Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Deep Learning with Improved Metaheuristic Optimization for Traffic Flow Prediction

Zhizhong Wu
College of Engineering, UC Berkeley, Berkeley, US
Corresponding Author: Zhizhong Wu, **E-mail**: ecthelion.w@gmail.com

| ABSTRACT
Aiming at the current dilemma of inaccurate prediction accuracy in the field of traffic flow prediction, this paper proposes a novel traffic flow prediction method using the Revised Enhanced Extreme Gray Wolf Optimizer (REEGWO) to optimize convolutional neural networks (CNNs) and combining with bi-directional long and short-term memory (BiLSTM) networks. The experimental results show that the model can effectively converge the training loss error and RMSE, and significantly outperforms the existing classical methods in terms of goodness-of-fit, average absolute error, average deviation error and average absolute percentage error, providing an efficient and accurate solution for traffic flow prediction.

| KEYWORDS
Metaheuristic Optimization, CNN, BiLSTM, Traffic Flow Prediction.

| ARTICLE INFORMATION
ACCEPTED: 28 August 2024 **PUBLISHED:** 12 September 2024 **DOI:** 10.32996/jcsts.2024.6.4.7

1. Introduction

Traffic flow prediction, as one of the core components of ITS, plays a crucial role in urban planning, traffic management and control, and road design. Accurate traffic flow prediction can not only help traffic managers to effectively allocate resources and formulate reasonable traffic management measures but also provide the public with accurate travel information, which can improve road use efficiency, ease traffic congestion, reduce the incidence of traffic accidents, and reduce the level of environmental pollution. With the acceleration of urbanization, traffic problems are becoming more and more prominent. Therefore, the development of efficient and accurate traffic flow prediction models has become a hotspot of concern for both academia and industry.

At present, scholars at home and abroad have carried out a lot of research work for traffic flow prediction, which is mainly categorized into two main groups: traditional statistical methods and machine learning methods. Traditional methods such as time series analysis (ARIMA model), Kalman filter, etc., which rely on the linear relationship of historical data for prediction, are easy to understand and implement but have limitations in dealing with nonlinear relationships and multivariate interactions. In recent years, with the development of big data technology and artificial intelligence, methods based on machine learning have gradually become mainstream, such as support vector machine (SVM)[1], artificial neural network (ANN) and deep learning models[3]. These methods can capture more complex data patterns and improve prediction accuracy, but they also face problems such as difficult parameter selection and long training time.

In view of the problems of existing methods, this paper proposes a traffic flow prediction method based on Improved Grey Wolf Optimizer (IGWO). The Grey Wolf Optimization algorithm[4], as an emerging swarm intelligence optimization algorithm, has demonstrated good performance in many optimization problems. However, the standard Gray Wolf Optimization algorithm may suffer from the problem of locally optimal solutions in some cases. For this reason, we propose the Gray Wolf Optimization Algorithm with Enhanced Exploration and Exploitation Mechanism (IGWO) to overcome this shortcoming and apply it to the hyper-parameter optimization process of Convolutional Neural Networks (CNN)[2], including the selection of the learning rate,

convolutional kernel size, and the number of neurons. With the IGWO algorithm, we are able to find the optimal or near-optimal hyperparameter combinations in a larger search space, resulting in better feature representations. Finally, we combine the optimized CNN network with BiLSTM[6] to construct an efficient traffic flow prediction model. The model can make full use of the spatial perception ability of CNN and the time series processing ability of BiLSTM, which can improve the prediction accuracy and provide a strong technical support for the construction of intelligent transportation system.

2. Methodology

2.1 BiLSTM

In the field of deep learning, recurrent neural networks (RNNs) are a particularly suitable model for processing sequential data because of its ability to memorize previous information and use it in subsequent time steps. However, traditional RNNs encounter the problem of gradient vanishing or gradient explosion when dealing with long sequence data, which limits their scope in practice. To solve these problems, Long Short-Term Memory Networks (LSTMs) were proposed, which effectively solved the gradient problem in long-sequence learning by introducing a special gating mechanism. However, for time series data, only considering the past information may not be sufficient to capture all useful information. Therefore, a Bidirectional Long and Short-Term Memory Network (Bidirectional LSTM, or BiLSTM for short) was born.

BiLSTM combines two independent LSTM layers, one processing the input sequence forward from the beginning to the end of the sequence (forward LSTM), and the other processing the input sequence in reverse from the end to the beginning of the sequence (reverse LSTM). In this way, the output of each time step is a combination of the hidden states of the LSTM in these two directions, thus enabling the model to simultaneously utilize the contextual information of the sequence and improve prediction accuracy.

Each LSTM unit in a BiLSTM comprises three gates: the input gate, the forget gate, and the output gate. These gates control the flow of information through the cell using sigmoid activation functions, while the candidate memory cell is updated via the tanh function. The forward LSTM hidden state h_t^f and the backward LSTM hidden state h_t^b are computed as follows:

1. **Forward LSTM**:

Forget gate: $f_t^f = \sigma(W_f^f [h_{t-1}^f, x_t] + b_f^f)$
Input gate: $i_t^f = \sigma(W_i^f [h_{t-1}^f, x_t] + b_i^f)$
Candidate memory cell: $\tilde{c_t^f} = tanh(W_c^f [h_{t-1}^f, x_t] + b_c^f)$
Cell state: $c_t^f = f_t^f \odot c_{t-1}^f + i_t^f \odot \tilde{c_t^f}$
Output gate: $o_t^f = \sigma(W_o^f [h_{t-1}^f, x_t] + b_o^f)$
Hidden state: $h_t^f = o_t^f \odot tanh(c_t^f)$

2. **Backward LSTM**:

Forget gate: $f_t^b = \sigma(W_f^b [h_{t+1}^b, x_t] + b_f^b)$
Input gate: $i_t^b = \sigma(W_i^b [h_{t+1}^b, x_t] + b_i^b)$
Candidate memory cell: $\tilde{c_t^b} = tanh(W_c^b [h_{t+1}^b, x_t] + b_c^b)$
Cell state: $c_t^b = f_t^b \odot c_{t+1}^b + i_t^b \odot \tilde{c_t^b}$
Output gate: $o_t^b = \sigma(W_o^b [h_{t+1}^b, x_t] + b_o^b)$
Hidden state: $h_t^b = o_t^b \odot tanh(c_t^b)$

Here, x_t is the input vector at time step t, W represents weight matrices, b denotes bias terms, σ is the sigmoid activation function, $tanh$ is the hyperbolic tangent function, and $\odot$ indicates element-wise multiplication.

The BiLSTM processing procedure is as follows:

1. **Initialization**: Initialize the initial hidden states and cell states for both the forward and backward LSTMs.
2. **Forward Propagation**: The forward LSTM processes the input data sequentially from the first time step to the last.
3. **Backward Propagation**: The backward LSTM processes the input data in reverse order, from the last time step to the first.
4. **Combination**: At each time step t, the forward and backward LSTM hidden states h_t^f and h_t^b are concatenated or averaged to form the final hidden state $h_t = [h_t^f; h_t^b]$.
5. **Output**: Generate predictions based on the final hidden state h_t.

By combining the forward and backward contexts, the BiLSTM can effectively utilize bidirectional information in the sequence, leading to improved performance in tasks such as natural language processing, speech recognition, and time series prediction.

2.2 CNN

Convolutional Neural Networks (CNN) have become a cornerstone in the field of deep learning, particularly excelling in tasks involving spatial hierarchies, such as image and video recognition, signal processing, and natural language processing. CNN are designed to automatically and adaptively learn spatial hierarchies of features from raw input data, making them highly effective for tasks where the input data has a grid-like topology, such as images or traffic flow data.

At the core of CNN lies the convolution operation, which is a mathematical operation used to extract features from the input data. The convolution operation involves sliding a filter over the entire input space, computing the dot product between the filter and the input at each location, and producing a feature map that highlights specific features within the input data. This process is repeated for multiple filters, generating a set of feature maps that capture different aspects of the input.

CNN architectures typically consist of several layers, including convolutional layers, pooling layers, and fully connected layers. Each layer serves a specific purpose in the overall architecture:

- **Convolutional Layers**: These layers apply a set of learnable filters to the input data, extracting features through the convolution operation. The filters are learned during training and are shared across the entire input space, reducing the number of parameters required.
- **Pooling Layers**: These layers downsample the spatial dimensions of the feature maps, reducing the computational complexity and controlling overfitting. Common pooling operations include max-pooling and average-pooling.
- **Fully Connected Layers**: These layers connect every neuron in one layer to every neuron in the next layer, allowing the network to make decisions based on the extracted features. They are often placed at the end of the network to perform classification or regression.

The processing of input data through a CNN follows a specific sequence:

1. **Input Layer**: The input data is fed into the network, typically as a tensor representing an image or another type of structured data.
2. **Convolutional Layers**: The input data passes through one or more convolutional layers, where filters are applied to detect local patterns and features. Each filter generates a new feature map.
3. **Activation Function**: After each convolutional layer, an activation function, such as ReLU (Rectified Linear Unit), is applied to introduce non-linearity into the network, enabling it to learn complex patterns.
4. **Pooling Layers**: Optionally, pooling layers are used to reduce the spatial dimensions of the feature maps, while preserving the most important information. This helps to reduce the computational load and enhances the robustness of the features against small translations.
5. **Flattening**: The output of the last convolutional or pooling layer is flattened into a one-dimensional vector to be passed to the fully connected layers.
6. **Fully Connected Layers**: The flattened vector is processed through one or more fully connected layers, which perform the final classification or regression task based on the learned features.

2.3 Reegwo

The Grey Wolf Optimizer (GWO) is a nature-inspired metaheuristic algorithm that mimics the leadership hierarchy and hunting behavior of grey wolves. Since its introduction, GWO has gained popularity due to its simplicity and effectiveness in solving various optimization problems. However, the standard GWO algorithm may encounter limitations, particularly in avoiding local optima in complex optimization scenarios. To address this issue, Yu et al. (2022) proposed the Reinforced Exploration and Exploitation Grey Wolf Optimizer (REEGWO)[5], aiming to enhance the global search capability and local precision of the algorithm.

The REEGWO algorithm builds upon the standard GWO framework but introduces several enhancements to improve the exploration and exploitation mechanisms:

- **Adaptive Learning Rate Adjustment**: REEGWO incorporates an adaptive learning rate adjustment mechanism, which dynamically adjusts the search step size based on the current iteration number and the characteristics of the search space. This adjustment helps expand the search range in the early stages and refine the search in later stages.

- **Random Neighborhood Search**: To further increase the exploration capability, REEGWO employs a random neighborhood search strategy. During the search process, each individual has a certain probability of conducting a random search around its current position, preventing premature convergence to local optima.
- **Elite Strategy**: REEGWO implements an elite strategy by retaining the best solution encountered during the search process. This retained solution guides the search direction in subsequent iterations, enhancing the search efficiency.
- **Dynamic Adjustment of α, β, and δ**: In GWO, α, β, and δ represent the leader, sub-leader, and third wolf, respectively. REEGWO dynamically adjusts these parameters to balance exploration and exploitation, ensuring that the algorithm can explore the search space widely while also intensively developing promising regions.
- The specific steps of the REEGWO algorithm are as follows:
- **Initialization**: Initialize the population of wolves, set the maximum number of iterations, and define other relevant parameters.
- **Evaluation**: Calculate the objective function values for each individual and determine the current α, β, and δ based on the fitness values.
- **Exploration and Exploitation**: Update the positions of the individuals according to the dynamically adjusted learning rate and random neighborhood search strategy to explore new solution spaces.
- **Elite Update**: Retain the best solution encountered during the search process and use it to guide the search direction in subsequent iterations.
- **Iterative Process**: Repeat the above steps until the maximum number of iterations is reached or other termination criteria are met.
- **Result Output**: Output the best solution as the final optimized result.

3. Experiment

3.1 Experiment Setup

This experiment uses MATLAB 2023a as the main simulation software. The deep learning toolbox and optimization toolbox provided by MATLAB provide strong support for the deep learning model construction and the implementation of the improved Gray Wolf optimization algorithm in this experiment. The experiments in this paper were conducted on a computer equipped with an Intel Core i9-12900K CPU, NVIDIA RTX 3090 GPU, 128 GB DDR5 RAM, and 2 TB NVMe SSD running Windows 11 Pro operating system.

3.2 Data sources

Minnesota DoT ATR Station 301[7] is located on westbound Interstate 94 between Minneapolis and St. Paul and is dedicated to monitoring hourly traffic volumes on that roadway. The monitoring data not only includes vehicle volumes, but also covers hourly weather characteristics (e.g., temperature, humidity, precipitation conditions, etc.) as well as whether it is a public holiday or special event day. This additional information contributes to a more comprehensive understanding of the changing patterns of traffic flow and the reasons behind them, which is important for traffic management and planning.

3.3 Experiment Results

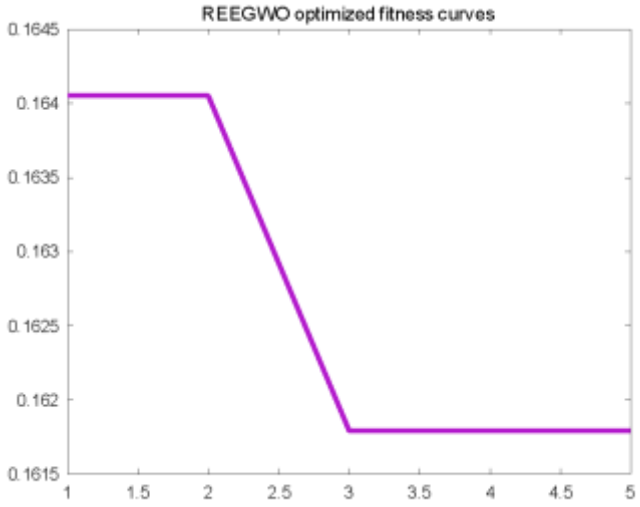
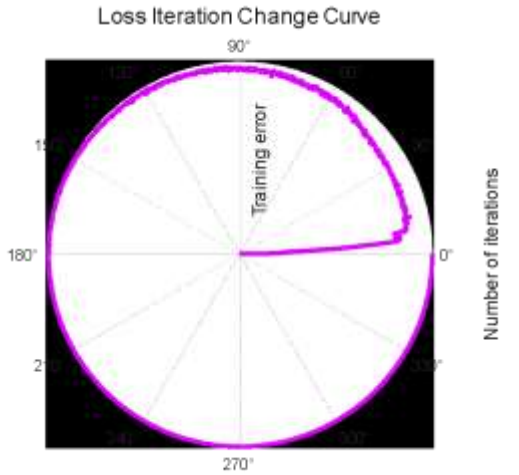


Fig. 1 REEGWO optimized fitness curves

From Fig. 1, it can be found that the proposed model adopts REEGWO to optimize the three most important parameters of learning rate, convolutional kernel size, and number of neurons in CNN network, and the final iterative rule of change of the fitness function is to maintain the same at the beginning, and then slowly reduce, and ultimately remain unchanged, so as to achieve the stability of the fitness function.



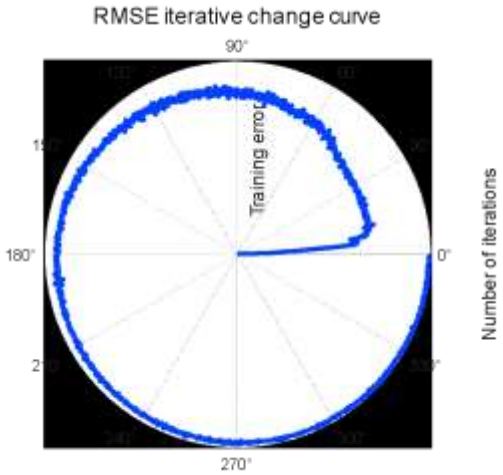


Fig. 2 Loss Iteration Change Curve
Fig. 3 RMSE Iteration Change Curve

According to Fig. 2 and 3, it can be noticed that this paper plots the change in training loss error and RMSE on the training set as the number of iterations increases. From the above figure, it can be concluded that the proposed model gradually converges in the training loss error and RMSE values with the number of iterations, indicating that the proposed model can achieve good results in the prediction of traffic flow.

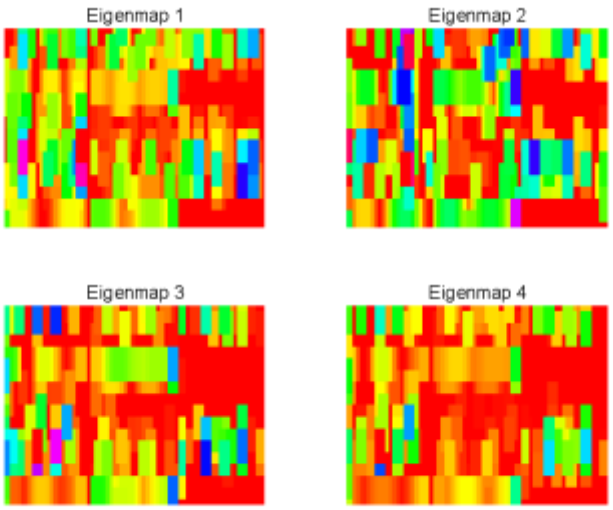


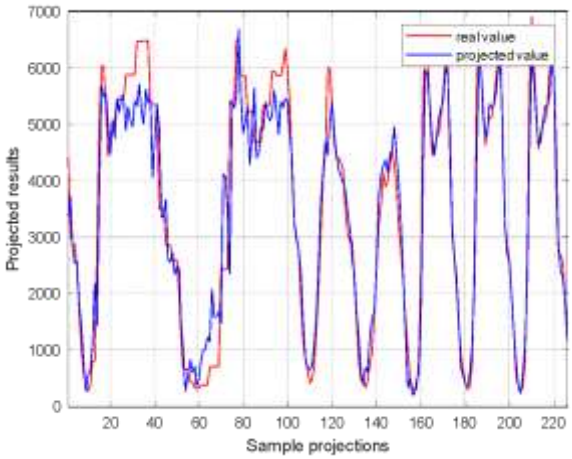
Fig. 4 Eigenmap for flatten layer

In Fig. 4, the characteristics of the flattened layer are plotted in this paper. Based on Fig. 4, it can be concluded that the optimization of CNN network using REEGWO in this paper was able to obtain the corresponding feature maps. This helps in information integration and model simplification but may also lead to a certain degree of spatial information loss. For the traffic flow prediction task, the flattened layer can effectively integrate the spatial features extracted by the CNN and pass them to the BiLSTM for time series prediction, thus improving the accuracy and efficiency of the overall prediction model.

3.4 Ablation Studies

In order to perform an ablation comparison test, the model proposed in this paper is validated and tested on the same dataset with Convolutional Neural Network CNN, BP Neural Network, BP Neural Network optimized using Genetic Algorithm GA, Long

and Short-Term Memory Neural Network LSTM, Radial Basis Function Network RBF. Firstly the test results of the proposed improved Grey Wolf algorithm optimization method on the given dataset are given in this paper:



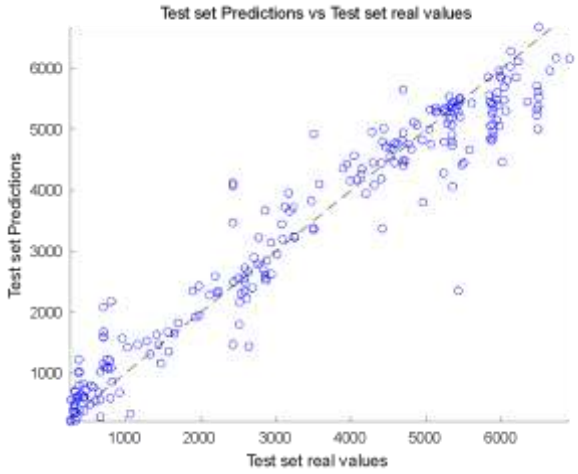


Fig. 5 Sample Projections

Fig. 6 Test Set Predictions vs. Test Set Label

As shown in Fig. 5, the proposed prediction method with improved gray wolf optimization is able to make a good fitting prediction for the given dataset, and it can be found from the fitting graph that the proposed method has some errors in the pole values, especially in the continuous variation of the place where there is some inaccuracy in the fitting. However, in most places, the proposed model is able to predict and analyze the changes in traffic flow well.

In this paper, in order to get a more intuitive feel for the comparative quantitative gap between the predicted results and the true values, Fig. 6 plots a scatter plot between the predicted results and the true results on the test set. According to the graph, it can be found that most of the distribution between the predicted results and the true values of the proposed method are on both sides of the standard straight line. Therefore, the model proposed in this paper is able to predict the traffic flow well.

In addition, it is important to compare the effectiveness gap between the method proposed in this paper and other currently available methods for traffic flow prediction more clearly. In this paper, each method is evaluated from a total of four indicators of goodness-of-fit R-square, MAE, MBE and MAPE, and the results are shown in the following table:

	R^2	MAE	MBE	MAPE
CNN	0.92598	398.006	-54.0353	0.22368
BP	0.84651	425.064	-68.2541	0.28650
GA-BP	0.86327	421.658	-64.3283	0.26874
LSTM	0.93571	390.128	-50.6912	0.20843
RBF	0.88652	413.985	-59.6723	0.23168
Our model	0.95632	375.912	-39.5172	0.15637

Table 1: Experiment Results

According to the above table, it can be found that the proposed model achieves the optimal results in the four indicators of goodness-of-fit R-square, MAE, MBE and MAPE, and reaches 95.632% in R-square, which is 10.951% higher than the lowest BP neural network. There is also a lot of improvement in all other indicators, achieving better than the currently available methods.

5. Conclusion

In this paper, a traffic flow prediction model combining the Revised Enhanced Extreme Gray Wolf Optimizer (REEGWO) with Convolutional Neural Networks (CNNs) and Bidirectional Long Short-Term Memory (BiLSTM) networks is proposed. Key parameters such as learning rate, convolutional kernel size, and the number of neurons in the CNN are optimized by REEGWO so that the fitness function gradually converges and reaches a steady state. Experimental results show that the model exhibits good convergence in terms of training loss error and root mean square error (RMSE) as the number of iterations increases. Meanwhile, the model optimized by the improved Gray Wolf optimization algorithm proposed in this study is able to fit the given dataset

better. Further quantitative comparisons show that the model outperforms other existing prediction methods in the four metrics of goodness-of-fit (R^2), mean absolute error (MAE), mean bias error (MBE) and mean absolute percentage error (MAPE).

Funding: This research received no external funding.
Conflicts of Interest: The authors declare no conflict of interest.
Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers.

References

[1] Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. IEEE Intelligent Systems and their applications, 13(4), 18-28.
[2] Huang, Z., Xu, W., & Yu, K. (2015). Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991.
[3] Hogue, J. (2019). Metro Interstate Traffic Volume. UCI Machine Learning Repository. https://doi.org/10.24432/C5X60B.
[4] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25.
[5] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.
[6] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46-61.
[7] Yu, X., Xu, W., Wu, X., & Wang, X. (2022). Reinforced exploitation and exploration grey wolf optimizer for numerical and real-world optimization problems. Applied Intelligence, 1-16.