mathematics

MDPI

Article

Flight Delay Propagation Prediction Based on Deep Learning

Jingyi Qu *, Shixing Wu and Jinjie Zhang

Tianjin Key Laboratory of Advanced Signal Processing, Civil Aviation University of China, Tianjin 300300, China
* Correspondence: jyqu@cauc.edu.cn

Abstract: The current flight delay not only affects the normal operation of the current flight, but also spreads to the downstream flights through the flights schedule, resulting in a wide range of flight delays. The analysis and prediction of flight delay propagation in advance can help civil aviation departments control the flight delay rate and reduce the economic loss caused by flight delays. Due to the small number of data samples that can constitute flight chains, it is difficult to construct flight chain data. In recent years, the analysis of the flight delay propagation problem is generally based on traditional machine learning methods with a small sample size. After obtaining a large amount of raw data from the China Air Traffic Management Bureau, we have constructed 36,287 pieces of three-level flight chain data. Based on these data, we tried to use a deep learning method to analyze and forecast flight delays. In the field of deep learning, there are CNN models and RNN models that deal with classification problems well. Based on these two classes of models, we modify and innovate the study of the problem of flight delay propagation and prediction. Firstly, the CNN-based CondenseNet algorithm is used to predict the delay level of the three-level flight chain data. Based on this, the CondenseNet network is improved by inserting CBAM modules and named CBAM-CondenseNet. The experimental results show that the improved algorithm can effectively improve the network performance, and the prediction accuracy can reach 89.8%. Compared with the traditional machine learning method, the average prediction accuracy increased by 8.7 percentage points. On the basis of the CNN model, we also considered the superiority of the LSTM (Long Short-Term Memory network) considering the processing time sequence information, and then constructed the CNN-MLSTM network and injected the SimAM module to enhance the attention of flight chain data. In the experiment of flight delay propagation prediction, the accuracy rate is 91.36%, which is a significant improvement compared to using the CNN or LSTM alone.

Keywords: flight delay propagation; deep learning; CBAM-CondenseNet; CNN-MLSTM

MSC: 68T07

Citation: Qu, J.; Wu, S.; Zhang, J. Flight Delay Propagation Prediction Based on Deep Learning. *Mathematics* **2023**, *11*, 494. https://doi.org/10.3390/math11030494

Academic Editor: Ioannis G. Tsoulos

Received: 28 November 2022
Revised: 6 January 2023
Accepted: 13 January 2023
Published: 17 January 2023

1. Introduction

As the same aircraft performs multiple flights in one day, there are close connections between the upstream and downstream flights, so delays in upstream flights may affect many other downstream flights, causing massive propagation of flight delays. It is necessary to deeply study the propagation path of flight delays in the airline network and predict the delay level of downstream flights. Therefore, it can provide a theoretical basis and data support for civil aviation departments to prevent and control delay propagation.

Many studies have been conducted by scholars on the prediction of flight delay propagation. Earlier, starting from the characteristics of delay propagation, they built flight delay propagation models to analyze the delay propagation impact [1–7]. Some researchers constructed a colored-time Petri net model for multiple flights at airports [1]. The model predicts whether flight delay will also occur at the downstream airport when the initial airports experience flight delays. Other researchers [2] proposed a flight delay propagation prediction model based on a Bayesian network. From the aspect of complex air transport

networks, the delay propagation phenomenon of hub airports in large-scale networks has been studied. Based on the queuing theory mechanism, researchers [3] proposed an analytical queuing and network decomposition model, and constructed an approximate network delay model to study the delay of 34 busy airports in the United States. The authors [4] proposed the analysis and prediction method of flight delay propagation based on complex network theory, and showed the specific classification of delay propagation. All the above methods are used to study the development of flight delay propagation by analyzing and modeling the existing historical data. Most of them use small sample data to study the causes of flight delay propagation problems. Traditional flight delay propagation analysis methods are easily influenced by model selection and subjective factors. As the volume of data in the civil aviation industry accumulates, more and more machine learning methods [8,9] are used for civil aviation delay prediction. Researchers [8] have proposed the use of machine learning to predict air traffic delay, and the use of shallow artificial neural networks to predict flight delay. Based on the problem of controllable delay in air traffic control, some researchers [9] used machine learning methods to predict the delay of individual aircrafts, taking into account the influencing factors such as weather, aircraft navigation, and control.

At present, the feature extraction ability of the CNN (Convolutional Neural Networks) is obviously more excellent than the traditional methods, and the practical application effect is remarkable [10–12]. Compared with the general CNN, CondenseNet proposed previously [10] can effectively solve the phenomenon of gradient disappearance during deep network training, with higher computational efficiency and less parameter storage. Researchers [11] have proposed a new Convolutional Block Attention Mechanism Module (CBAM) to improve the network accuracy by double feature weight calibration from spatial dimension and feature dimension. In this essay, the deep learning methods are used to study the flight delay propagation problem and prediction. The CBAM-CondenseNet algorithm is proposed by combining CondenseNet and the design ideas of CBAM. It is able to predict other flight departure delays caused by the spread of upstream flight delays.

In aviation networks, the data that make up the flight chain contain both rich spatial information and rich temporal information [13,14]. Although we use the deep learning method to build a model based on the convolutional neural network to predict the impact of flight delays, this model only extracts the spatial features of flight data and lacks the consideration of temporal information. Therefore, this paper combines the CNN network and Mogrifier LSTM (Mogrifier Long Short-Term Memory) to predict the problem of flight delay propagation. The Long Short-Term Memory (LSTM) network can remember the previous temporal information more profoundly in the time dimension, but the input x in the LSTM and the previous state h_{prev} are independent of each other before being input into the cell. The MLSTM makes these two inputs from completely independent to autonomic interaction, which greatly improves network performance. The SimAM proposed in literature [15] is different from the existing channel spatial attention module. The module deduces the 3D attention weight for the feature graph without additional parameters, which can better extract the feature of the flight chain data structure. Therefore, we fuse the attention module SimAM on the basis of CNN-MLSTM to improve the prediction accuracy. The final model is named SimAM-CNN-MLSTM.

According to the spatial and temporal characteristics of flight chains, in this paper, the method of flight delay propagation prediction based on deep learning [16–38] is proposed. This method not only uses the advantages of the CNN in spatial feature extraction, but also considers the advantages of the LSTM network in processing temporal information, and uses the attention mechanism module to enhance the feature matrix with important neurons. When the same aircraft performs multiple flight missions, it can predict the delay level of subsequent flights according to the propagation pattern of flight delays, and provide corresponding suggestions for the relevant civil aviation departments to control the delay propagation.

2. Flight Delay Propagation Prediction

The flight delay propagation prediction based on deep learning mainly includes the following parts: data preprocessing and flight chain data set construction, feature extraction, and classification prediction. Feature extraction is mainly introduced in the third and fourth parts of this paper. The following mainly introduces data preprocessing, flight chain data set construction, classification, and prediction.

2.1. Data Preprocessing

The flight data used in this project are the flight data of China from March 2018 to May 2019 provided by the Civil Aviation Administration of the China East China Regional Administration (ECRA). Among them, the key sample attributes include flight number, aircraft number, actual departure/arrival airport, flight path, planned departure/arrival time, actual departure/arrival time, planned departure/arrival airport, planned aircraft type, cruise altitude, cruise speed, military batch number, coverage type, and a total of 38 attributes. These characteristics are closely related to whether the flight is delayed or not, which not only contains important spatial features but also contains abundant time information. Since there are some abnormal values and null values in the flight data provided by the ECRA, the mainstream data analysis library Pandas is selected to clean the original flight data set. The characteristic attributes required by the model are defined as follows.

Definition 1. *Flight data F_f, including 38 characteristic attributes such as flight number, aircraft number, actual departure/arrival airport, flight path, planned departure/arrival time, actual departure/arrival time, etc.*

Definition 2. *Flight chain data F_c, within a certain time range, the same aircraft respectively performs different flight tasks from class 1 airport to class 2 airport and then to class 3 airport, and the time sequence is related. This is a flight chain. Multiple flight chain data constitute the flight chain data set.*

2.2. Construction of the Flight Chain Data Set

Flight delay has the characteristics of temporal and spatial distribution. When the same aircraft performs different flight missions in succession, it is common for subsequent flights to be delayed due to the previous flight delay. After the delay of the previous flight is passed along the flight plan step by step, it will lead to a large area of flight delays. The airport where the same aircraft takes off for the first time within a certain time range is defined as the class 1 airport. The airport where the aircraft arrives from the class 1 departure airport for flight task 1 is called the class 2 airport, also known as the class 1 arrival airport or the class 2 departure airport. By analogy, the same aircraft Z continuously performs flight tasks between multiple airports, which are connected in chronological order to form a flight chain relationship, as shown in Figure 1. Taking "Beijing-Tianjin-Shanghai" as an example, Beijing is defined as a class 1 airport. The same aircraft performs flight task 1 from Beijing to Tianjin. Tianjin is the class 2 airport in the flight chain, also known as the class 1 arrival airport or the class 2 departure airport. The plane starts from Tianjin and performs flight task 2. It flies from Tianjin to Shanghai. Shanghai is the class 3 airport in the flight chain, also known as the class 2 arrival airport or the class 3 departure airport.

Based on the above characteristic attributes, the flight chain data set is constructed. Firstly, a hub airport is selected as the class 1 airport. The airports with the number of flights from this class 1 airport are ranked from high to low, and the top 20 airports are selected as class 2 airports. Then, we directly select the airport with flights from each class 2 airport as class 3 airports. Thus, the air transport network is determined with the class 1 airport as the center and radiating outward. Secondly, taking the time and the flight tail number as key values, each flight chain is extracted from the aviation network to form a flight chain data set. Thirdly, the discrete data and continuous data in the original

data are encoded by different methods to avoid misleading the training process of the network. Lastly, the processed data are converted into a suitable characteristic matrix that feed into the network. In order to more clearly describe the flight chain data set, the *i-th* flight chain data in Definition 2 are represented by $f_i = (f_{i1}, f_{i2}, f_{i3})$, where f_{i1}, f_{i2}, and f_{i3}, respectively, represent the flight chain data f_i containing the information of three single flights that perform flight tasks before and after in the time dimension. The F_c dataset can be further represented by $F_c = \{(f_{11}, f_{12}, f_{13}), (f_{21}, f_{22}, f_{23}), \ldots, (f_{n1}, f_{n2}, f_{n3})\}$. The flight chain dataset description is shown in Figure 2.

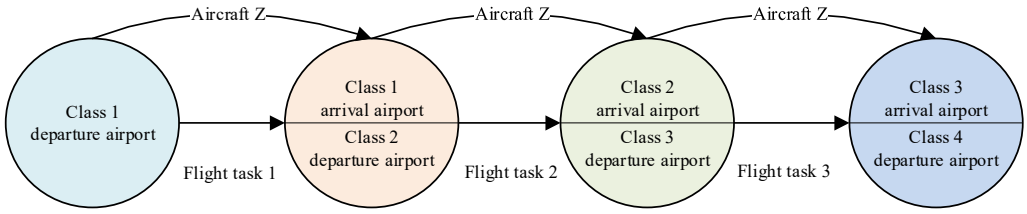


Figure 1. Flight chain model.

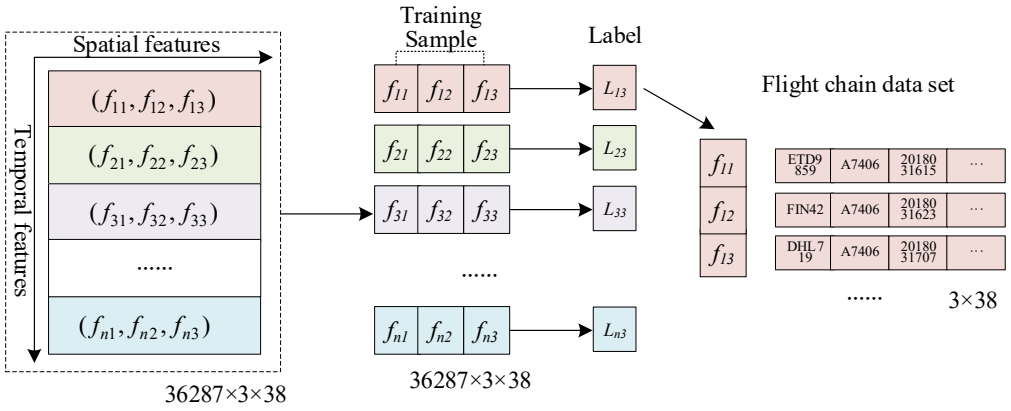


Figure 2. Flight chain data set description.

The flight data of three consecutive flights of the same aircraft within a certain time range constitute the flight chain data. There are 1,048,576 pieces of data in the original single flight. After data cleaning and construction of the flight chain data set, the data volume of the flight chain data used in the flight delay propagation prediction experiment are 36,287 pieces. The data set construction steps are as follows:

According to Definition 2, an aircraft performs continuous flight missions. In this paper, a three-class flight chain data set is formed according to the change of the same aircraft within 24 h. Firstly, select the four attributes of the aircraft number, flight execution date, class 1 arrival airport, and class 2 departure airport as the key values of data fusion; conduct the first data fusion on the cleaned flight data set; and remove the abnormal flight chain whose departure time of the secondary airport is earlier than that of the primary airport. At this time, in the flight chain data set, the aircraft performed two flight missions and turned around three airports in space.

The generation of the delayed propagation phenomenon has the characteristic of passing from one class to another, so we continue to fuse the flight chain data set for the second time. The aircraft number and flight execution date remain unchanged, and the class 2 arrival airport and class 3 departure airport are selected for the second data fusion. The abnormal flight chain whose departure time from the class 3 airport is earlier than the arrival time at the class 2 airport is removed. The flight chain data set of two consecutive flight tasks is obtained.

By analogy, the data are fused three times in this paper to form the final flight chain dataset. The aircraft in each data chain performed three consecutive flight missions, and

the spatial dimension involved the transit situation in four airports. A total of four airports including the first-class airport, second-class airport, third-class airport, and fourth-class airport are affected by flight delays. The delay label of the flight chain is the delay level of the class 3 flight mission. Most aircrafts fly one mission and do not fly another that day. As more flights are performed on the same day, the available data in the flight chain data set become smaller and smaller. Therefore, we focus on the delayed propagation of flight chains consisting of three consecutive flight missions. Finally, the characteristic attributes in the flight chain data set are divided into the numerical type and discrete type. The numerical type features are coded by Min-Max normalization, and the discrete type features are coded by CatBoost.

2.3. Classification and Prediction

Based on the relevant meaning of "flight delay" in the regulations on normal flight management, the flight delay is subdivided into five delay levels, and the number of delay levels is divided into different levels. The judging standard is shown in Table 1. Grade labels are calculated based on the difference value between the flight planned arrival time and the flight actual arrival time in the data set, and finally obtains the flight delay prediction grade with the Softmax classifier.

Table 1. Classification of flight delays.

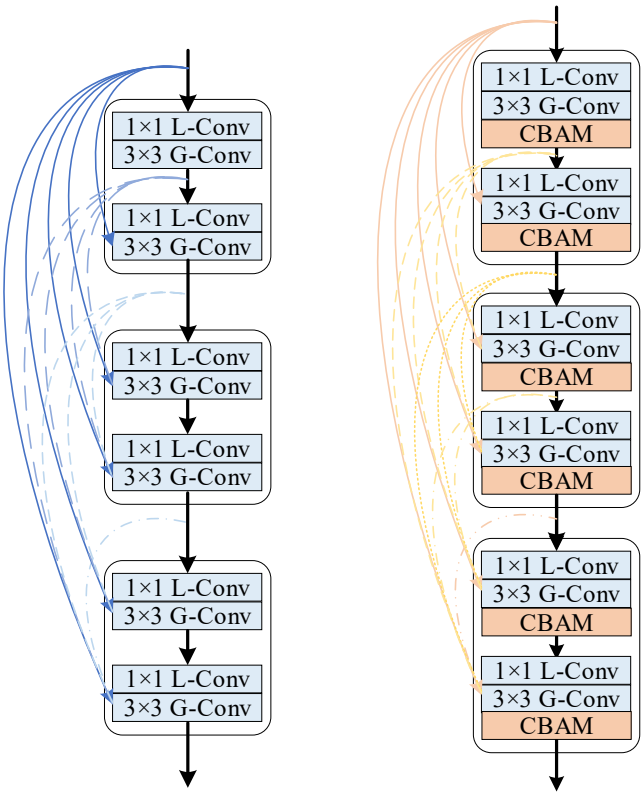
Delay Level	Delay Time T/min	Delay Level Classification
No delay	$T \leq 15$	0
Minor delay	$15 < T \leq 60$	1
Moderate delay	$60 < T \leq 120$	2
High delay	$120 < T \leq 240$	3
Significant delay	$T > 240$	4

3. CBAM-CondenseNet

CondenseNet [10] is a densely connected network based on the convolutional neural network. Based on the excellent feature extraction ability and higher computational efficiency of the CondenseNet network, we insert the CBAM [11] module on the CondenseNet network to improve the base network. The CBAM module adopts channel and spatial attention mechanisms to enhance the information transfer of the deep network structure. The CBAM-CondenseNet algorithm proposed in this paper combines the advantages of CondenseNet and CBAM. The improved CBAM-CondenseNet algorithm is used to extract features from the fused flight chain data to make it more adaptable to the task of flight delay propagation prediction. The experimental results show that the improved algorithm effectively improves the network performance.

3.1. Model Description

The traditional CondenseNet network structure is given in Figure 3a. Each network layer in each structural block is linked to all the following layers in a dense connection, and different structural blocks are also connected in a dense connection. The CBAM-CondenseNet proposed in this paper is to add a CBAM block after the convolutional layers (3×3) of each structural block, as shown in Figure 3b. After the integration of the CondenseNet and CBAM modules, the network can improve useful features and suppress useless features according to the different importance of channels and spaces, owing to which the model's ability of feature expression has been enhanced.



(a) CondenseNet (b) CBAM-CondenseNet

Figure 3. Network structure diagram. (**a**) CondenseNet network structure. (**b**) CBAM-CondenseNet network structure.

3.2. CBAM Convolution Module

CBAM mainly consists of two steps: first, the information is compressed into a channel descriptor using global max pooling and global average pooling in the spatial dimension, and the weight array of the aggregation in the compression operation is calibrated. Secondly, the importance degree between pixels is modeled based on the above operations. Two different channel descriptors are obtained by using global max pooling and global average pooling in the channel dimension, and the two channel descriptions are combined according to their channel dimension. Then, a hidden layer containing a single convolution kernel is used to carry out the convolution operation on the feature mapping to generate the final calibration of weights.

The CBAM module structure diagram is shown in Figure 4. For an input feature array $F \in R^{C \times H \times W}$ of an intermediate layer (F represents the input characteristic array of the CBAM, whose dimension is $C \times H \times W$. In general, C represents the number of channels, H represents the height, and W represents the width), the CBAM first undergoes a compression manipulation on a 1-dimensional channel. Then, the CBAM is multiplied with the feature array of inputs to obtain F'. Finally, the spatial weight array of $F' \in R^{C \times H \times W}$ is calculated by the 2-dimensional space compression operation to obtain F'', where $\odot$ indicates element-wise multiplication that the array elements in the corresponding positions are multiplied one by one.

$$F' = M_c(F) \odot F \tag{1}$$

$$F'' = M_S(F') \odot F' \tag{2}$$

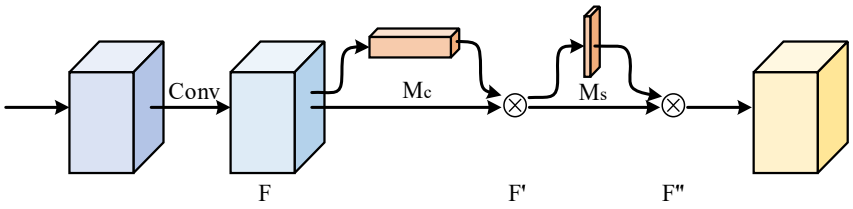


Figure 4. CBAM structure diagram.

Among them, $F \in R^{C\times H\times W}$, $F' \in R^{C\times H\times W}$, and $F'' \in R^{C\times H\times W}$ indicate the feature array of the input, the feature mapping via channel attention selection, and the feature mapping via spatial attention selection, respectively. Feature mapping $M_C \in R^{C\times 1\times 1}$ is the channel compression weight array, and $M_S \in R^{1\times H\times W}$ is the space compression weight array. $M_c(F)$ is used to represent the formula for calculating the channel attention characteristic array, and $M_s(F')$ is the formula for calculating the spatial attention characteristic array [11].

3.3. CBAM-CondenseNet Single Building Block

The computational unit structure of a single structural block in the CBAM-CondenseNet network is shown in Figure 5. Among them, X_1 is the array eigenvalue of the input of the corresponding layer, which means the nonlinear feature mapping after convolution transformation, X_2 represents the output array of the current layer, and M', H', C', M, H, and C in the figure represent three dimensions information of the array eigenvalue X_1, X_2.

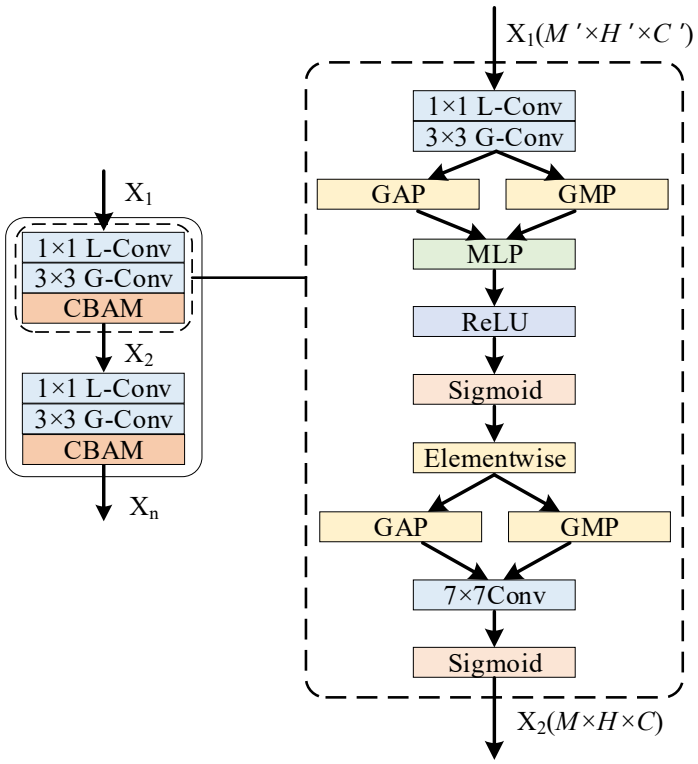


Figure 5. Network structure diagram.

In the CBAM-CondenseNet network structure, the feature mapping U after the convolution transformation of the *L-th* layer network is shown in formula (3).

$$U = W^L \otimes f(BN(W^{(L-1)} \otimes f(BN([X_0 X_1 \ldots X_{L-1}])))) \quad (3)$$

The outputs of all previous layers are first densely concatenated, and the outputs are obtained after batch normalization and activation function and then convolution opera-

tion with the weight array. Among them, $[X_0, X_1, \ldots, X_{L-1}]$ indicates that the feature mappings of all previous layers are took as input of the next layer network in the way of dense connection [10]; $W^{(L-1)}$ and W^L denote the 1×1, 3×3 convolutional weight matrix in turn; $BN(\cdot)$ represents batch normalization of the output data for each hidden layer; $f(\cdot)$ is the ReLU activation function; And $\otimes$ represents the convolution operation.

3.4. Back Propagation

The CBAM-CondenseNet model training process is mainly implemented by the Back Propagation (BP) algorithm. BP passes the error messages of the training samples back to the hidden layers to realize the continuous iterative update of the weight array between the hidden layers until the network converges. According to the BP algorithm, taking the first two network structure blocks of CBAM-CondenseNet as an example, the gradient value between each hidden layer is deduced. It is assumed that each structural block contains two groups of nonlinear transformations, and each group of transformations of the first structural block has one convolution layer and one CBAM. The second structure block has two convolution layers and one CBAM for each set of transformations, as shown in Figure 6.

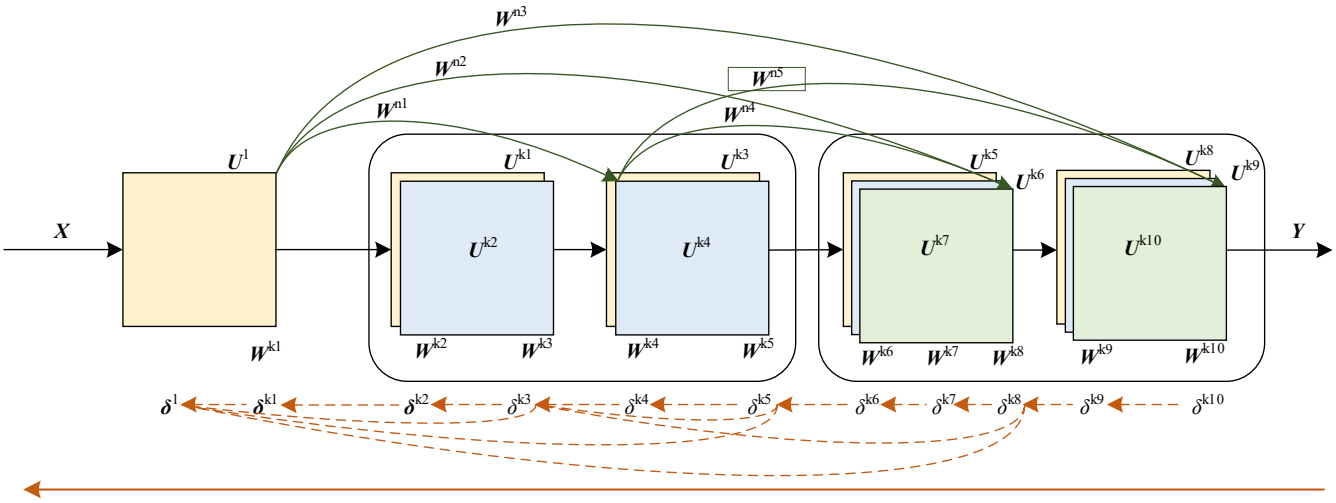


Figure 6. CBAM backpropagation.

Then, the calculation of the error term of each hidden layer in the structure block is shown in formulas (4)–(13).

$$\delta^{k10} = \partial J / \partial U^{k10} \tag{4}$$

$$\delta^{k9} = \delta^{k10} * W^{k10} \otimes f'(U^{k9}) \tag{5}$$

$$\delta^{k8} = \delta^{k9} * W^{k9} \otimes f'(U^{k8}) \tag{6}$$

$$\delta^{k7} = \delta^{k8} * W^{k8} \tag{7}$$

$$\delta^{k6} = \delta^{k7} * W^{k7} \otimes f'(U^{k6}) \tag{8}$$

$$\delta^{k5} = \delta^{k6} * W^{k6} \otimes f'(U^{k5}) \tag{9}$$

$$\delta^{k4} = \delta^{k5} * W^{k5} \tag{10}$$

$$\delta^{k3} = \delta^{k4} * W^{k4} + \delta^{k8} * W^{n5} + \delta^{k5} * W^{n4} \tag{11}$$

$$\delta^{k2} = \delta^{k3} * W^{k3} \otimes f\prime(U^{k2}) \tag{12}$$

$$\delta^{k1} = \delta^{k2} * W^{k2} \otimes f\prime(U^{k1}) \tag{13}$$

Among them, $\delta^{k1}, \delta^{k2}, \ldots, \delta^{k10}$ are the error terms corresponding to each layer in the two structural blocks, respectively. $U^{k1}, U^{k2}, \ldots, U^{k10}$ represent the output feature

mappings of each layer. W^{n5} represents the weight array between the *k3* and the *k5* layers. $\partial J/\partial U^{k10}$ represents the derivative of loss function J with respect to the last layer of network output characteristic mapping. * indicates flipping the convolution kernel in the convolution operation $\otimes$. The error terms of the remaining structural blocks can also be derived from Equations (4)–(13). The gradient value of the 1st hidden layer of the CBAM-CondenseNet network can be expressed as shown in formula (14).

$$\begin{aligned} \partial J/\partial W^1 = (&\delta^{k1} * W^{k1} + \delta^{k3} * W^{n1} + \delta^{k5} * W^{n2} + \\ &\delta^{k8} * W^{n3} + \delta^{k5} * W^{n4} + \delta^{k8} * W^{n5}) \otimes A^0 \end{aligned} \tag{14}$$

Among them, W^{n1}, W^{n2}, and W^{n3} represent the weight array between the *k3* layer, the *k5* layer, the *k8* layer, and the first layer, respectively, and A^0 represents the feature array of the input. From the above equation, the gradient information of the 1st hidden layer contains not only the gradient weights of the backpropagation of the next layer, but also the gradient information of each set of nonlinear transformations in the 1st structural block and the 2nd structural block. As a result, the gradient value of the hidden layer has been maintained in a stable range. Therefore, the CBAM-CondenseNet network can not only make efficient use of information features through dual attention mechanism strategy, but also reduce the decay of the error term in each hidden layer by its own backward conduction mechanism. It can also enhance the ability of learning and expression in deep networks, and improve the robustness of the network.

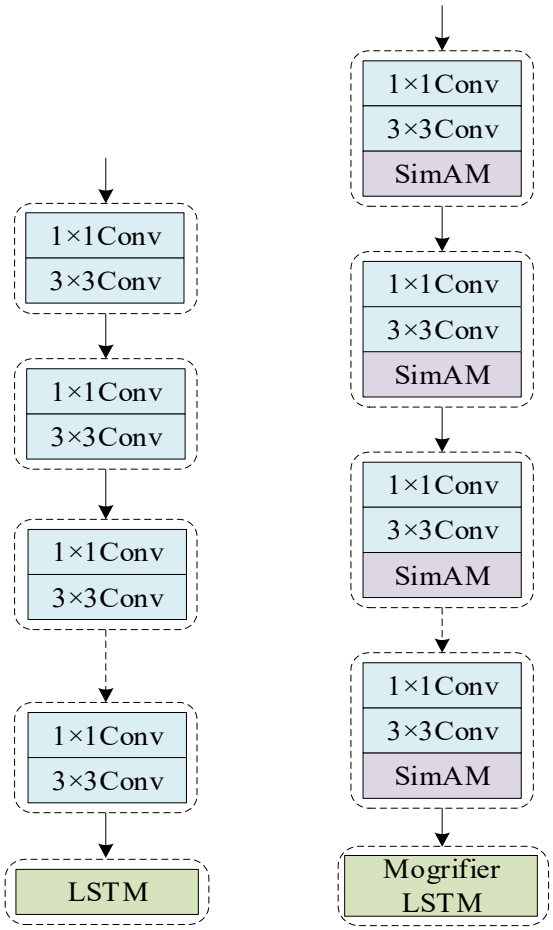
4. SimAM-CNN-MLSTM

Considering the temporal and spatial characteristics of flight delay propagation, we introduce the Mogrifier LSTM method on the basis of the CNN. The CNN has significant advantages in feature extraction, while the Mogrifier LSTM network is better at processing time sequence information. The SimAM-CNN-MLSTM integrating attention mechanism SimAM not only considers the spatial characteristics of flight tasks but also pays attention to the temporal relationship between flight chain data. The model also uses the attention mechanism module to enhance important neurons in the feature matrix. The improved model has great advantages in dealing with the task of flight delay and prediction.

4.1. Model Description

The CNN network model has achieved great success in the field of feature extraction. The key to feature extraction is the use of convolution kernel. It makes the network model have local receptive field, which can avoid the defect that the traditional feature extraction model is difficult to correlate the whole data. Therefore, the CNN convolutional layer is first used to extract the spatial features of the flight chain data set. However, the CNN has difficulty in learning the correlation between time-series data in prediction. Due to the time-series character of flight chain data, the prediction of the flight chain delay propagation problem requires the enhancement of recurrent neural network series methods. The traditional CNN-LSTM network structure diagram is shown in Figure 7a.

The key features enhanced by the attention mechanism make the prediction results more accurate. The proposed SimAM fusion convolutional layer in this paper adopts the addition of the SimAM attention module after each convolutional layer to carry out channel and space synchronization weighting for the extracted key feature information. The Mogrifier LSTM model is selected in the extraction part of time sequence features to better enable the interaction between the previous state and the input data before the cell input. The CNN-Mogrifier LSTM network model of the fusion attention mechanism SimAM proposed in this paper is shown in Figure 7b.



(a) CNN-LSTM (b) SimAM-CNN-MLSTM

Figure 7. Network structure diagram. (**a**) CNN-LSTM network structure. (**b**) SimAM-CNN-MLSTM network structure.

4.2. SimAM Attention Mechanism Module

At present, the commonly used attention mechanism CBAM usually carries out channel attention first and then spatial attention. It is impossible to pay attention to space and channel at the same time. Channel attention is shown in Figure 8a, and spatial attention is shown in Figure 8b.

However, the two types of attention in the human brain tend to work together. In neuroscience, information-rich neurons usually show different firing patterns from those of the surrounding neurons. Moreover, activating neurons usually suppresses surrounding neurons, i.e., spatial inhibition. Furthermore, neurons with spatial inhibition effects should be assigned higher importance. In order to better realize the attention, SimAM realizes an attention module with unified weights based on the neuroscience theory, and the structure diagram of SimAM is shown in Figure 8c. Taking the first feature extraction module in Figure 7b as an example, the single structural block of the CNN convolutional layer fused with SimAM is shown in Figure 9.

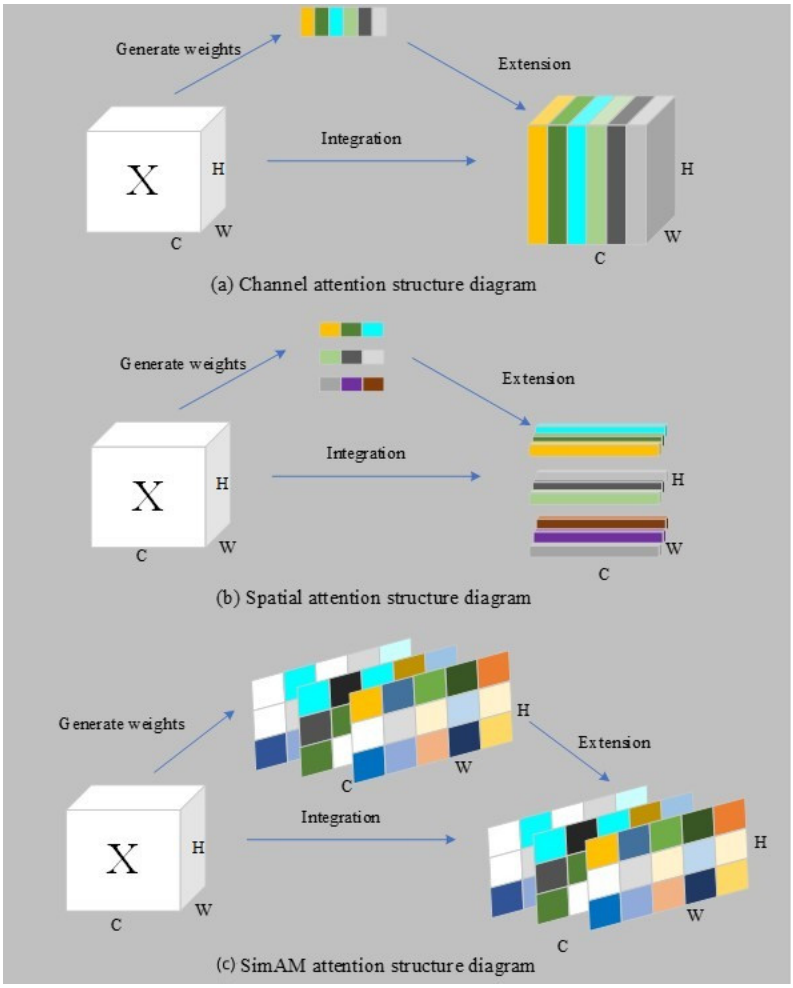


Figure 8. Attention comparison diagram. (**a**) Channel attention structure. (**b**) Spatial attention structure. (**c**) SimAM attention structure.

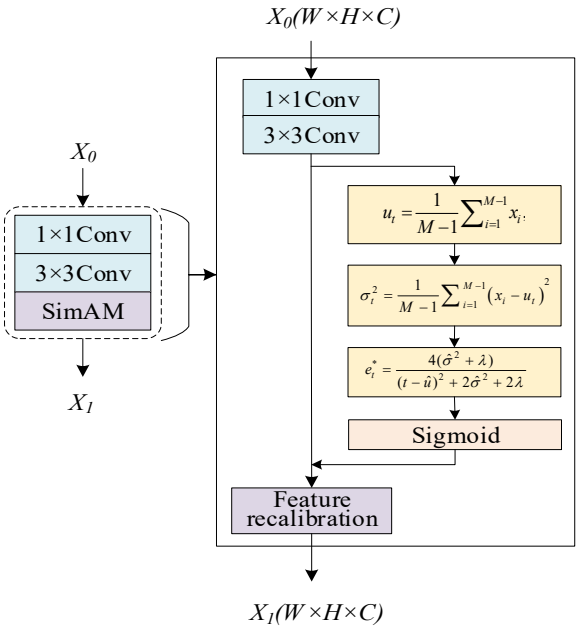


Figure 9. Integration of SimAM individual building blocks.

In order to evaluate the importance of each neuron, the easiest way is to distinguish the target neuron from other neurons. The energy function of SimAM [15] is defined as shown in formula (15).

$$e_t(w_t, b_t, y, x_i) = (y_t - \hat{t})^2 + \frac{1}{M-1}\sum_{i=1}^{M-1}(y_o - \hat{x}_i)^2 \tag{15}$$

where $\hat{t}$ and x_i represent the target neurons and other neurons of the input three-dimensional eigenvector x, and $\hat{t} = w_t t + b_t$, $\hat{x}_i = w_t x_i + b_t$, i represent the index in the spatial dimension, M denotes the number of all neurons in a channel, w_t and b_t, respectively, refer to the weight and paranoia of neurons during transformation. To facilitate the computation of the minimization of the energy formula, the binary label method is adopted. Let $y_t = 1$, $y_0 = -1$, and add regular items. λ is a super parameter. The energy function is shown in Equation (16).

$$e_t(w_t, b_t, y, x_i) = \frac{1}{M-1}\sum_{i=1}^{M-1}(-1 - (w_t x_i + b_t))^2 + (1 - (w_t t + b_t))^2 + \lambda w_t^2 \tag{16}$$

Each channel has $M = H \times W$ energy functions. We then find the analytical solution to formula (16). w_t and b_t are represented by formulas (17) and (18), respectively.

$$w_t = -\frac{2(t - u_t)}{(t - u_t)^2 + 2\sigma_t^2 + 2\lambda} \tag{17}$$

$$b_t = -\frac{1}{2}(t + u_t)w_t \tag{18}$$

where $u_t = \frac{1}{M-1}\sum_{i=1}^{M-1} x_i$, $\sigma_t^2 = \frac{1}{M-1}\sum_{i=1}^{M-1}(x_i - u_t)^2$. Therefore, the energy formula is simplified to Equation (19).

$$e_t^* = \frac{4(\hat{\sigma}^2 + \lambda)}{(t - \hat{u})^2 + 2\hat{\sigma}^2 + 2\lambda} \tag{19}$$

Equation (19) indicates that the greater the distinction between t neurons and peripheral neurons, the higher the importance, and the neuron importance can be calculated through $1/e^*$. Finally, after judging the importance of neurons according to formula (19), the feature matrix is enhanced according to the definition of the attention mechanism, as shown in formula (20).

$$X = \text{sigmoid}(\frac{1}{E}) \otimes X \tag{20}$$

4.3. Mogrifier LSTM Module

Since the increasing use of deep learning, the LSTM has been widely used in various time-series related tasks. The LSTM is a kind of RNN. The LSTM can relieve the gradient disappearance and information forgetting issues. However, in the LSTM, the current input is independent of the previous hidden layer state h_{prev}, and they only interact in the gate. The lack of previous interaction may lead to missing context information. The Mogrifier LSTM [14] allows the input and state to interact first without changing the structure of the LSTM itself, hoping to enhance the context modeling ability.

The unit structure of the Mogrifier LSTM is shown in Figure 10. The main approach of the Mogrifier LSTM is to alternately let x and h_{prev} interact for QR decomposition before ordinary LSTM computation. Let the input x and state h_{prev} first conduct multiple rounds of interaction, and then send them into the LSTM to participate in calculation. This simple modification achieves remarkable results, and its formula is shown in (7).

$$Mogrify(x, C_{prev}, h_{prev}) = Lstm\left(x^{\uparrow}, C_{prev}, h_{prev}^{\uparrow}\right) \tag{21}$$

where C_{prev} represents the previous Mogrifier LSTM unit cell state and h_{prev} represents the hidden layer state. $x^{\uparrow}$ and $h^{\uparrow}_{prev}$ are defined as the value with the largest superscript in x^i and h^i_{prev}, as shown in formulas (22) and (23). The number of alternating rounds r in the formula is a hyperparameter; if $r = 0$, then this is an ordinary LSTM.

$$x^i = 2\sigma\left(Q^i\, h^{i-1}_{prev}\right) \odot x^{i-2}, \text{odd } i \in [1 \ldots r] \tag{22}$$

$$h^i_{prev} = 2\sigma\left(R^i\, x^{i-1}\right) \odot h^{i-2}_{prev}, \text{even } i \in [1 \ldots r] \tag{23}$$

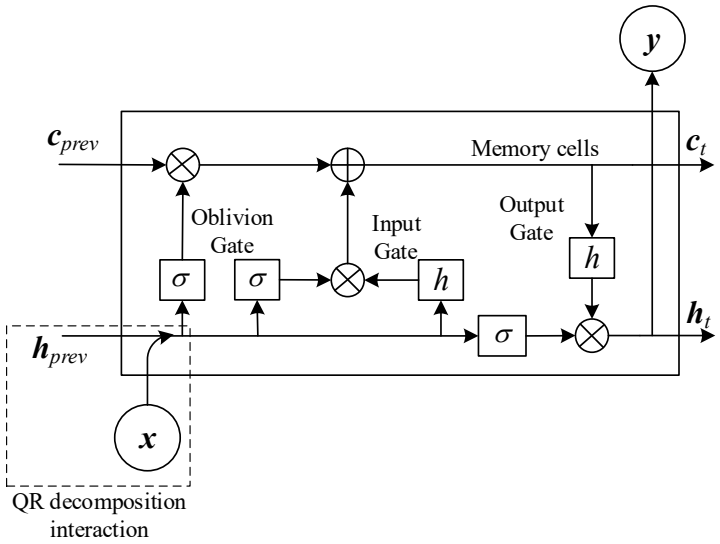


Figure 10. The Mogrifier LSTM cell structure.

The SimAM-CNN-Mogrifier LSTM model in this paper integrating the attention mechanism is compared and tested, and the number of alternating rounds $r = 6$ has the best effect. When $r = 6$, the interaction process of the Mogrifier LSTM is further shown in Figure 11.

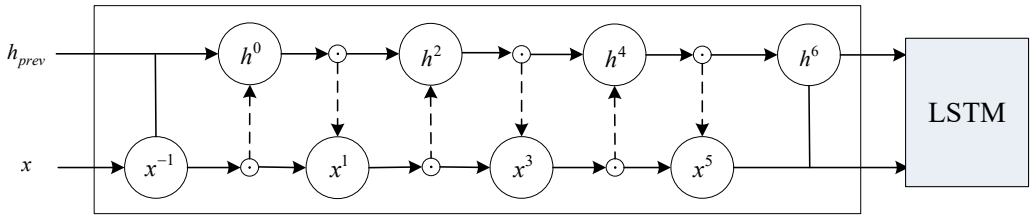


Figure 11. The Mogrifier LSTM interaction process.

The interaction between x and h_{prev} in this paper before entering the LSTM network is shown in formulas (24)–(29), where $x^{-1} = x$, $h^0_{prev} = h_{prev}$.

$$h^6_{prev} = 2\sigma\left(R^6 x^5\right) \odot h^4_{prev} \tag{24}$$

$$x^5 = 2\sigma\left(Q^5 h^4\right) \odot x^3 \tag{25}$$

$$h^4_{prev} = 2\sigma\left(R^4 x^3\right) \odot h^2_{prev} \tag{26}$$

$$x^3 = 2\sigma\left(Q^3 h^2\right) \odot x^1 \tag{27}$$

$$h^2_{prev} = 2\sigma\left(R^2 x^1\right) \odot h^0_{prev} \tag{28}$$

$$x^1 = 2\sigma\left(Q^1 h^0\right) \odot x^{-1} \tag{29}$$

The final result after the interaction is input into LSTM cells, and its unit structure formula is expressed as formulas (30)–(35).

$$f = \sigma\left(W^{fx} x^5 + W^{fh} \mathrm{h}^6_{prev} + b^f\right) \tag{30}$$

$$i = \sigma\left(W^{ix} x^5 + W^{ih} \mathrm{h}^6_{prev} + b^i\right) \tag{31}$$

$$j = \tanh\left(W^{jx} x^5 + W^{jh} \mathrm{h}^6_{prev} + b^j\right) \tag{32}$$

$$o = \sigma\left(W^{ox} x^5 + W^{oh} \mathrm{h}^6_{prev} + b^o\right) \tag{33}$$

$$c = f \odot c_{prev} + i \odot j \tag{34}$$

$$h = o \odot \tanh(c) \tag{35}$$

where f is the forget gate, it is set to manage how much the previous memory cell C_{prev} retains. i is the input gate used to manage how much the current information should be input. O is the output gate to manage how much the current memory cell should output.

4.4. Backpropagation

The backpropagation process of the network is iterated layer by layer through a gradient descent algorithm, and parameters are updated according to the error term until the network converges. The CNN-MLSTM fused with the attention mechanism SimAM is divided into the Mogrifier LSTM module and the CNN module fused with SimAM in the process of backpropagation. The backpropagation process is shown in Figure 12.

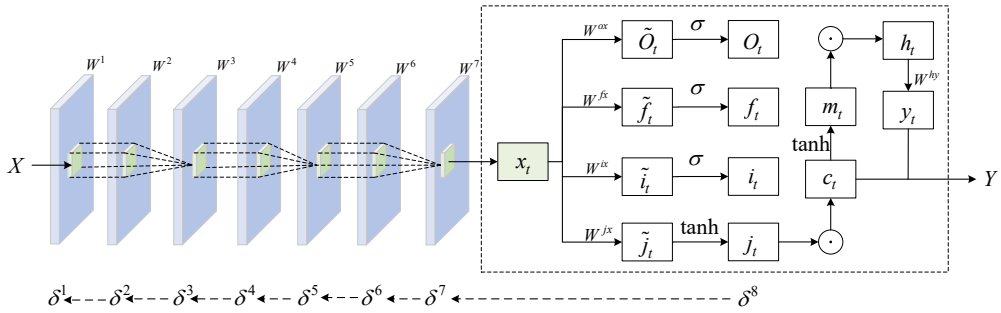


Figure 12. Network reverse propagation map.

According to the gradient descent algorithm, the error term in the backpropagation process of the Mogrifier LSTM network is first deduced. According to the principle of error backpropagation, it is derived that the backpropagation process of the error along the number of network layers is shown in formula (36).

$$\delta_t^8 = \delta_{i,t}^l W^{ix} + \delta_{f,t}^l W^{fx} + \delta_{\underset{\sim}{j},t}^l W^{\widetilde{jx}} + \delta_{o,t}^l W^{ox} \tag{36}$$

where $\delta_{i,t}^l, \delta_{f,t}^l, \delta_{\underset{\sim}{j},t}^l, \delta_{o,t}^l$, respectively, represent the error terms of each gate in each memory cell as shown in Equations (37)–(40). It represents the weight matrix of the input gate, forget gate, and output gate in turn.

$$\delta_{i,t}^l = \delta_t^l o_t^l f\left(j_t^l\right) \widetilde{j}_t^l i_t^l \left(1 - i_t^l\right) \tag{37}$$

$$\delta_{f,t}^l = \delta_t^l o_t^l f\left(j_t^l\right) j_{t-1}^l f_t^l \left(1 - f_t^l\right) \tag{38}$$

$$\delta_{\widetilde{j},t}^{l} = \delta_t^l o_t^l f\left(j_t^l\right) i_t^l \left(1 - \left(\widetilde{j}_t^l\right)^2\right) \tag{39}$$

$$\delta_{o,t}^{l} = \delta_t^l f\left(c_t^l\right) o_t^l \left(1 - o_t^l\right) \tag{40}$$

where $f(\cdot)$ is the activation function and $f(\cdot)$ is the derivative of the activation function. The superscript l represents the current layer, and the subscript t represents the current moment. The error calculation of each hidden layer in the convolution module is shown in Equations (41)–(47).

$$\delta^7 = \delta_t^8 * f\left(U^6\right) \tag{41}$$

$$\delta^6 = \delta^7 * W^6 \otimes f\left(U^5\right) \tag{42}$$

$$\delta^5 = \delta^6 * W^5 \otimes f\left(U^4\right) \tag{43}$$

$$\delta^4 = \delta^5 * W^4 \otimes f\left(U^3\right) \tag{44}$$

$$\delta^3 = \delta^4 * W^3 \otimes f\left(U^2\right) \tag{45}$$

$$\delta^2 = \delta^3 * W^2 \otimes f\left(U^1\right) \tag{46}$$

$$\delta^1 = \delta^2 * W^1 \otimes f\left(U^1\right) \tag{47}$$

Among them, δ^1, $\delta^2 \ldots \delta^7$ represent the error term of the corresponding layer, respectively; U^1, U^2, $\ldots U^6$ is the output feature mapping of each layer; and W represents the mapping matrix between each layer.

5. The Discussion about Simulation Results

In this paper, the improved CBAM-CondenseNet network based on the CNN, and the improved SimAM-CNN-MLSTM network, are constructed respectively. The prediction experiments are conducted separately for the departure flight delay levels of class 3 airports affected by the propagation of preceding flight delay. The experiment shows the result of predicting the departure delay level of class 3 airports affected by the departure delay of class 1 and class 2 airports. Therefore, flight delay levels of class 3 airports are used as labels for training on the training set, and flight delay level predictions are performed on the test set. Next, the data, experimental environment, and parameter configuration are introduced, and then, the performance of the model before and after improvement is compared using various indicators.

5.1. Dataset Description

The dataset used for the experiments in this paper is the national flight data from March 2018 to May 2019 provided by the ECRA. Flight chain data are formed by the flight data of an aircraft flying three times in a certain time range. The original flight data used in the experiments are 1,048,576 items. In the original data, five levels of delay are divided according to the delay time, among which the proportion of no delay, minor delay, moderate delay, high delay, and significant delay is 65:20:8:4:3. After data cleaning and flight chain dataset construction, the final flight chain dataset used in the flight delay propagation includes 36,287 data items. Subsequent experiments are conducted and verified on this flight chain dataset. The training set and validation set are divided in a ratio of 5:1. The experimental environment is a Dell PoweredgeR370 rackmount server with 16G video memory, double Intel XeonE5-2630 CPUs with 2.20 GHz CPU frequency, and NVIDIA P100 GPU accelerated graphics card. The model is run on the Pytorch deep learning framework built on the Ubuntu 16.04 operating system.

5.2. Parameters Selection

After several experiments and parameter adjustments, the parameter selection information for the CBAM-CondenseNet network and the SimAM-CNN-MLSTM network is shown in Table 2. The CBAM-CondenseNet network is initialized with weight orthogonality. The optimizer introduces the Momentum's Stochastic Gradient Descent method, where the momentum factor is set to 0.9. The coalescence factor in the grouped convolution is set to 4. The learning rate is set to 0.1, and the learning rate is adjusted by cosine annealing. The number of batches during training is 128, and the maximum number of iterations is 69,000.

Table 2. Experimental environment parameters.

Parameter Name	CBAM-CondenseNet	SimAM-CNN-MLSTM
Loss function	Cross entropy loss function	Cross entropy loss function
Learning rate	0.1	0.001
Optimizer	SGD	Adam
Regular term λ	-	1×10^{-5}
Alternate rounds r	-	6
Dropout	0	0.2
Number of training rounds	100	100

The CNN-MLSTM fused with SimAM uses four-layer convolution in the convolution layer. The convolution layer is configured as (3×3, 64), indicating that the size of the convolution kernel is set to 3×3, the number of convolution filters is set to 64, and the step size is the default value of 1. At the same time, padding 0 is performed on the boundary to ensure that the output size does not change after the input of the convolution layer. The pooling layer is averaged pooling with a pooling dimension of 2×2, and the step size is also set to 1. Then, the next layer is the Mogrifier LSTM network with hidden layer dimension 256.

5.3. Evaluation Metrics

The experiment in this paper is a typical multi-classification task. The main common evaluation metrics are accuracy, precision, recall, and F1 value. Each evaluation metric is explained below using a confusion matrix, as shown in Figure 13.

Confusion Matrix		True Value	
		Positive	Negative
Predicted Value	Positive	TP	FP
	Negative	FN	TN

Figure 13. Confusion matrix.

TP: True Positive, indicates that the prediction value is a positive case and the true value is a positive case.

TN: True Negative, indicates that the prediction value is a negative case and the true value is a negative case.

FP: False Positive, indicates that the prediction value is a positive case and the true value is a negative case.

FN: False Negative, indicates that the prediction value is a negative case and the true value is a positive case.

The accuracy rate represents the proportion of the number of samples with correct classification to the total number of samples, which reflects the overall performance of

the model. However, when the number of positive samples and negative samples are extremely unbalanced, the accuracy rate is not a good evaluation of the model's performance. The calculation formula is shown in Equation (48)

$$Accuracy = \frac{n_{correct}}{N} \tag{48}$$

The precision rate is the proportion of true positive classes among all results predicted to be positive classes. The accuracy rate is calculated as shown in Equation (49).

$$P = \frac{TP}{TP + FP} \tag{49}$$

Recall indicates the proportion of the number of correctly predicted positive samples to the total number of true positive samples. The recall's calculation method is shown as follows:

$$R = \frac{TP}{TP + FN} \tag{50}$$

The F1 value is the result of a combination of considerations, and is the summed average of the precision and recall rates. Since P and R can easily go up and down, the numerator of F1 is P * R. This makes that blindly increasing either P or R will not improve the F1 index. F1 will only be high when both are high. Equation (51) following the second equal sign also suggests that the F1 index is designed to lower both FP and FN (false positives and false negatives).

$$F1 = \frac{2 * P * R}{P + R} = \frac{TP}{TP + (FP + FN)/2} \tag{51}$$

5.4. Analysis of Experimental Results

Before training the neural network, the data set needs to be divided into two categories: the training set and test set. The training set data are used for neural network learning. The generalization ability of the trained model is examined by using the test set. Table 3 shows the loss values, accuracy, precision, recall, and F1 value of the CBAM-CondenseNet and SimAM-CNN-MLSTM models in the data set.

Table 3. Experimental indicators.

Indicators	CBAM-CondenseNet	SimAM-CNN-MLSTM
Loss value	0.3	0.2
Accuracy	0.898	0.9136
Precision	0.913	0.825
Recall	0.892	0.874
F1	0.904	0.849

The performance of model prediction is measured by the loss value of the training set. The smaller the loss value, the more the model converges. It also means that the prediction result is closer to the true value and the robustness of the model is better. With the same training dataset, the faster the model converges, the better the learning ability of the model is. Figure 14 shows the changes of the loss values of the CBAM-CondenseNet and SimAM-CNN-MLSTM with the number of iterations. The SimAM-CNN-MLSTM model converges after 20 rounds of training, and the CBAM-CondenseNet model converges after 40 rounds of training. The SimAM-CNN-MLSTM network performs better in terms of the model learning capability.

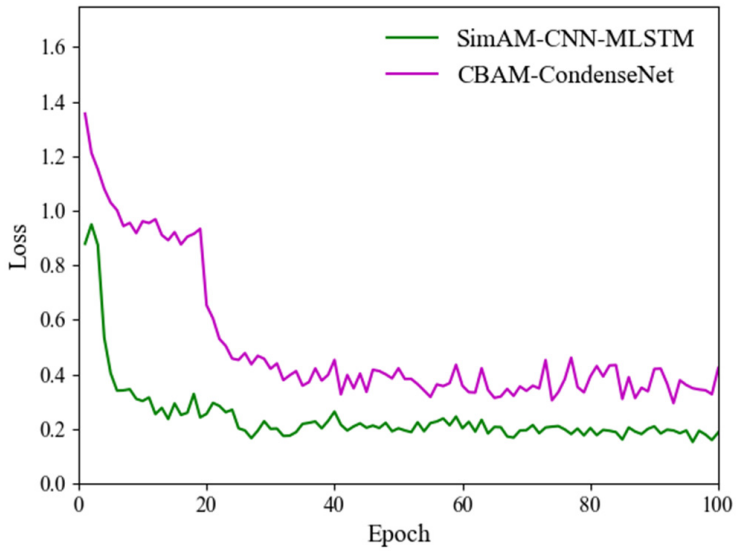


Figure 14. Variations of loss values of the CBAM-CondenseNet and SimAM-CNN-MLSTM with the number of iterations.

The accuracy rate represents the overall performance of the model and evaluates the generalization capability of the model classification. The CBAM-CondenseNet is able to achieve an accuracy of 89.8%, while the SimAM-CNN-MLSTM performs even better, with an accuracy that is 1.56 percentage points higher than that of the CBAM-CondenseNet network. The flight delay prediction in this paper is a typical multiple classification task. However, the number of flights of each delay level in the flight data set is not evenly distributed. Therefore, accuracy is not the only indicator of the overall performance of the model. Precision represents the degree of accuracy in predicting the positive sample results. Recall indicates how many of the positive cases in the sample are predicted correctly. Precision and recall reflect the assessment of the classification ability for each category in a multiclassification problem. The F1 value is the summed average of the precision and recall. From Table 3, the SimAM-CNN-MLSTM outperforms CBAM-CondenseNet in the evaluation of model accuracy, but the CBAM-CondenseNet network performs better in terms of precision and recall, which represented classification capabilities of each category. This means that CBAM-CondenseNet has a more balanced prediction performance in multiple categories.

5.5. Effect of Network Layers on CBAM-CondenseNet

In order to compare the effect of model improvement before and after, the influence of different number of network layers on the accuracy of the model is explored. Table 4 shows the accuracy of the improved CBAM-CondenseNet and CondenseNet models with different number of layers on the dataset. The experimental results show that the improved CBAM-CondenseNet model has a higher model accuracy than the CondenseNet model for the same number of layers. When the network reaches 44 layers, the accuracy rate is 89.81%. To further verify the stability of the improved model and the trainable depth, the network is tested with deeper training. When the number of layers of the network is 70, 102, and 126, the experimental results show that the CBAM-CondenseNet network could maintain good stability, and the accuracy rate is maintained stable at about 89.8% as the network deepened.

Table 4. Comparison of classification accuracy (%).

Number of Network Layers	CondenseNet	CBAM-CondenseNet
18	83.13	87.52
28	85.72	89.36
36	86.66	89.75
44	86.68	89.81
70	85.56	89.82
102	86.66	89.82
126	86.65	89.82

To verify that the CBAM-CondenseNet model is more advantageous in terms of data processing and prediction accuracy, the accuracy of the CBAM-CondenseNet and CondenseNet algorithms with different layers is analyzed and compared with the algorithm model proposed previously [39] on the flights chain dataset, as shown in Table 5. When the network has 18, 36, and 44 layers, the accuracy of the CondenseNet algorithm is higher than that of the DenseNet, SE-DenseNet, and CBAM-CondenseNet algorithm models. When the network reaches 44 layers, the CBAM-CondenseNet model is 4.25% more accurate than the CondenseNet model. It indicates that the CBAM-CondenseNet network has better performance and higher classification accuracy.

Table 5. Comparison of classification accuracy of different models (%).

Number of Network Layers	DenseNet	SE-DenseNet	CondenseNet	CBAM-CondenseNet
18	80.81	82.03	83.93	89.19
36	81.28	82.37	86.52	91.31
44	82.57	83.28	87.11	91.36

5.6. Effect of Alternate Rounds on SimAM-CNN-MLSTM

The number of alternating rounds r value is an important hyperparameter in the Mogrifier LSTM network. The larger the r value, the more fully the input x of the LSTM network interacts with the state of the previous cells, and the better the network can explore the correlation between the temporal information. However, when r value increases by 1, each update of the LSTM cell state requires one more QR matrix decomposition, and the amount of network calculation and training time will greatly increase. Due to the limited arithmetic power of the experimental hardware facilities, the experimental r is set to 6 at most. Table 6 shows the experimental results of the prediction accuracy of flight delay propagation and the training time per round when the number of alternating rounds r increases. The experiments in this paper are not very demanding in terms of time consumption, and priority is given to the impact on accuracy, so the r value is set to 6.

Table 6. Comparison of alternating rounds r accuracy and training time.

Alternate Rounds	Accuracy	Training Time per Round
1	88.45%	26.22 s
2	88.91%	28.41 s
3	90.42%	32.47 s
4	90.28%	40.86 s
5	90.97%	56.53 s
6	91.31%	71.34 s

5.7. Comparison of Model Complexity

Space complexity and time complexity are two important metrics to indicate the complexity of an algorithm. Space complexity is used to calculate the degree of resource consumption. The model parameters are measured by Params, and the more complex the

algorithm is, the more parameters are involved. The time complexity is measured by the number of floating-point operations (FLOPs), and the higher the complexity of the model, the longer the model training and prediction time will be. To test the performance of the improved model, Table 7 shows the complexity of the model in this paper compared with several other models, where Mogrifier LSTM is abbreviated as MLSTM.

Table 7. Comparison of model complexity of different network models.

Model	FLOPs (M)	Params (M)
CNN	3.23	0.32
LSTM	1.56	0.64
MLSTM	1.57	0.68
CNN-LSTM	4.80	0.98
CNN-MLSTM	4.82	1.05
SimAM-CNN-MLSTM	4.82	1.05
CondenseNet	20.55	1.39
CBAM-CondenseNet	20.77	1.46

The comparison shows that, with the same amount of data input, the computation amount of the CBAM-CondenseNet network increased slightly compared with the model before the improvement. The growth of the improved network model parameters is not significant compared to the improved network model. Therefore, the embedding of CBAM brings about a negligible growth in the overall parameters and computation of the model, and the algorithm complexity is basically the same as before the improvement.

As can be seen in Table 7, the MLSTM does not have a great increase in complexity compared to the LSTM algorithm. The parametric increase basically does not change after incorporating the attention mechanism SimAM module in the CNN-MLSTM, which further verifies the parametric-free property of the SimAM module.

5.8. Comparison with Traditional Models

To test the performance of the CBAM-CondenseNet network and SimAM-CNN-MLSTM network, this section uses CNN, LSTM, MLSTM, CNN-LSTM, CNN-MLSTM, and CBAM-CondenseNet network models on the flight chain dataset to conduct accuracy and loss value comparison experiments. For the same flight chain data set, the comparative experimental results of different models are shown in Figures 15 and 16. Figure 15 is the comparison of loss values, and Figure 16 is the comparison of accuracy.

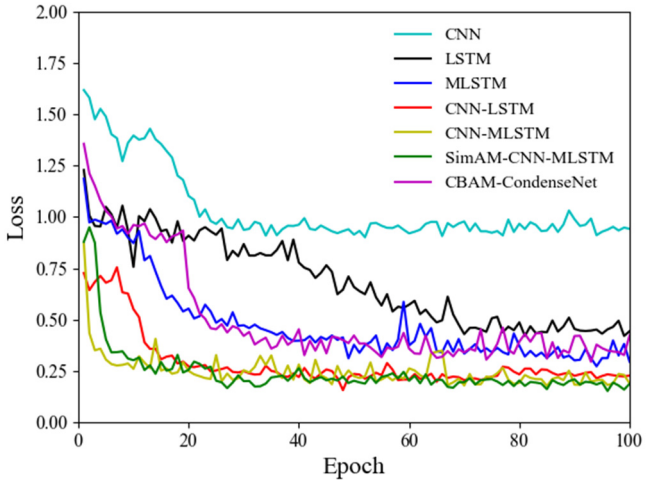


Figure 15. Comparison of the loss values of different network models.

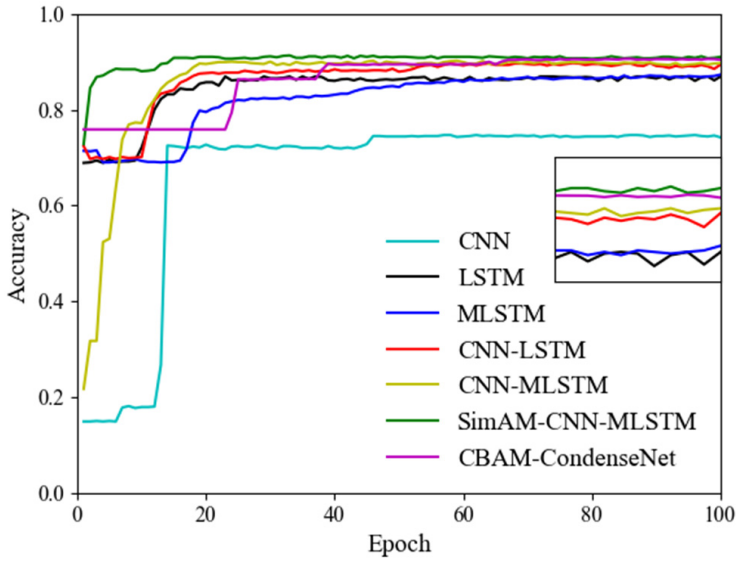


Figure 16. Comparison of the accuracy of different network models.

From the experimental results, we can see that the CNN-MLSTM network with the addition of the convolutional idea has a significant improvement in the accuracy on the flight chain dataset compared to the CNN network alone or the LSTM network, with an improvement of 16.46% and 3.16%, respectively. In this paper, the addition of the SimAM attention mechanism module to CNN-MLSTM improves 1.56% to 91.36% on the flight chain dataset, which is significantly higher than the accuracy of other networks and has the lowest loss function value.

It is comprehensively shown that when performing flight delay propagation prediction, the SimAM-CNN-MLSTM network with fused attention mechanism proposed in this paper predicts the closest flight delay propagation classification results to the actual ones, and the network performance is the best in terms of accuracy prediction.

In order to further verify that the accuracy prediction of flight delay propagation based on big data using the deep learning methods is greatly enhanced compared with the traditional algorithms, this experiment uses several different flight delay propagation prediction models [8,40,41] to compare with the models in this paper on the same flight chain data set. The experimental results are shown in Table 8. The experimental results demonstrate that the deep learning models have better performance in handling the task of flight delay propagation prediction compared with the traditional models. In particular, the improved CBAM-CondenseNet network and SimAM-CNN-MLSTM network achieve the best prediction performance.

Table 8. Comparison of accuracy of traditional models.

Network Model	Accuracy (%)
C4.5 Decision tree	78.05
Support vector machine	80.00
ATD Bayesian network	80.00
Artificial Neural Network	86.30
CBAM-CondenseNet	89.80
SimAM-CNN-MLSTM	91.36

6. Conclusions

In this paper, the chain spread model of flight delay propagation is established by analyzing the characteristics of chain spread. Two models of flight delay propagation prediction based on deep learning methods are presented. Many experiments have verified the effectiveness of the models, and the conclusions are as follows:

(1) Based on the study of flight delay propagation characteristics, a chain model of flight delay propagation effect is established. It can predict the delay level of subsequent departing flights affected by the delay of previous departing flights.

(2) The improved CBAM-CondenseNet overcomes the problem of gradient disappearance in deep network. It also combines spatial and channel attention mechanisms to achieve adaptive weight calibration. After several experiments, the prediction accuracy of the improved network is improved.

(3) According to the space-time characteristics of the flight chain data set, a SimAM-CNN-MLSTM network model integrating attention mechanisms is proposed. The CNN network layer is used to extract spatial information for the first time. Then, the important features are enhanced by the simultaneous attention of space and channel through the SimAM attention mechanism module. Finally, the Mogrifier LSTM is input to further extract the temporal characteristics in flight delay propagation, which effectively improves the accuracy of flight delay propagation prediction.

In summary, two methods of flight delay propagation prediction based on deep learning are presented to realize the prediction of flight delay propagation. In the next stage, we will consider how to use regression models to make predictions on the specific duration of flight delays, and also add more influencing factors to the analysis of flight delay propagation when the data allows.

Author Contributions: Methodology, J.Q.; validation, S.W., J.Z.; investigation, J.Q.; writing—original draft preparation, S.W., J.Z.; writing—review and editing, S.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Scientific Research Project of the Tianjin Educational Committee, grant number XJ2022000301; the National Natural Science Foundation of China, grant number U1833105; and the Fundamental Research Funds for the Central Universities, grant number 3122019185.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ding, J.L.; Chen, T.T.; Liu, Y.J. Colored-timed Petri nets model of flight delays and propagated analysis. *Comput. Integr. Manuf. Syst.* **2008**, *14*, 2334–2340.
2. Liu, Y.; Pilian, H.E.; Liu, C.; Cao, W. Flight delay propagation research based on Bayesian net. *Comput. Eng. Appl.* **2008**, *44*, 242–245.
3. Pyrgiotis, N.; Malone, K.M.; Odoni, A. Modelling delay propagation within an airport network. *Transp. Res. Pt. C Emerg. Technol.* **2013**, *27C*, 60–75. [CrossRef]
4. Shao, Q.; Zhu, Y.; Jia, M.; Zhang, H.J. Analysis of flight delay propagation based on complex network theory. *Aeronaut. Comput. Techn.* **2015**, *45*, 24–28.
5. Campanelli, B.; Fleurquin, P.; Arranz, A.; Etxebarria, I.; Ciruelos, C.; Eguíluz, V.M.; Ramasco, J.J. Comparing the modeling of delay propagation in the US and European air traffic networks. *J. Air Transp. Manag.* **2016**, *56*, 12–18. [CrossRef]
6. Wu, W.; Wu, C.L. Enhanced delay propagation tree model with Bayesian Network for modelling flight delay propagation. *Transp. Plan. Technol.* **2018**, *41*, 319–335. [CrossRef]
7. Baspinar, B.; Ure, N.K.; Koyuncu, E.; Inalhan, G. Analysis of delay characteristics of European air traffic through a data-driven airport-centric queuing network model. *IFAC-PapersOnLine* **2016**, *49*, 359–364. [CrossRef]
8. Khanmohammadi, S.; Tutun, S.; Kucuk, Y. A new multilevel input layer artificial neural network for predicting flight delays at JFK airport. *Procedia Comput. Sci.* **2016**, *95*, 237–244. [CrossRef]

9. Takeichi, N. Prediction of delay due to air traffic control by machine learning. In Proceedings of the AIAA Modeling and Simulation Technologies Conference, Grapevine, TX, USA, 9–13 January 2017; pp. 191–199.
10. Huang, G.; Liu, S.; Laurens, V.; Weinberger, K.Q. CondenseNet: An efficient DenseNet using learned group convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 2752–2761.
11. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
12. Rebollo, J.J.; Balakrishnan, H. Characterization and prediction of air traffic delays. *Transp. Res. Pt. C Emerg. Technol.* **2014**, *44*, 234–241. [CrossRef]
13. Graves, A. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780.
14. Melis, G.; Koisk, T.; Blunsom, P. Mogrifier LSTM. *arXiv* **2019**, arXiv:1909.01792.
15. Yang, L.; Zhang, R.Y.; Li, L.; Xie, X. SimAM: A simple, parameter-free attention module for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, PMLR, Online, 18–24 July 2021; pp. 125–137.
16. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
17. Wan, R.; Mei, S.; Wang, J.; Liu, M.; Yang, F. Multivariate temporal convolutional network: A deep neural networks approach for multivariate time series forecasting. *Electronics* **2019**, *8*, 876. [CrossRef]
18. Bai, S.; Kolter, J.Z.; Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv* **2018**, arXiv:1803.01271.
19. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
20. Jie, H.; Li, S.; Gang, S.; Albanie, S. Squeeze-and-Excitation Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; p. 99.
21. Arikan, M.; Deshpande, V.; Sohoni, M.G. Building reliable air-travel infrastructure using empirical data and stochastic models of airline networks. *Oper. Res.* **2013**, *61*, 45–64. [CrossRef]
22. Sha, M.Y.; Chi, H.; Gao, M.G. Estimation of flight delays and propagation under the airport capacity constraints. *Math. Pract. Theory.* **2019**, *49*, 96–105.
23. Xu, B.G.; Liu, Q.Q.; Gao, M.G. The flight delay propagation analysis based on airport busy state. *Chin. J. Manag. Sci.* **2019**, *27*, 87–95.
24. Qiu, S.; Wu, W.W.; Hou, M. Correlation analysis of flight delay based on copula function. *J. Wuhan Univ. Technol.* **2015**, *39*, 117–120.
25. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. *arXiv* **2017**, arXiv:1706.03762.
26. Ahmadbeygi, S.; Cohn, A.; Guan, Y.; Belobaba, P. Analysis of the potential for delay propagation in passenger airline networks. *J. Air Transp. Manag.* **2008**, *14*, 221–236. [CrossRef]
27. Ahmadbeygi, S.; Cohn, A.; Lapp, M. Decreasing airline delay propagation by re-allocating scheduled slack. *IIE Trans.* **2010**, *42*, 478–489. [CrossRef]
28. Sternberg, A.; Soares, J.; Carvalho, D.; Ogasawara, E. A review on flight delay prediction. *arXiv* **2017**, arXiv:1703.06118.
29. Shao, W.; Prabowo, A.; Zhao, S.; Tan, S.; Salim, F.D. Flight delay prediction using airport situational awareness map. In Proceedings of the 27th ACM SIGSPATIAL International Conference, Chicago, IL, USA, 5–8 November 2019; pp. 432–435.
30. Yu, B.; Guo, Z.; Asian, S.; Wang, H.Z.; Chen, G. Flight delay prediction for commercial air transport: A deep learning approach. *Transp. Res. Pt. e-Logist. Transp. Rev.* **2019**, *125*, 203–221. [CrossRef]
31. Alvaro, R.S.; Fernando, G.C.; Rosa, A.V.; Javier, P.C.; Rock, B.M.; Sergio, C.S. Assessment of airport arrival congestion and delay: Prediction and reliability. *Transp. Res. Pt. C Emerg. Technol.* **2019**, *98*, 255–283.
32. Khanna, S.; Tan, V. Economy statistical recurrent units for inferring nonlinear granger causality. *arXiv* **2019**, arXiv:1911.09879.
33. Nicholas, G.P.; Vadim, O.S. Deep learning for short-term traffic flow prediction. *Transp. Res. Pt. C-Emerg. Technol.* **2017**, *79*, 1–17.
34. Kim, Y.J.; Sun, C.; Briceno, S.; Mavris, D. A deep learning approach to flight delay prediction. In Proceedings of the Digital Avionics Systems Conference, Sacramento, CA, USA, 25–29 September 2016; pp. 203–221.
35. Tsoi, A.C.; Tan, S. Recurrent neural networks: A constructive algorithm, and its properties. *Neurocomputing* **1997**, *15*, 309–326. [CrossRef]
36. Cornegruta, S.; Bakewell, R.; Withey, S.; Montana, G. Modelling radiological language with bidirectional Long short-term memory networks. *arXiv* **2016**, arXiv:1609.08409.
37. Klein, A.; Craun, C.; Lee, R.S. Airport delay prediction using weather-impacted traffic index (WITI) model. In Proceedings of the Digital Avionics Systems Conference (DASC), 2010 IEEE/AIAA 29th, Salt Lake City, UT, USA, 3–7 October 2010; pp. 2111–2119.
38. Gai, S.; Bao, Z. Banknote recognition research based on improved deep convolutional neural network. *J. Electron. Inf. Technol.* **2019**, *41*, 1992–2000.
39. Wu, R.B.; Zhao, T.; Qu, J.Y. Flight delay prediction model based on deep SE-DenseNet. *J. Electron. Inf. Technol.* **2019**, *41*, 1510–1517.

40. Cheng, H.; Li, Y.M.; Luo, Q.; Li, C. Study on flight delay with C4.5 decision tree based prediction method. *Syst. Eng. Theory Pract.* **2014**, *34*, 239–247.
41. Xu, T.; Ding, J.; Gu, B.; Wang, J. Forecast warning level of flight delays based on incremental ranking support vector machine. *Acta Aeronaut. Et Astronaut. Sin.* **2009**, *30*, 1256–1263.