Journal of Network Intelligence ©2025 ISSN 2414-8105 (Online)
Taiwan Ubiquitous Information Volume **10**, Number **2**, May **2025**

Urban Traffic Flow Prediction and Optimization Control Based on Convolutional Neural Network-Bidirectional Long Short-Term Memory

Pei-Peng Cong*

CRRC Dalian Locomotive & Rolling Stock Co., Ltd., Dalian 116000, China
013500012677@crrcgc.cc, 18584654946@163.com.

Tian-Yu Wang

CRRC Dalian Locomotive & Rolling Stock Co., Ltd., Dalian 116000, China
013500012375@crrcgc.cc

Chuang Sun

CRRC Dalian Locomotive & Rolling Stock Co., Ltd., Dalian 116000, China
013500012745@crrcgc.cc

*Corresponding author: Pei-Peng Cong

Received March 15, 2024, revised July 28, 2024, accepted December 12, 2024.

Abstract. *As urbanization accelerates, the growing problem of urban traffic congestion underscores the importance of regional traffic optimization for signal timing. However, current research has limitations, such as providing only static optimization solutions and optimizing for individual intersections. To handle these issues, this paper introduces an urban traffic flow prediction (TFP) and optimization solution based on (convolutional neural networks- bidirectional long short-term memory) CNN-BiLSTM and beluga whale optimization (BWO) algorithm. First, to enhance the real-time control of regional traffic signal (RTS), a multi-intersection TFP model, CNN-BiLSTM, is introduced to predict the inbound flow at intersections. Secondly, in order to enhance the overall coordination, a dynamic RTS coordination optimization control strategy is proposed. This method is based on the predicted inbound flow at intersections, combined with an improved BWO to find the optimal scheme for the average delay model of regional vehicles, thereby achieving dynamic optimization control. Finally, through experimental simulations, the proposed approach demonstrates superior predictive and optimization performance compared to existing models.*

Keywords: TFP; RTS control; CNN; BiLSTM; Improved BWO

1. **Introduction.** In recent years, the rapid increase in the number of urban automobiles has led to persistent challenges such as traffic congestion, extended travel times, increased fuel consumption, and elevated emissions [1–3]. The close proximity and frequent acceleration and braking between vehicles have sharply increased the likelihood of traffic accidents, significantly impacting travel safety. To enhance urban road traffic efficiency, safety, and environmental friendliness, studies on traffic optimization has emerged [4]. This research employs traffic control technologies, information technologies, and other means to comprehensively optimize and control traffic signal systems. However, most existing studies can only offer static optimization solutions,

failing to adapt to variations in traffic flow at different times. Additionally, many of these studies focus on individual intersections, lacking a holistic approach to the planning and optimization of regional traffic systems [5,6]. This limitation hinders the effective improvement of traffic system efficiency, and issues such as congestion and safety persist. Therefore, deploying intelligent traffic systems correctly, enabling control systems to efficiently manage dynamic vehicles in high-density traffic with limited infrastructure, holds significant practical significance [7].

Traffic flow prediction (TFP) is a crucial component of intelligent transportation systems, offering significant value in alleviating traffic congestion, reducing the risk of traffic accidents, improving traffic control efficiency, and enhancing timely traffic resource dispatching [8–10]. Initially, scholars utilized shallow machine learning methods for TFP. For instance, Kumar et al. [11] proposed the use of seasonal autoregressive integrated moving average models to achieve short-term TFP with limited input data. Chen et al. [12] employed popular smoothing models to denoise raw traffic flow data and introduced artificial neural networks for TFP. Sharaf AlKheder et al. [13] suggested applying Bayesian Combined Neural Network for short-term TFP at neighboring intersections. Although machine learning-based models achieve high accuracy in short-term traffic flow prediction, they are still limited in extracting deep and implicit spatiotemporal correlations in large traffic datasets with the rapid development of intelligent transportation systems. With technological advancements, numerous studies have shown that deep learning has powerful nonlinear feature extraction capabilities in capturing deep-level features in spatiotemporal sequence data [1,14]. Therefore, an increasing number of researchers are focusing on TFP models based on deep learning. Daeho et al. [15] proposed a cooperative traffic signal control with TFP method on the basis of multi-intersection TFP. This approach uses long short-term memory (LSTM) networks to achieve TFP at multiple intersections (MI). Wan et al. [16] introduced a novel deep network, utilizing Convolutional Neural Networks (CNN) and LSTM to extract spatial and temporal features. However, these models still exhibit limitations in their ability to learn long dependencies in complex time series data. Therefore, this study aims to investigate a model with stronger predictive capabilities.

After achieving accurate TFP at MI, improving the coordination and optimization control of urban regional traffic signals (RTS) is also a crucial research direction. Currently, many researchers have focused on the optimization of regional traffic by applying population-based metaheuristic algorithms [17–22]. For example, the particle swarm optimization (PSO) algorithm [17] was improved combining with the method of genetic algorithm [18], and then was employed to find the optimal scheme for the control of traffic flow, thereby achieving reduction of the average delay time [19]. In addition, this team also introduced an adaptive PSO algorithm with mutation to solve the optimization model, leading to an improvement in regional traffic efficiency [21]. Subsequently, a novel regional signal optimization scheme using a multi-objective PSO algorithm was introduced to optimize a multi-objective model, making it suitable for integer programming problems [22]. Although these schemes are improvements relied on existing swarm intelligence algorithms, these enhancements only focus on individual optimization methods, without addressing the problem of unequal distribution of initial population groups. This limitation may result in less pronounced effects when applied to extensive road networks with numerous variables. Hence, this study endeavors to explore ways to enhance existing swarm intelligence algorithms from various angles, aiming to augment the overall optimization capabilities of the algorithm.

Driven by above analysis, this paper proposes an urban-region TFP based on CNN-BiLSTM and an optimization control scheme based on improved BWO. The specific contributions of this paper are as follows

(1) Firstly, based on previous research, an optimization model is established using the average delay time of regional vehicles as an evaluation criterion to achieve RTS coordination and optimization control.

(2) Secondly, a multi-intersection TFP model based on CNN and BiLSTM is proposed for predicting intersection approach flow. CNN-BiLSTM utilizes CNN and BiLSTM to extract spatial and temporal features of traffic flow. Additionally, based on the periodicity of traffic flow, CNN-BiLSTM proposes the use of an average pooling layer to learn the periodic features of traffic flow.

(3) Finally, to enhance the overall coordination of RTS control, this paper introduces an improved BWO based on a combination of chaotic reverse learning strategy and water wave optimization (WWO). Compared to other intelligent algorithms, this algorithm demonstrates superior performance in finding the optimal scheme for the model constructed with the average delay time of regional vehicles as the evaluation criterion.

2. Related Theory.

2.1. Optimal control of RTS coordination.

RTS coordination optimization control involves coordinating and controlling conflicting traffic flows within a region by adjusting the traffic signal lights set at intersections. The fundamental parameters of signal control are as follows

1) Signal phases: refers to the allocation of specific green light durations to different directions of traffic flow during signal timing at intersections. The purpose of introducing signal phases is to appropriately separate traffic flows that conflict or cause significant interference, thereby enhancing the efficiency of traffic at intersections. In the regional road network studied in this paper, right-turning vehicles are disregarded at intersections. Instead, a four-phase mode of control is employed, as illustrated in Figure 1, encompassing north-south straight, east-west straight, north-south left turn, and east-west left turn movements.

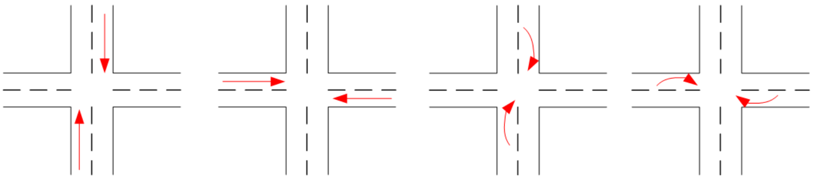


Figure 1. Four-phase mode diagram

2) Phase transition time: refers to the time interval between the end of one green phase and the beginning of the next green phase. The purpose of setting the phase transition time is to ensure that vehicles that have entered the intersection beyond the stop line during the current phase can safely pass through the conflict point and exit the intersection before the leading vehicle of the next phase reaches the conflict point. The phase transition time is typically composed of the yellow light duration or the sum of the yellow light and clearance red time.

3) Signal phase difference: refers to the time difference between the green signal phases at two different intersections, categorized into absolute phase difference and

relative phase difference. Absolute phase difference refers to the time difference between a specific intersection taken as the reference and the green signal phases at other intersections relative to this reference intersection. Relative phase difference refers to the time difference between adjacent intersections.

4) Signal cycle (SC): refers to the total time required for all signal phases to operate at an intersection, denoted by the symbol T, and can be divided into long cycle and short cycle. For intersections with lower traffic flow, adopting a short cycle allows vehicles from all directions to flow quickly, resulting in shorter waiting times for drivers and a better travel experience. However, at intersections with higher traffic flow, using a "short cycle" may not quickly clear queued vehicles, leading to a decrease in operational efficiency. Therefore, a "long cycle" is used to increase the maximum number of vehicles passing through the intersection per hour. However, having a cycle that is too long can increase drivers' waiting times and reduce the travel experience. Hence, the size of the SC needs to be set based on the travel characteristics of the intersection and roadway, considering factors such as operational efficiency and safety comprehensively.

5) Green split: refers to the ratio of the green time g for a specific signal phase to the SC time T within one SC, denoted by the symbol λ, which is given by

$$\lambda = \frac{g}{T} \tag{1}$$

6) Saturation flow rate: refers to the maximum number of vehicles passing through within one effective green time, measured using the method of saturation headway. It is denoted by the symbol S.

7) Road capacity: refers to the maximum number of vehicles that can pass through a specific road section within a unit of time under certain road traffic and environmental conditions. It is denoted by the symbol Q. It can be expressed as

$$Q = S \cdot \lambda \tag{2}$$

7) Saturation degree: it refers to the ratio of the actual traffic flow to the capacity of a specific entrance road. It is one of the important indicators reflecting the level of road service, denoted by the symbol x. A higher value indicates lower road traffic efficiency. The calculation is done by the formula:

$$x = \frac{q}{Q} = \frac{q}{S \cdot \lambda} = \frac{q}{S} \cdot \frac{T}{g} \tag{3}$$

2.2. **CNN and BiLSTM Model.**

2.2.1. *CNN Model.*

The CNN possesses two important characteristics: local perception and shared weights [23]. CNN processes images through local perception, taking advantage of the close relationships between adjacent pixels and weaker connections between pixels that are farther apart. Instead of perceiving the entire image, CNN focuses on local regions. This approach allows for better feature extraction and reduces computational complexity by combining local information at higher levels to form global image information [21, 24]. As shown in Figure 2, each hidden neuron in the convolutional layer is connected only to a portion of the region; shared weights mean that neurons in the same feature map share a common convolutional kernel. The convolutional kernel starts as a random matrix of small numbers and undergoes continuous changes in weight during the training process. Shared weights enable neurons in the feature map to detect the same features hidden in different positions within the image [25].

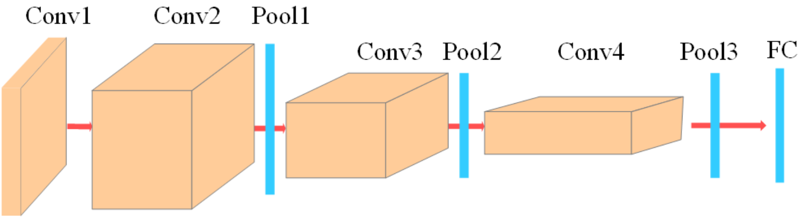


FIGURE 2. CNN Model

Compared to feedforward neural networks that use a “fully connected” structure, CNNs process images using local perception and shared weights, significantly reducing the number of parameters required for computation. Additionally, CNNs employ pooling layers to downsample images, sparsely process features, reduce computational load, and prevent the model from overfitting the dataset [26]. However, CNNs also face certain challenges: firstly, as the depth of the neural network increases, modifying parameters using the backpropagation algorithm can result in slower changes to parameters near the input layer, requiring more data to train the network; secondly, during training with gradient descent, CNNs may get stuck in local optima; thirdly, pooling layers can lead to the loss of important information and overlook correlations between the whole and parts of the image.

2.2.2. *LSTM Model.* LSTM is a specialized variant of RNNs designed to address the challenge of capturing long-term dependencies that traditional RNNs struggle with [27]. As shown in Figure 2, by introducing memory units and three crucial gates (input gate, forget gate, and output gate), LSTM can effectively learn and retain information from sequential data [28]. The memory unit serves as the internal state of the network, the input gate controls the influx of new information, the forget gate manages when to clear unnecessary information, and the output gate determines how much information to output from the memory unit.

The input gate I_t, forget gate F_t, and output gate O_t are given by

$$I_t = \sigma\left(W_{in}x_t + W_{ih}h_{t-1} + b_{in}\right) \tag{4}$$

$$F_t = \sigma\left(W_{fn}x_t + W_{fh}h_{t-1} + b_f\right) \tag{5}$$

$$o_t = \sigma\left(W_{on}x_t + W_{oh}h_{t-1} + b_{out}\right) \tag{6}$$

where W represents the learnable weight parameters, b is the bias term, and h_{t-1} denotes the hidden layer state from the previous time step. Based on this, the long-term memory unit C_t and the final hidden layer state h_t can be expressed as

$$C_t = F_t \odot C_{t-1} + I_t \odot \widetilde{C}_t \tag{7}$$

$$h_t = O_t \odot \tanh\left(C_t\right) \tag{8}$$

with $\odot$ denoting element-wise Hadamard multiplication, and $\widetilde{C}_t$ is the candidate memory unit at the current time step expressed as

$$\widetilde{C}_t = \tanh\left(W_c x_t + W_{ch}h_{t-1} + b_c\right) \tag{9}$$

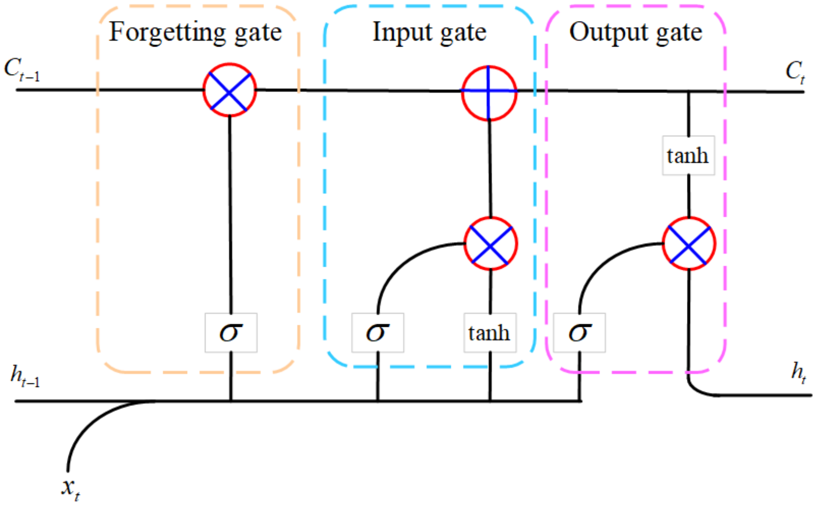


FIGURE 3. LSTM Model

2.3. **BWO Algorithm.** The BWO algorithm is an intelligent optimization algorithm that consists of initialization, the exploration phase, exploitation phase, and the convergence phase [29]. Unlike traditional algorithms such as PSO and genetic algorithm, the BWO features a simple structural design, fast convergence speed, and high solution precision. It has demonstrated outstanding performance in many practical optimization problems, particularly in high-dimensional optimization problems and multimodal function optimization, showcasing strong global search capabilities and rapid convergence.

2.3.1. *Initialization.* The mechanism based on the whale population considers each whale as a search agent, where each whale represents a candidate solution that is continuously updated during the optimization process. The model is established as

$$x = \begin{bmatrix} x_{1,1} & \cdots & x_{1,d} \\ \vdots & \ddots & \vdots \\ x_{n,1} & \cdots & x_{n,d} \end{bmatrix} \tag{10}$$

where n represents the size of the whale population, and d is the dimension of the variables. The storage of fitness is as follows

$$F(x) = \begin{bmatrix} f\left(x_{1,1}, \cdots, x_{1,d}\right) \\ f\left(x_{2,1}, \cdots, x_{2,d}\right) \\ f\left(x_{3,1}, \cdots, x_{3,d}\right) \end{bmatrix} \tag{11}$$

In the BWO algorithm, the selection between the exploration and exploitation phases depends on the balance factor B_f, and whose mathematical model is given by:

$$B_f = B_0 \left(1 - \frac{T}{2T_{\max}}\right) \tag{12}$$

Here, T represents the current iteration number, $T_{\max}$ is the maximum number of iterations, and B_0 is a random number within the range $(0, 1)$ that changes with each iteration. When the balance factor $B_f < 0.5$, the algorithm enters the exploitation phase, and when $B_f > 0.5$, the algorithm enters the exploration phase. As the iteration number increases, the fluctuation range of B_f gradually decreases from $(0, 1)$ to $(0, 0.5)$, and the probability of entering the exploitation phase increases continuously with the number of iterations.

2.3.2. *Exploitation phase.* In the exploration phase of the BWO algorithm, the position update is determined by the paired swimming behavior of the whales, and the update rule is as follows

$$\begin{cases} X_{i,j}^{T+1} = X_{i,P_j}^{T} + \left(X_{r,P_1}^{T} - X_{i,P_j}^{T}\right)(1+r_1)\sin\left(2\pi r_2\right), j = even \\ X_{i,j}^{T+1} = X_{i,P_j}^{T} + \left(X_{r,P_1}^{T} - X_{i,P_j}^{T}\right)(1+r_1)\cos\left(2\pi r_2\right), j = odd \end{cases} \tag{13}$$

where T is the current iteration number, $X_{i,j}^{T+1}$ is the new position of the i-th whale in the j-dimensional space, $P_j(j = 1, 2, \cdots, d)$ represents a randomly selected integer from the d dimensions, X_{i,P_j}^{T} is the position of the i-th whale in the P_j dimension, X_{i,P_j}^{T} and X_{r,P_1}^{T} denote the current positions of the i-th and r-th whales respectively, r represents the randomly selected whale, r_1 and r_2 are random numbers between $(0, 1)$, and $\sin(2\pi r_2)$ and $\cos(2\pi r_2)$ represent the orientation of the mirrored whale, with two choices based on whether the dimension is odd or even.

2.3.3. *Exploitation phase.* The foraging behavior of whales is designed to be the development phase of the BWO algorithm. In the development phase, a Levy flight strategy is introduced to enhance convergence. The mathematical model is expressed as follows

$$X_i^{T+1} = r_3 X_{best}^{T} - r_4 X_i^{T} + C_1 \cdot L_F\left(X_r^{T} - X_i^{T}\right) \tag{14}$$

where X_i^T and X_r^T represent the current positions of the i-th whale and a randomly chosen whale, respectively. X_i^{T+1} is the new position of the i-th whale, and X_{best}^T is the best position in the population. r_3 and r_4 are random numbers between $(0, 1)$, and $C_1 = 2$. $r_4(1 - T/T_{\max})$ is the random jump strength measuring the Levy flight intensity. Lr is a random number following the Levy distribution, with the formula as follows

$$L_F = 0.05 \times \frac{u \times \sigma}{|v|^{1/\beta}} \tag{15}$$

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin(\pi\beta/2)}{\Gamma((1+\beta)/2) \times \beta \times 2^{(\beta-1)/2}}\right)^{1/\beta} \tag{16}$$

u and v are random numbers following a normal distribution, with β defaulting to 1.5.

2.3.4. *Whale Fall phase.* In the ecological environment, white whales are also subject to external life threats. To maintain the population, the positions are updated using the positions and descent steps of the white whales. The mathematical model is represented as follows

$$X_i^{T+1} = r_5 X_i^{T} - r_6 X_r^{T} + r_7 X_{step} \tag{17}$$

$$X_{step} = (u_b - l_b)\exp\left(-C_2 T/T_{\max}\right) \tag{18}$$

where r_5, r_6, and r_7 are random numbers between $(0, 1), X_{step}$ is the step length of the whale's fall, u_b and l_b are the upper and lower limits of the variables, and C_2 is a step factor related to the descent probability of white whales and the population size $(C_2 = 2W_f \times n)$.

The fall probability W_f of the white whale is calculated as a linear function, given by

$$W_f = 0.1 - 0.05T/T_{\max} \tag{19}$$

The probability of the white whale falling gradually decreases from the initial 0.1 to 0.05, indicating that as the optimization progresses and the whale gets closer to the food, the probability of the whale falling decreases.

3. **Method.** The urban transportation network, as a complex system highly susceptible to various uncertain factors such as weather and unforeseen events, exhibits strong dynamic characteristics in traffic flow, including nonlinearity, non-stationarity, and randomness. Therefore, the design of TFP models often needs to consider two typical features.

Firstly, there is spatiotemporal correlation: there exists a direct or indirect correlation between adjacent roads in the road network, making their traffic flows closely interconnected. Additionally, regions with close geographical proximity tend to exhibit similar trends in traffic flow, and the connections between different roads in the road network and interactions among traffic entities contribute to the spatiotemporal correlation of traffic flow on the roads.

Secondly, there is periodicity: through the analysis of a large amount of traffic flow data, it is evident that traffic flow demonstrates strong periodicity over time. With predetermined periodic lengths such as yearly, monthly, weekly, and daily cycles, traffic flow data often exhibit significant regularity and correlation.

Therefore, to address these issues, the paper proposes a city TFP algorithm based on CNN-BiLSTM and optimization of RTS coordination method based on TWBWO.

3.1. Regional average vehicle delay.

3.1.1. *Vehicle Delay Modeling Calculation Method.* Vehicle delay, is a most frequently used evaluation metric, refers to the time loss experienced by vehicles in motion due to disruptions caused by other vehicles, traffic control facilities, and similar interferences. Experts, through extensive theoretical research and numerical simulation analyses, have derived patterns and regularities in vehicle delay.

The earliest delay model was proposed by Webster as the steady-state theory delay model. This model consists of uniform delay time and random delay time. However, it is not suitable for high-saturation traffic conditions. Especially when the road saturation approaches 1, the random delay time tends to infinity, which is inconsistent with actual traffic conditions. To adapt the delay model to both undersaturated and oversaturated conditions, Miller [30], Rouphail [31], and others have successively improved the Webster delay model. In 2000, the transportation research board of the United States introduced the highway capacity manual 2000 (HCM2000). The delay model in HCM2000 draws on the latest research findings and accurately describes traffic conditions under different saturation levels. The calculation method for the HCM2000 vehicle delay model is expressed as the following equation

$$d_u = \frac{T(1-\lambda)^2}{2(1-y)} \tag{20}$$

$$d_r = 900C\left[(x-1) + \sqrt{(x-1)^2 + \frac{4x}{QC}}\right] \tag{21}$$

$$d = d_u + d_r \tag{22}$$

where d_u represents the uniform delay time for vehicles and d_r represents the random delay time for vehicles. T denotes the SC duration. λ stands for the green signal ratio (GSR) for the inbound lane in Equation (1). y signifies the traffic flow ratio for the inbound lane, i.e., the ratio of the inbound lane flow to the road's saturated flow. Q

denotes the road capacity in Equation (2). x is the saturation degree for the inbound lane. C is the duration of the analysis period, typically set at 0.25 hours.

3.1.2. *Analysis of model constraints.* In order to enact RTS control strategy, it is essential to impose constraints on two parameters: SC and GSR.

SC Constraint: The meticulous configuration of the SC stands as a pivotal stage in signal timing, exerting a direct influence on the efficiency of traffic flow. An excessively long SC results in prolonged waiting times for vehicles, affecting the overall travel experience. Conversely, a SC that is too short leads to frequent stops for vehicles, reducing traffic efficiency. Hence, the constraint on the SC is given by

$$T_{\min} \leq T_i \leq T_{\max} i \in L_{insecs} \tag{23}$$

$$T_i = T_j \tag{24}$$

where L_{insecs} is the set of intersections; T_i denotes the SC duration for the i-th intersection in the region. To achieve effective coordinated control across intersections in the region, it is required that the SC durations for all intersections be the same.

GSR Constraint: The GSR plays a pivotal role in shaping the efficiency of traffic flow. Varied GSR dictate the scientific and rational timing for different signal phases. A higher GSR for a specific signal phase corresponds to an extended green light duration for that phase, consequently causing a reduction in the GSR for other signal phases. This, in turn, leads to diminished travel time for vehicles and an overall decline in road efficiency. Hence, the constraint on the GSR is given by

$$0 < \lambda_{\min} \leq \lambda_{i,j} \leq \lambda_{\max} < 1 \tag{25}$$

with $\lambda_{i,j}$ denoting the GSR for the j-th phase of the i-th intersection in the region.

3.1.3. *Regional average vehicle delay model.* Based on the above analysis, an optimization period consisting of N cycles is set, and a regional average vehicle delay model is established. It is represented as

$$PI = \min\left(\frac{\sum_{n=1}^{N}\sum_{i}^{L_{insecs}}\sum_{j=1}^{N_{i,phase}}\sum_{k=1}^{L_{i,j,flow}} d_{i,j,k}}{\sum_{n=1}^{N}\sum_{i}^{L_{insecs}}\sum_{j=1}^{N_{i,1iphase}}\sum_{k=1}^{L_{i,j,flow}} q_{i,j,k}}\right) \tag{26}$$

$$\begin{cases} T_{\min} \leq T_i \leq T_{\max}, i \in L_{insecs} \\ T_i = T_j \\ 0 < \lambda_{\min} \leq \lambda_{i,j} \leq \lambda_{\max} < 1 \end{cases} \tag{27}$$

where PI represents the optimization objective function, $N_{i,phase}$ indicates the number of phases at the i-th intersection, $L_{i,j,flow}$ represents the set of traffic volumes for the j-th phase at the i-th intersection, $d_{i,jk}$ denotes the average delay time for the k-th direction of the j-th phase at the i-th intersection in the region, and $q_{i,j,k}$ represents the corresponding traffic flow. The numerator denotes the total delay time for vehicles in the region, while the denominator represents the total traffic flow in the region.

3.2. **CNN-BiLSTM Model.** As shown in Figure 3, the CNN-BiLSTM model's input comprises three temporal windows of traffic flow data: the traffic flow data $X_n = (X_{t-H}, X_{t-H+1}, \ldots, X_t)$ from the previous temporal window, the data $X_w = (X_{t-H-w}, X_{t-H-w+1}, \ldots, X_{t-w})$ from the same time in the previous week, and the data $X_m = (X_{t-H-m}, X_{t-H-m+1}, \ldots, X_{t-m})$ from the same time in the previous month. Each temporal window consists of H time steps, and the traffic flow data for each time step can be represented as a two-dimensional matrix. The input for each of the three temporal windows is separately processed, with the H two-dimensional matrices from each time window stacked and fed into the CNN.

After extracting spatial features from the input data using CNN, the resulting spatial feature vector sequence (Y_n, Y_w, Y_m) is input into the BiLSTM. The BiLSTM is utilized to extract temporal features from the data, generating the spatiotemporal feature vector sequence Z. Subsequently, the outputs (Z_n, Z_w, Z_m) from the three temporal windows are stacked and input into an average pooling layer to extract the periodic features of the data. The final output of the model is the predicted traffic flow data for the next temporal window.

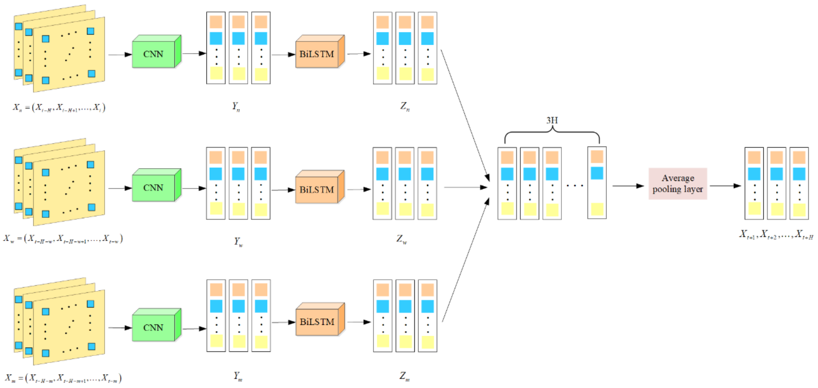

CNN
BiLSTM
CNN
BiLSTM
CNN
BiLSTM
$X_n = (X_{t-H}, X_{t-H+1}, \dots, X_t)$
Y_n
Z_n
$X_w = (X_{t-H-w}, X_{t-H-w+1}, \dots, X_{t-w})$
Y_w
Z_w
$X_m = (X_{t-H-m}, X_{t-H-m+1}, \dots, X_{t-m})$
Y_m
Z_m
3H
Average pooling layer
$X_{t+1}, X_{t+2}, \dots, X_{t+H}$

FIGURE 4. The structure of CNN-BiLSTM model

3.2.1. *Spatial Feature.* Intersections in a road network are interconnected, influencing TFP for a specific intersection by both upstream and downstream intersections. TFP relies on traffic flow data from the previous temporal window to predict patterns for the most recent window. This study focuses on a regional road network with nine intersections, illustrated in Figure 4(a), and represented as a two-dimensional matrix based on the intersection arrangement. Utilizing CNN, spatial features of traffic flow from MI are extracted, with input containing traffic flow data for H time steps.

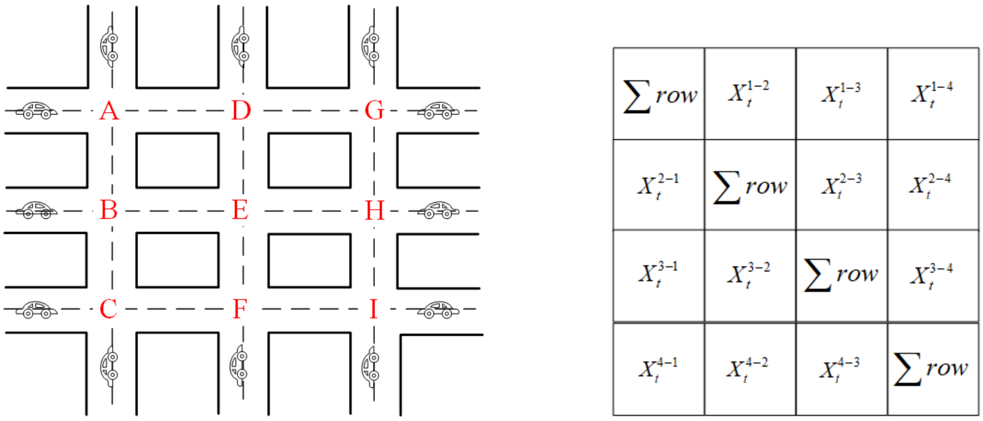
In Figure 4(b), inbound traffic flow data for an intersection at time step t is described as an OD matrix, associating each intersection with 12 traffic flow directions. Elements in the OD matrix represent traffic flow for specific directions at time step t. The $\sum$ row indicates the cumulative flow on that road, promoting interconnections among traffic directions.

When describing the data at time step t as a two-dimensional matrix, it is possible to consider using this matrix as the input for CNN, denoted as X_t, with X_t^n serving as the input
for the $(n+1)$-th convolutional layer. Therefore, the feature representation extracted at time step t can be expressed as follows

$$X_t^n = \sigma_n \left(X_t^{n-1} * W_t^n + b_t^n \right) \tag{28}$$

where σ_n represents the activation function for the n-th layer, $*$ represents the convolution operation, and ReLU is selected as the activation function. In addition, W_t^n and b_t^n represent the weights and bias terms in the nth convolutional layer, respectively.

As depicted in Figure 5, the CNN model extracts spatial features of traffic flow using a 2D convolutional layer with a convolutional kernel size of (3,3). Following the convolution, the ReLU activation function and a Dropout layer are applied to enhance the neural network's performance. ReLU introduces non-linearity to the neuron



(a) Transportation Network Schematic

$\sum row$	X_t^{1-2}	X_t^{1-3}	X_t^{1-4}
X_t^{2-1}	$\sum row$	X_t^{2-3}	X_t^{2-4}
X_t^{3-1}	X_t^{3-2}	$\sum row$	X_t^{3-4}
X_t^{4-1}	X_t^{4-2}	X_t^{4-3}	$\sum row$

(b) OD Matrix

FIGURE 5. Regional Transportation Road Network Structure Map

outputs, effectively mitigating the vanishing gradient problem, thereby improving the training effectiveness and generalization ability of the neural network. The Dropout layer randomly discards the output of some neurons to prevent overfitting, thereby enhancing the generalization and robustness of the neural network.

The output X_t^n of the nth convolutional layer at time step t is connected via residual connections to better fit the training data. Finally, the spatial feature vector Y_t is obtained as the input to the fully connected layer. This vector is then used as input to the BiLSTM model to extract temporal features of traffic flow from MI.

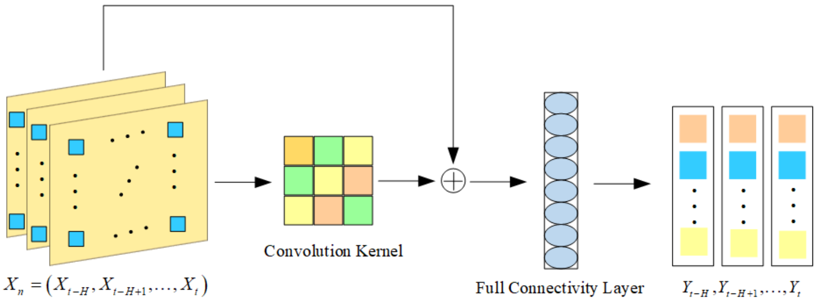

$X_n = (X_{t-H}, X_{t-H+1}, \ldots, X_t)$
Convolution Kernel
Full Connectivity Layer
$Y_{t-H}, Y_{t-H+1}, \ldots, Y_t$

FIGURE 6. The structure of CNN model

3.2.2. *Time Series Feature.* TFP is a typical time series forecasting task aimed at using historical traffic flow data to predict future traffic flow data. This paper employs the BiLSTM model to extract features from time series and the input $Y = (Y_{t-H}, Y_{t-H+1}, \ldots, Y_t)$ is a spatial feature vector sequence containing H time steps. Here, Y_t represents the spatial feature vector obtained as output after inputting the traffic flow data at time step t into CNN, and H is the time step length.

While much valuable information can be learned from historical traffic flow data, directly inputting this data into the model increases the dimensionality, resulting in computationally complex calculations. To mitigate this, an AutoEncoder is utilized firstly for feature extraction from historical traffic flow data, with the extracted features incorporated as part of the input data. Specifically, the raw data is first subjected to feature extraction through the encoder, yielding $Y_{AE} \in R^h$, where h is the dimensionality of the extracted features. Subsequently, the encoder's output serves as the input to the decoder, and through the decoding process, the effectiveness of the extracted features is validated, as illustrated in Figure 6. Finally, the AutoEncoder is trained by minimizing the reconstruction error. It can be given by

$$L\left(Y', Y'_{AE}\right) = \frac{1}{2}\sum_{i=1}^{n}\left|Y'_i - Y'_{AE_i}\right|^2 \tag{29}$$

where Y' denotes the input features after tiling and Y'_{AE} represents the decoded results.

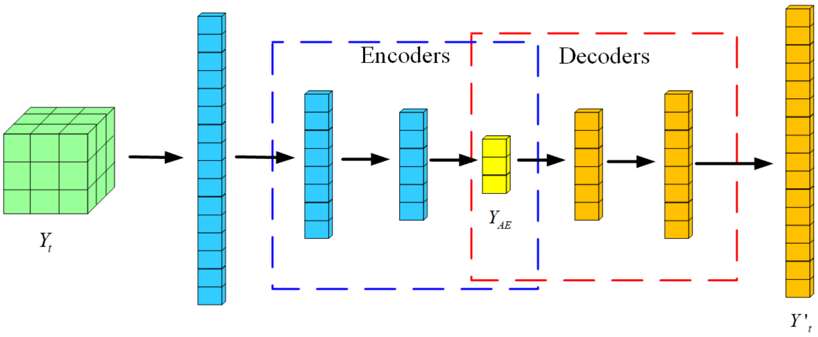


Figure 7. AutoEncoder schematic

As shown in Figure 7, BiLSTM is a variant of RNNs designed to handle sequential data more effectively. It extends the LSTM architecture by introducing two sets of hidden layers. The forward hidden layer processes input sequences in chronological order, while the backward hidden layer processes input sequences in reverse order. These two sets of hidden layers operate in both chronological and reverse chronological order. This allows the network to learn information from both past and future contexts simultaneously, enabling the model to better understand the overall context of the input sequence. With the input vector Y'_{AE}, the predicted result Z_t can be obtained, which is expressed as follows

$$Z_t = f\left(O_t \odot \tanh\left(C_t\right) + O'_t \odot \tanh\left(C'_t\right)\right) \tag{30}$$

$$O_t = \sigma\left(WX'_t + WH_{t-1} + b\right) \tag{31}$$

$$O'_t = \sigma\left(WX'_t + WH'_{t-1} + b\right) \tag{32}$$

where O_t and O'_t represent the forward output gate and the backward output gate, respectively. C_t and C'_t represent the forward hidden state and the hidden state for the backward process, respectively. H_{t-1} and H'_{t-1} represents the previous layer's forward hidden state and the previous layer's backward hidden state, respectively.

Generally, BiLSTM and BiGRU perform similarly in handling sequence data, effectively capturing long-term dependencies. However, when dealing with large-scale data or significant noise, BiLSTM tends to be more stable than BiGRU. This is because BiLSTM can better control and maintain long-term memory. Given the large scale of traffic flow data and the presence of sensor noise in this study, the BiLSTM is ultimately chosen.

3.2.3. *Periodicity Feature.* The final outputs from BiLSTM consist of three distinct spatiotemporal feature vector sequences: Z_n, Z_w, and Z_m. These three vectors are stacked and fed into the average pooling layer to extract the periodic features of traffic flow from MI. The calculation method for the average pooling layer is as follows

$$\hat{q}_{t+1} = AvgPooling\left(Z_{now}, Z_{week}, Z_{month}\right) \tag{33}$$

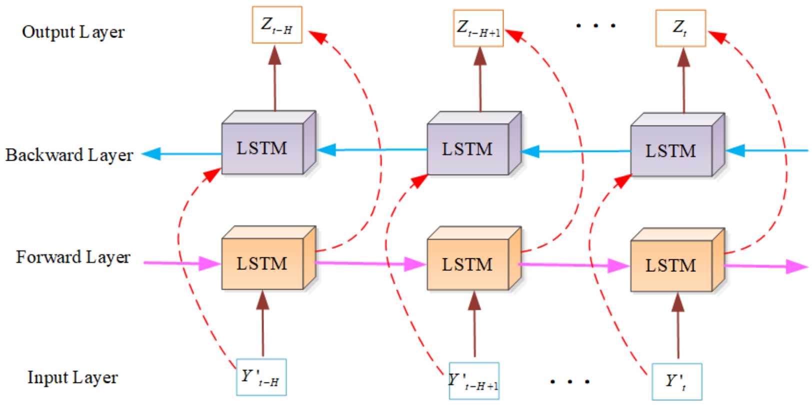


Figure 8. The structure of BiLSTM model

where $\hat{q}_{t+1}$ denotes the predicted traffic flow data. The structure of the average pooling layer is illustrated in Figure 8. It calculates the average value of feature points within a neighborhood and integrates these values to obtain new features. In comparison to fully connected layers, the average pooling layer significantly reduces the number of network parameters, thereby mitigating the occurrence of overfitting. The final output of the average pooling layer is the predicted traffic flow for the next temporal window.

It is noteworthy that the activation function chosen for the average pooling layer is the Gaussian error linear units (GELU). GELU is a high-performance neural network activation function with a non-linear transformation that aligns with the desired random regularization, leading to superior performance. Its computation is represented as follows:

$$xP(X \leq x) = x\Phi(x) \tag{34}$$

where $\Phi(x)$ represents the cumulative distribution function of the Gaussian normal distribution for x. The GELU activation function, based on the concept of random regularization, provides a probabilistic description of neuron inputs, aligning more closely with natural cognitive patterns. Therefore, this function can more accurately and intuitively describe the activation state of neurons, offering certain theoretical advantages.

3.3. **Optimization of RTS coordination.** The BWO algorithm, as a global search algorithm, is characterized by its simple structure and fast computation speed, enabling rapid convergence to the global optimum. However, as the iterations progress, individuals in the population tend to converge towards the global optimum, resulting in decreased diversity. This reduction in diversity makes it challenging to escape local optima, leading to slower convergence speeds and decreased computational accuracy over time.

The quality of the initial population and its reasonable distribution play a crucial role in allowing the algorithm to find excellent results from the beginning, significantly reducing convergence time. The issue of local optima has always been a significant factor affecting optimization precision and speed. The WWO algorithm possesses a strong ability to escape local optima, thanks to its unique reflection operation that can reflect contrasting results for local optimum solutions. The selection between the two solutions enhances the algorithm's capability to escape local optima.

To improve the computational accuracy, convergence speed, and the ability to escape local optima, and to enhance the overall performance of the algorithm, a strategy is

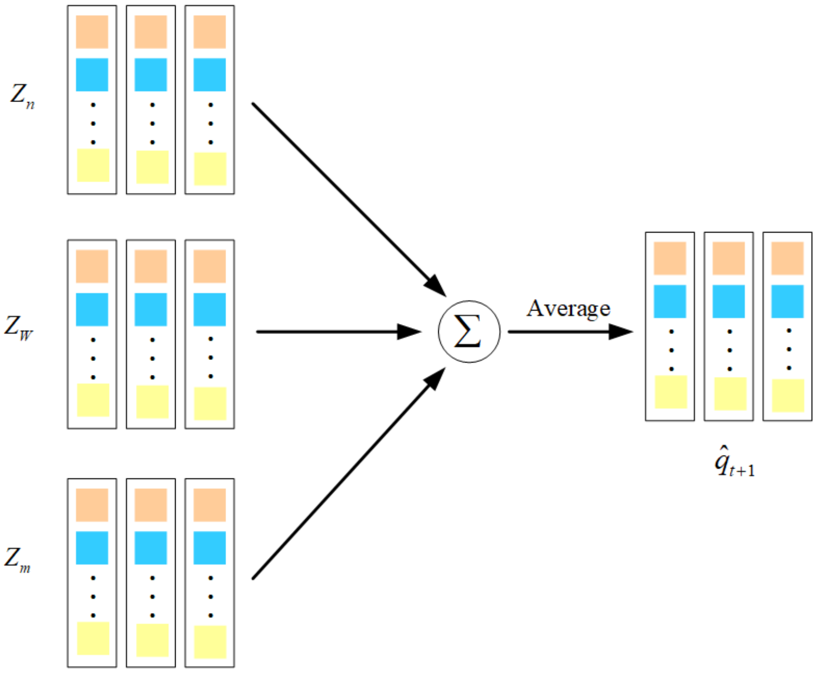


Figure 9. Schematic structure of the average pooling layer

employed that leverages chaotic mapping and backward learning to enhance the quality of the initial population and accelerate convergence. Additionally, the introduction of the reflection operation from the WWO algorithm aims to reduce the probability of the algorithm falling into local optima, ultimately improving computational accuracy.

3.3.1. *Chaotic reverse learning strategy.* The BWO algorithm employs random variables that adhere to a standard probability distribution for generating the initial population, introducing considerable randomness and uncertainty in the produced individuals. This inherent randomness, coupled with the algorithm's limited optimization and convergence capabilities, presents challenges. Consequently, the TWBWO algorithm proposed in this paper addresses these issues by integrating chaotic mapping and backward learning into the process of generating the initial population. Furthermore, it adopts a multi-population approach to augment the diversity of the evolving population.

Chaotic mapping possesses excellent randomness and traversal characteristics. Its mapping of population members helps enhance the algorithm's global optimization capability, thereby improving overall algorithm performance. Hence, in this paper, we employ the tent mapping method from chaotic mapping. Tent mapping exhibits better distributional characteristics, providing a more uniform distribution that effectively generates a diverse initial population. Additionally, the simple structure of the tent mapping enhances its adaptability and compatibility with algorithms. Tent mapping is a piecewise linear mapping, and its function is defined as follows

$$x_{n+1} = \begin{cases} x_n/\alpha, x_n \in [0, \alpha) \\ (1 - x_n)/(1 - \alpha), x_n \in [\alpha, 1] \end{cases} \tag{35}$$

with d being the index of the chaotic variable, equivalent to the dimension of individuals in the algorithm. The parameter α belongs to the interval $(0, 1)$, and it is typically optimal when α is set to 0.5 .

Following the Tent mapping relationship, we can obtain the following initial function

$$X_{i,d}^{t} = X_{\min}^{t} + \left(X_{\max}^{t} - X_{\min}^{t}\right) \cdot Z_{i,d}^{t} \tag{36}$$

where $Z_{i,d}^{t}$ represents the chaotic sequence obtained after i iterations, $X_{\min}^{t}$ is the lower boundary, and $X_{\max}^{t}$ is the upper boundary.

Secondly, the generated numerous individuals are evenly divided into several sub-populations. Reverse populations for each sub-population are then generated through backward learning. The calculation method for backward learning is as follows

$$OBX_{i,d}^{t} = X_{\min}^{t} + X_{\max}^{t} - X_{i,d}^{t} \tag{37}$$

with $X_{i,d}^{t}$ represents the population generated through Tent mapping, and $OBX_{i,d}^{t}$ is the new population generated through backward learning. It is crucial to note that in this step, the fitness values of the original individuals and the reverse individuals need to be compared. If the reverse individuals exhibit superior fitness, they replace the original individuals. The fitness values of individuals are calculated based on the fitness function, and a selection process is employed by sorting individuals according to their fitness values to obtain the final initial sub-population.

3.3.2. *WWO algorithm.* To improve the efficiency of finding the optimal solution, the WWO algorithm was introduced in 2015, inspired by the shallow water wave theory [32]. This algorithm simulates the propagation, refraction, and breaking wave movements observed in water wave motion. In this algorithm, the fitness value of the water wave is inversely proportional to the depth of the seabed and the wavelength. The closer the distance to the static water level, the higher the fitness value, and the shorter the wavelength. By controlling the wavelength, the algorithm simulates the search range of water waves. Water waves with low fitness values conduct extensive searches, maximizing the possibility of finding better solutions. On the other hand, water waves with high fitness values perform localized searches within a small range, ensuring the quality of the solution.

In each iteration, the population undergoes a propagation operation where each water wave individual x generates a new individual solution by adding random displacements to each dimension d. After the propagation operation updates the wavelength, the wave height of water wave individuals decays. When the wave height gradually decreases to zero, the algorithm enters the refraction operation, allowing the water wave individual to learn from the current best individual, preventing the algorithm from stagnating. Upon discovering a new optimal water wave, the algorithm performs a breaking wave operation to decompose it, enabling a more detailed local search to ensure the superiority of the solution.

In order to enhance the ability of the BWO algorithm to escape local optima, improve convergence speed, and increase computational accuracy, the WWO is integrated with the BWO algorithm. After the BWO algorithm finds the current global optimum, a refraction operation is introduced. This refraction learning is applied to the optimal solution, and by comparing the fitness values of the new individual with the current optimal individual, the superior value is selected to replace and enter the next iteration. The improved formula is as follows:

$$X_d = N\left(\frac{X_{best} + X_m}{2}, \frac{|X_{best} - X_m|}{2}\right) \tag{38}$$

$$X_m = mean\left(X_{best} + X_{worst}\right) \tag{39}$$

where X_{best} and X_{worst} represent the best and worst individuals in the current population, and X_m is the average position of the two. $N(\mu, \sigma)$ denotes a Gaussian random number with a mean μ and standard deviation σ.

3.3.3. *TWBWO algorithm.* The TWBWO algorithm is an improved BWO algorithm that combines chaotic reverse learning strategy and WWO algorithm. The flow chart is shown in Figure 9 and the specific steps are as follows

Step 1: Initialize the parameters.

Step 2: Use the chaotic backward learning strategy for initialization to obtain a new initial population. Check if it exceeds the search boundary, calculate the fitness values, and sort them to obtain the current best and worst individual solutions.

Step 3: Calculate the balance factor B_f and whale fall probability W_f according to Formulas (12) and (19).

Step 4: If $B_f < 0.5$, enter the development phase, update individual positions according to Equation (14); if $B_f > 0.5$, enter the exploration phase, update individual positions according to Equation (13).

Step 5: After completing Step 3 or Step 4, compare the balance factor B_f and whale fall probability W_f. If $B_f < W_f$, enter the whale fall phase, and generate new white whale individuals according to Equation (17).

Step 6: After all individuals have undergone one iteration, calculate fitness values, find new current optimal and worst solutions, perform reflection operations on the optimal solution according to Equations (40) and (41), calculate fitness values, and compare to save the optimal solution.

Step 7: Check if the maximum number of iterations or stopping conditions are met. If not, return to Step 3 until the conditions are met, then output the final result.

4. Experiments Results.

4.1. Simulation Dataset.

AnyLogic is a versatile simulation tool built on the Java platform. It excels in discrete event simulation, system dynamics simulation, and agent-based simulation. The Road Traffic Library within AnyLogic is specifically designed to facilitate the modeling, simulation, and visualization of vehicular traffic. This library enables a detailed and efficient representation of vehicle motion at the physical layer, making AnyLogic a powerful choice for modeling scenarios involving road traffic, street traffic, transportation within production sites, parking lots, or any system where vehicles, roads, and lanes are essential components.

In this study, AnyLogic serves as the foundation for constructing a microsimulation model of a regional road network, enabling the realistic simulation of traffic conditions. The model encompasses nine intersections, and the simulation dataset is generated by statistically capturing traffic flow data from all intersections in the region. The dataset spans three months, with hourly intervals for statistical collection of traffic flow data at every intersection. In the simulation of real-world traffic scenarios, considerations such as morning peak hours, weekends, and holidays are factored in to introduce randomness, enhancing the simulation data's resemblance to actual traffic situations.

4.2. Data preprocessing.

Prior to inputting data into the model, it is crucial to perform data standardization. Data standardization refers to scaling data with different attributes to the same range, eliminating dimensional differences between various attributes and enhancing the accuracy and reliability of data analysis.

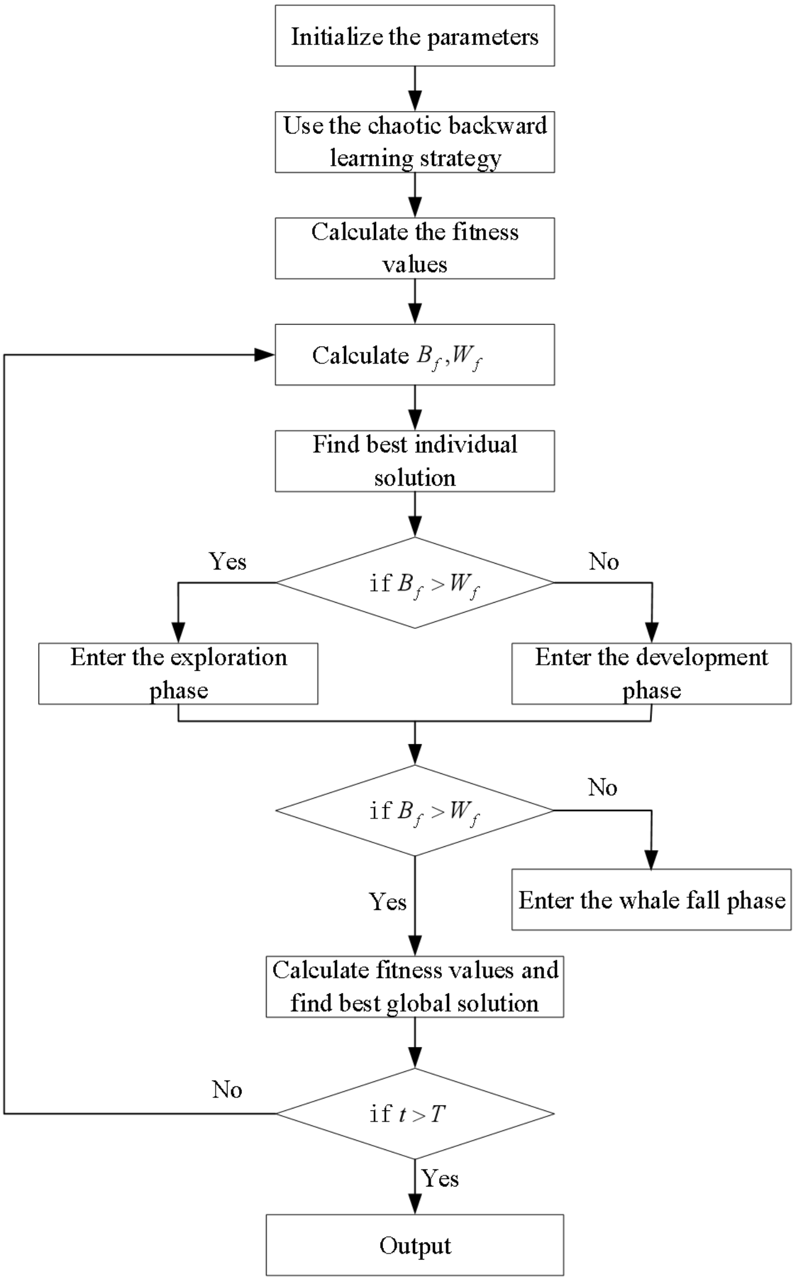

Initialize the parameters
Use the chaotic backward learning strategy
Calculate the fitness values
Calculate B_f, W_f
Find best individual solution
Yes
No
if $B_f > W_f$
Enter the exploration phase
Enter the development phase
if $B_f > W_f$
No
Yes
Enter the whale fall phase
Calculate fitness values and find best global solution
No
if $t > T$
Yes
Output

Figure 10. The flow chart of TWBWO

In this paper, we adopt the min-max normalization method to preprocess the data. Specifically, this method first centralizes the data x based on the minimum value and then scales it using the range (the difference between the maximum and minimum values), adjusting the data range to [0, 1.0]. This standardization process reduces the search range during optimization, resulting in a smoother optimization process and facilitating the identification of the optimal solution. The calculation formula for min-max normalization is as follows

$$x = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{40}$$

4.3. **Experimental Setting.** The experiments in this paper were conducted using the Python programming language. PyCharm was employed as the local compiler, and the specific experimental environment configurations are outlined in Table 1.

Table 1. Configuration of experimental environment

Name	Model or Version
CPU processor	Intel(R) Xeon(R) Silver 4210
RAM	256 GB
Python	3.8 .15
Pytorch	1.12 .1
Cuda	11.2
Numpy	1.23 .5
Pandas	1.5 .2

In terms of network configuration, the BiLSTM network had 3 layers in the recurrent part, with a hidden dimension of 64*2. During the training of the model, RMSE was selected as the loss function, Adam was chosen as the optimizer, the batch size was set to 128, the learning rate was 0.01, and the training lasted for 500 epochs with early stopping configured to trigger after 50 steps. Additionally, for a more comprehensive evaluation of the algorithm's predictive performance, this paper utilized Mean Square Error (MSE) and Mean Absolute Error (MAE) to measure the errors between predicted values and actual values. Smaller MSE and MAE values indicate higher predictive accuracy of the model.

4.4. **Experiments Results and Analysis.** In this section, CNN-BiLSTM is compared with some typical models, involving CNN, LSTM, DISTN [16], BiLSTM, BiGRU, and Informer [33], where Informer is a time series prediction model based on Transformer. As depicted in Figure 10, CNN-BiLSTM proves to be more suitable for multi-intersection TFP tasks compared to DISTN. This advantage arises from BiLSTM's proficiency in establishing long-distance dependencies and parallel computation, effectively overcoming the limitation of LSTM, which relies on the computation of the previous time step. Similarly, BiGRU is an improved model of GRU that effectively overcomes the limitation of GRU's dependency on the previous time step calculation. BiGRU generally performs almost identically to BiLSTM, but BiLSTM exhibits better stability in large data scenarios. Informer has demonstrated outstanding performance in time series prediction tasks across diverse domains, showcasing its strengths in forecasting sequential data over time. However, in this experiment, the relatively lower predictive accuracy of Informer may be attributed to its weaker capability in capturing the features of multi-intersection traffic flow data.

The results depicted in the figure illustrate that CNN-BiLSTM outperforms other models in multi-intersection traffic flow prediction tasks, as evidenced by its lowest values in both MAE and MSE metrics. This highlights the significant advantage of CNN-BiLSTM in terms of predictive accuracy. Specifically, CNN-BiLSTM surpasses other models such as LSTM, DISTN, BiLSTM, BiGRU, and Informer in both MAE and MSE performance. The key to CNN-BiLSTM's superior performance lies in the precision of CNN in extracting spatial features from multi-intersection traffic flow data, enabling the model to capture the complex characteristics of traffic flow more accurately and achieve more precise TFP. Overall, the experimental results convincingly demonstrate the effectiveness and superiority of the CNN-BiLSTM model in multi-intersection traffic flow prediction tasks.

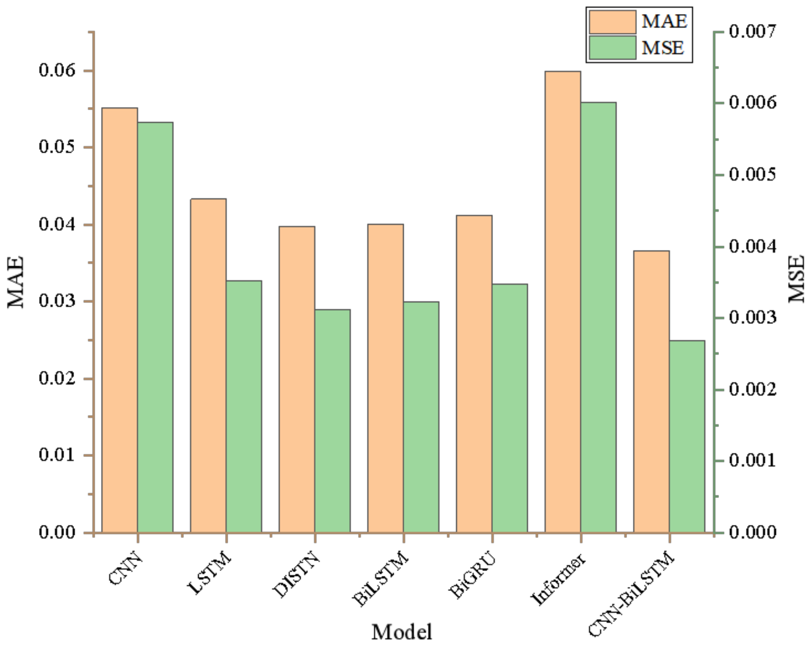
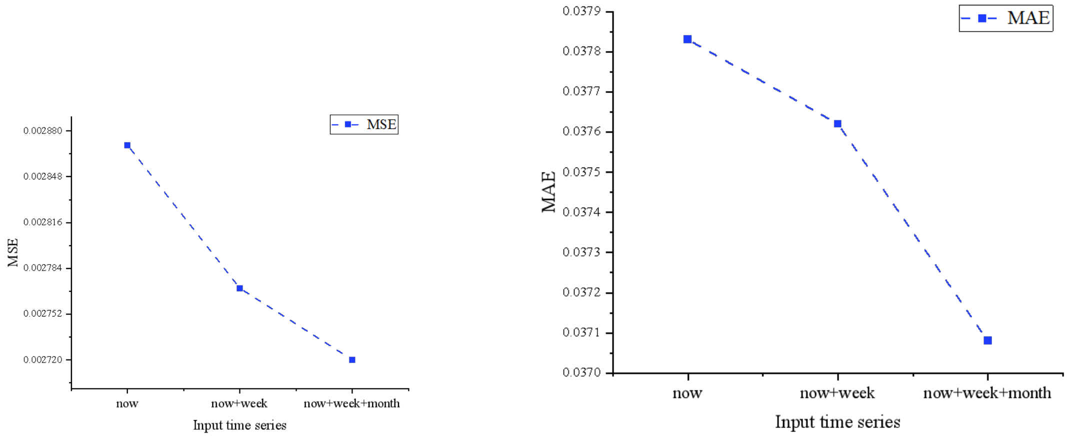


Figure 11. Comparison of experimental results

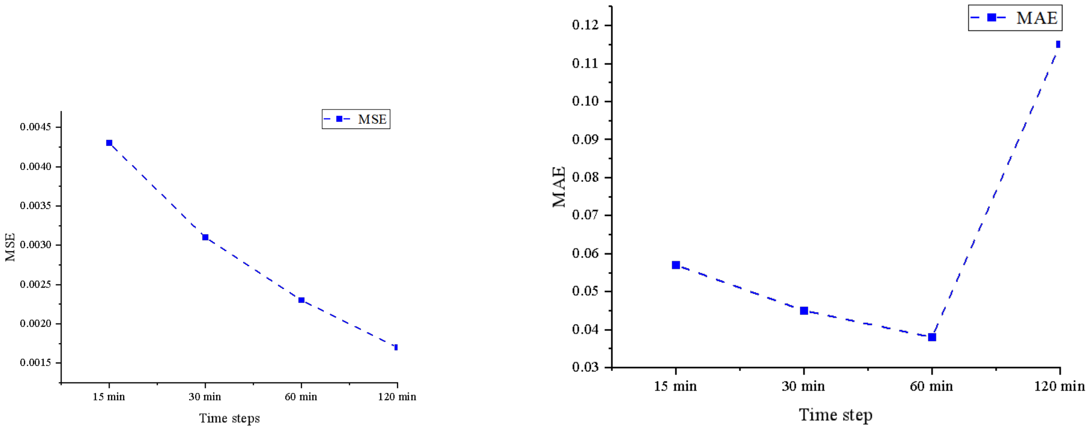
Figure 11 illustrates the impact of different time series on prediction accuracy. Here, now denotes the traffic flow data X_{now}, week represents the traffic flow data X_{week}, and "month" denotes the traffic flow data X_{month}. The results indicate that the prediction accuracy is highest when the model is provided with a time series containing X_{now}, X_{week}, and X_{month}. This suggests that learning the periodic characteristics of traffic flow contributes to improved prediction accuracy.



(a) MSE (b) MAE

Figure 12. Impact of different time series on prediction accuracy

Figure 12 presents the prediction accuracy of traffic flow at different time steps. The graph illustrates that as the time step increases from 15 minutes to 1 hour, the prediction accuracy improves. This is attributed to the fact that with a larger time step, the variation in traffic volume becomes more uniform, thereby reducing the complexity of prediction. However, as the time step further increases from 1 hour to 2 hours, the prediction accuracy decreases. The analysis suggests that BiLSTM requires a substantial amount of training data, and as the time step increases, the model captures fewer temporal features of traffic flow, leading to increased prediction difficulty.



(a) MSE

(b) MAE

Figure 13. Prediction accuracy of traffic flow at different time steps

Subsequently, in order to validate the superior performance of TWBWO in optimizing RTS coordination control, this paper will conduct a comparative analysis of the optimization outcomes achieved by TWBWO, PSO, and BWO, respectively. To ensure consistency in iterations, the maximum iteration times for all three algorithms are set to 400 , with a population size of 500 . The algorithms uniformly adopt integer encoding, rounding variables to the nearest integer during the iteration process. The TWBWO and BWO utilize a greedy selection strategy, replacing the original individual with the updated one if it is superior, otherwise retaining the original individual. Parameters for the PSO are chosen as follows: particle velocity: $V_{\max} = 1.5, V_{\min} = 0.5$; regional traffic constraint parameters: SC: $T_{\max} = 180$, $T_{\min} = 80$, phase green time: $X_{\max} = 70, X_{\min} = 10$, green time ratio: $\lambda_{\max} = 0.5, \lambda_{\min} = 0.1$.

The comparative optimization performance of the PSO, BWO, and TWBWO is depicted in Figure 13. The graph reveals that, in contrast to the PSO, the BWO exhibits heightened convergence accuracy and precision. Consequently, this paper incorporates an enhancement strategy into the BWO, leading to the proposal of the TWBWO. Relative to the BWO, the preliminary results of the TWBWO showcase the efficacy of the refined population initialization method. Moreover, the algorithm's global optimization capability experiences a notable enhancement, ultimately yielding the optimal signal timing plan. This outcome further validates the usefulness of the proposed TWBWO.

Simulation experiments were executed employing dynamic RTS control methods relied on PSO, BWO, and TWBWO, with a comparison made against the fixed timing control method. The efficacy of the optimization control methods was showcased by computing the average number of stops during each hourly time period. The simulation results are depicted in Figure 14. The graph illustrates that, in comparison to the fixed timing control method, dynamic RTS control methods can significantly decrease the

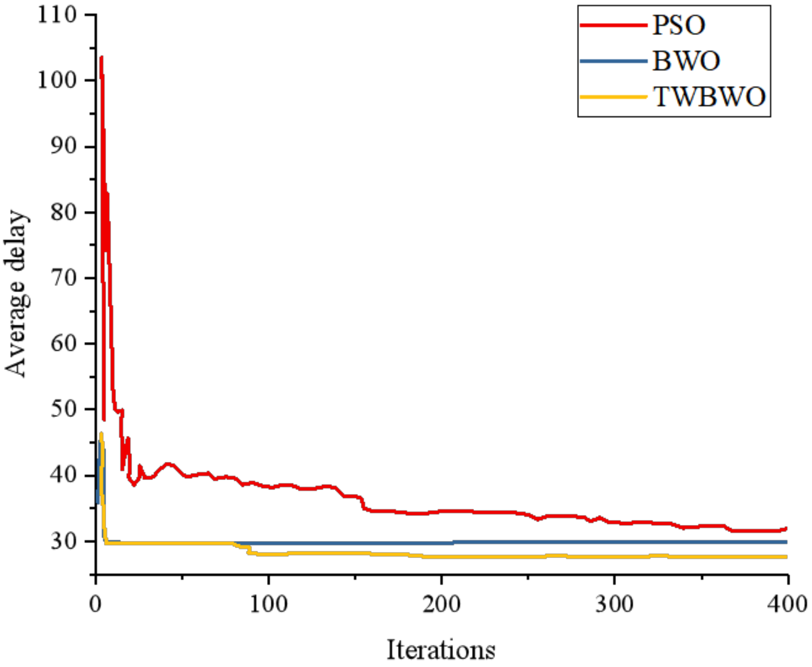


Figure 14. Comparative optimization performance of PSO, BWO and TWBWO

average number of stops for vehicles in the region. This improvement is attributed to the multi-intersection TFP method outlined in Section 3, which adeptly captures changes in traffic conditions. When coupled with the regional average delay model introduced in Section 3, the signal phase timing is dynamically adjusted to enhance regional traffic efficiency. This outcome confirms the validity of CNN-BiLSTM.

In contrast to the PSO, the BWO consistently delivers superior optimization results across all time periods. The analysis implies that the PSO is more susceptible to local optima and exhibits lower convergence accuracy compared to the BWO. Over the entire time span, the utilization of TWBWO leads to the fewest average number of vehicle stops. This is attributed to the TWBWO's exceptional optimization performance, producing optimal results and thereby enhancing regional traffic efficiency.

Upon calculation, it is observed that when employing the fixed signal timing control method, the average number of stops for vehicles in the region amounts to 2.18 times per day. In contrast, with the implementation of PSO and BWO, the average number of stops decreases to 1.93 and 1.84 times, respectively. Comparative to the fixed signal timing control method, the optimization rates for PSO and BWO are 11.2% and 15.3%, respectively, underscoring the superiority of dynamic RTS control methods. With the application of TWBWO, the average number of stops further reduces to 1.72 times, representing a 7.4% improvement compared to BWO. This highlights that the proposed algorithm yields the most optimal results among the considered methods.

5. **Conclusion.** This study introduces a deep learning model, CNN-BiLSTM, for achieving TFP at intersection entrances to enhance the real-time capabilities of RTS control. Additionally, by establishing a model for the average delay of regional vehicles, the TWBWO is proposed to optimize the model and achieve coordinated RTS control. The combination of TFP and RTS coordination optimization methods is employed to realize dynamic RTS control, thereby improving the overall coordination of RTS control. Experimental results demonstrate that CNN-BiLSTM exhibits higher prediction

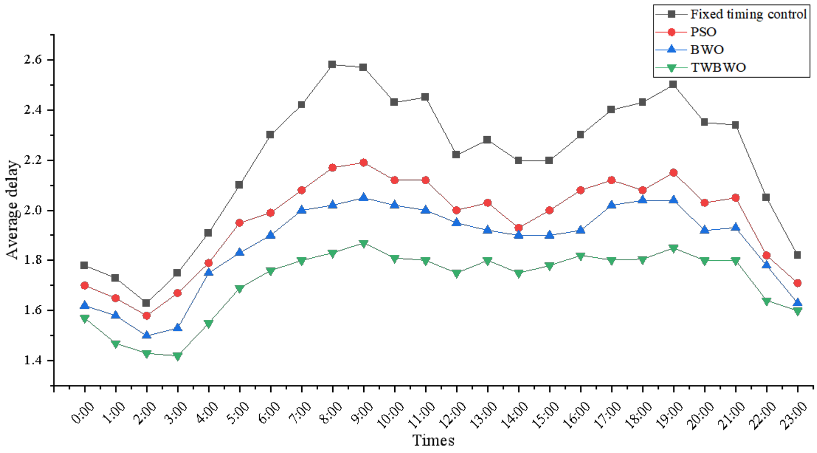


Figure 15. Comparison of simulation results

accuracy compared to existing models for predicting traffic flow at MI. Furthermore, the proposed TWBWO outperforms popular intelligent optimization algorithms, showcasing superior global search capabilities and rapid convergence. However, this study focuses on a small regional road network consisting of nine intersections, with relatively uniform spacing between the intersections. When the structure of the regional road network is complex and traffic conditions are chaotic, achieving the expected prediction results may be challenging. Additionally, the delay characteristics between different network structures have not been considered. Future research directions include introducing reinforcement learning strategies to achieve traffic flow control.

Acknowledgment. This work is partially supported by the Natural Science Foundation of Fujian Province of China (No. 2021J011187), the National Social Science Fund of China (No. 21BTJ011).

REFERENCES

[1] L. Guo, Y. Cui, H. Liang, and Z. Zhou, "Spectral bisection community detection method for urban road networks," in *2021 40th Chinese Control Conference (CCC)*. IEEE, 2021, pp. 806–811.

[2] F. Qi, R. He, L. Yan, J. Yao, P. Wang, and X. Zhao, "Traffic signal control with deep q-learning network (dqn) algorithm at isolated intersection," in *2022 34th Chinese Control and Decision Conference (CCDC)*. IEEE, 2022, pp. 616–621.

[3] P. Yu and J. Luo, "Minimize pressure difference traffic signal control based on deep reinforcement learning," in *2022 41st Chinese Control Conference (CCC)*. IEEE, 2022, pp. 5493–5498.

[4] N. Wu, D. Li, and Y. Xi, "Distributed weighted balanced control of traffic signals for urban traffic congestion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3710–3720, 2018.

[5] S. Shahriari, M. Ghasri, S. A. Sisson, and T. Rashidi, "Ensemble of arima: combining parametric and bootstrapping technique for traffic flow prediction," *Transportmetrica A: Transport Science*, vol. 16, no. 3, pp. 1552–1573, 2020.

[6] P. Wang, Y. Zhang, T. Hu, and T. Zhang, "Urban traffic flow prediction: A dynamic temporal graph network considering missing values," *International Journal of Geographical Information Science*, vol. 37, no. 4, pp. 885–912, 2023.

[7] N. Kumar, S. S. Rahman, and N. Dhakad, "Fuzzy inference enabled deep reinforcement learning-based traffic light control for intelligent transportation system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 8, pp. 4919–4928, 2020.

[8] I. Cvitić, D. Peraković, M. Periša, and M. D. Stojanović, "Novel classification of iot devices based on traffic flow features," *Journal of Organizational and End User Computing (JOEUC)*, vol. 33, no. 6, pp. 1–20, 2021.

[9] S. Fang, V. Prinet, J. Chang, M. Werman, C. Zhang, S. Xiang, and C. Pan, "Ms-net: Multi-source spatio-temporal network for traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 7142–7155, 2021.

[10] C. Wu, S. Zhou, C. Fan, and L. Qi, "A method for short-term traffic flow prediction and control on urban road," in *2021 7th International Conference on Big Data Computing and Communications (BigCom)*. IEEE, 2021, pp. 263–270.

[11] S. V. Kumar and L. Vanajakshi, "Short-term traffic flow prediction using seasonal arima model with limited input data," *European Transport Research Review*, vol. 7, no. 3, pp. 1–9, 2015.

[12] X. Chen, S. Wu, C. Shi, Y. Huang, Y. Yang, R. Ke, and J. Zhao, "Sensing data supported traffic flow prediction via denoising schemes and ann: A comparison," *IEEE Sensors Journal*, vol. 20, no. 23, pp. 14 317–14 328, 2020.

[13] S. AlKheder, W. Alkhamees, R. Almutairi, and M. Alkhedher, "Bayesian combined neural network for traffic volume short-term forecasting at adjacent intersections," *Neural Computing and Applications*, vol. 33, pp. 1785–1836, 2021.

[14] E. Ning, C. Wang, H. Zhang, X. Ning, and P. Tiwari, "Occluded person re-identification with deep learning: A survey and perspectives," *Expert Systems with Applications*, vol. 239, p. 122419, 2024.

[15] D. Kim and O. Jeong, "Cooperative traffic signal control with traffic flow prediction in multi-intersection," *Sensors*, vol. 20, no. 1, p. 137, 2019.

[16] W. Li, X. J. Ban, J. Zheng, H. X. Liu, C. Gong, and Y. Li, "Real-time movement-based traffic volume prediction at signalized intersections," *Journal of Transportation Engineering, Part A: Systems*, vol. 146, no. 8, p. 04020081, 2020.

[17] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[18] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–73, 1992.

[19] Y. Zhang, H. B. Zhu, X. Q. Liu, and X. B. Chen, "Optimal control for region of the city traffic signal base on selective particle swarm optimization algorithm," in *2017 36th Chinese Control Conference (CCC)*. IEEE, 2017, pp. 2723–2728.

[20] Y. Zhang and H. B. Zhu, "Optimal control for region of the city traffic signal based on apsowm," in *2017 29th Chinese Control And Decision Conference (CCDC)*. IEEE, 2017, pp. 2412–2417.

[21] A. S. N. Zainab, I. Soesanti, and D. R. Utomo, "Detection of covid-19 using cnn's deep learning method," in *2022 4th International Conference on Biomedical Engineering (IBIOMED)*. IEEE, 2022, pp. 59–64.

[22] D. Ma, J. Xiao, X. Song, X. Ma, and S. Jin, "A back-pressure-based model with fixed phase sequences for traffic signal optimization under oversaturated networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 9, pp. 5577–5588, 2020.

[23] C. C. J. Kuo, "Understanding convolutional neural networks with a mathematical model," *Journal of Visual Communication and Image Representation*, vol. 41, pp. 406–413, 2016.

[24] K. Duvvuri, H. Kanisettypalli, and S. Jayan, "Detection of brain tumor using cnn and cnn-svm," in *2022 3rd International Conference for Emerging Technology (INCET)*. IEEE, 2022, pp. 1–7.

[25] T. Bao, N. Ren, R. Luo, B. Wang, G. Shen, and T. Guo, "A bert-based hybrid short text classification model incorporating cnn and attention-based bigru," *Journal of Organizational and End User Computing (JOEUC)*, vol. 33, no. 6, pp. 1–21, 2021.

[26] X. Li and H. Li, "Feature point detection method of pig face based on convolutional neural network," *Journal of Jilin University Science Edition*, vol. 60, no. 3, pp. 609–616, 2022.

[27] J. Chen, H. Wang, X. Zhu, Y. Jiang, and X. Chen, "Time aware sequence recommendation algorithm based on long-term memory enhancement," *Journal of Jilin University Science Edition*, vol. 60, no. 4, pp. 919–928, 2022.

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

[29] C. Zhong, G. Li, and Z. Meng, "Beluga whale optimization: A novel nature-inspired metaheuristic algorithm," *Knowledge-Based Systems*, vol. 251, p. 109215, 2022.

[30] A. J. Miller, "Settings for fixed-cycle traffic signals," *Journal of the Operational Research Society*, vol. 14, no. 4, pp. 373–386, 1963.

[31] R. Akçelik and N. M. Rouphail, “Overflow queues and delays with random and platooned arrivals at signalized intersections,” *Journal of Advanced Transportation*, vol. 28, no. 3, pp. 227–251, 1994.

[32] Y. J. Zheng and B. Zhang, “A simplified water wave optimization algorithm,” in *2015 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2015, pp. 807–813.

[33] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, “Informer: Beyond efficient transformer for long sequence time-series forecasting,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, 2021, pp. 11 106–11 115.