Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 3673085, 10 pages
https://doi.org/10.1155/2020/3673085

Research Article

Traffic Flow Anomaly Detection Based on Robust Ridge Regression with Particle Swarm Optimization Algorithm

Mingzhu Tang,[1,2] **Xiangwan Fu**,[1] **Huawei Wu**,[2] **Qi Huang**,[3] **and Qi Zhao**[1]

[1]*School of Energy and Power Engineering, Changsha University of Science & Technology, Changsha 410114, China*
[2]*Hubei Key Laboratory of Power System Design and Test for Electrical Vehicle, Hubei University of Arts and Science, Xiangyang 441053, China*
[3]*School of Transport Management, Hunan Communication Polytechnic, Changsha 410132, China*

Correspondence should be addressed to Huawei Wu; whw_xy@163.com and Qi Huang; huang.qiqi.813@163.com

Received 31 July 2020; Accepted 2 September 2020; Published 30 September 2020

Academic Editor: Yong Chen

Traffic flow anomaly detection is helpful to improve the efficiency and reliability of detecting fault behavior and the overall effectiveness of the traffic operation. The data detected by the traffic flow sensor contains a lot of noise due to equipment failure, environmental interference, and other factors. In the case of large traffic flow data noises, a traffic flow anomaly detection method based on robust ridge regression with particle swarm optimization (PSO) algorithm is proposed. Feature sets containing historical characteristics with a strong linear correlation and statistical characteristics using the optimal sliding window are constructed. Then by providing the feature sets inputs to the PSO-Huber-Ridge model and the model outputs the traffic flow. The Huber loss function is recommended to reduce noise interference in the traffic flow. The L_2 regular term of the ridge regression is employed to reduce the degree of overfitting of the model training. A fitness function is constructed, which can balance the relative size between the k-fold cross-validation root mean square error and the k-fold cross-validation average absolute error with the control parameter η to improve the optimization efficiency of the optimization algorithm and the generalization ability of the proposed model. The hyperparameters of the robust ridge regression forecast model are optimized by the PSO algorithm to obtain the optimal hyperparameters. The traffic flow data set is used to train and validate the proposed model. Compared with other optimization methods, the proposed model has the lowest RMSE, MAE, and MAPE. Finally, the traffic flow that forecasted by the proposed model is used to perform anomaly detection. The abnormality of the error between the forecasted value and the actual value is detected by the abnormal traffic flow threshold based on the sliding window. The experimental results verify the validity of the proposed anomaly detection model.

1. Introduction

Traffic flow anomaly detection plays an essential role in the traffic field. Traffic jams have become a common thing in big cities and have received considerable critical attention. The traffic flow anomaly detection model can detect the abnormal traffic flow and can be achieved by constructing a traffic flow forecast model, which is helpful to avoid traffic congestion in time. The accurate forecast of traffic flow can not only provide a basis for real-time traffic control but also provide support for the alleviation of traffic jams and the effective use of traffic networks, and the forecast result of traffic flow can directly affect the accuracy of traffic anomaly detection. Useful information can be extracted from massive traffic flow data through the traffic flow forecast model so as to quickly forecast the short-term traffic flow in the future and detect the traffic flow abnormalities in time, thus improving the traffic operation efficiency.

In recent years, many experts and scholars have studied traffic flow forecasting. The ARIMA model is a classic time series model that is often used in traffic flow forecasts. Kumar and Vanajakshi proposed a SARIMA-based traffic flow forecast scheme, which effectively solved the problem of massive data required for model training [1]. Shahriari et al. combined bootstrap with the ARIMA model, which overcame the shortcomings of nonparametric methods lacking

theoretical support and improved the forecast accuracy of the model [2]. Luo et al. combined the improved SARIMA model with the genetic algorithm and used the real traffic flow to test the model. The model forecast results were good [3]. The ARIMA model forecasts the traffic flow based on historical values. If the model training data contain noise, the model's performance will be greatly reduced.

The neural network model can fit complex data relationships, which can learn the nonlinear relationships implicit in traffic flow. Qu et al. proposed a batch learning method to solve the time-consuming problem of the traffic flow neural network prediction model, which effectively reduced the training time of the neural network [4]. Zhang et al. used the spatiotemporal feature extraction algorithm to extract the temporal and spatial features in traffic flow. The features were input into the recurrent neural network for modeling and forecast, which effectively improved the forecast performance of the model [5]. Zhang et al. proposed a multitask learning deep learning model to forecast the traffic network flow. The nonlinear Granger causality analysis was used to select features for the model. The Bayesian optimization algorithm was used to optimize the model parameters. The forecast performance was better than that of the single deep learning model [6]. Do et al. used temporal and spatial attention mechanisms to help neural network models fully explore the temporal and spatial characteristics of the traffic flow, which not only effectively improved the prediction performance of the model but also enhanced the interpretability of the model [7]. The use of neural network models can cause overfitting easily with a calculation cost much higher than that of the traditional traffic flow forecast model. As neural networks can fit nonlinear relationships of data, it is easy to use the wrong noise as the implicit nonlinear relationship in the data, which will reduce the generalization ability of the model.

The support vector regression machine can fit data based on the strategy of structural risk minimization, which is a common model in the field of traffic flow forecasts. Wang et al. proposed an adaptive traffic flow forecast framework, which used the Bayesian optimization algorithm to optimize the parameters of the support vector machine model. The forecast performance was better than that of the SARIMA model [8]. Luo et al. used the discrete Fourier transform to extract the trend information in traffic flow and used the support vector machines for error compensation, which improved the forecast accuracy of the model [9]. The support vector regression machine solves the optimization problem based on quadratic programming. When the sample size is large, the model training time will be greatly increased. The support vector regression machine is very sensitive to the noise in the data. When the support vector regression machine selects the noise as the support vector, the forecast performance of the model will be poor.

Traffic flow anomaly detection plays an important role in the field of urban traffic control. Many studies have done related work in the field of traffic flow anomaly detection. Djenouri et al. proposed a framework for detecting temporal and spatial traffic anomalies. The KNN algorithm was applied to the space-time traffic database, and the traffic flows at ten different locations were experimented. Experimental results showed that the performance of the proposed framework is better than the baseline model [10]. Yujun et al. proposed a hybrid model that contained the Poisson mixture model and coupled hidden Markov model. The proposed model considered the spatial correlation of traffic flow and the degree of traffic congestion. Semisynthetic and real traffic anomaly data were used to verify the validity of the model [11]. Zhang et al. employed the dictionary-based compression theory to identify the spatial and temporal characteristics of traffic flow and used anomaly index to quantify the degree of traffic anomalies [12]. The proposed method can clearly detect the location of traffic flow spatial anomalies. Noise in traffic data may lead to false detection results of traffic anomaly detection models, which may affect the normal operation of traffic networks.

Influenced by factors such as mechanical damage, line aging, signal loss, and environmental interference, the data detected by the traffic flow sensor contain a lot of noise. Huber loss function is a mixture of L_1 and L_2 loss functions, which is insensitive to noise [13], the L_2 regular term of the ridge regression can effectively avoid overfitting caused by model training [14]. To improve the generalization performance of the model, the sum of $\mathrm{RMSE}_{k_{cv}}$ and $\eta * \mathrm{MAE}_{k_{cv}}$ on the training set based on k-fold cross-validation is constructed as the fitness function and the PSO algorithm is used to optimize the model hyperparameters. The PSO algorithm originated from the research on the foraging process of birds [15]. It has a simple structure. Each particle in the particle swarm has three main parameters: position, velocity, and fitness. In recent years, many pieces of literature have achieved good results using the particle swarm optimization algorithm [16–20].

To solve the problem of noise in traffic flow data, a Huber-Ridge traffic flow anomaly detection model with the particle swarm optimization (PSO) algorithm is proposed. The Huber-Ridge model is used to reduce the negative impact of noise in the data. Huber-Ridge model performance depends on model hyperparameters. Therefore, it is very important to determine the optimal model hyperparameters. A PSO algorithm based on the proposed fitness function is used to search for the optimal hyperparameters of the model so that the model has the best performance.

The remaining part of the paper proceeds as follows: Section 2 introduces the theoretical information of the Huber-Ridge algorithm; Section 3 proposes the data preprocessing steps and the steps using PSO algorithm to optimize the Huber-Ridge model parameters; Section 4 illustrates the model evaluation indexes; Section 5 presents the experimental content which contains the comparison of the forecast models and the results of traffic flow anomaly detections; Section 6 is conclusions.

2. Huber-Ridge Algorithm

2.1. Huber Function. The combination of the Huber function with the L_1 loss function and the L_2 loss function can effectively avoid the interference of noise in the data during the data fitting [21]. Its robustness is better than that of L_1

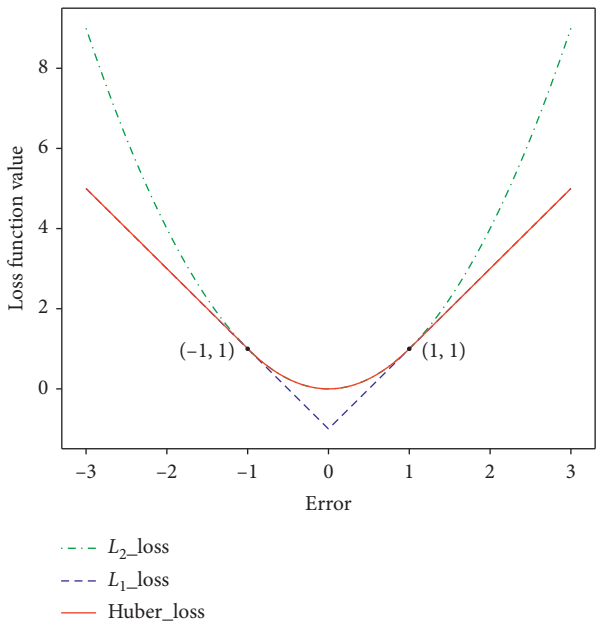

Figure 1: When the threshold M is 1, comparison of Huber function with L_1 loss function and L_2 loss function.

and L_2 loss functions. The definition of the Huber loss function is

$$\phi_{\text{hub}}(u) = \begin{cases} u^2, & |u| \le M, \\ M(2|u| - M), & |u| > M. \end{cases} \tag{1}$$

The definitions of L_1 loss function and L_2 loss function are shown in equations (2) and (3):

$$\phi_{L_1}(u) = M(2|u| - M), \tag{2}$$

$$\phi_{L_2}(u) = u^2, \tag{3}$$

where u is the error between the actual value and the estimated value, and M is the threshold. When the threshold M is 1, the comparison of the Huber loss function, the L_1 loss function, and the L_2 loss function is shown in Figure 1. Compared with the L_1 loss function, when u is smaller than the threshold M, the Huber loss function penalizes the model for making large errors. Compared with the L_2 loss function, when u is greater than the threshold M, the Huber loss function penalizes the model for making small error Therefore, the Huber loss function is quadratic for smaller errors and is linear for larger errors.

2.2. Ridge Regression Model. The ridge regression model is first proposed by Hoerl and Kennard. The ridge regression objective function adds the L_2 regular term based on the least square objective function [22]. Its definition is as follows:

$$\hat{w}_j = \text{argmin}_w \left(\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{k} (w_j)^2 \right), \tag{4}$$

where $\sum_{j=1}^{k} (w_j)^2$ is the L_2 regular term and λ is the ridge parameter, which is the weight of the L_2 regular term.

For the linear regression model $y = wx + \varepsilon$, the least square estimation of the regression coefficient is defined as follows:

$$\hat{w} = \left(x^T x\right)^{-1} x^T y, \tag{5}$$

where x is the independent variable matrix and y is the dependent variable vector.

The mean square error of the least square estimation is defined as follows:

$$\hat{w}_{\text{mse}} = E\left(||w - \hat{w}||^2\right) = \sigma^2 tr\left(x^T x\right)^{-1} = \sigma^2 \sum_{i=1}^{q} \frac{1}{k_i}. \tag{6}$$

If there is a linear correlation between independent variables, the matrix $x^T x$ is a singular matrix. Some characteristic roots k_i of the singular matrix are close to zero, resulting in a large $\hat{w}_{\text{mse}}$. This indicates that there is a large error between the least-squares estimated value and the actual value. The addition of the disturbance term $\lambda I (\lambda > 0)$ on the matrix $x^T x$ will weaken the singularity, thereby reducing $\hat{w}_{\text{mse}}$. The least square estimation with the disturbance term added is the ridge estimation. The ridge estimate is defined as follows:

$$\hat{w}(\lambda) = \left(x^T x + \lambda I\right)^{-1} x^T y, \tag{7}$$

where λ is the ridge parameter and I is the identity matrix. $\hat{w}(\lambda)$ indicates the ridge estimation of the regression parameter w when the ridge parameter is λ. When $\lambda = 0$, the ridge estimation is the least square estimation. In the case of linear correlation of independent variables, the ridge estimation provides improved efficiency in parameter estimation problems, that is, biased but has lower variance than the least square estimator.

2.3. Huber-Ridge Regression. Owen uses the Huber loss function to replace the least-squares loss function and converted the ridge regression to the Huber-Ridge regression [23]. The definition of the Huber-Ridge model is as follows:

$$\hat{w}_j = \text{argmin}_w \left(\phi_{\text{hub}}(u) + \frac{\lambda}{2} \sum_{j=1}^{k} (w_j)^2 \right), \tag{8}$$

where w is the weight vector of the regression when the objective function is the smallest, w_j represents the estimate for each regression coefficient, $\sum_{j=1}^{k} (w_j)^2$ is the L_2 regular term, and $\lambda/2$ is the weight of the L_2 regular term, which is used to balance the relationship between the Huber loss function and the L_2 regular term. The Huber loss function can help the model avoid the influence of the data noise. The L_2 regular term helps the model have a proper sparsity and avoid overfitting of the model. The Huber-ridge regression combines the robustness of the Huber regression to noise with the regularization of the Ridge regression, which not only ensures the robustness of the model but also makes the regression model more stable.

$\sum_{j=1}^{k}(w_j)^2$ can be considered as $||w||_2^2$, which is the L_2 norm square of the weight vector w. The objective function $f(w)$ is defined as follows:

$$f(w) = \phi_{\text{hub}}(u) + \frac{\lambda}{2}||w||_2^2, \tag{9}$$

where u is the error. The objective function $f(w)$ is used to take the partial derivative of the weight vector w and let it to be zero. It can be obtained that the expression of the weight vector w is at the minimum value of the objective function in the direction of the weight vector w. The solution process of equation (9) is as follows:

$$\frac{\partial f(w)}{\partial w} = \frac{\partial \phi_{\text{hub}}(u)}{\partial u}\frac{\partial u}{\partial w} + \frac{\mathrm{d}(\lambda/2)||w||_2^2}{\mathrm{d}w} = 0, \tag{10}$$

where $u = xw - y$, xw is the estimated value, and y is the actual value. The first term of equation (10) can be simplified as

$$\frac{\partial \phi_{\text{hub}}(u)}{\partial u}\frac{\partial u}{\partial w} = \frac{\partial \phi_{\text{hub}}(u)}{\partial u}x, \tag{11}$$

$$\frac{\partial \phi_{\text{hub}}(u)}{\partial u} = \left[\frac{\partial \phi_{\text{hub}}(u_1)}{\partial u_1}, \frac{\partial \phi_{\text{hub}}(u_2)}{\partial u_2}, \ldots, \frac{\partial \phi_{\text{hub}}(u_n)}{\partial u_n}\right]^T = [h(u_1), h(u_2), \ldots, h(u_n)],$$
$$\frac{\partial \phi_{\text{hub}}(u_i)}{\partial u_i} = \begin{cases} 2|u_i|, & |u_i| \le M, \\ 2M, & |u_i| > M. \end{cases} \tag{12}$$

Let $\omega(u) = \partial h(u)/\partial u$, equation (11) can be simplified as

$$x^T\frac{\partial \phi_{\text{hub}}(u)}{\partial u} = x^T\varphi u,$$
$$\varphi = \text{diag}[\omega(u_1), \omega(u_2), \ldots, \omega(u_n)]. \tag{13}$$

The second term of equation (10) can be simplified as

$$\frac{\mathrm{d}(\lambda/2)w_2^2}{\mathrm{d}w} = \frac{\mathrm{d}(\lambda/2)\left(w^Tw\right)^2}{\mathrm{d}w} = \lambda w. \tag{14}$$

In summary, the solution process of equation (10) is as follows:

$$x^T\varphi u + \lambda w = 0,$$
$$x^T\varphi(xw - y) + \lambda w = 0, \tag{15}$$

$$w = \left(x^T\varphi x + \lambda I\right)^{-1}x^T\varphi y, \tag{16}$$

where I is the identity matrix. The optimal threshold M and the ridge parameter λ can be found in a fixed interval through the optimization algorithm. The weight vector w can be obtained by substituting the threshold value M, the ridge parameter λ, and the sample data into equation (16).

3. PSO-Huber-Ridge Model

3.1. PSO Algorithm. The core idea of the PSO algorithm comes from the foraging process of birds. For the PSO algorithm, the candidate solution of the optimization problem is a particle in the hyperparameter space. Each particle has its corresponding fitness value, speed, and position. The speed of the particle determines the direction and the displacement of the particle to look for the candidate solution. The PSO algorithm can find the optimal solution by iterating a group of initialized random particles.

For the PSO algorithm, there are m particles in the D-dimensional space. The speed of each particle can be expressed as $\overrightarrow{v}_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$, and the position of each particle can be expressed as $\overrightarrow{s}_i = (s_{i1}, s_{i2}, \ldots, s_{iD})$, where $i \in [1, 2, \ldots, m]$. In the loop iteration, each particle represents a candidate solution. The corresponding fitness value can be obtained through the fitness function. The individual optimal particle and the global optimal particle can be selected based on the fitness value. The personal optimal particle ($p$best) is expressed as $\overrightarrow{p}_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$, and the global optimal particle ($g$best) is expressed as $\overrightarrow{p}_g = (p_{g1}, p_{g2}, \ldots, p_{gD})$. Before the next iteration, each particle will update its speed and position through equations (17)–(19):

$$\overrightarrow{v}_i^{(t+1)} = \omega * \overrightarrow{v}_i^{(t)} + c_1 * r_1 * \left(\overrightarrow{p}_i - \overrightarrow{s}_i^{(t)}\right) + c_2 * r_2 * \left(\overrightarrow{p}_g - \overrightarrow{s}_i^{(t)}\right), \tag{17}$$

$$v_{ij}^{(t+1)} = \begin{cases} v_{\text{Max}}, & \left|v_{ij}^{(t+1)}\right| > v_{\text{Max}}, \\ v_{ij}^{(t+1)}, & \text{otherwise}, \end{cases} \tag{18}$$

$$\overrightarrow{s}_i^{(t+1)} = \overrightarrow{s}_i^{(t)} + \overrightarrow{v}_i^{(t+1)}, \tag{19}$$

$$i \in [1, 2, \ldots, m],$$
$$j \in [1, 2, \ldots, D], \tag{20}$$

where ω is the inertia factor ($\omega > 0$), c_1 is the local learning factor, and c_2 is the global learning factor ($c_1, c_2 > 0$). r_1 and r_2 are random numbers uniformly distributed between $[0, 1]$. t and $t + 1$ represent the number of iterations. v_{Max} represents the maximum speed of the particle.

For equation (17), where $\omega * \overrightarrow{v}_i^{(t)}$ is called the memory item, which refers to the influences of the speed on the particle when it is updated; $c_1 * r_1 * (\overrightarrow{p}_i - \overrightarrow{s}_i^{(t)})$ is called the

self-cognition term, which means that when the particle is updated, it is biased toward the individual optimal particle; $c_2 * r_2 * (\overrightarrow{p}_g - \overrightarrow{s}_i^{(t)})$ is called the group-cognition term, which means that when the particles are updated, they are biased toward the group optimal particle. It represents the result of collaboration among multiple particles.

3.2. Fitness Function. The PSO algorithm can find the optimal hyperparameters for the model based on the fitness function. The smaller the particle fitness value, the lower the forecast error of the hyperparameters. To improve the generalization ability of the model, the k-fold cross-validation [24] is added to the fitness function. The fitness function is defined as the sum of RMSE and MAE of k-fold cross-validation on the model training set. The expression equation for the fitness function is as follows:

$$\text{fitness} = \text{RMSE}_{k_{cv}} + \eta * \text{MAE}_{k_{cv}}. \tag{21}$$

$\text{RMSE}_{k_{cv}}$ is a root mean square error based on k-fold cross-validation and its expression is as follows:

$$\text{RMSE}_{k_{cv}} = \sqrt{\frac{\sum_{j=1}^{k_{cv}} \sum_{i=1}^{n} \left(y_{ij} - \widehat{y}_{ij}\right)^2}{nk_{cv}}}. \tag{22}$$

$\text{MAE}_{k_{cv}}$ is based on the average absolute error of k-fold cross-validation, and its expression equation is as follows:

$$\text{MAE}_{k_{cv}} = \sum_{j=1}^{k_{cv}} \sum_{i=1}^{n} \frac{\left|y_{ij} - \widehat{y}_{ij}\right|}{nk_{cv}}, \tag{23}$$

where n is the number of training samples, k_{cv} is the number of cross-validated subsets. $\widehat{y}_{ij}$ and y_{ij} are the model estimated value and the true value, respectively. The smaller the fitness function value, the better the corresponding particle.

The weight of $\text{MAE}_{k_{cv}}$ is η $(\eta > 0)$, which is also the control parameter used to balance the size of $\text{RMSE}_{k_{cv}}$ and $\text{MAE}_{k_{cv}}$. When $0 < \eta < 1$, $\text{MAE}_{k_{cv}}$ has less weight than $\text{RMSE}_{k_{cv}}$; when $1 < \eta < +\infty$, $\text{MAE}_{k_{cv}}$ has more weight than $\text{RMSE}_{k_{cv}}$; when $\eta = 1$, $\text{MAE}_{k_{cv}}$ has the same weight as $\text{RMSE}_{k_{cv}}$. $\text{RMSE}_{k_{cv}}$ has a small penalty for small errors. The degree of $\text{MAE}_{k_{cv}}$ penalty for errors remains unchanged. However, it does not punish large errors as much as $\text{RMSE}_{k_{cv}}$. The fitness function controls the degree of which the fitness function penalizes errors by adjusting the size of the control parameter η. As the control parameter η increases, the degree of penalty for small errors by the fitness function increases. Combining $\text{MAE}_{k_{cv}}$ and $\text{RMSE}_{k_{cv}}$, the problem of insufficient penalty for small errors for $\text{RMSE}_{k_{cv}}$ and insufficient penalty for large errors for $\text{MAE}_{k_{cv}}$ can be improved, which not only increases the penalty for model prediction errors but also improves the generalization ability of the model.

3.3. Data Preprocessing. Good data quality can improve the performance of the model. The missing values and the dimensional differences in the data will reduce the forecast performance of the model. Therefore, it is significant to preprocess the data. The data preprocessing can be divided into the following steps:

(1) Data cleaning. The previous value of the missing value should be used to fill in the missing value.

(2) Construction of model feature sets and output samples. For the traffic flow data set, the historical characteristics based on the linear correlation and the statistical characteristics based on the sliding window should be constructed. The model output sample is the traffic flow at the next time point in the sliding window.

(3) Data standardization. There are dimensional differences between different features. To prevent dimensional errors from reducing the model performance, the data distribution is transformed into a standard distribution with a mean of 0 and a variance of 1 through the standardized equation. The standardized equation is as follows:

$$\begin{cases} x_{ki} = \dfrac{X_{ki} - \overline{X}_i}{\sigma_i}, \\ \sigma_i = \dfrac{\left(X_{1i} - \overline{X}_i\right)^2 + \left(X_{2i} - \overline{X}_i\right)^2 + \cdots + \left(X_{ni} - \overline{X}_i\right)^2}{n}. \end{cases} \tag{24}$$

For the feature matrix, x_{ki} is the standardized data of the k-th row and the i-th column, $\overline{X}_i$ is the mean value of the i-th column, σ_i is the standard deviation of the i-th column, and n is the number of samples.

3.4. PSO-Huber-Ridge Model Optimization Process. The optimization steps of the PSO-Huber-Ridge model are as follows:

Step 1. Start the optimization.

Step 2. Determine the model inputs and outputs. The feature set is used as the model input and the model output the traffic flow.

Step 3. PSO-Huber-Ridge model parameter settings. The number of particles m, the inertial factor ω, the local learning factor c_1, and the global learning factor c_2 are input into the PSO algorithm. Initialize the speed $\overrightarrow{v}$ and the position $\overrightarrow{s}$ of each particle. Set the maximum number of iterations of the PSO algorithm i_{Max} and the value range of the model hyperparameters.

Step 4. $i = i + 1$.

Step 5. Particles update. Use equations (17)~(19) to update the speed $\overrightarrow{v}$ and position $\overrightarrow{s}$ of each particle.

Step 6. Fitness evaluation. Use equation (21) to calculate the fitness value of the particle based on the threshold value M and the ridge parameter λ of each particle.

Step 7. Optimal particle selection. Select the individual optimal particle and the global optimal particle according to the fitness value of the particles.

Step 8. Terminate training judgment. If the number of iterations i does not meet the termination condition ($i > i_{\text{Max}}$), return to Step 4. Otherwise, continue to the next step.

Step 9. Output optimization results. Output the threshold M and the ridge parameter λ in the global optimal particle.

Step 10. End the optimization.

4. Evaluation Indexes

The average absolute error (MAE), root mean square error (RMSE), and average absolute percentage error (MAPE) were used to evaluate the forecast performance of the model. The definition equations of MAE, RMSE, and MAPE are as follows:

$$\begin{aligned}
\text{MAE} &= \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \widehat{y}_i\right|, \\
\text{RMSE} &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \widehat{y}_i\right)^2}, \\
\text{MAPE} &= \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \widehat{y}_i}{y_i}\right|,
\end{aligned} \tag{25}$$

where n is the number of samples in the test set, $\widehat{y}_i$ is the model forecast value, y_i is the true value. MAE and RMSE can reflect the forecast error of the model. The value range of MAPE is $[0, +\infty]$. The closer its value is to 0, the better the model performance.

5. Experimental Results and Analysis

5.1. Data Description. The traffic flow data set used in the experiment came from a highway intersection in Changsha City and was collected by a single detector with a data interval of 5 minutes. There were a small number of missing values in the traffic flow data set and the previous value of the missing value was used to fill in the missing points. The data sets containing 5 days of traffic flow were divided into the training set and the test set. The traffic flow from Saturday to Tuesday was used as the training set for the training model. The traffic flow on Wednesday was used as the test set to verify the performance of the trained model.

5.2. Feature Extraction and Selection. Historical characteristics based on the linear correlation from the traffic flow data were selected. The statistical characteristics based on the optimal sliding window were extracted.

The historical characteristics were selected. The Pearson correlation coefficient was used to judge the strength of the linear correlation between the data. The range of the correlation coefficient r was $[-1, 1]$. The closer to 1, the stronger the positive correlation between the data; the closer to -1, the stronger the negative correlation between the data; the closer to 0, the weaker the linear correlation between the data. The historical value of r greater than 0.9 was selected as historical characteristics. See Table 1 for the correlation coefficients of traffic flow with delays of 1–9.

According to Table 1, the historical characteristics with delays of 1–6 were selected as historical characteristics. To fully consider the periodicity of the traffic flow, the historical characteristics at the same time point last week were selected. The set of historical characteristics included the historical values with delays of 1–6 and the historical values at the same time point last week.

The statistical characteristics of the optimal sliding window were extracted. The maximum, minimum, median, mean, standard deviation, skewness, and kurtosis of the data set within the length of the sliding window were taken as the statistical characteristics. The sliding window length L had a value range of $[6, 150]$. The Huber-Ridge model with default hyperparameters ($\lambda = 0.0001, M = 1.35$) was used for the exhaustive operation on the traffic flow training set. The optimal window length was selected with the MAPE evaluation index as the standard. It can be seen from Figure 2 that when the MAPE value was the smallest, the sliding window length was 34 as the optimal sliding window length.

5.3. Experimental Results. The state transition algorithm (STA) [25], grey wolf optimizer (GWO) [26], genetic algorithm (GA) [27], and PSO algorithm were used to optimize the hyperparameters of the Huber-Ridge model. The range of model parameters is shown in Table 2:

The parameter values of the optimization algorithm are shown in Table 3:

The model training was performed using the standardized traffic flow training set. The fitness function based

Table 1: Pearson correlation coefficient of traffic flow with delays of 1–9.

Number of delays	9	8	7	6	5	4	3	2	1	0
R	0.85	0.87	0.89	0.90	0.92	0.93	0.95	0.96	0.97	1.00

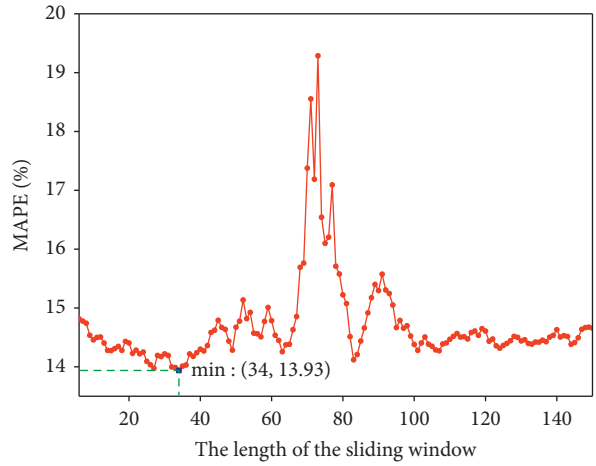

Figure 2: Relationship between sliding window length and MAPE (%).

Table 2: Model hyperparameters and value ranges.

Hyperparameter	Value range
Threshold M	$M \in [1, 4]$
Ridge parameter λ	$\lambda \in [0.0001, 4]$

Table 3: Optimization algorithm parameters.

Optimization algorithm	Parameter value
STA	$m = 30$; $\alpha_{max} = 1$; $\alpha_{min} = 10^{-4}$; $\beta = 1$; $\gamma = 1$; $\delta = 1$; $fc = 2$; maxital = 100
GWO	$m = 30$; $a \in [0, 2]$; $r_1, r_2 \in [0, 1]$; maxital = 100
GA	$m = 30$; $\text{prob}_{mut} = 0.001$; maxital = 100
PSO	$m = 30$; $\omega = 0.5$; $c_1 = 0.5$; $c_2 = 0.5$; $v_{Max} = 2$; maxital = 100

where m represents the number of seeds of each optimization algorithm, the maxital represents the maximum number of the iterations of the optimization algorithms. For the STA: the value range of the rotation factor α is $[\alpha_{max}, \alpha_{min}]$, which decreases in the form of an exponential function with $1/fc$ as the base with the increasing number of iterations; β indicates the translation factor; γ indicates the expansion factor; δ indicates the axesion factor. For the GWO: a is called the convergence factor and decreases from 2 linear to 0 with the increase of iterations; r_1 and r_2 are random numbers evenly distributed over an interval $[0, 1]$. For the GA: prob_{mut} represents the mutation probability, and the Partial-Mapped crossover is used as the crossover operator. For the PSO algorithm: ω indicates the inertia factor; c_1 indicates the local learning factor; c_2 indicates the global learning factor; v_{Max} represents the maximum speed of the particle.

Table 4: Comparison of optimization results between 4 model parameters.

	Threshold M	Ridge parameter λ	Fitness function value
STA-huber-ridge	1.54822021	3.64338165	0.203787303
GWO-huber-ridge	1.67576409	3.48931938	0.230480862
GA-huber-ridge	1.54444195	3.47396986	0.203815397
PSO-huber-ridge	1.55445167	3.99935861	0.203780642

on 10-fold cross-validation was used. The control parameter η of the fitness function was taken as 1. The performances of the STA-Huber-Ridge model, the GWO-Huber-Ridge model, the GA-Huber-Ridge model, and the PSO-Huber-Ridge model were compared and analyzed using RMSE, MAE, and MAPE evaluation functions. The optimization results of the four model parameters are shown in Table 4. The iterative comparison of their fitness values is shown in Figure 3.

It can be seen from Table 1 and Figure 3 that the fitness value of the STA algorithm dropped rapidly in the early stage of the iteration and then fell into the search for the local

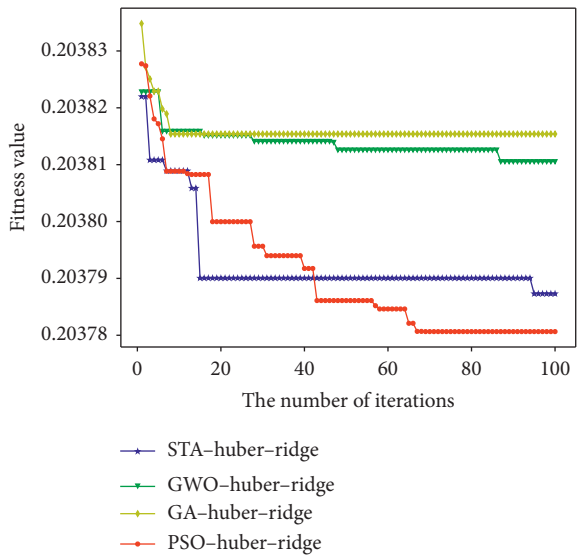

Figure 3: Comparison of four optimization iterations.

Table 5: Comparison of forecast results of 4 models.

	MAE	RMSE	MAPE (%)
STA-huber-ridge	7.06437	9.31449	13.9243
GWO-huber-ridge	7.06517	9.31559	13.9235
GA-huber-ridge	7.06418	9.31511	13.9238
PSO-huber-ridge	7.06393	9.31346	13.9230

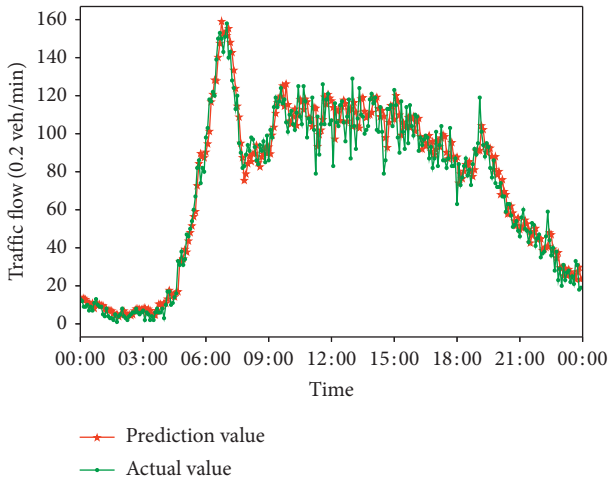

Figure 4: Forecast results of traffic flow by PSO-huber-ridge model.

optimum; after that, it dropped slowly in the later stage. The state transition algorithm used four transform operators to search. The search range was large and the early convergence was fast. However, transform operators with fixed values limited the global search capability of the state transition algorithm [28]. The fitness value of the GWO algorithm decreased slowly in the iterative process. The global optimization efficiency was not high. The GWO algorithm may easily fall into the local optimum and be unsuccessful in finding the global best [29]. The control parameters of the GWO algorithm decreased linearly with the iterative process, which cannot satisfy the complex search process [30]. The fitness value of GA stagnated in the early stage of the iteration and fell into the search for the local optimum. This is because the genetic algorithm has a premature phenomenon [31], making it difficult to jump out of the local optimum. Compared with the GWO, GA, and STA optimization algorithms, the PSO algorithm has a better iterative update strategy. It updates the particle position based on the individual experience of particles and the global experience

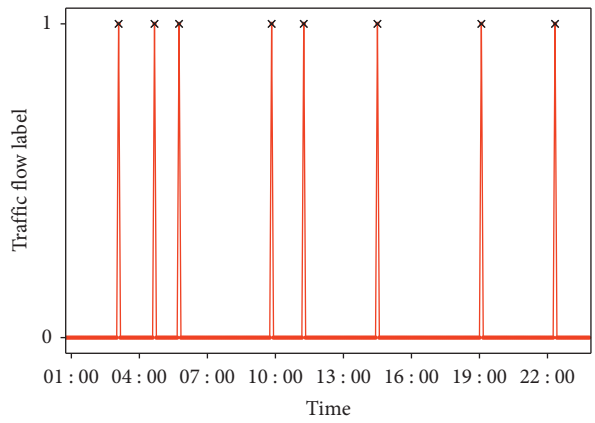

Figure 5: Traffic flow anomaly detection based on PSO-huber-ridge model.

of the particle swarm so that it will not all into the search for the local optimum easily.

The forecast evaluation results of the four models are shown in Table 5. The forecast result of the PSO-Huber-Ridge model is shown in Figure 4.

It can be seen from Table 5 that the PSO-Huber-Ridge model had the lowest MAE, RMSE, and MAPE; that is, the forecast performance of the PSO-Huber-Ridge model was the best. It can be seen from Figure 4 that the PSO-Huber-ridge model can well forecast the trend of the traffic flow at most time points.

Based on the error between the predicted value of the PSO-Huber-Ridge model and the actual value, the anomaly detection was performed on the traffic flow using the threshold (mean $\pm 2\sigma$) by calculating the mean value (mean) and variance (σ) of error data in a sliding window with a length of 10. If the forecast error at the next time point of the sliding window was greater than the anomaly detection threshold, the traffic flow at this time point was defined as an abnormal flow. The abnormal warnings would be reported to relevant traffic departments to avoid possible traffic jams. The label for abnormal traffic flow was defined as 1 and the label for normal traffic flow was defined as 0. The traffic flow anomaly detection based on the PSO-Huber-Ridge model is shown in Figure 5. It can be seen from Figure 5 that the proposed model can well detect the abnormal traffic flow in each period time.

6. Conclusions

To solve the problem of the large data noises in traffic flow, the traffic flow anomaly detection based on PSO-Huber-Ridge model is proposed. The strong robustness of the Huber function enables it to effectively reduce the influence of noise in traffic flow data on model training. The addition of the L_2 regular term of the ridge regression in the objective function can reduce the risk of model overfitting. The sum of $\mathrm{RMSE}_{k_{cv}}$ and $\mathrm{MAE}_{k_{cv}}$ based on 10-fold cross-validation is constructed as the fitness function to improve the generalization ability of the model. The optimal model parameters can be obtained through the particle swarm optimization algorithm so as to improve the model performance. Compared with the STA-Huber-Ridge, GA-Huber-Ridge, and GWO-Huber-Ridge models, the experimental results show that the PSO-Huber-Ridge model has the best model forecast performance. The traffic flow anomaly detection is performed using the traffic flow forecasted by the PSO-Huber-Ridge model. The error between the forecasted and actual traffic flow at a certain time point is large, which indicates that the regular pattern of traffic flow at that time point is different from that of history and may cause traffic congestion. The anomaly detection is performed on the traffic flow using the threshold (mean $\pm 2\sigma$). The experimental results verify the validity of the proposed traffic flow anomaly detection model.

The information contained in the traffic flow is complex. The PSO-Huber-Ridge model is limited to explore the linear information in the traffic flow. The nonlinear information needs further analysis and exploration. When extracting statistical features in feature engineering, the optimal sliding window is determined by the method of exhaustion. Its disadvantage is that it takes a long time and is not easy to apply. Using an adaptive method to extract features will greatly reduce the time of feature engineering. The Huber loss function reduces the negative impact of the data noise on the model training by reducing the penalty for large errors. Combining the Huber function with outlier detection method in data preprocessing can further improve the robustness of the model. Using adaptive feature extraction to mine linear and nonlinear information on the basis of improving model robustness is the next step.

Data Availability

The data used to support the findings of this study are currently under embargo, while the research findings are commercialized. Requests for data, 6/12 months after publication of this article, will be considered by the corresponding author.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Authors' Contributions

All authors contributed equally to this work.

Acknowledgments

This research was funded by the National Natural Science Foundation of China (Grant no. 61403046), the Natural Science Foundation of Hunan Province, China (Grant no. 2019JJ40304), Changsha University of Science and Technology "The Double First Class University Plan" International Cooperation and Development Project in Scientific Research in 2018 (Grant no. 2018IC14), the Research Foundation of Education Bureau of Hunan Province (Grant no. 19K007), Hunan Provincial Department of Transportation 2018 Science and Technology Progress and

Innovation Plan Project (Grant no. 201843), the Key Laboratory of Renewable Energy Electric-Technology of Hunan Province, the Key Laboratory of Efficient and Clean Energy Utilization of Hunan Province, Innovative Team of Key Technologies of Energy Conservation, Emission Reduction and Intelligent Control for Power-Generating Equipment and System, CSUST, Hubei Superior and Distinctive Discipline Group of Mechatronics and Automobiles (Grant no. XKQ2020009), National Training Program of Innovation and Entrepreneurship for Undergraduates (Grant no. 202010536016), Major Fund Project of Technical Innovation in Hubei (Grant no. 2017AAA133), and Hubei Natural Science Foundation Youth Project (Grant no. 2020CFB320).

References

[1] 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, p. 21, 2015.

[2] 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.

[3] X. Luo, L. Niu, and S. Zhang, "An algorithm for traffic flow prediction based on improved SARIMA and GA," *KSCE Journal of Civil Engineering*, vol. 22, no. 10, pp. 4107–4115, 2018.

[4] L. Qu, W. Li, W. Li, D. Ma, and Y. Wang, "Daily long-term traffic flow forecasting based on a deep neural network," *Expert Systems with Applications*, vol. 121, pp. 304–312, 2019.

[5] W. Zhang, Y. Yu, Y. Qi, F. Shu, and Y. Wang, "Short-term traffic flow prediction based on spatio-temporal analysis and CNN deep learning," *Transportmetrica A: Transport Science*, vol. 15, no. 2, pp. 1688–1711, 2019.

[6] K. Zhang, L. Zheng, Z. Liu, and N. Jia, "A deep learning based multitask model for network-wide traffic speed prediction," *Neurocomputing*, vol. 396, pp. 438–450, 2020.

[7] L. N. N. Do, H. L. Vu, B. Q. Vo, Z. Liu, and D. Phung, "An effective spatial-temporal attention based neural network for traffic flow prediction," *Transportation Research Part C: Emerging Technologies*, vol. 108, pp. 12–28, 2019.

[8] D. Wang, C. Wang, J. Xiao, Z. Xiao, W. Chen, and V. Havyarimana, "Bayesian optimization of support vector machine for regression prediction of short-term traffic flow," *Intelligent Data Analysis*, vol. 23, no. 2, pp. 481–497, 2019.

[9] X. Luo, D. Li, and S. Zhang, "Traffic flow prediction during the holidays based on DFT and SVR," *Journal of Sensors*, vol. 2019, Article ID 6461450, 10 pages, 2019.

[10] Y. Djenouri, A. Belhadi, J. C. Lin, and A. Cano, "Adapted *K*-nearest neighbors for detecting anomalies on spatio-temporal traffic flow," *IEEE Access*, vol. 7, pp. 10015–10027, 2019.

[11] Y. Chen, J. Pu, J. Du, Y. Wang, and Z. Xiong, "Spatial-temporal traffic outlier detection by coupling road level of service," *IET Intelligent Transport Systems*, vol. 13, no. 6, pp. 1016–1022, 2019.

[12] Z. Zhang, Q. He, H. Tong, J. Gou, and X. Li, "Spatial-temporal traffic flow pattern identification and anomaly detection with dictionary-based compression theory in a large-scale urban network," *Transportation Research Part C: Emerging Technologies*, vol. 71, pp. 284–302, 2016.

[13] P. Petrus, "Robust Huber adaptive filter," *IEEE Transactions on Signal Processing*, vol. 47, no. 4, pp. 1129–1133, 1999.

[14] D. W. Marquardt and R. D. Snee, "ridge regression in practice," *The American Statistician*, vol. 29, no. 1, pp. 3–20, 1975.

[15] P. Smets and R. Kennes, "The transferable belief model," *Artificial Intelligence*, vol. 66, no. 2, pp. 191–234, 1994.

[16] Z. Zhang, J. Yin, N. Wang, and Z. Hui, "Vessel traffic flow analysis and prediction by an improved PSO-BP mechanism based on AIS data," *Evolving Systems*, vol. 10, no. 3, pp. 397–407, 2019.

[17] C. Luo, C. Huang, J. Cao et al., "Short-term traffic flow prediction based on least square support vector machine with hybrid optimization algorithm," *Neural Processing Letters*, vol. 50, no. 3, pp. 2305–2322, 2019.

[18] Q. Ma, "Design of BP neural network urban short-term traffic flow prediction software based on improved particle swarm optimization," *AIP Conference Proceedings*, vol. 2073, no. 1, Article ID 020085, 2019.

[19] W. Cai, J. Yang, Y. Yu, Y. Song, T. Zhou, and J. Qin, "PSO-ELM: a hybrid learning model for short-term traffic flow forecasting," *IEEE Access*, vol. 8, pp. 6505–6514, 2020.

[20] L. Lin, J. C. Handley, Y. Gu, L. Zhu, X. Wen, and A. W. Sadek, "Quantifying uncertainty in short-term traffic prediction and its application to optimal staffing plan development," *Transportation Research Part C: Emerging Technologies*, vol. 92, pp. 323–348, 2018.

[21] P. J. Huber, "Robust estimation of a location parameter,"‘-Robust estimation of a location parameter," in *Breakthroughs in Statistics: Methodology and Distribution*, S. Kotz and N. L. Johnson, Eds., pp. 492–518, Springer New York, New York, NY, USA, 1992.

[22] A. E. Hoerl and R. W. Kennard, "Ridge regression: biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[23] A. B. Owen, "A robust hybrid of lasso and ridge regression," *Contemporary Mathematics*, vol. 443, pp. 59–72, 2006.

[24] J. D. Rodriguez, A. Perez, and J. A. Lozano, "Sensitivity analysis of *k*-fold cross validation in prediction error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, pp. 569–575, 2010.

[25] X. Zhou, C. Yang, C. Yang, and W. Gui, "State transition algorithm," *Journal of Industrial & Management Optimization*, vol. 8, no. 4, pp. 1039–1056, 2012.

[26] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[27] D. Whitley, "A genetic algorithm tutorial," *Statistics and Computing*, vol. 4, no. 2, pp. 65–85, 1994.

[28] X. Zhou, J. Long, C. Xu, and G. Jia, "An external archive-based constrained state transition algorithm for optimal power dispatch," *Complexity*, vol. 2019, Article ID 4727168, 11 pages, 2019.

[29] W. Long, J. Jiao, X. Liang, and M. Tang, "An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization," *Engineering Applications of Artificial Intelligence*, vol. 68, pp. 63–80, 2018.

[30] W. Long, J. Jiao, X. Liang, and M. Tang, "Inspired grey wolf optimizer for solving large-scale function optimization problems," *Applied Mathematical Modelling*, vol. 60, pp. 112–126, 2018.

[31] S. Yu and S. Kuang, "Fuzzy adaptive genetic algorithm based on auto-regulating fuzzy rules," *Journal of Central South University of Technology*, vol. 17, no. 1, pp. 123–128, 2010.