Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

Low-cost and high-performance abnormal trajectory detection based on the GRU model with deep spatiotemporal sequence analysis in cloud computing

Guohao Tang[1], Huaying Zhao[2] and Baohua Yu[1*]

Abstract

Trajectory anomalies serve as early indicators of potential issues and frequently provide valuable insights into event occurrence. Existing methods for detecting abnormal trajectories primarily focus on comparing the spatial characteristics of the trajectories. However, they fail to capture the temporal dimension's pattern and evolution within the trajectory data, thereby inadequately identifying the behavioral inertia of the target group. A few detection methods that incorporate spatiotemporal features have also failed to adequately analyze the spatiotemporal sequence evolution information; consequently, detection methods that ignore temporal and spatial correlations are too one-sided. Recurrent neural networks (RNNs), especially gate recurrent unit (GRU) that design reset and update gate control units, process nonlinear sequence processing capabilities, enabling effective extraction and analysis of both temporal and spatial characteristics. However, the basic GRU network model has limited expressive power and may not be able to adequately capture complex sequence patterns and semantic information. To address the above issues, an abnormal trajectory detection method based on the improved GRU model is proposed in cloud computing in this paper. To enhance the anomaly detection ability and training efficiency of relevant models, strictly control the input of irrelevant features and improve the model fitting effect, an improved model combining the random forest algorithm and fully connected layer network is designed. The method deconstructs spatiotemporal semantics through reset and update gated units, while effectively capturing feature evolution information and target behavioral inertia by leveraging the integration of features and nonlinear mapping capabilities of the fully connected layer network. The experimental results based on the GeoLife GPS trajectory dataset indicate that the proposed approach improves both generalization ability by 1% and reduces training cost by 31.68%. This success do provides a practical solution for the task of anomaly trajectory detection.

Keywords Trajectory anomalies, Anomaly detection, Deep spatiotemporal sequence analysis, Improved GRU

Introduction

With the development of sensor network technology, communication and positioning technology becoming increasingly mature, and the wide application of various positioning devices and mobile intelligent terminals, large-scale collection of location-related information of mobile objects (people, vehicles, ships, animals, etc.) is possible. Various sensor devices can be utilized to accurately obtain spatiotemporal data, encompassing

*Correspondence:
Baohua Yu
ybh-sharp@foxmail.com
[1] Department of Software Engineering, Shihezi University, Shihezi 832000, China
[2] Department of Tourism and Geography, Shihezi University, Shihezi 832000, China

geographic coordinates, speed, direction, and time stamps. Sequential spatiotemporal data describe the user's temporal and spatial attributes and contain significant semantic information such as target behavior, state and preference [1]. Effectively utilizing these data has emerged as a prominent research focus for scholars both domestically and internationally. Areas of investigation include spatiotemporal data semantic annotation [2, 3], trajectory clustering [4], pattern recognition [5], spatiotemporal recommendation [6], and spatiotemporal prediction [7]. Understanding the semantics of spatiotemporal data expression, especially trajectory semantic information, plays an important role in urban planning, traffic logistics management and emergency prevention. As an effective means of data utilization, anomaly trajectory detection is widely applied in various domains and holds significant research significance in practice [8, 9].

Anomaly discovery is an important aspect of data management and analysis. Unlike noise, which hampers the quality of datasets and affects data analysis, anomalies [10] often indicate specific events and thus have higher research value. Trajectory anomalies of individuals or clusters, such as public security emergencies, urban traffic congestion, and paralysis, can be identified in advance. In this paper, abnormal trajectories meet the following conditions [11]: (1) the same number or a similar number of trajectories in the trajectory dataset are rare; (2) there is a great difference between the trajectory characteristics and its own space–time domain: the trajectory does not follow an expected pattern or behaves differently from other objects based on similarity criteria (travel time, motion characteristics, data distribution, etc.).

Due to the inherent characteristics of trajectory data, including uncertainty, sparsity, skewed distribution, large scale, and fast updates [12, 13], researching anomaly detection is relatively complicated, and real-time anomaly detection for trajectory data remains relatively scarce. Trajectory data exhibit temporal evolution, and trajectory anomalies are not constant. Even if the current trajectory is similar to its space–time neighbors, it may still evolve into an anomaly with time. Cloud computing [14] is a computational paradigm endowed with robust real-time processing capabilities. Through centralized management and scheduling of network resources, virtualization services can extract abnormal evolution in time, and have been widely used in intelligent transportation tasks [15]. Additionally, deep learning methods have been proven to be able to automatically learn features directly from big data. Based on this feature, phased iterative classification models can more accurately identify evolutionary anomalies within trajectory sequences [16, 17]. As an excellent implementation unit in the field of deep learning, recurrent neural networks (RNNs) process sequence data well. By recursively calling cyclic units in the direction of sequence evolution, RNNs have certain advantages over other network models in learning nonlinear features of sequences. At the same time, the recursive call process can overlay input, update, forget and other types of gate control units. From a microscale perspective, recursion strengthens the independent information storage and processing capacity of each cycle unit. From a macro perspective, the network characteristics of coherent memory and selective forgetting are given to the RNNs ontology. Benefitting from their neural unit structure design, RNNs have better basic conditions for time series correlation analysis and abnormal feature capture. In addition, this network is widely used in natural language processing, such as speech recognition, language modeling, image captioning, machine translation and other sequence fields.

In this study, a novel approach to anomaly trajectory detection that leverages an improved gate recurrent unit (GRU) neural network is proposed. Since space–time logic is strictly continuous and the trajectory data have an absolutely linear causal performance, any bidirectional RNNs correlation models that can guide the past with the future are not considered in this paper. Unlike other deep learning models that handle temporal and spatial features separately, making it challenging to capture spatiotemporal dependencies, the structure of the GRU is more flexible. Its recurrent gating units are capable of dynamically adjusting internal weights with the sequence variations, allowing it to better adapt to the spatiotemporal evolution features at different time steps. Simultaneously, leveraging the provided flexible, efficient, reliable, and cost-effective computing services, the cloud computing environment [18] emerges as the primary infrastructure choice to support the proposed detection model in this study. To present the innovative contributions and practical significance of this paper more clearly, we summarize our contributions in the following points:

i) We utilized three models and three combination models based on a recurrent neural network (RNN) for the task of abnormal trajectory detection. We also improved the above models and conducted comparative experiments.
ii) To enhance the integration of hidden features, a network model that combines GRU with a fully connected layer has been designed. This design aims to enhance the classification and extraction abilities of the original model's hidden features.
iii) The proposed improved GRU model addresses the potential impact of irrelevant track features on training and detection by employing the random forest algorithm for feature pruning.iv) The outstanding

detection capability validates the effectiveness of RNNs in spatiotemporal sequence processing tasks, while the improvement in generalization ability and training efficiency makes the model more conducive to widespread applications.

Related work

Current trajectory anomaly detection technologies can be broadly classified into five categories based on varying application requirements and implementation principles: classification-based detection methods, historical similarity-based detection methods, distance-based detection methods, grid division-based detection methods, and deep learning-based detection methods [19]. This paper explores some specific advancements in these fields, which are summarized as follows.

Classification-based and historical similarity-based abnormal trajectory detection

The classification-based and historical similarity-based detection methods excel in scenarios involving missing training label datasets. Lei et al. [20] proposed the MT-MAD method for anomaly detection in massive maritime vessel trajectory data. By exploring the behavioral patterns of extracted moving sequences in the target region and modeling similar behaviors, the method is capable of leveraging this classification model to compare and identify suspicious trajectories in the detected trajectory data. Similarly, Wang et al. [21] proposed an anomalous trajectory detection and classification (ATDC) method to identify anomalies under different abnormal patterns, which employed the difference and intersection set (DIS) distance metric to evaluate the similarity between any two trajectories. Based on this distance, the method devised an anomaly score function to quantify the differences between different types of anomalous trajectories and normal trajectories. This matrix is then used to train a one-class support vector classifier for outlier detection. Zhu et al. [22] proposed a time-dependent popular path (TPRO) method based on the analysis of track outlier points based on the historical similarity of tracks. By conducting group analysis on historical tracks, popular paths for each group within each period are extracted, and tracks deviating from these popular paths are identified as anomalies. Navarro et al. proposed a methodology to detect outliers by combining several sources of information at the similarity level. The authors defined a similarity measure for each source of information and combined them with clustering results to produce a general similarity matrix.

Distance-based and grid division-based abnormal trajectory detection

Anomaly determination dependent on direction and density is widely used in detection methods based on distance and grid division. Román et al. [23] presented an original method to detect anomalous human trajectories based on a new and promising context-aware distance, which highlights a context-aware distance measure between trajectories. The authors used the distance matrix to extract homogeneous groups (clusters) of trajectories using an unsupervised learning method. Subsequently, an outlier detection method is applied to the trajectories in each cluster. Chen et al. [24] proposed an abnormal-trajectory detection method based on a variable grid to detect fraudulent behavior by taxi drivers when carrying passengers. By using a kernel density analysis method to divide the urban road network into three regions to set grids with different sizes, the authors obtained trajectory tuples and developed a trajectory-abnormality probability function to calculate the deviation of each trajectory from the benchmark trajectory to detect abnormal trajectories. Similarly, Xia et al. [25] proposed a cost-factor-based anomaly score model (ASM-CF) to detect anomalous trajectories of detour behavior. The authors first designed an urban road network rasterization approach to solve the problem of driving the same path but recoding different trajectory points. Then constructed a cost factor based on distance and duration to improve the accurate detection of anomalous trajectories of taxi drivers' detours.

State-of-the-art DL networks

The remarkable advancements in deep learning technology and computing power have made it possible to train detection models based on massive data. Wang et al. [26] combined geographic information and proposed an anomaly trajectory detection algorithm based on a combination of a recurrent neural network and a convolutional neural network (CNN). The algorithm accounts for the geospatial constraints of the trajectory by embedding the geographic information and topological constraints into a structured vector space, thereby avoiding the influence of sparse trajectories on anomaly detection. Zhou et al. [27] presented an abnormal trajectory detection framework for the online warning of highway traffic events. The framework is built on a long short-term memory (LSTM) autoencoder and incorporates an adversarial learner to introduce additional adversarial loss, enabling adversarial learning (AL) to learn better normal trajectory patterns. Additionally, an anomaly trajectory discriminator (ATD) is utilized to establish and train the model for detecting

small distance average displacements and filtering out transient false alarms. Ji et al. [28] proposed a Seq2Seq model based on an LSTM prediction network for trajectory modeling (SL-Modeling). The authors employed SL modeling to directly obtain sequence-type trajectory models of normal trajectory groups and then calculated the similarity between the models and the trajectory to be detected to detect abnormal trajectories by introducing the concepts of distance and semantic interest sequence.

In the scenario of massive data, the excessive computational cost brought about by a large number of nearest neighbor queries severely restricts the performance of distance-related detection methods. Meanwhile, detection models based on density, classification, and other methods impose high requirements on the distribution of data samples, limiting their generalization ability. With outstanding perceptual and learning capabilities, anomaly trajectory detection methods based on deep learning are gradually becoming mainstream. However, deep learning models are often regarded as black-box models, making it challenging to interpret the criteria underlying the model's judgment of anomalous trajectories. Moreover, the training and inference processes of these models demand substantial computational resources, rendering them unsuitable for real-time applications in certain high-demand scenarios of anomaly trajectory detection. This presents an opportunity for the investigation undertaken in this study.

The proposed model

In this paper, a model based on an improved GRU model is implemented to identify and detect the abnormal trajectory of a specific target. The improved GRU model enables deeper spatiotemporal sequence analysis, thereby comprehensively extracting trend information reflected by the long-distance dependencies. Figure 1 shows the overall solution of the proposed anomaly trajectory detection model. The detection scheme can be roughly divided into four stages: data preprocessing, feature extraction, trajectory clustering and model validation, and the pruning and clustering algorithms involved are described in the following sections.

Network structure design

To comprehensively analyze the internal relationship of trajectory features while preserving temporal memory, the proposed model adopts a combination structure comprising two layers of a GRU neural network with a stacked fully connected layer. As a type of RNN, GRU addresses gradient disappearance and explosion in RNN by introducing two gating units, an update gate and a reset gate. This mechanism ensures long-term memory of the characteristic information of the input model. When trajectory data are fed into the model, they are iteratively analyzed within the two-layer GRU network. At each time step, the neuron output is determined by the previous moment's neuron output and the current moment's neuron input, and the reset gate selects

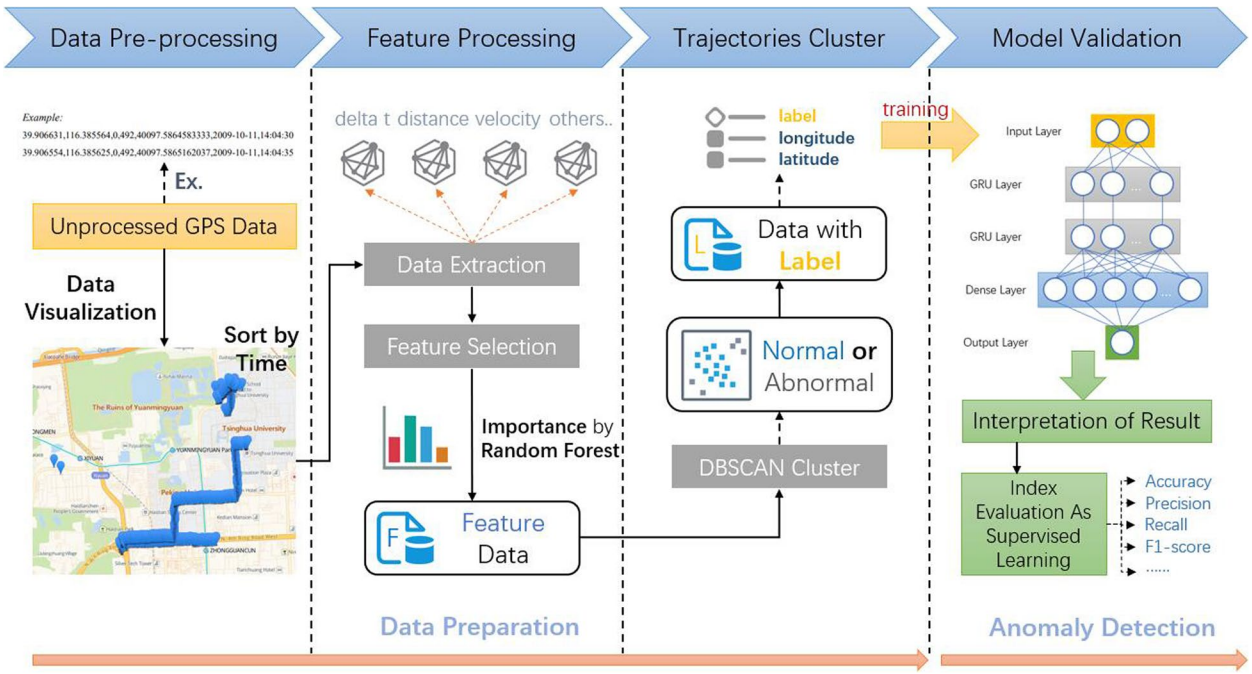

Fig. 1 Schematic diagram of the abnormal trajectory detection scheme

relevant historical information while the update gate preserves the most recent state. The processed feature information continues to flow to the fully connected layer, and the fully connected layer integrates and maps all the data characteristics, which can effectively solve the data nonlinear problem and make the model fit better. The combination of the two-layer GRU network and the fully connected layer forms the hidden layer of the model. Figure 2 shows the network structure legend of the model, where U, V,W is the weight matrix, and x_t, h_t and Y_t represent the input vector, hidden vector and output vector, respectively.

Forward propagation

The GRU network uses a time-based forward propagation algorithm. When its neural unit processes information, the processing formula at the reset gate can be expressed as:

$$r_t = \sigma(\lambda_r \cdot [h_{t-1}, x_t] + b_r) \tag{1}$$

Where b_r and λ_r represent the bias vector and weight matrix at the reset gate, respectively. σ is the sigmoid activation function, ensuring that the weight calculation results are compressed to [0, 1] so that historical information irrelevant to the prediction information can be selectively discarded in subsequent calculations. Similarly, the processing formula at the update gate can be expressed as:

$$u_t = \sigma(\lambda_u \cdot [h_{t-1}, x_t] + b_u) \tag{2}$$

Where $b_v$andλ_v represent the bias vector and the weight matrix at the update gate, respectively. Then, the state of the candidate hiding layer is calculated, and the final output to the fully connected layer at time t is calculated according to the updated gating result and the candidate hiding state:

$$h_t = tanh(\lambda_h \cdot [r_t \cdot Y_{t-1}, x_t] + b_h) \tag{3}$$

$$y_t = (1 - v_t) \cdot Y_{t-1} + v_t \cdot h_t \tag{4}$$

The neuronal nodes of the fully connected layer and the hidden layer are fully connected. This expression is shown in formula (5):

$$a_i^{(k)} = f\left(\sum_{j=1}^{n} \lambda_{ij}^{k-1} x_j^{k-1} + \varepsilon_i^{k-1}\right) \tag{5}$$

Where k indicates the number of layers corresponding to the layer network.

Backpropagation

As shown in formulas (1) to (4), the parameters involved in forward propagation that need to be continuously learned and updated by the network during the training process are λ_r,λ_u and λ_h. Taking the weight matrix λ_r of the reset gate r_t as an example, the specific formula is shown in formula (6).

$$\lambda_r = \lambda_{rx} + \lambda_{ry} \tag{6}$$

By decomposing the target parameter λ_r into two subparameters λ_{rx} and λ_{ry}, the update gate r_t can better control its learning process. λ_{rx} controls the importance of the update gate on the input information, allowing the model to flexibly learn the information from the input sequence. On the other hand, λ_{ry} controls the importance of the update gate on the previous hidden state, enabling the model to better retain historical information and learn the dependencies between sequential

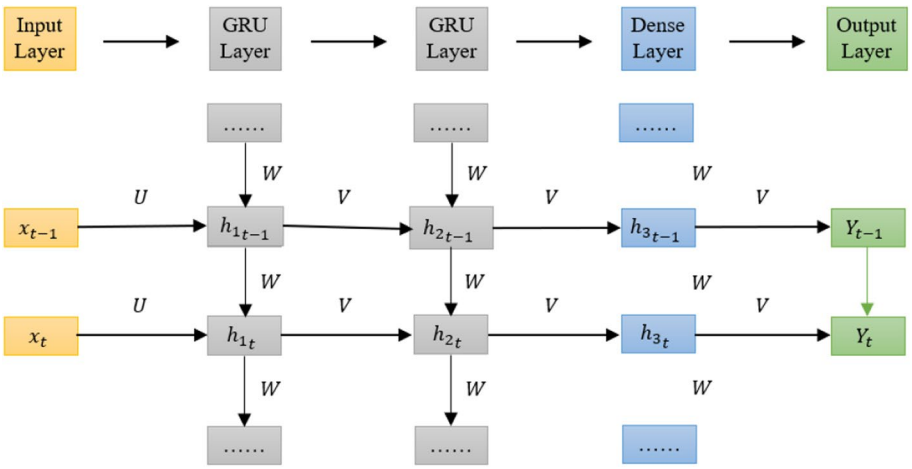

Fig. 2 The structural design of the improved GRU network model

data. The output of the output layer at time step t is shown in formula (7).

$$Y_t = \sigma(\lambda_o \cdot y_t) \tag{7}$$

Where λ_o represent the weight matrix at the output layer. After obtaining the predicted model output, the cross-entropy loss function is selected as the training objective to adjust and update various parameters in the model by checking the error between the trajectory prediction results and the actual trajectory labels. The loss function is shown in formula (8).

$$L_{loss} = -\left[ylg\left(f(x,\alpha)\right) + (1-y)\lg\left(1-f(x,\alpha)\right)\right] \tag{8}$$

Where $f(x,\alpha)$ represents the predicted value of the model and α represents the model parameters. After calculating the loss function, the error terms of various parameters at time step t are calculated. The specific formulas used are shown in formulas (9) to (11).

$$\varepsilon_t = {}^{\partial L}/_{\partial y_t} \tag{9}$$

$$n_{r,t} = \lambda_r[y_{t-1}, x_t] = \lambda_{ry} y_{t-1} + \lambda_{rx} x_t \tag{10}$$

$$\varepsilon_{r,t} = {}^{\partial L}/_{\partial n_{r,t}} \tag{11}$$

As shown in formulas (12) and (13), the error terms ε_{t-1} and ε_t at time steps t − 1 and t are calculated using backpropagation.

$$\varepsilon_{t-1}^T = {}^{\partial L}/_{\partial y_{t-1}} = \left({}^{\partial L}/_{\partial y_t}\right) \cdot \left({}^{\partial y_t}/_{\partial y_{t-1}}\right) = \varepsilon_t^T \left({}^{\partial y_t}/_{\partial y_{t-1}}\right) \tag{12}$$

$$\varepsilon_t^T \left({}^{\partial y_t}/_{\partial y_{t-1}}\right) = \varepsilon_{u,t}^T \left({}^{\partial n_u}/_{\partial y_{t-1}}\right) + \varepsilon_{r,t}^T \left({}^{\partial n_r}/_{\partial y_{t-1}}\right) +$$

$$\varepsilon_{h,t}^T \left({}^{\partial n_h}/_{\partial y_{t-1}}\right) = \varepsilon_{u,t}^T \lambda_{uy} + \varepsilon_{r,t}^T \lambda_{ry} + \varepsilon_{h,t}^T \lambda_{hy} r_t + \varepsilon_t^T (1 - v_t) \tag{13}$$

Based on the calculated error terms, the weight gradients of various neurons are obtained. The specific formulas used are shown in formulas (14) to (15).

$${}^{\partial L}\Big/_{\partial \lambda_{yr}} = \sum_{i=1}^{k} \varepsilon_{r,i} y_{i-1}^T \tag{14}$$

$${}^{\partial L}/_{\partial \lambda_{xr}} = \sum_{i=1}^{k} \varepsilon_{r,i} x_r^T \tag{15}$$

After calculating the weight gradients, the parameters of various neurons are updated accordingly, and the iteration continues until the loss function converges.

Abnormal Trajectory detection process based on an improved GRU neural network

This paper proposes an improved GRU model for detecting abnormal trajectories. This model judges whether the trajectory of a user is abnormal based on the trajectory feature input. Figure 3 shows the implementation scheme of anomaly trajectory detection based on the improved GRU neural network.

Step 1. Trajectory preprocessing. The original trajectory dataset undergoes preprocessing, including cleaning and denoising operations, per the specific requirements of the proposed model.

Step 2. Trajectory feature extraction. The trajectory feature extraction algorithm is invoked to extract time, distance, speed, and other feature attributes that may affect model training and prediction from the preprocessed trajectory dataset, which are organized into an m-dimensional vector form $\left(m \in N^+\right)$.

Step 3. Trajectory clustering. Trajectories are clustered to distinguish normal and abnormal trajectories. The DBSCAN algorithm is employed to perform density-based clustering on the trajectory dataset by utilizing longitude and latitude as clustering features. This categorizes the trajectory data as normal or abnormal and provides supervised labels for feature dimensionality reduction and model training.

Step 4. Trajectory feature dimensionality reduction. Using the m-dimensional vector processed in the previous step and the data label as input, the random forest algorithm analyzes the importance of each feature and discards irrelevant features to accelerate model training and improve overall performance. The reduced features are in the form of an n-dimensional vector, serving as the input for subsequent model training $\left(n \in N^+ \cup n \leq m\right)$.

Step 5. Training the improved GRU neural network for abnormal trajectory detection. The reduced trajectory dataset is partitioned into training and testing sets, with the training set serving as input for training the improved GRU neural network. The optimizer is selected, and the loss function is defined. The parameters of the hidden layers are iteratively adjusted until the model tends to be stable.

Step 6. Abnormal trajectory recognition capability assessment based on the trained model. The testing set is fed into the trained model, and the prediction results are organized and compared to the actual label information to verify the recognition accuracy of the model and evaluate the abnormal object detection ability.

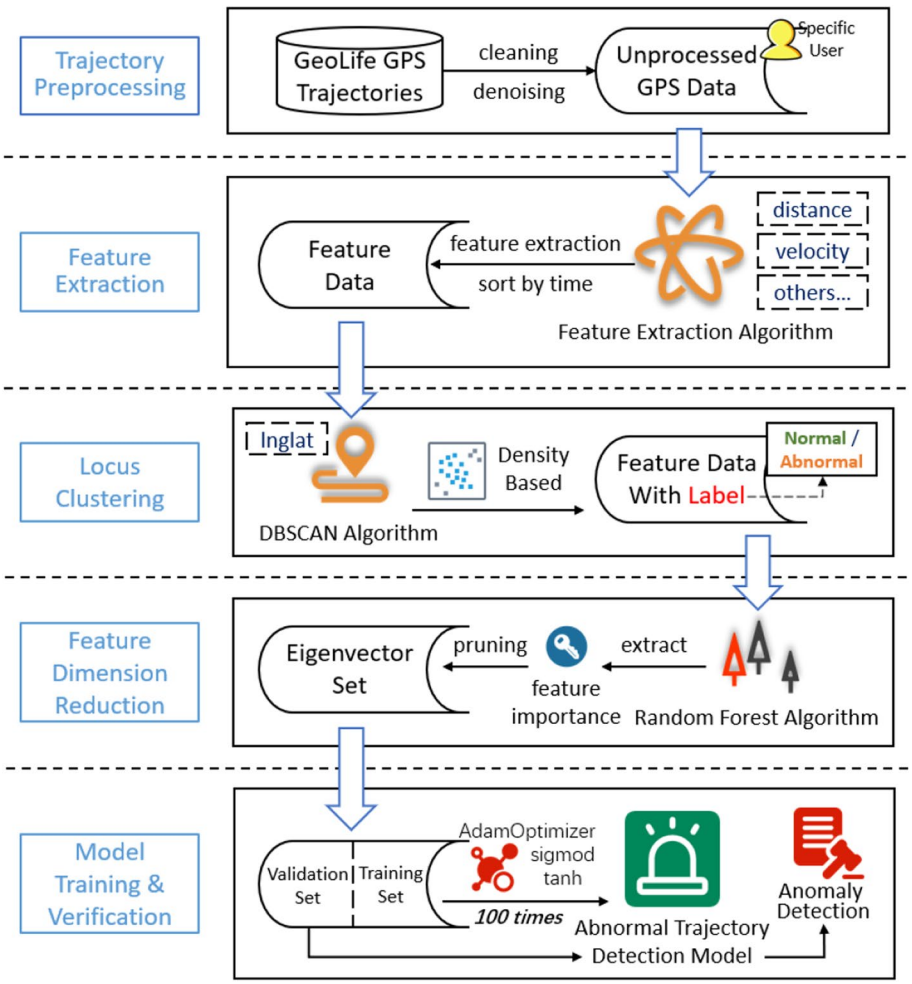

Fig. 3 Implementation scheme of anomaly trajectory detection based on the improved GRU model

Experimental results and analysis

Experiment preparation

Trajectory dataset acquisition

The experiment utilizes the GeoLife GPS trajectories dataset from the Microsoft Research Asia GeoLife project. This dataset comprises trajectory data collected from 182 users between April 2007 and August 2012. The dataset includes 17,621 trajectories with a total distance of over 1.2 million kilometers and a total time of over 48,000 h. Each trajectory consists of a sequence of points ordered by time, with each point characterized by information such as longitude, latitude, and altitude. These trajectory data record the spatial and temporal trajectories of users not only at home and work but also through a wide range of outdoor activities, such as shopping, hiking, tourism, and cycling [29].

The source data for this experiment consist of all the trajectory information of user 000 in the Geolife GPS trajectories dataset from October 23, 2008, to November 23, 2008, for a total of 31 days, which includes 11,370 trajectories. The source data contain six categories and seven features: longitude, latitude, default value column, altitude, trajectory date, and collection time. However, it does not contain specific trajectory classification labels, so it cannot be directly input into the network in the form of supervised training and must undergo subsequent processing. In this study, human activity trajectories are not significantly influenced by altitude factors. Therefore, considering only the latitude and longitude information is sufficient for the identification and analysis of anomalous trajectories. For this experiment, the unnecessary features in the source dataset are removed, retaining only longitude, latitude, trajectory date and collection time. Combining trajectory date and collection time into trajectory time and defining longitude as a, latitude as b and trajectory time as c, a single trajectory can be shown as T = (a, b, c). The trajectory dataset is represented as T_s, where $T_s = \{T_1, T_2, \cdots T_n\}$.

Trajectory feature processing

Trajectory feature extraction T_s already contains feature vectors such as latitude, longitude, and acquisition time. For any two consecutive trajectories T_i and T_{i+1} in the trajectory dataset T_s, the following features can be extracted from the existing features.

Extracting the time vector of the trajectory called Dts:

$$\text{Dts} = c_{i+1} - c_i \tag{16}$$

Extracting the travel distance vector called Ds:

$$\text{Ds} = R * \cos^{-1}(\cos b_i * \cos b_{i+1} * \cos(a_i - a_{i+1}) + \sin b_i * \sin b_{i+1}) \tag{17}$$

Where R is the radius of the earth and set to R = 6371.0 km.

Extracting speed vector called Vs:

$$\text{Vs} = \frac{Ds}{Dts} \tag{18}$$

For each trajectory in T_s, the corresponding feature information is calculated according to formulas (16, 17), and (18). The existing features in T are obtained after summarizing and organizing, which include longitude, latitude, travel time, distance, and speed. Defining travel time as d, distance as e, and speed as f, update and eventually obtain $\text{T} = (a, b, d, e, f)$.

Trajectory clustering The feature information of any trajectory T_i does not include the normal/abnormal classification label of the trajectory, so T_s cannot be directly input into the model training as supervised samples. In this paper, the DBSCAN algorithm is used to cluster all trajectories in T_s based on longitude and latitude, taking the clustering density reflecting the trajectory frequency as the main consideration for normal/abnormal classification of any trajectory T_i. The trajectory dataset is clustered as shown in Algorithm 1.

Algorithm 1. Density clustering algorithm

Input: Trajectory dataset T_s. ($T_s = \{T_1, T_2, \cdots T_n\}$)
Output: Trajectory classification labels X. ($\text{X} = [x_1, x_2, \cdots, x_n]^T$)
1 Initialize $T_s' = \emptyset$, k, Eps, MinPts
2 for i=1 to T_n do
3 add $T_i = (a_i, b_i)$ to T_s'
4 end for
5 k = 2 * dimension(T_1) − 1
6 Eps by k-distance(T_s', k)
7 MinPts = k + 1
8 X = DBSCAN(Eps, MinPts)
9 return X

When utilizing the DBSCAN algorithm for recursive clustering, the choice of parameters Eps and MinPts often directly affects the clustering results [30]. In this paper, a simple and effective heuristic algorithm [31] is adopted to determine the parameter selection and cluster the trajectories: for k defined by the feature dimension, the k-distance function is defined, and for any longitude and latitude point T_i, the distance from each point in T_s to its k-th nearest neighbor is calculated and sorted in descending order to form a k- distance graph. The inflection point in the graph, as shown in Fig. 4, is the threshold point Eps sought in this research.

The resulting inflection point value is Eps = 0.0024373 61893523881. By rounding Eps to 0.0024 and incrementing k by 1, the value of MinPts is determined to be 4. With the finalized parameters, clustering is performed. The clustering results are shown in Fig. 5.

Then, add the clustering label x to any trajectory T_i, and now $T_i = (a, b, d, e, f, x)$.

Feature selection Incorporating irrelevant and unnecessary features into the network training can have varying degrees of impact on its efficiency and even produce negative feedback. Therefore, appropriate data dimensionality reduction can not only improve the comprehensive performance of constructing regression models on the feature subset but also avoid the adverse effects of useless features on the model in certain situations. The random forest algorithm calculates the importance score of each feature by analyzing the error before and after feature random permutation. The higher the score is, the more important the feature. Unlike other feature selection algorithms, the random forest algorithm can not only reflect the interactions between features but also has advantages such as high accuracy and strong robustness [32]. Therefore, in this paper, the random forest algorithm is used to perform feature selection on the dataset. The feature importance results are shown in Fig. 6, where the horizontal axis represents the feature and the vertical axis represents the importance factor.

As shown in Fig. 6, the importance factors of speed, distance, and time are only approximately 0.01, and their impact on the trajectory label is not significant. Therefore, this can be used as a dimension reduction standard to update trajectory T_i and obtain $T_i = (a, b, x)$.

Model construction

In the experiments of this paper, there are 2 neurons in the input layer that input the valuable trajectory

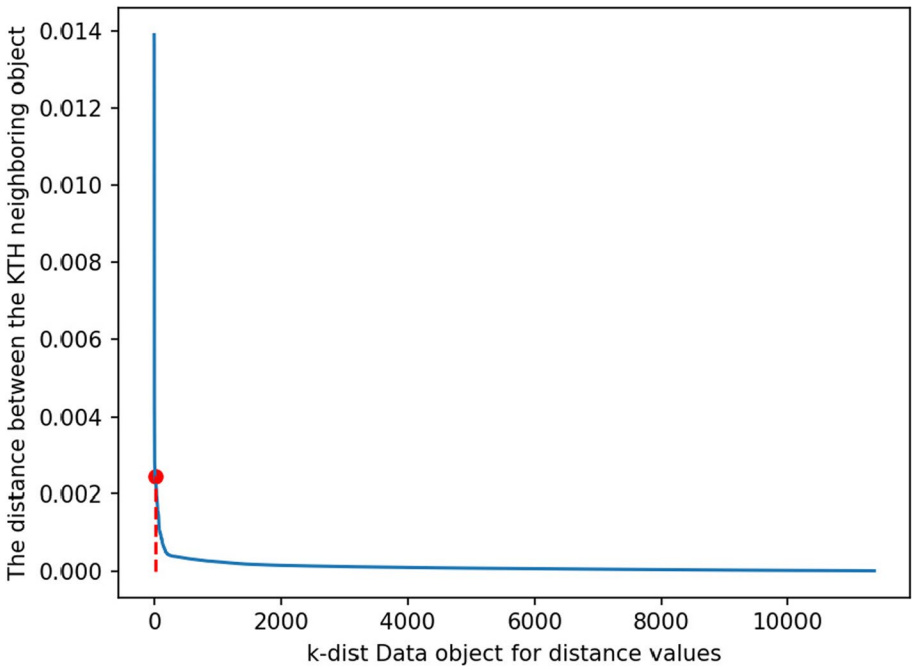

Fig. 4 k- distance diagram

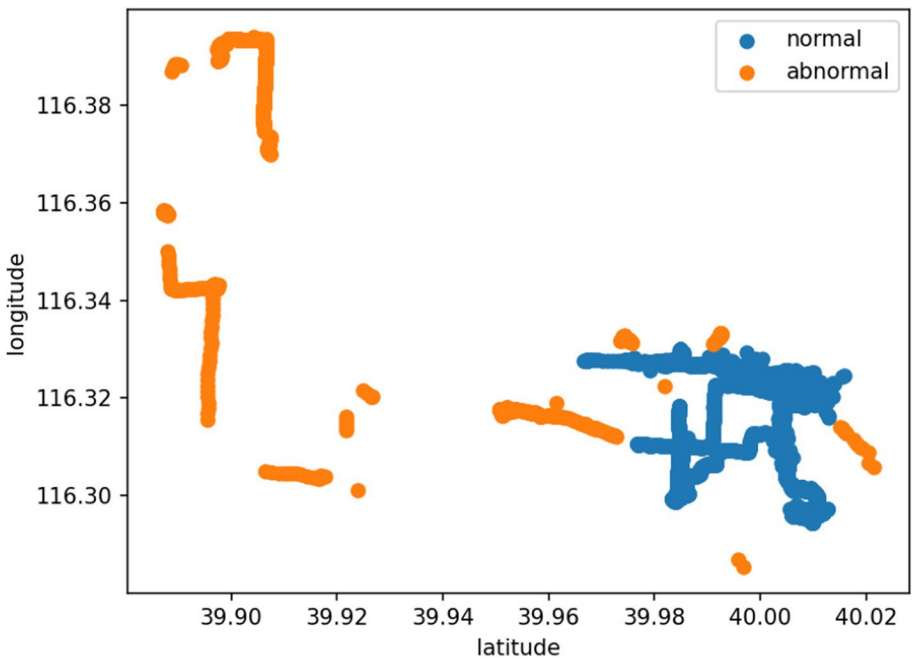

Fig. 5 DBSCAN clustering results

longitude and latitude attributes after feature selection. There is 1 neuron in the output layer, which is limited to output 0 or 1 to represent the user trajectory category: normal or abnormal. Due to the trajectory data collection interval being only 5 s, the variations in longitude and latitude in the dataset are not significant. Using a relatively low learning rate during model training allows for smoother parameter updates, avoiding large fluctuations in the parameters and reducing oscillations and instability in the training process. When evaluating the model

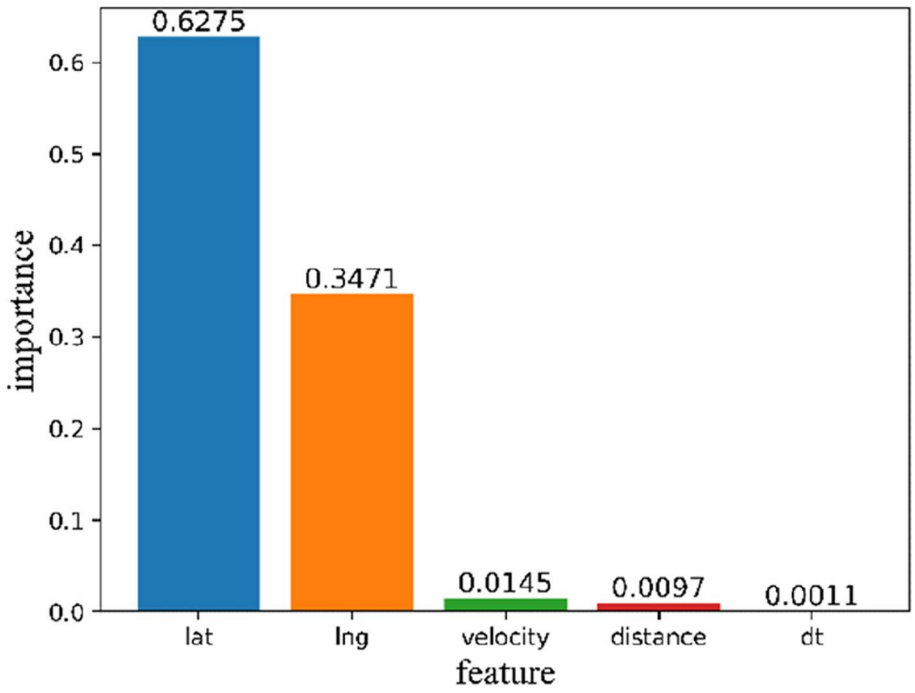

Fig. 6 Feature selection results

using the unimproved RNNs, the network parameters are set as indicated in Table 1.

Table 1 Recurrent neural network model parameter settings

Training parameters	Parameter input
The number of hidden neurons in the first layer of network	10
The number of hidden neurons in the second layer network	10
The number of hidden neurons in the third layer network	10
Learning Rate	0.0006
Activation Function	Sigmod OR tanh
Iterations	100

Model verification

The input dataset contains 11,370 trajectory information from selected users from October 23, 2008, to November 23, 2008, for a total of 31 days. Among these, the first 8,000 trajectories serve as the training dataset for the model, while the remaining 3,370 trajectories serve as the test set. Setting the step size to 20 (predicting the 21st data point based on the preceding 20 data points), and concurrently considering the model's training efficiency and adaptability, AdamOptimizer was selected as the model optimizer for controlling weight updates after a comprehensive assessment of optimizer scenario performance, as presented in Table 2. Evaluate various aspects of the model's performance using common machine learning metrics such as accuracy, precision, recall, F1 score, AUC, etc. The training model is evaluated on the test set based on the training set as the model input, with weight parameters initialized randomly.

Comparison of various recurrent neural network models

First, the original three-layer RNN and its combination models without improvement are tested. The confusion matrix of each model's classification results is shown in Fig. 7, where 0 represents abnormal points and 1 represents normal points of trajectories.

Based on the confusion matrix results of each model, the model prediction evaluation indicators are calculated as shown in Table 3, where the best results are in bold. Additionally, the ROC curve and DCA curve results are visually represented in Figs. 8 and 9, respectively.

From Table 2, it is evident that the three-layer GRU network model performs better than other models under the same conditions in abnormal trajectory detection under various evaluation indicators. Due to long-term dependencies, the performances of RNN and its combination networks are not ideal, and their generalization ability is also weak. The LSTM network has more gate

Table 2 Performance comparison of optimizers in different scenarios

Algorithm	Characteristics	Advantages	Disadvantages
Adagrad [33]	Dynamically adjusts the learning rate based on the historical gradient	Adaptive learning rate, suitable for sparse data	Learning rate decay is fast, may result in small early parameter updates
RMSprop [34]	An improvement over Adagrad, adjusts the learning rate with an exponential moving average	Adapts well to different learning rates for each parameter	Requires tuning hyperparameters, may converge slowly
Adam [35]	Combines momentum and RMSprop, with adaptive learning rates and momentum	Efficient, suitable for various data and models	Requires tuning of additional hyperparameters, may be unstable at times
Adadelta [36]	Further improvement over RMSprop, eliminates learning rate decay issues	Adaptive learning rate, reduces hyperparameter reliance	Sensitive to initialization, requires tuning of the initial learning rate

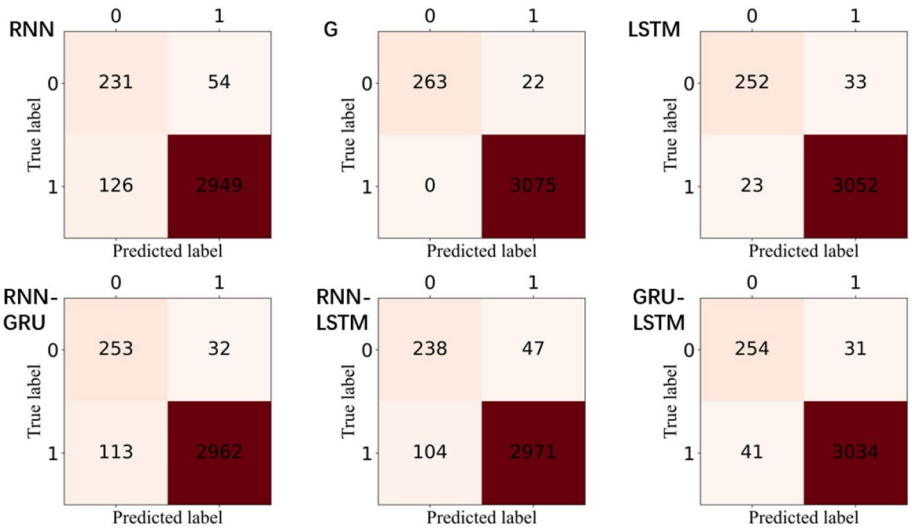

Fig. 7 Confusion matrix of the classification results of various recurrent neural networks

Table 3 Result of various recurrent neural networks with the same parameters

Model	Precision	Accuracy	Recall	F1 Score	Cost(s)
RNN	0.946429	0.647059	0.810526	0.719626	**44**
GRU	**0.993452**	**1**	**0.922807**	**0.959854**	101
LSTM	0.983333	0.916364	0.884211	0.9	81
RNN-GRU	0.956845	0.691257	0.887719	0.777266	56
RNN-LSTM	0.95506	0.695906	0.835088	0.759171	50
GRU-LSTM	0.978571	0.861017	0.891228	0.875862	96

units, increasing the risk of overfitting to some extent, which is one possible factor in it being slightly weaker than the GRU network in this scenario. The GRU network performs the best, and its simplified gate structure performs better on small sample data. The GRU combination model with other networks restricts its own advantages and weakens its performance.

In terms of generalization ability, the ROC and DCA performances of the three-layer GRU network are the best, and its generalization ability is the strongest. Due to the similarity of neural unit structures, the generalization ability of the LSTM network is also considerable, but the GRU network still performs better. The RNN network fails to effectively handle problems such

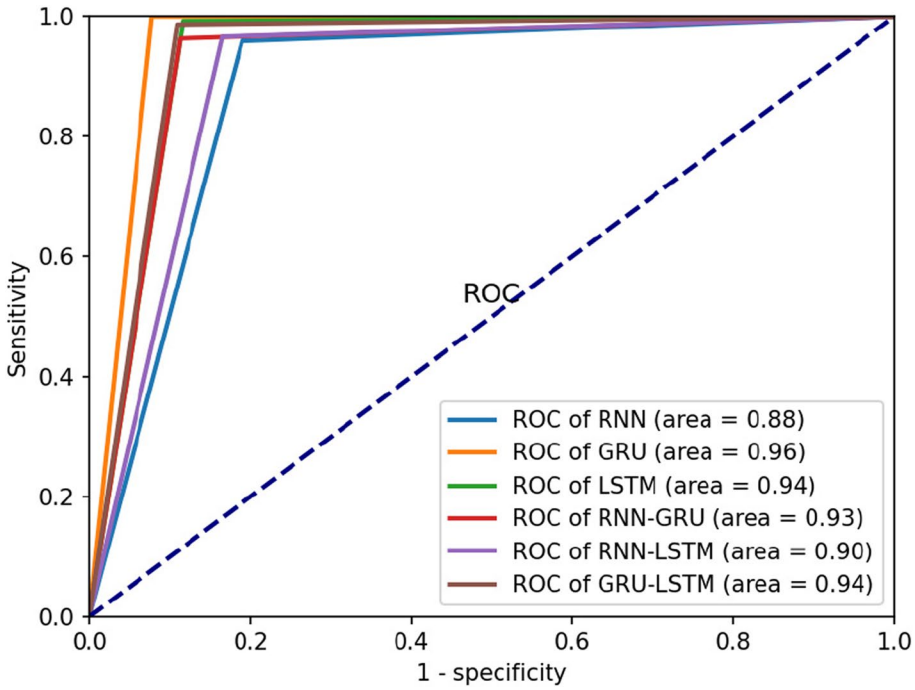

Fig. 8 Comparison of ROC curves of various recurrent neural networks

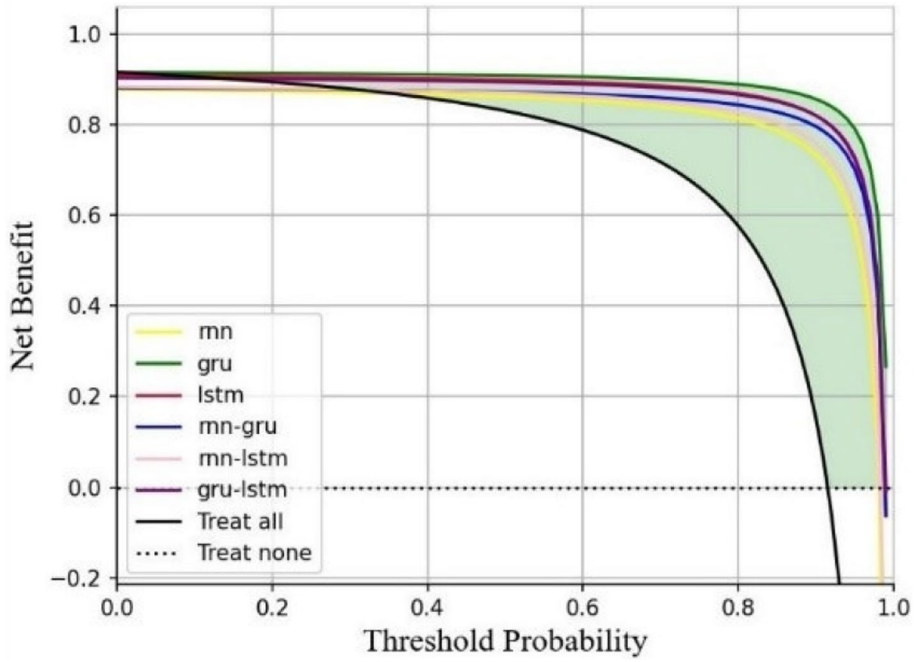

Fig. 9 Comparison of DCA curves of various recurrent neural networks

as gradient disappearance and explosion. It also easily forgets information in long-time sequence processing, leading to the training network tendency to fall into local optima, thereby affecting various indicators of the network. In terms of training efficiency, there is an inverse correlation between the training efficiency and performance of various recurrent neural networks and their combined models. Specifically, the best-performing GRU model takes the longest training time, indicating the lowest efficiency, while the moderately

performing RNN model has the shortest training time and the best training efficiency.

Comparison of various improved recurrent neural network models

When utilizing the improved three-layer model for testing the network, the network parameters are set as shown in Table 4.

Continuing the experiment, the model performances of various improved RNN models are compared. The confusion matrix of the classification results of various models is shown in Fig. 10.

The evaluation indexes of model prediction were calculated according to the confusion matrix results of each model. These indexes are shown in Table 5, with the optimal results shown in bold. The ROC curve and DCA curve are plotted as shown in Figs. 11 and 12, respectively.

Table 4 Improved three-layer network model parameter settings

Training parameters	Parameter input
The number of hidden neurons in the first layer of network	10
The number of hidden neurons in the second layer network	10
The number of hidden neurons in the fully connected layer	24
Learning Rate	0.0006
Activation Function	sigmod/tanh
Iterations	100

As shown in Table 2 and Table 4, various evaluation indexes of the improved RNN, GRU and their combined models have been improved compared with the original model, while the performance of LSTM and its combined models has fluctuated. Considering that the LSTM neural unit itself has the most gated units, there is a large overfitting risk. More model parameters increase the possibility of overfitting after superposing the fully connected layer, resulting in greatly increased stability and uncertainty of the model. For other models, the superposed fully connected layer can better capture the complex features of input data, increase the connections between hidden states, and improve the overall expressiveness of the model. The recall ability of the improved RNN model is greatly enhanced, even exceeding that of the model in this paper, which can also prove the advantages of this improvement.

In addition, the performance of the GRU model and the improved model in this paper is evaluated, and the indicators of the improved GRU model have been improved to varying degrees compared with the three-layer GRU network. Moreover, the training time of the network is reduced by 31.68% compared to the three-layer GRU network, significantly enhancing the training efficiency of the model. The ROC and DCA curves shown above demonstrate that the improved GRU model also exhibits greater generalization ability than the original network. By using the fully connected layer as a feature extractor, the network is allowed to learn higher-level representations of the input data. These higher-level features capture more abstract and meaningful information, which can improve the network's ability to generalize to different instances. As shown

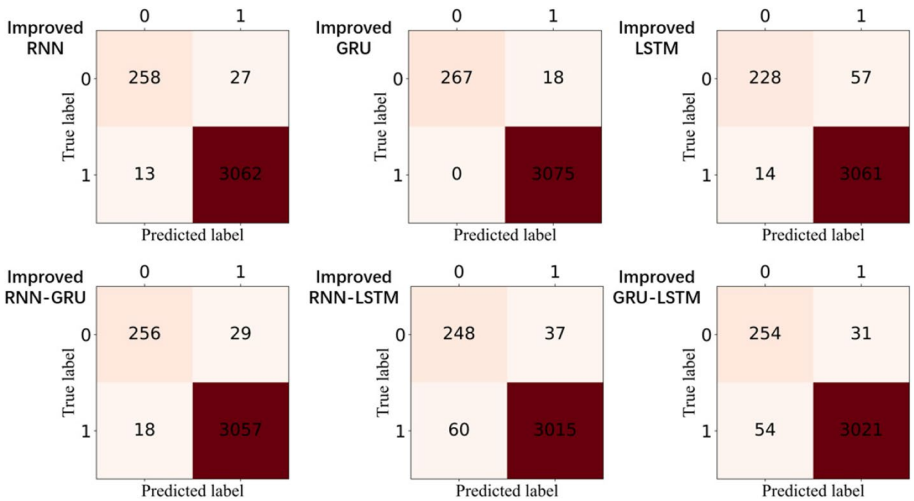

Fig. 10 Confusion matrix of the classification results of various improved recurrent neural networks

Table 5 Result of various improved recurrent neural networks with the same parameters

Model	Precision	Accuracy	Recall	F1 Score	Cost(s)
Improved RNN	0.988095	0.905263	**0.95203**	0.928058	**40**
Improved GRU	**0.994643**	**1**	0.936842	**0.967391**	69
Improved LSTM	0.978869	0.8	0.942149	0.865275	63
Improved RNN-GRU	0.986012	0.898246	0.934307	0.915921	51
Improved RNN-LSTM	0.971131	0.870175	0.805195	0.836425	49
Improved GRU-LSTM	0.974702	0.891228	0.824675	0.856661	67

in Fig. 13, the loss curves of the two types of models continue to be drawn, and the convergence speed of the two models is evaluated.

In Fig. 13, although improving the random value of the parameter matrix initialization of the GRU model is detrimental to model convergence, the convergence efficiency of the improved GRU model is still better than that of the three-layer GRU network. This shows that the model in this paper has a better training effect and better detection performance in abnormal trajectory recognition.

In this paper, the random forest algorithm is used to analyze the importance of input features that may affect the prediction results. The vector after dimensionality reduction performs well when the input features are only longitude and latitude. Therefore, when dealing with abnormal trajectory detection or similar tasks, we can simplify the tedious feature extraction process in the data preprocessing stage, as discussed in the conclusions of this paper. After replacing the original network with the fully connected layer, the processing flow of the gate control unit is reduced, which greatly improves the training efficiency of the proposed model. Additionally, when the new fully connected layer replaces the original layer network for feature integration, the feature information from the initial two hidden layers can be more comprehensively and effectively integrated, improving various performance metrics of the model to varying degrees.

Conclusion

Aiming to address the low spatiotemporal correlation and limited recognition accuracy of the current anomaly trajectory detection model, this paper proposes an anomaly

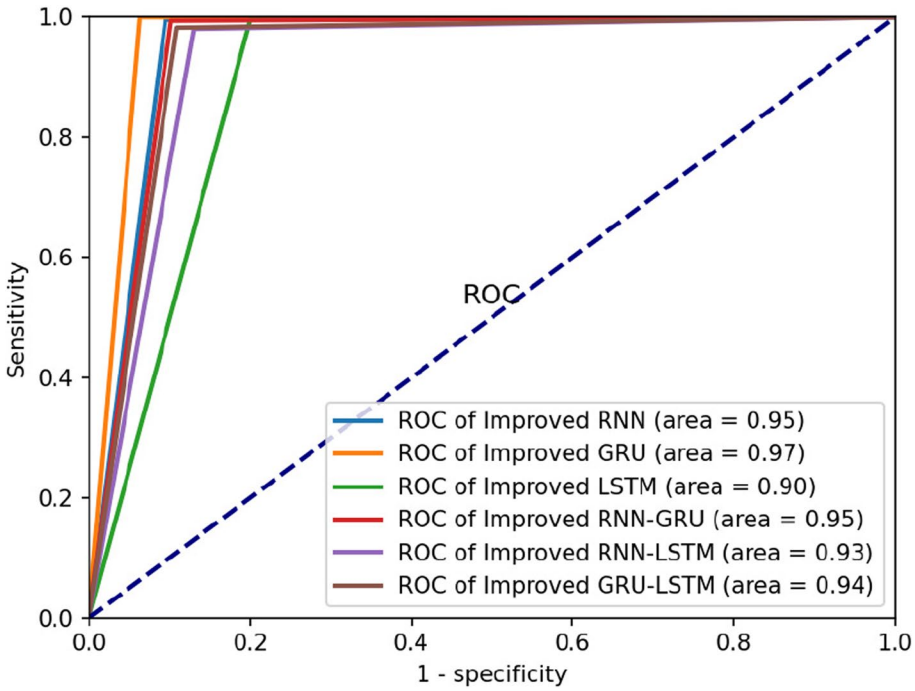

Fig. 11 Comparison of ROC curves of various improved recurrent neural networks

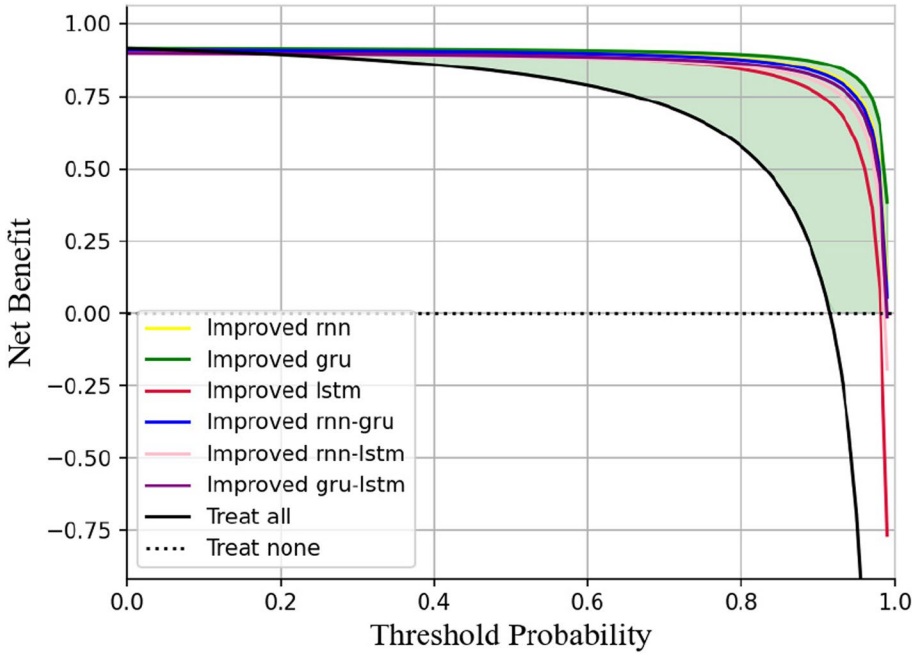

Fig. 12 Comparison of DCA curves of various improved recurrent neural networks

trajectory detection method based on an improved GRU model in cloud computing and describes in detail the data preprocessing, network structure design, network parameter selection and model training. Experiments are completed on the GeoLife GPS trajectories dataset from Microsoft Research Asia. The results demonstrate that the proposed model outperforms other RNNs and their combinations in anomaly recognition accuracy and training cost while also exhibiting a certain level of generalization ability. In future work, the author will try to

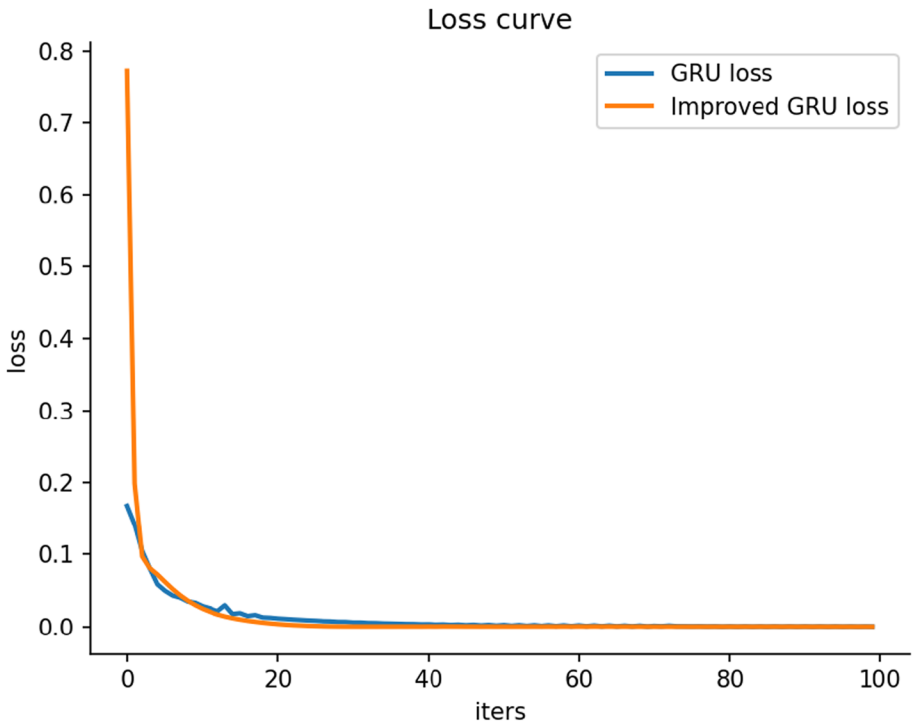

Fig. 13 Comparison of loss curves between GRU and improved GRU

introduce the attention mechanism to improve the proposed model, fully use the important factors extracted by the random forest algorithm to determine the feature input form, and further improve the accuracy and efficiency of the anomaly trajectory detection model.

Authors' contributions

B.Y. designed the primary content of the experiments and provided all the support for the experimental environment. G.T. wrote the main manuscript text and created all the figures. H.Z. assisted in processing the trajectory data and helped organize the table information. All authors reviewed the manuscript.

Funding

This research is supported by the Financial and Science Technology Plan Project of Xinjiang Production and Construction Corps under grant 2020DB005.

Availability of data and materials

The data that support the findings of this study are available on request from the corresponding author, [B.Y.], upon reasonable request.

Declarations

Ethics approval and consent to participate

Not applicable.

Competing interests

The authors declare no competing interests.

Received: 28 August 2023 Accepted: 10 February 2024
Published online: 05 March 2024

References

1. Yao D, Zhang C, Huang JH, Chen YX, Bi JP (2018) Semantic understanding of spatio-temporal data: technology & application. Ruan Jian Xue Bao/J Softw 29(7):2018–2045. in Chinese. http://www.jos.org.cn/1000-9825/5576.htm
2. Wang B, Wei H, Li R, Liu S, Wang K (2023) Rumor detection model fused with static spatiotemporal information. J Intelligent Fuzzy Syst 44(2):2847–2862. https://doi.org/10.3233/jifs-220417
3. Wang J et al (2022) STHGCN: a spatiotemporal prediction framework based on higher-order graph convolution networks. Knowl-Based Syst 258:109985. https://doi.org/10.1016/j.knosys.2022.109985
4. Yuan G, Sun P, Zhao J, Li D, Wang C. A review of moving object trajectory clustering algorithms. Artif Intell Rev. 2017:123–144. https://doi.org/10.1007/s10462-016-9477-7
5. Zhao E, Du P, Sun S (2022) Historical pattern recognition with trajectory similarity for daily tourist arrivals forecasting. Expert Syst Appl 203:117427. https://doi.org/10.1016/j.eswa.2022.117427
6. Guan Q, Reich BJ, Laber EB (2020) A spatiotemporal recommendation engine for malaria control. Cornell University - arXiv,Cornell University – arXiv
7. Zang H, Zhu J, Gao Q (2022) Deep learning architecture for flight flow spatiotemporal prediction in airport network. Electronics 11(23):4058. https://doi.org/10.3390/electronics11234058
8. Zheng Li, Xiaolong Xu, Tian Hang, Haolong Xiang, Yan Cui, Lianyong Qi, Xiaokang Zhou (2022) A knowledge-driven anomaly detection framework for social production system https://doi.org/10.1109/TCSS.2022.3217790
9. Liu W, Xu X, Wu L, Qi L, Jolfaei A, Ding W, Khosravi MR (2023) Intrusion detection for maritime transportation systems with batch federated aggregation. IEEE Trans Intell Transp Syst 24(2):2503–2514
10. Kwak SK, Kim JH (2017) Statistical data preparation: management of missing values and outliers. Korean J Anesthesiol. 70(4):407. https://doi.org/10.4097/kjae.2017.70.4.407
11. Shengjun Q, Ting L (2021) An anomaly detection algorithm for traffic trajectory data based on long short term memory model
12. Zhou W, Yu Y, Zhan Y, Wang C (2022) A vision-based abnormal trajectory detection framework for online traffic incident alert on freeways. Neural Comput Appl 14945–14958. https://doi.org/10.1007/s00521-022-07335-w
13. Wang H et al (2020) Abnormal trajectory detection based on geospatial consistent modeling. IEEE Access 184633–184643. https://doi.org/10.1109/access.2020.3028847
14. Zheng Li, Xiaolong Xu, Xuefei Cao, Wentao Liu, Yiwen Zhang, Dehua Chen, Haipeng Dai (2022) Integrated CNN and federated learning for COVID-19 detection on chest x-ray images. IEEE/ACM Trans Comput-Biol Bioinform https://doi.org/10.1109/TCBB.2022.3184319
15. Xu X, Liu Z, Bilal M, Vimal S, Song H (2023) Computation offloading and service caching for intelligent transportation systems with digital twin. IEEE Trans Intell Transp Syst. 23(11):20757–20772
16. Liang Yao, Xiaolong Xu* (2022) Muhammad Bilal, Huihui Wang. Dynamic edge computation offloading for internet of vehicles with deep reinforcement learning. IEEE Trans Intell Transp Syst 24(11):12991 - 12999
17. Hao Tian, Xiaolong Xu*, Tingyu Lin, Yong Cheng, Cheng Qian, Lei Ren, Muhammad Bilal (2021) DIMA: distributed cooperative microservice caching for internet of things in edge computing by deep reinforcement learning. World Wide Web J 25(5):1769–1792.
18. Xiaolong Xu, Li H, Li Z, Zhou X (2023) Safe: synergic data filtering for federated learning in cloud-edge computing. IEEE Trans Industr Inf 19(2):1655–1665
19. Mao JL, Jin CQ, Zhang ZG, Zhou AY (2017) Anomaly detection for trajectory big data: advancements and framework. Ruan Jian Xue Bao/J Softw 28(1):17–34. in Chinese. http://www.jos.org.cn/1000-9825/5151.htm
20. Lei P-R (2016) A framework for anomaly detection in maritime trajectory behavior. Knowl Inf Syst 47(1):189–214. https://doi.org/10.1007/s10115-015-0845-4
21. Wang Jet al (2020) Anomalous trajectory detection and classification based on difference and intersection set distance. IEEE Trans Vehicular Technol 2487–2500. https://doi.org/10.1109/tvt.2020.2967865
22. Zhu J, Jiang W, Liu A, Liu G, Zhao L (2015) Time-dependent popular routes based trajectory outlier detection," in Lecture Notes in Computer Science,Web Information Systems Engineering – WISE 2015 16–30. https://doi.org/10.1007/978-3-319-26190-4_2
23 San Román I, de Martín Diego I, Conde C, Cabello E (2019) Outlier trajectory detection through a context-aware distance. Pattern Anal Appl. 22(3):831–839. https://doi.org/10.1007/s10044-018-0732-1
24. Chen C et al (2023) Abnormal-trajectory detection method based on variable grid partitioning. ISPRS Int J Geo Inf 12(2):40. https://doi.org/10.3390/ijgi12020040
25. Dawen X, Shunying J, Yunsong L et al (2023) An ASM-CF model for anomalous trajectory detection with mobile trajectory big data. Phys A Stat Mechanics Appl 621:128770
26. Wang H et al (2020) Abnormal trajectory detection based on geospatial consistent modeling. IEEE Access 8:184633–184643. https://doi.org/10.1109/access.2020.3028847
27. Zhou W, Yu Y, Zhan Y, Wang C (2022) A vision-based abnormal trajectory detection framework for online traffic incident alert on freeways. Neural Comput Appl 34(17):14945–14958. https://doi.org/10.1007/s00521-022-07335-w
28. Ji Y, Wang L, Wu W, Shao H, Feng Y (2020) A method for LSTM-based trajectory modeling and abnormal trajectory detection. EEE Access 104063–104073. https://doi.org/10.1109/access.2020.2997967
29. Zheng Y, Xie X, Ma WY (2010) GeoLife: A Collaborative Social Networking Service among User, Location and Trajectory. IEEE Data Eng Bull 33(2):32–9
30. Hassan B, Zineb R, Amine L, Elhoussine L (2021) A complexity survey on density based spatial clustering of applications of noise clustering algorithms. Int J Adv Comput Sci Appl 12(2). https://doi.org/10.14569/ijacsa.2021.0120283

31. Mestria M (2018) New hybrid heuristic algorithm for the clustered traveling salesman problem. Comput Ind Eng 116:1–12. https://doi.org/10.1016/j.cie.2017.12.018
32. Fischer J, Wirtz S, Scherer V (2023) Random forest classifier and neural network for fraction identification of refuse-derived fuel images. Fuel 341:127712. https://doi.org/10.1016/j.fuel.2023.127712
33. Duchi J, Hazan E, Singer Y (2010) Adaptive subgradient methods for online learning and stochastic optimization. Conference on Learning Theory, Conference on Learning Theory
34. Tieleman T, Hinton G (2021) Lecture 6.5- rmsprop: divide the gradient by a running average of its recent magnitude. Coursera: Neural Networks for Machine Learning
35. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv: Learning, arXiv: Learning
36. Zhang R, Gong W, Grzeda V, Yaworski A, Greenspan M (2013) An adaptive learning rate method for improving adaptability of background models. IEEE Signal Process Lett 20(12):1266–1269. https://doi.org/10.1109/lsp.2013.2288579

Publisher's Note