time Prediction. *ICCK Transactions on Intelligent Systematics*, 3(3), 196–208.

© 2026 ICCK (Institute of Central Computation and Knowledge)

prediction [1–3]. In practice, however, bus travel time is affected by irregular sampling, stop-and-go behavior, traffic congestion, signal delay, and route-specific disturbances. These factors make the prediction problem nonlinear and strongly time dependent.

Traditional approaches such as historical averages, regression models, time-series forecasting, and artificial neural networks are easy to deploy, but they are weak at representing dynamic traffic variation [4–7]. Conventional recurrent networks can also be difficult to train over long temporal spans [8]. Long short-term memory (LSTM) networks [9] improve temporal modeling by learning dependencies from previous trajectory observations. Nevertheless, a standard LSTM still has three important limitations for bus arrival time prediction. First, it usually treats traffic-state variables only as ordinary input features, so congestion or speed fluctuation cannot directly regulate memory updating. Second, many models operate on a single window length, which limits their ability to jointly represent short-term vehicle maneuvers and longer route-level trends. Third, treating all input dimensions equally may weaken key variables under complex traffic conditions.

This paper presents a Spatial Attention Multi-Scale Traffic-State Gated Long Short-Term Memory (SA-MS-TG-LSTM) network. Unlike attention-based recurrent models that only reweight their inputs, the proposed architecture transforms three explicit traffic-state variables and injects them into the input, forget, output, and candidate-memory computations. It also encodes three nested temporal windows in parallel and fuses the resulting representations by concatenation. The main contributions are as follows:

1. A traffic-state feature construction scheme is developed from raw bus GPS trajectories, including acceleration, congestion index, and speed volatility.
2. A traffic-state gated LSTM unit is designed so that traffic-state variables directly modulate input, forget, output, and candidate memory gates.
3. A multi-scale temporal architecture is used to model short-, medium-, and long-range traffic dynamics in parallel.
4. A feature-dimension attention module is introduced to adaptively emphasize important variables and suppress less relevant information.

The remainder of this paper is organized as follows. Section 2 reviews bus arrival prediction and recent deep spatiotemporal approaches. Section 3 presents the proposed methodology. Section 4 reports the experimental setting and results, Section 5 discusses the study limitations, and Section 6 concludes the paper.

2 Related Work

Bus arrival time prediction has been studied with statistical and machine-learning methods. A recent comprehensive review organizes this literature around data acquisition, travel-time patterns, route discretization, implementation strategy, and performance evaluation [10]. Early systems based on Automatic Vehicle Location (AVL), Automatic Passenger Counter (APC), and GPS data used historical travel times, dynamic regression, filtering, and artificial neural networks to estimate downstream arrival or departure times [1–3, 5, 6]. Multi-route interactions were subsequently considered in stop-level prediction [7], while bus-specific time-series models remained useful as interpretable references [4]. Such methods are generally inexpensive and transparent, but fixed functional assumptions can limit their response to nonlinear congestion and rapidly changing operations. Recent non-recurrent work has also examined multiple extreme-learning machines with dynamic, road, trajectory, signal, weather, and passenger variables [11].

Recurrent deep networks provide greater capacity for sequential representation. The difficulty of learning long-term dependencies with conventional gradient-based recurrent networks was established in early work [8]. Long short-term memory (LSTM) networks [9] and gated recurrent units (GRUs) [12] address this difficulty through learned gates, and convolutional LSTM models have captured correlations across multiple bus travel-time outputs [13]. Attention mechanisms provide a general means of selecting informative input or sequence components [14]. More recent bus-specific studies have optimized LSTM hyperparameters [15], used dual-stage attention with dynamic and static factors [16], or combined LightGBM feature selection with LSTM temporal modeling [17]. These approaches show the value of nonlinear temporal modeling and feature selection, but attention-based recurrent models typically reweight inputs or hidden states rather than allowing a compact traffic-state vector to participate explicitly in every recurrent gate.

Bus-specific spatial and Transformer architectures

provide complementary alternatives. Lee and Yoon encode short-term inbound and outbound bus flows at individual stations and combine the resulting flow-centrality representation with recurrent predictors [18]. MetaSTNN combines graph and temporal convolution with meta-learning to represent segment-dependent bus dynamics [19], while FEN-MRMGCN models multiple bus-network relations and incorporates passenger flow, congestion, and weather information [20]. These bus-specific spatial models represent stop, route, and passenger-related structure more directly than generic road-sensor traffic-flow forecasting, but they require reliable network context or synchronized multi-route observations. BAT-Transformer uses self-attention and positional encoding [21] to learn longer dependencies from large-scale bus operation records [22]. Transformers offer parallel sequence modeling, although their data and computational requirements can be disproportionate for a single-route dataset.

The present study addresses a narrower setting in which only trajectory-derived variables are available. It combines feature-dimension attention, three nested temporal windows, and a traffic-state gated recurrent unit. The distinction from conventional attention LSTMs is that acceleration, congestion index, and speed volatility are first transformed and then included in all four gate computations. Unlike the temporal convolution and self-attention alternatives discussed above, which learn their temporal receptive fields implicitly, the proposed model makes the three historical ranges explicit through fixed 5-, 10-, and 15-step recurrent windows and concatenates their outputs. The distinction from graph-based methods is that no road-network adjacency matrix or synchronized multi-route data are required. External variables such as weather, signal timing, and passenger demand remain complementary inputs for future multi-route extensions rather than components evaluated in the present experiment.

3 Methodology

3.1 Overall Architecture

The proposed model is named SA-MS-TG-LSTM. As shown in Figure 1, the model takes a historical trajectory sequence $\boldsymbol{X}$ as input and predicts the remaining travel time to the target station. The input sequence is first processed by a Spatial Feature Attention module. The attended sequence is then fed into a multi-scale TG-LSTM module composed of three parallel temporal branches. These branches model short-term, mid-term, and long-term temporal dependencies with window lengths of 5, 10, and 15 time steps, respectively. Traffic state features, including acceleration, congestion index, and speed volatility, are used as auxiliary information for the TG-LSTM branches. The outputs of the three branches are combined by feature concatenation and then passed through a fully connected layer to obtain the final remaining travel time prediction.

3.2 Input Representation and Spatial Feature Attention

For each GPS record, the implementation uses a seven-dimensional feature vector

$$\boldsymbol{x}_t = [v_t, \theta_t, d_t, \Delta\tau_t, a_t, \rho_t, \phi_t]^\top, \tag{1}$$

where $\boldsymbol{x}_t \in \mathbb{R}^7$, v_t is speed (km/h), θ_t is heading angle (degrees), d_t is distance to the station (m), $\Delta\tau_t$ is the recorded time difference (min), a_t is acceleration (km/h/s), ρ_t is a dimensionless congestion index, and ϕ_t is speed volatility (km/h). This notation distinguishes station distance d_t from acceleration a_t.

The three traffic-state variables are calculated separately within each trip. Let s_t denote the timestamp in seconds. For two consecutive valid records, the elapsed time and finite-difference acceleration are

$$\delta_t = \max(s_{t+1} - s_t, 10^{-6}), \tag{2}$$

$$a_t = \frac{v_{t+1} - v_t}{\delta_t}. \tag{3}$$

The last record of a trip reuses the latest available acceleration. In the implementation, a single-record trip retains zero for all three derived traffic-state variables and cannot generate a 15-step sample. For trips containing at least two records, the congestion index is normalized by the maximum observed speed of trip q:

$$\rho_t = 1 - \frac{v_t}{v_{\max}^{(q)}}, \qquad v_{\max}^{(q)} = \max_{j \in q} v_j, \tag{4}$$

where a fallback reference speed of 40 km/h is used only when all recorded speeds in a trip are non-positive. Speed volatility is the sample standard deviation over the current and two preceding speed records:

$$\phi_t = \begin{cases} \sqrt{\dfrac{1}{n_t - 1} \displaystyle\sum_{j \in \mathcal{W}_t} (v_j - \bar{v}_t)^2}, & n_t > 1, \\ 0, & n_t = 1, \end{cases} \tag{5}$$

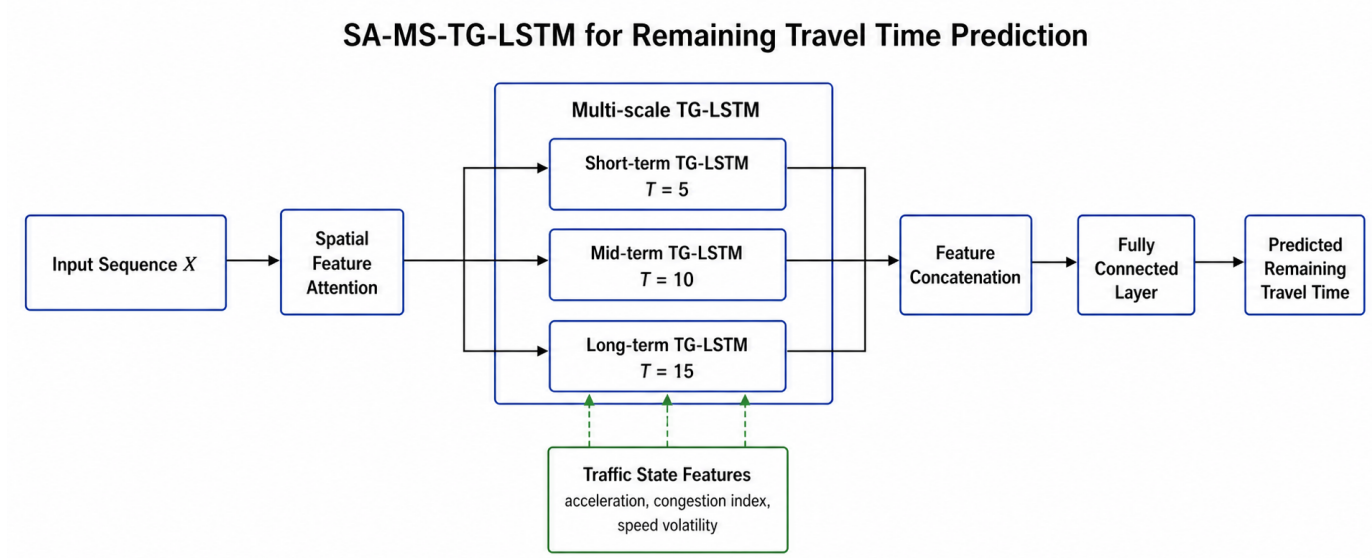


Figure 1. Overall architecture of SA-MS-TG-LSTM.

where $\mathcal{W}_t = \{\max(1, t-2), \ldots, t\}$, $n_t = |\mathcal{W}_t|$, and $\bar{v}_t$ is the corresponding mean speed. Missing values in the four original numerical features are forward-filled and then backward-filled within each trip before these variables are calculated.

A supervised sample is formed from a 15-step historical trajectory segment:

$$\boldsymbol{X}_k = [\boldsymbol{x}_k, \boldsymbol{x}_{k+1}, \ldots, \boldsymbol{x}_{k+14}], \qquad y_k = r_{k+15}, \tag{6}$$

where r_{k+15} is the remaining travel time at the next recorded point after the input window.

In the architecture, Spatial Feature Attention is applied before temporal modeling. For each time step, the attention module computes feature-dimension weights as

$$\boldsymbol{e}_t = \boldsymbol{W}_{a2}\,\mathrm{Dropout}\left(\tanh(\boldsymbol{W}_{a1}\boldsymbol{x}_t + \boldsymbol{b}_{a1})\right) + \boldsymbol{b}_{a2}, \tag{7}$$

$$\boldsymbol{\alpha}_t = \mathrm{softmax}(\boldsymbol{e}_t), \tag{8}$$

$$\boldsymbol{x}_t^a = F\boldsymbol{x}_t \odot \boldsymbol{\alpha}_t, \tag{9}$$

where $F = 7$ is the feature dimension, $\boldsymbol{W}_{a1} \in \mathbb{R}^{16\times 7}$, $\boldsymbol{b}_{a1} \in \mathbb{R}^{16}$, $\boldsymbol{W}_{a2} \in \mathbb{R}^{7\times 16}$, and $\boldsymbol{b}_{a2} \in \mathbb{R}^7$. Multiplication by F preserves the input scale: uniform attention gives $F\alpha_{t,j} = 1$ for every feature, while nonuniform weights still emphasize or suppress individual dimensions. The softmax operation is performed over the seven feature dimensions, and the attention multilayer perceptron uses a dropout rate of 0.1. The attended sequence

$$\boldsymbol{X}_k^a = [\boldsymbol{x}_k^a, \boldsymbol{x}_{k+1}^a, \ldots, \boldsymbol{x}_{k+14}^a] \tag{10}$$

is then used by the multi-scale TG-LSTM branches.

3.3 TG-LSTM Unit

The internal mechanism of a TG-LSTM unit is illustrated in Figure 2. At time step t, the unit receives the input vector from the current recurrent layer, the previous hidden state $\boldsymbol{h}_{t-1}$, and the previous cell state $\boldsymbol{c}_{t-1}$. The traffic state vector is selected from the branch input and contains acceleration, congestion index, and speed volatility:

$$\boldsymbol{T}_t = \begin{bmatrix} a_t \\ \rho_t \\ \phi_t \end{bmatrix} \in \mathbb{R}^3. \tag{11}$$

The traffic state vector is first transformed by a Traffic State multilayer perceptron (MLP). In the implementation, this MLP maps the three traffic-state variables through an eight-dimensional hidden layer and a rectified linear unit (ReLU) activation to a hidden-size traffic representation:

$$\boldsymbol{z}_t = \boldsymbol{W}_{T2}\,\mathrm{Dropout}\left(\mathrm{ReLU}(\boldsymbol{W}_{T1}\boldsymbol{T}_t + \boldsymbol{b}_{T1})\right) + \boldsymbol{b}_{T2}, \tag{12}$$

where $\boldsymbol{W}_{T1} \in \mathbb{R}^{8\times 3}$, $\boldsymbol{b}_{T1} \in \mathbb{R}^8$, $\boldsymbol{W}_{T2} \in \mathbb{R}^{H\times 8}$, $\boldsymbol{b}_{T2} \in \mathbb{R}^H$, $\boldsymbol{z}_t \in \mathbb{R}^H$, H is the TG-LSTM hidden dimension, and the MLP dropout rate is 0.1. The unit diagram indicates that the transformed traffic features, the current input vector, and the previous hidden state jointly determine the gate values. In the code, this dependency is implemented through separate affine projections for the three sources.

Following the gate computation shown in Figure 2, the input gate, forget gate, output gate, and candidate

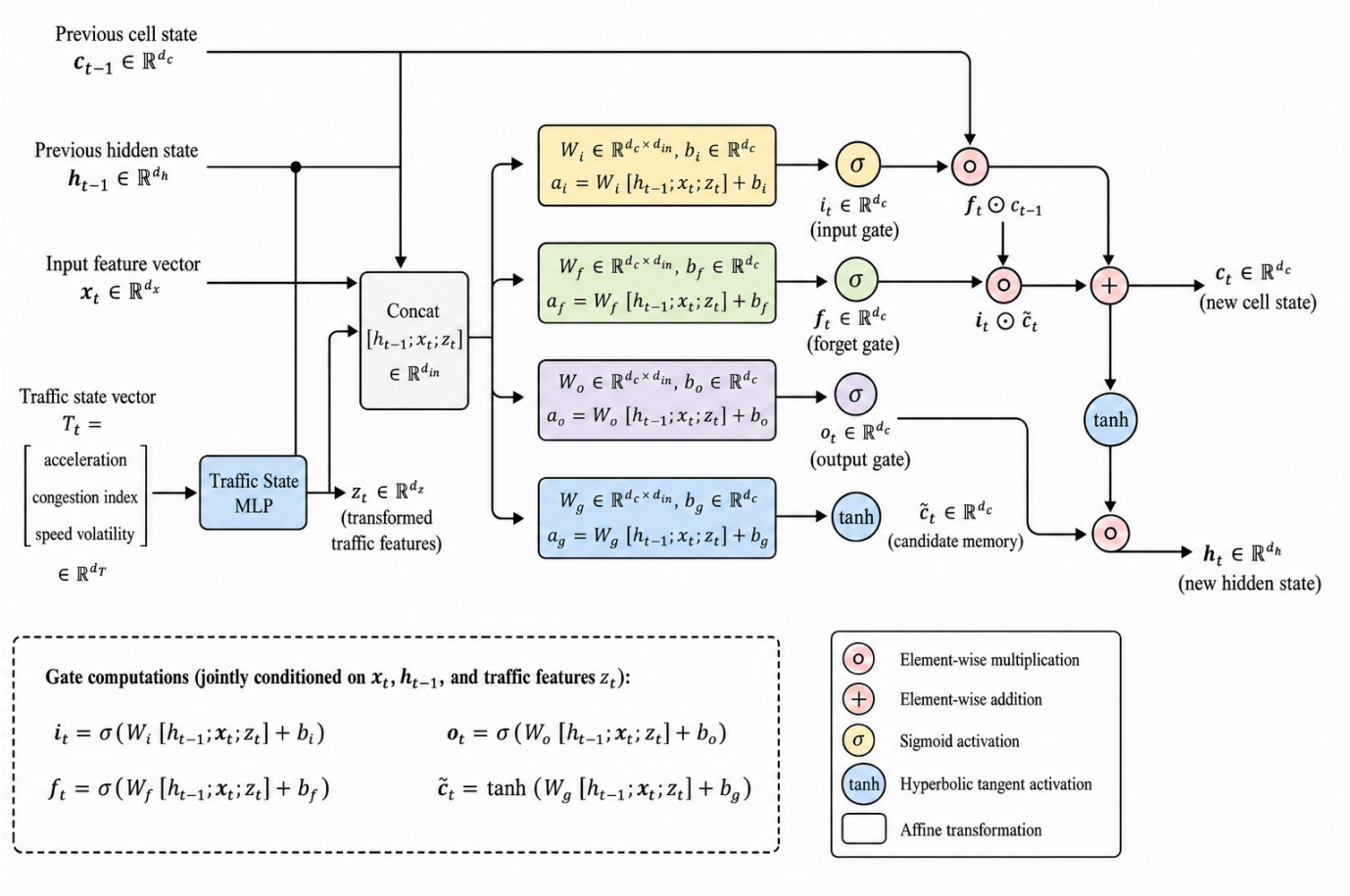


Figure 2. Internal structure of the TG-LSTM unit.

memory are computed as

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i \boldsymbol{x}_t + \boldsymbol{U}_i \boldsymbol{h}_{t-1} + \boldsymbol{V}_i \boldsymbol{z}_t + \boldsymbol{b}_i), \tag{13}$$

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f \boldsymbol{x}_t + \boldsymbol{U}_f \boldsymbol{h}_{t-1} + \boldsymbol{V}_f \boldsymbol{z}_t + \boldsymbol{b}_f), \tag{14}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o \boldsymbol{x}_t + \boldsymbol{U}_o \boldsymbol{h}_{t-1} + \boldsymbol{V}_o \boldsymbol{z}_t + \boldsymbol{b}_o), \tag{15}$$

$$\tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{W}_c \boldsymbol{x}_t + \boldsymbol{U}_c \boldsymbol{h}_{t-1} + \boldsymbol{V}_c \boldsymbol{z}_t + \boldsymbol{b}_c). \tag{16}$$

The cell state and hidden state are then updated by

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t, \tag{17}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t), \tag{18}$$

where $\sigma(\cdot)$ is the sigmoid activation and $\odot$ denotes element-wise multiplication. This formulation follows the LSTM update logic in the unit diagram while explicitly conditioning the gates on the transformed traffic features.

For recurrent layer ℓ, let its input dimension be D_ℓ, where $D_1 = 7$ and $D_2 = H$. For each gate, $\boldsymbol{W}_* \in \mathbb{R}^{H \times D_\ell}$, $\boldsymbol{U}_*, \boldsymbol{V}_* \in \mathbb{R}^{H \times H}$, and $\boldsymbol{b}_* \in \mathbb{R}^H$. The hidden and cell states are also in $\mathbb{R}^H$. These dimensions make the trajectory input, recurrent state, and transformed traffic state compatible through separate affine projections rather than through an implicit addition of unlike variables.

3.4 Multi-Scale Temporal Window Construction

The multi-scale temporal construction is shown in Figure 3. For each attended 15-step sequence, three windows are extracted. The short-term branch uses the latest 5 time steps, the mid-term branch uses the latest 10 time steps, and the long-term branch uses the full 15-step sequence:

$$\boldsymbol{X}_k^{short} = [\boldsymbol{x}_{k+10}^a, \boldsymbol{x}_{k+11}^a, \ldots, \boldsymbol{x}_{k+14}^a], \tag{19}$$

$$\boldsymbol{X}_k^{mid} = [\boldsymbol{x}_{k+5}^a, \boldsymbol{x}_{k+6}^a, \ldots, \boldsymbol{x}_{k+14}^a], \tag{20}$$

$$\boldsymbol{X}_k^{long} = [\boldsymbol{x}_k^a, \boldsymbol{x}_{k+1}^a, \ldots, \boldsymbol{x}_{k+14}^a]. \tag{21}$$

Each window is encoded by a corresponding TG-LSTM branch. The three branches are designed to capture temporal dependencies at different historical ranges rather than to duplicate the same sequence length.

The lengths 5, 10, and 15 are nested subdivisions of the fixed 15-step input. They provide progressively broader contexts while preserving the latest observation as the common endpoint of all branches. The selected values are an empirical, compact multi-scale setting rather than a claim of globally optimal window lengths; their interpretation in clock time varies with the irregular GPS sampling interval.

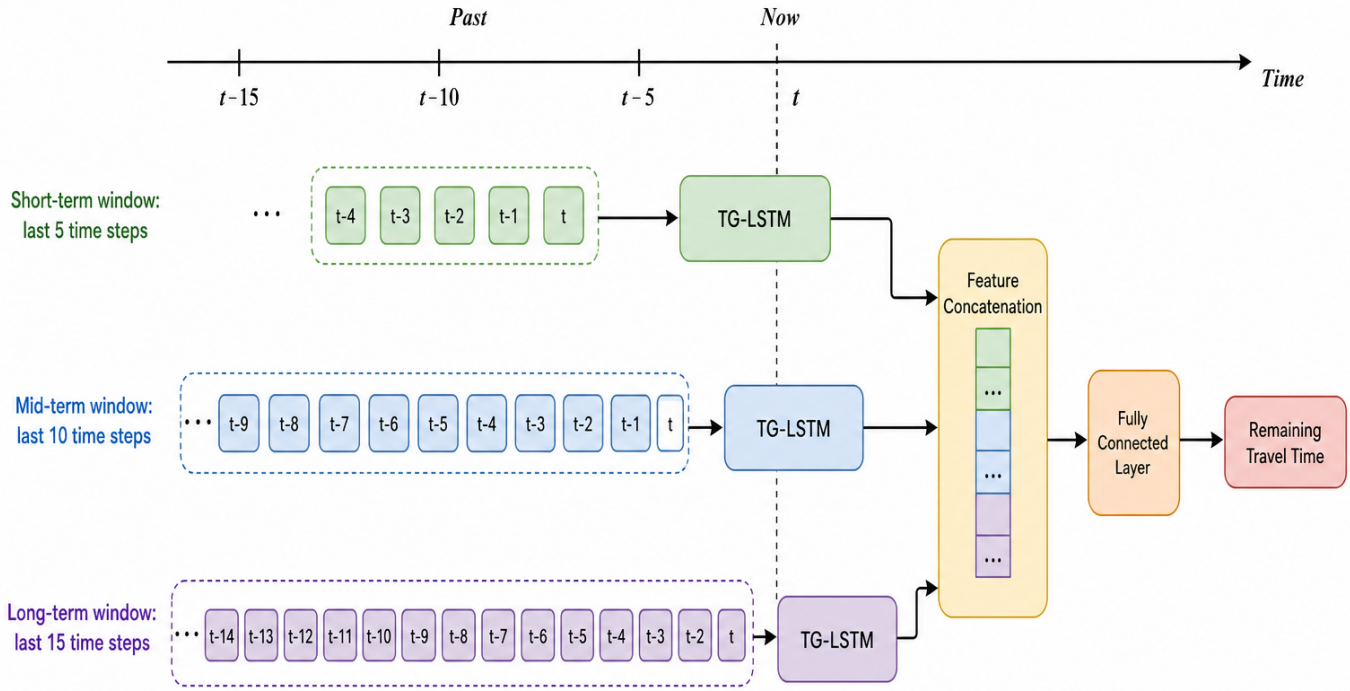


Figure 3. Multi-scale temporal window construction. The model uses the latest 5, 10, and 15 time steps to represent short-term, mid-term, and long-term temporal dependencies, respectively.

3.5 Feature Concatenation and Prediction

Let $\boldsymbol{h}_k^{short}$, $\boldsymbol{h}_k^{mid}$, and $\boldsymbol{h}_k^{long}$ denote the outputs of the three TG-LSTM branches. Consistent with Figure 1 and Figure 3, these outputs are fused by concatenation:

$$\boldsymbol{h}_k^{cat} = [\boldsymbol{h}_k^{short}; \boldsymbol{h}_k^{mid}; \boldsymbol{h}_k^{long}]. \tag{22}$$

Because each branch output is in $\mathbb{R}^H$, $\boldsymbol{h}_k^{cat} \in \mathbb{R}^{3H}$. The concatenated representation is passed to a fully connected layer to generate the remaining travel time prediction:

$$\hat{y}_k = \boldsymbol{W}_p \boldsymbol{h}_k^{cat} + \boldsymbol{b}_p. \tag{23}$$

Here, $\boldsymbol{W}_p \in \mathbb{R}^{1\times 3H}$ and $\boldsymbol{b}_p \in \mathbb{R}$. Thus, the final prediction is based on spatially attended trajectory features, traffic-state-conditioned TG-LSTM encoding, and multi-scale temporal feature concatenation. The complete computational procedure is summarized in Algorithm 1.

3.6 Training Objective

The model is trained by minimizing mean squared error (MSE) on the standardized remaining-time target:

$$\mathcal{L} = \frac{1}{M}\sum_{k=1}^{M}(\hat{y}_k - y_k)^2. \tag{24}$$

The implementation uses dropout regularization [23] and Adam with decoupled weight decay (AdamW) [24, 25]. AdamW uses an initial learning rate of 0.001, weight decay of 10^{-5}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and `amsgrad=false`. Gradient clipping limits the maximum norm to 1.0. A ReduceLROnPlateau scheduler monitors held-out MSE in minimization mode and multiplies the learning rate by 0.5 after five consecutive epochs without improvement; the relative threshold is 10^{-4}, cooldown is zero, and minimum learning rate is zero. Early stopping requires a strictly lower held-out MSE and uses a patience of 10 epochs. Random seeds for NumPy and PyTorch are both fixed at 42. For the controlled comparison, every recurrent model uses a hidden size of 16, one recurrent layer, and at most 50 epochs. A dropout rate of 0.3 is used where an explicit dropout operation is present; the traffic-state and feature-attention MLPs use dropout of 0.1. Because

Algorithm 1: SA-MS-TG-LSTM Training Procedure

Data: Raw GPS records with timestamps and destination times
Result: Predicted remaining travel time $\hat{y}_k$ for each sample k

`// Stage 1: Feature Construction`
Compute a_t, ρ_t, ϕ_t from consecutive GPS records;
Assemble seven-dimensional feature vectors $\boldsymbol{x}_t = [v_t, \theta_t, d_t, \Delta\tau_t, a_t, \rho_t, \phi_t]^\top$;
Forward-fill then backward-fill missing values within each trip;
Generate 15-step sliding-window samples $\boldsymbol{X}_k$ and targets $y_k = r_{k+15}$;
Standardize feature and target arrays;

`// Stage 2: Forward Pass`
Apply Spatial Feature Attention: $\boldsymbol{X}_k^a \leftarrow \mathrm{SFA}(\boldsymbol{X}_k)$;
Extract nested windows of 5, 10, and 15 steps from $\boldsymbol{X}_k^a$;
Encode each window with its TG-LSTM branch: $\boldsymbol{h}^{short}$, $\boldsymbol{h}^{mid}$, $\boldsymbol{h}^{long}$;
Concatenate branch outputs: $\boldsymbol{h}^{cat} \leftarrow [\boldsymbol{h}^{short}; \boldsymbol{h}^{mid}; \boldsymbol{h}^{long}]$;
Predict remaining travel time: $\hat{y}_k \leftarrow \boldsymbol{W}_p \boldsymbol{h}^{cat} + \boldsymbol{b}_p$;

`// Stage 3: Optimization`
Compute MSE loss: $\mathcal{L} \leftarrow \frac{1}{M}\sum_{k=1}^{M}(\hat{y}_k - y_k)^2$;
Update parameters with AdamW and gradient clipping (max norm $= 1.0$);
Reduce learning rate on plateau; apply early stopping (patience $= 10$);

each recurrent stack has one layer, inter-layer recurrent dropout is inactive.

4 Experiments

4.1 Data and Evaluation Metrics

The experiments use GPS trajectories collected from an actual Route 86 bus service. After timestamp parsing and invalid-record removal, the dataset contains 8,834 records from 293 trips. The implementation reads the message time and destination time fields, computes the remaining travel time in minutes, and sorts the records by trip identifier and timestamp. The original feature columns are speed, heading angle, distance to station, and recorded time difference; acceleration, congestion index, and speed volatility are then added as traffic-state features, resulting in seven input features. A 15-step sliding window is used to construct sequence samples, and the target is the remaining travel time at the next recorded point. The train-test split is performed at the trip level, producing 3,579 training samples and 889 test samples from 57 held-out trips. All compared models use the same preprocessing pipeline, feature standardization, sliding-window sample construction, and evaluation metrics.

Four metrics are used: mean absolute error (MAE), root mean square error (RMSE), mean absolute percentage error (MAPE), and coefficient of determination (R^2):

$$\mathrm{MAE} = \frac{1}{M}\sum_{i=1}^{M}|\hat{y}_i - y_i|, \tag{25}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(\hat{y}_i - y_i)^2}, \tag{26}$$

$$\mathrm{MAPE} = \frac{100\%}{M}\sum_{i=1}^{M}\left|\frac{\hat{y}_i - y_i}{y_i}\right|, \tag{27}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{M}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{M}(y_i - \bar{y})^2}. \tag{28}$$

Here, M is the number of evaluated samples, y_i and $\hat{y}_i$ are the observed and predicted remaining travel times, respectively, and $\bar{y}$ is the mean observed remaining travel time. Lower MAE, RMSE, and MAPE indicate smaller prediction errors, while a higher R^2 indicates stronger goodness of fit. The 889 held-out targets range from approximately 1 to 21 min; the minimum stored value is 0.9999997 min because of floating-point conversion, so the evaluated set contains no zero or near-zero target. The implementation adds 10^{-8} to the MAPE denominator for numerical protection, but this term does not materially affect the reported values at the observed target range.

4.2 Experimental Protocol

The experimental protocol is designed to evaluate prediction ability on trips that are not used for parameter updates. Therefore, the data are divided at the trip level rather than by randomly splitting individual sequence samples. Adjacent samples from one trip are highly correlated; placing them in both subsets would create a less demanding evaluation. All models use the same standardized feature and target arrays, the same trip assignment, and the same sequence samples.

All models are implemented in PyTorch and trained with the same standardized input tensors. The sequence length is fixed to 15, while the proposed multi-scale branches internally select the latest 5, 10, and 15 time steps. The training batch size is 256, and the evaluation batch size is 512. To control the recurrent configuration, LSTM, GRU, CNN-GRU, TG-LSTM, MS-TG-LSTM, and SA-MS-TG-LSTM all

Table 1. Controlled comparison with hidden size 16, one recurrent layer, and at most 50 epochs for every model.

Model	Params	Best epoch	RMSE (min)	MAE (min)	MAPE (%)	R^2
LSTM	1,617	48	1.238	0.897	16.76	0.9493
TG-LSTM	2,817	50	1.264	0.897	15.61	0.9472
MS-TG-LSTM	8,449	50	1.209	0.848	17.31	0.9517
SA-MS-TG-LSTM	8,696	49	**1.056**	**0.762**	**12.06**	**0.9631**
GRU	1,217	47	1.234	0.836	16.82	0.9496
CNN-GRU	2,001	48	1.127	0.807	13.59	0.9580

use a hidden dimension of 16, one recurrent layer, and a maximum of 50 epochs. They also share the same optimizer, learning rate, weight decay, gradient clipping, scheduler, early-stopping patience, and random seed. Total trainable parameters remain architecture dependent because traffic-state projections, parallel temporal branches, and SFA are components of the evaluated models; these counts are reported in Table 1 rather than hidden through a common nominal budget.

The software environment is Python 3.11.9 with PyTorch 2.4.1 and CUDA 11.8 on Windows. Computation uses an NVIDIA GeForce RTX 4060 Laptop graphics processing unit (GPU) with 8 GB of graphics memory.

4.3 Baselines

Six models are compared:

1. LSTM: a standard recurrent baseline using the constructed feature sequence.
2. GRU: a lighter gated recurrent baseline.
3. CNN-GRU: a hybrid model that extracts local feature patterns before recurrent modeling.
4. TG-LSTM: an LSTM variant with traffic-state gate modulation.
5. MS-TG-LSTM: a multi-scale version of TG-LSTM.
6. SA-MS-TG-LSTM: the proposed complete model with feature attention, multi-scale encoding, and traffic-state gating.

Every model is trained for at most 50 epochs with early-stopping patience 10. This controlled width, depth, and training-budget setting removes the former 32-versus-16 hidden-size, two-versus-one-layer, and 50-versus-30-epoch differences.

4.4 Results and Discussion

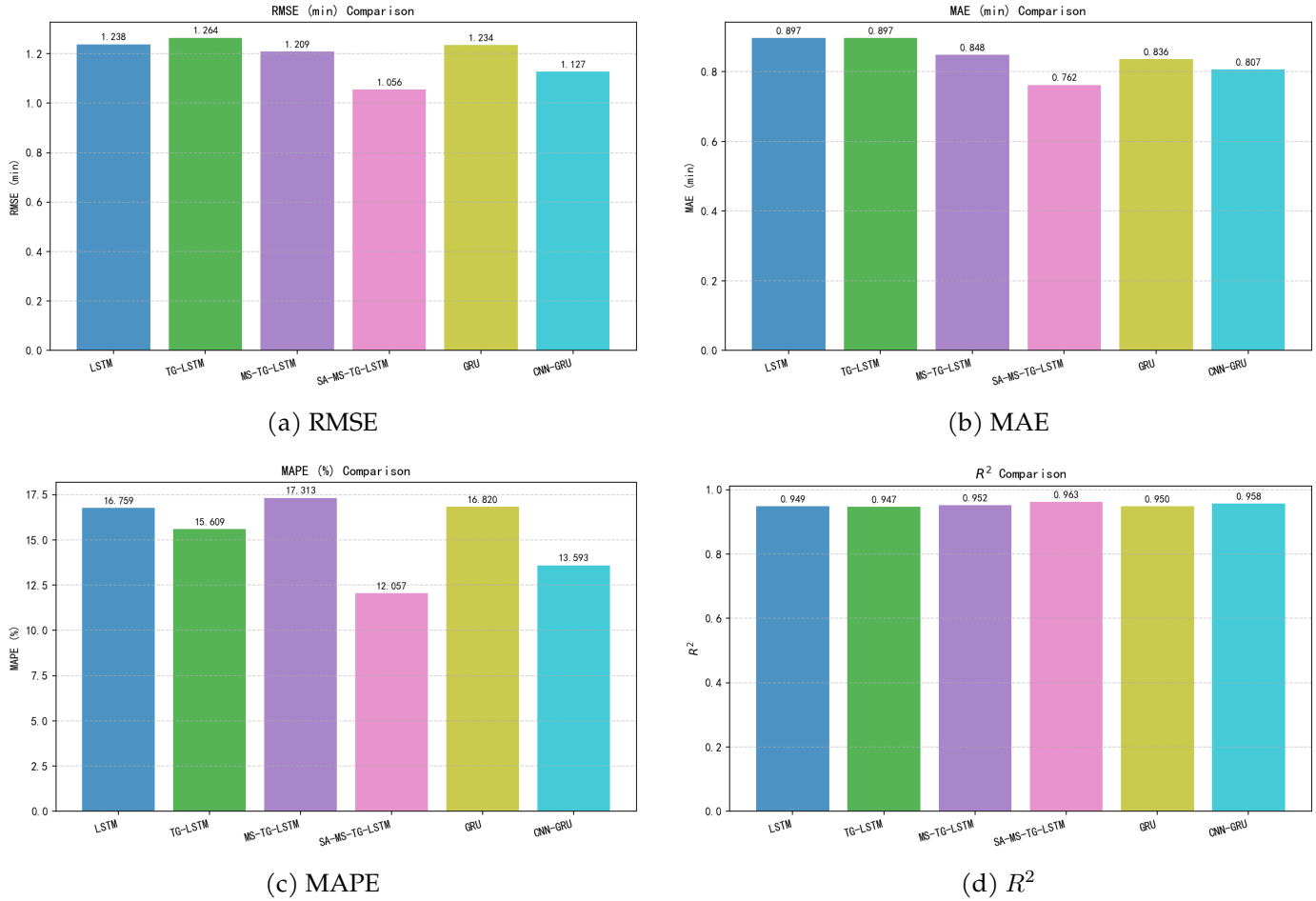
Table 1 reports prediction performance under the controlled recurrent configuration. SA-MS-TG-LSTM obtains the lowest point estimates of RMSE, MAE, and MAPE and the highest R^2, with an RMSE of 1.056 min, MAE of 0.762 min, MAPE of 12.06%, and R^2 of 0.9631. Compared with the standard LSTM, it reduces RMSE, MAE, and MAPE by 14.70%, 15.05%, and 28.04%, respectively. Compared with CNN-GRU, the strongest non-traffic-gated reference, it reduces RMSE by 6.30%, MAE by 5.57%, and MAPE by 11.26%, based on the rounded values in Table 1.

To assess whether the observed error differences are systematic across trips, absolute errors were averaged within each of the 57 held-out trips. A two-sided Wilcoxon signed-rank test compared trip-level MAE with SA-MS-TG-LSTM, and a 10,000-resample trip-level bootstrap estimated the 95% confidence interval (CI) of $\Delta\mathrm{MAE} = \mathrm{MAE}_{reference} - \mathrm{MAE}_{SA}$. Holm correction was applied across the five comparisons. Table 2 shows significant adjusted differences relative to LSTM and TG-LSTM. The differences relative to MS-TG-LSTM, GRU, and CNN-GRU are not significant at the 0.05 level; consequently, the aggregate ranking is reported as a point-estimate result rather than conclusive superiority over every reference.

Table 2. Paired trip-level MAE comparison with SA-MS-TG-LSTM under the controlled configuration.

Reference	ΔMAE (min)	95% CI	Adjusted p
LSTM	0.139	[0.050, 0.251]	0.002
TG-LSTM	0.138	[0.062, 0.231]	0.003
MS-TG-LSTM	0.090	[0.023, 0.174]	0.243
GRU	0.079	[−0.033, 0.226]	0.615
CNN-GRU	0.044	[−0.027, 0.124]	0.615

Figure 4 visualizes the same metric comparison from four perspectives. SA-MS-TG-LSTM has the lowest point estimates of RMSE, MAE, and MAPE and



(a) RMSE (b) MAE

(c) MAPE (d) R^2

Figure 4. Metric comparison under the controlled hidden size, layer, and training-epoch setting.

the highest R^2 under the evaluated configurations, consistent with Table 1. Statistical qualification of the trip-level MAE differences is provided in Table 2.

The component sequence gives a qualified ablation view. TG-LSTM improves MAPE from 16.76% to 15.61% relative to LSTM, but its RMSE and MAE are not lower, so traffic-state gating alone is not uniformly beneficial in the one-layer setting. MS-TG-LSTM reduces RMSE and MAE to 1.209 and 0.848 min, respectively, but has a higher MAPE. After scale-preserving SFA is added, SA-MS-TG-LSTM reduces RMSE to 1.056 min, MAE to 0.762 min, and MAPE to 12.06%, while adding only 247 parameters to MS-TG-LSTM. This pattern is consistent with feature reweighting reducing redundant branch inputs, but the adjusted trip-level difference between MS-TG-LSTM and the full model is not statistically significant.

Figure 5 shows that all models reduce training and held-out loss during optimization. SA-MS-TG-LSTM reaches the lowest best held-out MSE of 0.0347 at epoch 49. CNN-GRU is the strongest non-traffic-gated reference with a best held-out MSE of 0.0400, while the remaining models settle at higher minima.

Figure 6 presents a representative 16-point test trip. All models capture the overall decrease in remaining travel time. SA-MS-TG-LSTM remains close to the observed curve through the middle and later points, while the reference predictions generally fall below the observed values near the end of the trip. This case is illustrative rather than a substitute for the aggregate comparison in Table 1.

The station-level comparison in Figure 7 shows that prediction difficulty varies across stops, where S1–S10 identify the ten most frequent test stations in decreasing order of sample frequency. SA-MS-TG-LSTM has relatively low MAE at several frequent stations but is not uniformly best; at S1, for example, its error is the largest among the six models. The full model therefore improves the aggregate held-out metrics without eliminating stop-specific variation.

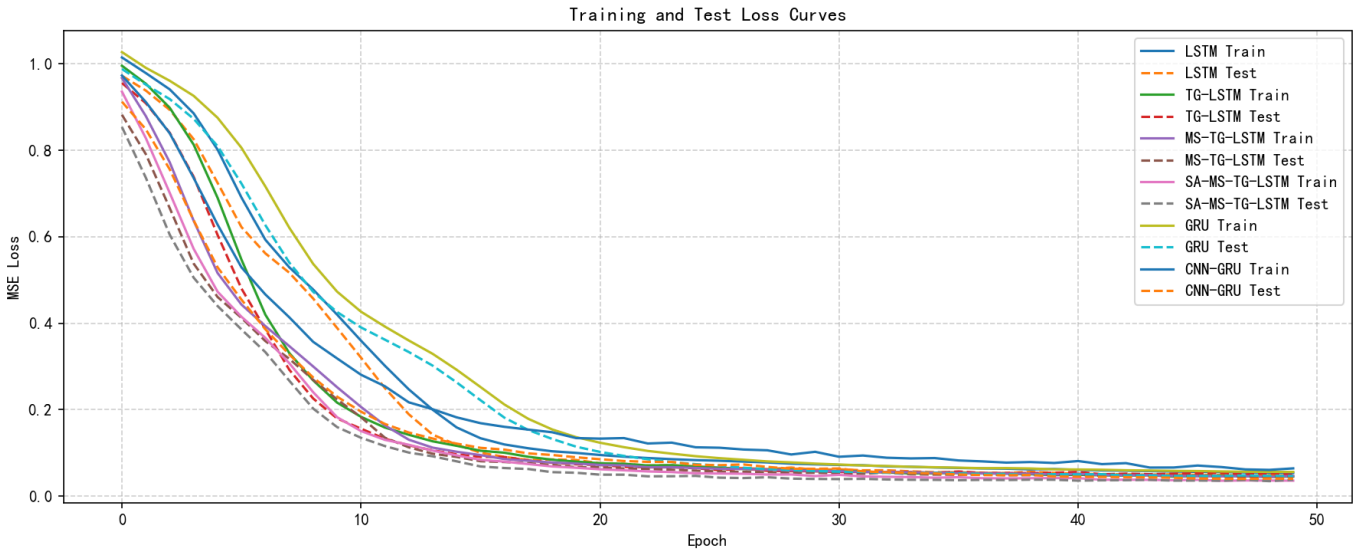


Figure 5. Training and held-out MSE loss curves under the controlled configuration.

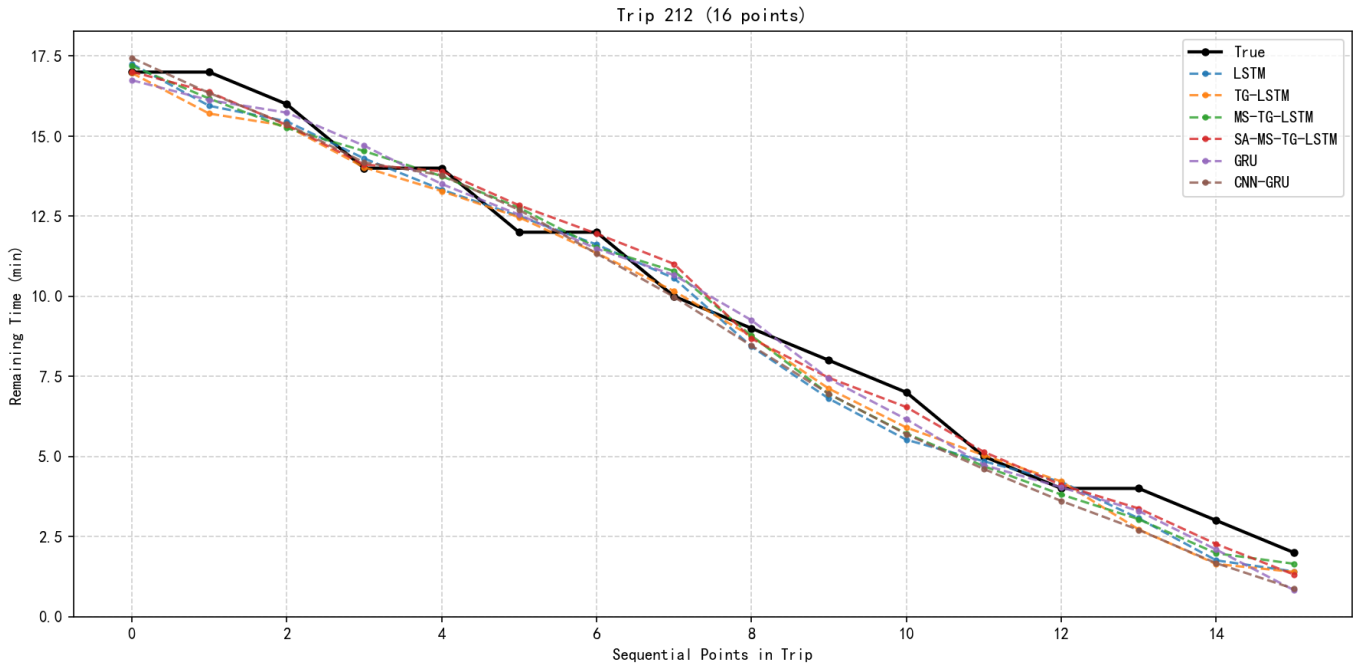


Figure 6. Remaining travel time prediction on a representative test trip.

4.5 Discussion of Model Components

The comparison among LSTM, TG-LSTM, MS-TG-LSTM, and SA-MS-TG-LSTM provides a progressive view under the same hidden width, recurrent depth, and maximum epoch count. In the LSTM reference, acceleration, congestion index, and speed volatility are only part of the input vector. In TG-LSTM, the transformed traffic state contributes to the input, forget, output, and candidate-memory computations. TG-LSTM lowers MAPE but does not lower RMSE or MAE relative to LSTM, indicating that gate conditioning alone is insufficient in this setting. MS-TG-LSTM improves RMSE and MAE relative to TG-LSTM but increases MAPE, showing that additional temporal branches do not improve every criterion. The lowest aggregate errors are obtained only after scale-preserving SFA is added. The factor F prevents uniform attention from shrinking all standardized inputs, while nonuniform weights still reweight the feature dimensions before the three temporal encoders. This mechanism is consistent with the observed point estimates, but the non-significant adjusted comparisons with MS-TG-LSTM, GRU, and CNN-GRU mean that the ranking should be verified

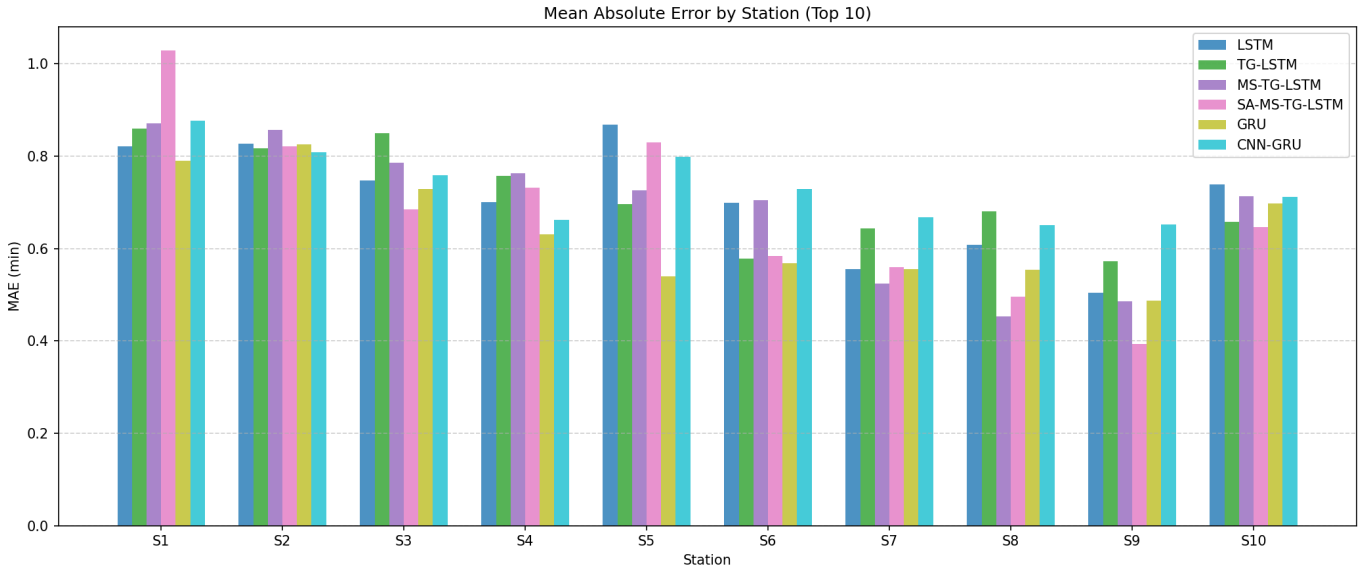


Figure 7. Station-level MAE comparison for the ten most frequent stations in the test set, denoted by S1–S10 in decreasing order of sample frequency.

with repeated runs and additional routes.

From an operational perspective, the reduction in MAE from 0.897 min for LSTM and 0.807 min for CNN-GRU to 0.762 min for SA-MS-TG-LSTM corresponds to smaller average deviations in the displayed configuration. This aggregate advantage does not eliminate station-level uncertainty, as shown in Figure 7; future deployment should therefore consider stop-specific calibration or additional route context.

5 Limitations

This study has several limitations. First, the experiment uses GPS trajectories from one bus route. Although the trip-level split avoids sample leakage between training and testing trips, the evaluation does not yet verify cross-route generalization. A model trained on a single route may learn route-specific speed patterns, stop spacing, and signal-delay characteristics. Testing the same architecture on multiple routes would provide stronger evidence of robustness.

Second, the current feature set is derived from GPS records and simple traffic-state variables. External factors such as weather, incidents, road works, signal timing, holidays, and passenger boarding demand are not included. These factors may explain some of the station-level error differences observed in Figure 7. Incorporating such contextual variables may improve prediction during abnormal operating conditions.

Third, the model is evaluated as a point predictor of remaining travel time. In practical passenger information systems, uncertainty estimates are also important because two predictions with the same expected value may have different reliability. Future work can extend SA-MS-TG-LSTM to probabilistic prediction, for example by predicting quantiles or confidence intervals for remaining travel time.

Fourth, the controlled comparison aligns hidden size, recurrent depth, maximum epochs, optimization settings, and random seed, but total parameter counts remain architecture dependent because the proposed model contains traffic projections, three recurrent branches, and SFA. Exact parameter-budget matching would require architecture-specific hidden widths and would answer a different capacity question. The experiment also uses one fixed random seed. Although the trip-level analysis quantifies uncertainty across held-out trips, repeated training and a separate validation split are still needed to quantify optimization variance and model-selection uncertainty.

6 Conclusion

This paper presented a multi-scale traffic-state gated attention network for bus arrival time prediction. The method constructs traffic-state features from GPS records, applies scale-preserving feature-dimension attention, divides the input into three nested temporal windows, and uses traffic-state gated LSTM branches to model recurrent dynamics. In the controlled Route 86 comparison, every model uses hidden

size 16, one recurrent layer, and at most 50 epochs. SA-MS-TG-LSTM obtains the lowest aggregate point errors, with an RMSE of 1.056 min, MAE of 0.762 min, MAPE of 12.06%, and R^2 of 0.9631. Adjusted trip-level tests support significant MAE reductions relative to LSTM and TG-LSTM, but not relative to MS-TG-LSTM, GRU, or CNN-GRU. Together with station-level heterogeneity and the single-route scope, these results motivate repeated-seed evaluation, additional routes, and external context such as weather, signal timing, and passenger demand.

Data Availability Statement

The bus GPS data used in this study are not publicly available because they were obtained from an operational transit route. Processed data and implementation details are available from the corresponding author upon reasonable request.

Funding

This work was supported without any funding.

Conflicts of Interest

The authors declare no conflicts of interest.

AI Use Statement

The authors declare that no generative AI was used in the preparation of this manuscript.

Ethical Approval and Consent to Participate

Not applicable.

References

[1] Cathey, F. W., & Dailey, D. J. (2003). A prescription for transit arrival/departure prediction using automatic vehicle location data. *Transportation Research Part C: Emerging Technologies, 11*(3-4), 241-264. [CrossRef]

[2] Patnaik, J., Chien, S., & Bladikas, A. (2004). Estimation of bus arrival times using APC data. *Journal of public transportation, 7*(1), 1-20. [CrossRef]

[3] Shalaby, A., & Farhan, A. (2004). Prediction model of bus arrival and departure times using AVL and APC data. *Journal of Public Transportation, 7*(1), 41-61. [CrossRef]

[4] Mete, S., Çelik, E., & Gül, M. (2022). Predicting the time of bus arrival for public transportation by time series models. *Journal of Transportation and Logistics, 7*(2), 541-555. [CrossRef]

[5] Chien, S. I. J., Ding, Y., & Wei, C. (2002). Dynamic bus arrival time prediction with artificial neural networks. *Journal of transportation engineering, 128*(5), 429-438. [CrossRef]

[6] Jeong, R., & Rilett, L. R. (2005). Prediction model of bus arrival time for real-time applications. *Transportation Research Record, 1927*(1), 195-204. [CrossRef]

[7] Yu, B., Lam, W. H. K., & Tam, M. L. (2011). Bus arrival time prediction at bus stop with multiple routes. *Transportation Research Part C: Emerging Technologies, 19*(6), 1157-1170. [CrossRef]

[8] Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks, 5*(2), 157-166. [CrossRef]

[9] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation, 9*(8), 1735-1780. [CrossRef]

[10] Kumar, B. A., Singh, R., Shaji, H. E., & Vanajakshi, L. (2025). Bus arrival time prediction: A comprehensive review. *IEEE Transactions on Intelligent Transportation Systems, 26*(6), 7362-7379. [CrossRef]

[11] Jalaney, J., & Ganesh, R. S. (2023). Multiple extreme learning machines based arrival time prediction for public bus transport. *Intelligent Automation & Soft Computing*, 36(3), 2819-2834. [CrossRef]

[12] Cho, K., Van Merriënboer, B., Gulçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014, October). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1724-1734). [CrossRef]

[13] Petersen, N. C., Rodrigues, F., & Pereira, F. C. (2019). Multi-output bus travel time prediction with convolutional LSTM neural network. *Expert Systems with Applications, 120*, 426-435. [CrossRef]

[14] Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*. [CrossRef]

[15] Zhang, B., Tang, L., Zhou, D., Liu, K., & Xue, Y. (2024). Bus Arrival Time Prediction Based on the Optimized Long Short-Term Memory Neural Network Model With the Improved Whale Algorithm. *Journal of Advanced Transportation, 2024*(1), 6997338. [CrossRef]

[16] Li, Z. (2024). DA-RNN-based bus arrival time prediction model. *International Journal of Intelligent Transportation Systems Research, 22*(3), 660-674. [CrossRef]

[17] Yu, C., Chong, Y. W., Budi, A. S., Setiawan, E., & Keoh, S. L. (2024, August). Bus Arrival Time Prediction Using Hybrid LightGBM-LSTM. In *2024 International Conference on Platform Technology and Service (PlatCon)* (pp. 44-49). IEEE. [CrossRef]

[18] Lee, C., & Yoon, Y. (2022). A novel bus arrival time prediction method based on spatio-temporal flow centrality analysis and deep learning. *Electronics, 11*(12), 1875. [CrossRef]

[19] Sun, H., Li, F., & Liu, L. (2024, August). A spatial-temporal neural network model based on meta learning for bus arrival time prediction. In *2024 10th International Conference on Big Data Computing and Communications (BigCom)* (pp. 57-64). IEEE. [CrossRef]

[20] Qiu, T., Lam, C. T., Liu, B., Ng, B. K., Yuan, X., & Im, S. K. (2025). FEN-MRMGCN: A frontend-enhanced network based on multi-relational modeling GCN for bus arrival time prediction. *IEEE Access, 13*, 5296-5307. [CrossRef]

[21] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems, 30*.

[22] Jeong, S., Oh, C., & Jeong, J. (2024). BAT-transformer: Prediction of bus arrival time with transformer encoder for smart public transportation system. *Applied Sciences, 14*(20), 9488. [CrossRef]

[23] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research, 15*(56), 1929-1958. https://jmlr.org/papers/v15/srivastava14a.html

[24] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. [CrossRef]

[25] Loshchilov, I., & Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*. [CrossRef]

Ruping Liu received the B.S. degree in Electronic Science and Technology from Huai'an University, Huai'an, China. She is currently pursuing the M.S. degree at Huai'an University, where her research focuses on Traffic Information Engineering and Control. (Email: 1774179967@qq.com).

Xiaoqi Yin is a Professor and M.S. supervisor with the School of Electronic Science and Technology, Huai'an University, Huai'an, China. Her research interests include signal processing and measurement technology. She is the corresponding author of this article. (Email: hy_xuebao2009@126.com).

Jing Zhou received the B.S. degree in Electronic Science and Technology from Huai'an University, Huai'an, China. She is currently pursuing the M.S. degree at Huai'an University, where her research focuses on Traffic Information Engineering and Control. (Email: 2960088350@qq.com).