Hindawi
Computational Intelligence and Neuroscience
Volume 2022, Article ID 7400833, 11 pages
https://doi.org/10.1155/2022/7400833

Research Article

Analysis on Time-Series Data from Movie Using MF-DCCA Method and Recurrent Neural Network Model Under the Internet of Things

Ruomu Miao[1] and Boyuan Zhang[2]

[1]*School of Media and Communication, Shanghai Jiao Tong University, Shanghai 200240, China*
[2]*School of Music, Film, and Television, Tianjin Normal University, Tianjin 300382, China*

Correspondence should be addressed to Ruomu Miao; miaoruomu@sjtu.edu.cn

Received 5 May 2022; Revised 9 June 2022; Accepted 14 June 2022; Published 8 July 2022

Academic Editor: Vijay Kumar

The present work aims to analyze the time-series data (TSD) from movies and support constructing the movie recommendation system. Referencing the Internet of Things (IoT) technology as the framework, a time-series data analysis system for movies is built based on the recurrent neural network (RNN) and multifractal detrended mobility cross-correlation analysis (MF-DCCA) method. First, the traditional RNN model is improved by replacing the conventional convolution operation with spatial adaptive convolution. Specifically, an additional convolution layer is used to obtain the position parameters required for adaptive convolution to improve the model performance to capture the characteristics of spatial-temporal transformation. Then, the MF-DCCA method is optimized to reduce the interference of noise signals to the analysis processing of TSD from movies. Finally, the TSD analysis system is tested for performance verification. The test results indicate that the method proposed here has outstanding stability and runs smoothly. When the prediction scheme is long short-term memory (LSTM) ($L = 20$), the similarity of the LSTM ($L = 20$) network under one frame is 0.977; the similarity of the LSTM ($L = 20$) network under nine frames is 0.727. This system provides a specific idea for applying the RNN model and MF-DCCA method in analyzing TSD from movies.

1. Introduction

With the continuous development of Internet technologies, computers are becoming even more familiar in daily life. People can use computers to search for necessary information on the Internet. However, it is challenging to quickly find the specific information in the current network environment with a tremendous amount of data, known as information overload [1, 2]. Many researchers have researched this phenomenon and related approaches, such as the search engine, which can search information in the network resource database through the input keywords and retrieve it through a specific sorting mechanism. However, this search method also has some problems: an excess of noise in the searched results. In this case, there is a personalized recommendation system [1, 3]. The personalized recommendation system can actively provide recommendation services for users. It integrates collecting, analyzing, and predicting information and recommends projects to users according to the predicted data. Movie websites contain a large number of movie resources. However, it will take users a lot of time to manually search movies on domestic and foreign websites, or even fruitless. At present, most websites provide search engines with excellent performance to let users precisely locate target movies. This method can help users quickly find movies of interest. However, most users usually do not have a clear goal. Increasing relevant recommendation systems emerge with the increase of online videos and movies. Douban Movie is a popular online movie-sharing community in China. Users can see the movie recommendation based on user evaluation through the “movies of interest” module on the website [4].

Researchers have done much work in the automatic recommendation of movies and TV series. Qiao and Cheng [5] believed that the time-series data (TSD) were critical in the present world, which gradually accumulated with time, with high dimensions and large data scales. However, the traditional feature extraction method was inadequate for the cluster analysis of the TSD. Hence, the authors trained the import data by using the recurrent neural network (RNN) to enhance the clustering function of TSD. First, they utilized the long short-term memory (LSTM) network for the feature extraction of TSD. Then, they used the pooling technology to conduct dimensionality reduction on the output characteristics in the bottom layer of the LSTM network. Finally, they employed the imbalanced K-means algorithm to cluster the reduced-dimensionality characteristics because of the imbalance of most TSD. Meanwhile, experiments were carried out on numerous open, acquirable TSD sets. The authors found that the TSD could be efficiently treated through the combination of the dimensionality reduction via the pooling technology, the cluster analysis of imbalanced data, and feature extraction through the LSTM network. Majumdar and Laha [6] proposed a new time-series classification and clustering method based on the extension of topological data analysis technology. Zhu and Xiao [7] developed a novel precise TSD integration algorithm based on evaluation laboratory models and decision tests after discussing the relation between data. The authors validated the effectiveness and feasibility of the algorithm via numerical examples. Geng and Luo [8] reduced the influence of time intervals on fusion results by analyzing various factors in the model, combined with an ordered weight aggregation operator algorithm. Hu et al. [9] presented a straightforward and practical neural layer to normalize the input time series adaptively. The neural layer used backpropagation to train the model end-to-end, significantly improving the performance compared with other standard schemes. Unlike the traditional normalization method, this method could learn to normalize a specified task rather than utilizing a stationary normalization approach. Besides, it was suitable for any novel TSD without retraining. Le et al. [10] reported that the development of the IoT had popularized all kinds of sensors in various industrial fields and led to a massive increase in sensor data. Because of sensor data's high capacity and dimension, there is a limitation in direct deep data analysis and original DST mining of sensors. Therefore, the authors proposed a multiresolution representation and dimensionality reduction method of sensor data, using a right quantity of critical data points in a specific sensing TSD to generate the equivalent multiresolution segmented linearization portrayal. Theoretical analysis and experimental results showed that the multiresolution piecewise linear representation could reduce the dimension while keeping the critical features of TSD.

After summarizing the current situation worldwide, it is found that traditional algorithms consume a lot of time and space when performing recommendation calculations and cannot cope with the increasing number of users and items. In the past few years, deep learning has been used in image processing. Breakthroughs have been made in the field of classification and speech recognition. Artificial neural network has the characteristics of high computing power, nonlinear mapping ability, high parallelism ability, and generalization ability. As a kind of artificial neural network, convolutional neural network (CNN) has both the advantages of traditional artificial neural network and its unique advantages, so its application in the field of recommendation has important research value.

The feasibility of this study lies in that after researching some existing knowledge points, and the CNN and the RNN are combined and applied in movie rating and recommendation according to the relevant characteristics of deep learning. Finally, the structure of the two neural networks is analyzed and combined accordingly. In this paper, the user's historical rating record behavior is regarded as a sequence of information. An RNN is used to analyze the user's interest and perform corresponding modeling operations combined with the user's rating information. The information of the user and the movie is extracted by a CNN for the subsequent combination.

This paper uses the RNN model and the multifractal detrended mobility cross-correlation analysis (MF-DCCA) method to construct a TSD analysis system for movies. The research innovation lies in two aspects. On the one hand, a spatial adaptive convolution LSTM algorithm is proposed. Inspired by the spatial transformation network, this paper changes the traditional convolution operation to spatial adaptive convolution during the "input-to-state" calculation process inside the convolutional LSTM. An additional convolutional layer is utilized to obtain the position required for the adaptive convolution parameter. Then, the adaptive convolution selects the convolution position according to the spatiotemporal information and improves the model's performance to capture the spatiotemporal transformation features. On the other hand, the construction process and metrics of the MF-DCCA method are studied and used for the analysis of TSD of movies.

2. Analysis System of TSD from Movies

2.1. Optimization of LSTM Network and Construction of Internet of Things System. A neuron in the LSTM model contains a cell state and three gate mechanisms. The cell state is the basis of the LSTM model, equivalent to the model memory. The LSTM model uses the forget, update, and output gates to protect and control the cell state. The input gate controls the candidate state, the forget gate manages the information to be overlooked, and the output gate controls the output information. Figure 1 shows the architecture of the LSTM model.

Combining LSTM with convolution operation can input image-level features into the LSTM network [11, 12]. But such a procedure does not solve the pain point of video sequence analysis in a practical sense. Only using convolution operation cannot filter the information features of the image sequence and cannot meet the performance requirements of video sequence analysis [13–15]. Therefore, researchers design an explicit processing module for the network, explicitly handling the above transformations. This paper proposes a spatial adaptive convolutional LSTM model based on the above ideas, as presented in Figure 2.

The network structure reported here is similar to the classical video prediction network structure, namely the

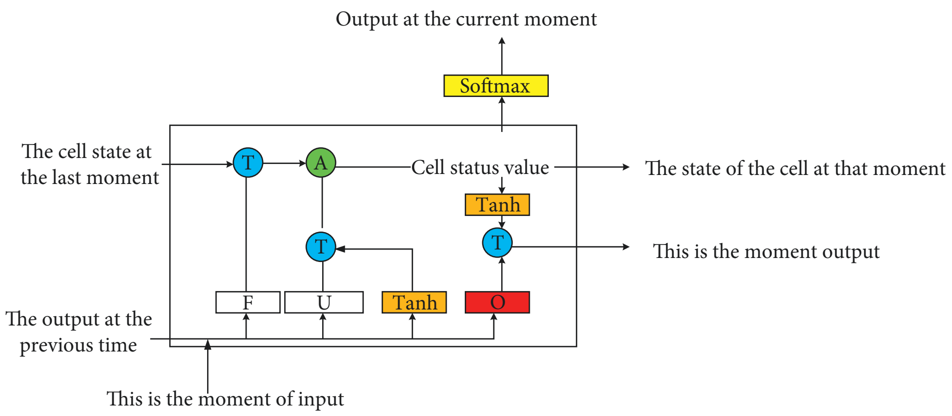

Output at the current moment
Softmax
The cell state at the last moment
T
A
Cell status value
The state of the cell at that moment
Tanh
T
T
This is the moment output
F
U
Tanh
O
The output at the previous time
This is the moment of input

Figure 1: Structure of LSTM model. In Figure 1, *T* means take, *A* represents add, *F* denotes forget the door, *U* means update the door, and *O* indicates output the door.

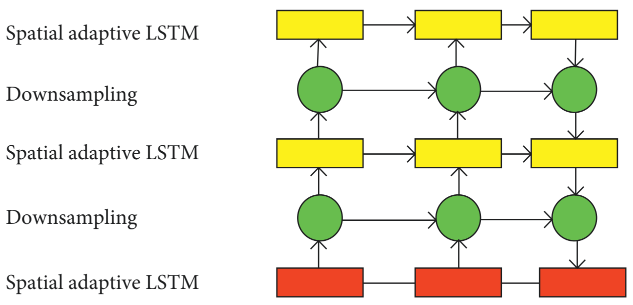


Figure 2: Spatially adaptive convolution LSTM network model.

encoder-predictor structure. The network stacks three hidden layers, i.e., the spatial adaptive convolutional LSTM layer [16, 17]. A downsampling/upsampling layer is inserted between the hidden layers. The sampling layer is essentially a convolution operation to enable the network to characterize the low-level local detail dynamics and high-level global dynamic information in a targeted manner. The network output terminal is placed at the bottom layer of the network. Therefore, high-level spatiotemporal features can guide the calibration and update of low-level local spatiotemporal features from top to bottom and use low-level state information to improve the prediction performance of details. The convolution calculation is to map the target position of the input image and the pixel information of several fixed places around it to the corresponding position of the output image. The mathematical expression of the calculation process of the 3×3 convolution operation is shown in.

$$y_{i,j} = \sum_{l}^{L} W_l \cdot x_{p_{l,i,j}, q_{l,i,j}}. \tag{1}$$

In equation (1), L represents the number of connections per point of the output relative to the input. In the traditional convolution operation with the 3×3 convolution kernel, $L = 9$. $p_{l,i,j}$ and $q_{l,i,j}$ denote the location parameters of lth connection with the output location of (i, j).

The network structure of this paper is similar to the classical video prediction network structure, that is, the encoder-predictor structure. The network stacks three hidden layers, namely the spatially adaptive convolutional LSTM layer. Besides, a downsampling layer or an upsampling layer is inserted between the hidden layers. The sampling layer is a convolution operation, enabling the network to characterize the low-level local detail dynamics and high-level global dynamic information in a targeted manner. The network output is placed at the bottom layer of the network, so high-level spatiotemporal features can guide the calibration and update of low-level local spatiotemporal features from top to bottom and use low-level state information to improve the prediction performance of details.

In the convolutional LSTM, the objects of the convolution operation are the input of the current time step and the state variables of the previous time step. The spatial features of the input and state are extracted through the multilayer convolution operation to determine the trade-offs of state variables and input information at each spatial location. The convolution calculation is to map the target position of the input image and the pixel information of several fixed positions around it to the corresponding position of the output image. Takes the 3×3 convolution operation as an example. Its essence is the mapping from input to output. The pixel value of each position of the output is related to the 9 points around the corresponding position of the input. After finding the corresponding input positions of all target positions, the sum of different weights is given to different channels in the same position. Finally, the sum of the weighted results of different positions is the output.

Figure 3 demonstrates an ordinary $3 * 3$ convolutional neural network (CNN).

The information contained in the current time step is not necessarily the same as that in the previous time step when time and space changes are complex. Based on this, this paper does not fix the convolution kernel size. In this way, the spatial position of each convolution in the convolution operation can change with time to improve the model's ability to capture spatiotemporal correlations [18]. A spatial adaptive convolution operation is introduced, inspired by equation (1) and the spatial transformation network. First, the number of convolution connections L is determined. The position parameter U_t and V_t represent all the positions

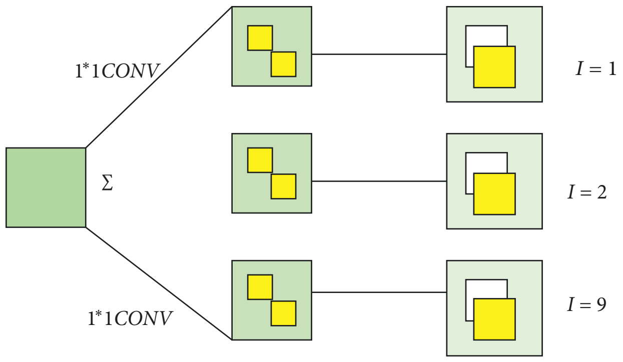


Figure 3: An ordinary $3*3$ CNN. In Figure 3, I represents the number of channels.

related to the output in the input. The input position can be found according to the corresponding position parameter. Then, each position in the output image corresponds to several places in the input image. Equations (2)~(5) describe the new convolution equation to realize adaptive convolution.

$$f_t = \sigma\left(\begin{array}{c}\sum_{l=1}^{L} W_{fx}^{l} * \mathrm{trans}\left(x_t, \mathrm{U}_{t,l}, \mathrm{V}_{t,l}\right) + \\ W_{fh} * h_{t-1} + b_f\end{array}\right), \tag{2}$$

$$i_t = \sigma\left(\begin{array}{c}\sum_{l=1}^{L} W_{ix}^{L} * \mathrm{trans}\left(x_{\mathrm{t}}, U_{t,l}, V_{r,l}\right) + \\ W_{ih} * h_{t-1} + b_i\end{array}\right), \tag{3}$$

$$\widetilde{C}_t = \tanh\left(\begin{array}{c}\sum_{i=1}^{L} W_{cx}^{l} * \mathrm{trans}\left(x_t, U_{t,l}, V_{t,l}\right) + \\ W_{ch} * h_{t-1} + b_c\end{array}\right), \tag{4}$$

$$o_t = \sigma\left(\begin{array}{c}\sum_{i=1}^{L} W_{ox}^{l} * \mathrm{trans}\left(x_t, V_{t,l}, U_{t,l}\right) + \\ W_{oh} * h_{t-1} + b_o\end{array}\right). \tag{5}$$

In equations (2)~(5), $U_{t,l}$ and $V_{t,l}$ represent the horizontal and vertical coordinates of the lth connection, respectively; W_{fh}, W_{ih}, W_{ch}, and W_{oh} are the weights of each threshold layer obtained through training and learning. C stands for the number of channels of the input image. Each threshold layer has L weights. Therefore, the parameter quantity is $C \times L$.

The position parameters used here need to be obtained by deep network training. Because the position parameters are discrete, the reverse derivation method cannot get the position parameters. Therefore, the bilinear difference method is introduced. Here, the coordinate of a position of the output feature is denoted as (i, j), and the convolution position input to the feature map is expressed as (u, v). The pixel value of (u, v) is determined by the bilinear difference method and then input to the convolution. Here, the pixel value calculation is expressed by "wrap". $Y = \mathrm{warp}(X, U, V)$ represents the relationship among parameters. Equation (6) illustrates the mathematical relationship.

$$\begin{gathered} Y_{c,i,j} = \sum_{h=1}^{H}\sum_{w=1}^{W} X_{c,m,n} \max\left(0, 1 - \left|i + V_{i,j} - h\right|\right) \\ \max\left(o, 1 - \left|j + U_{i,j} - w\right|\right). \end{gathered} \tag{6}$$

An explicit processing module is designed to learn the position parameters, as shown in.

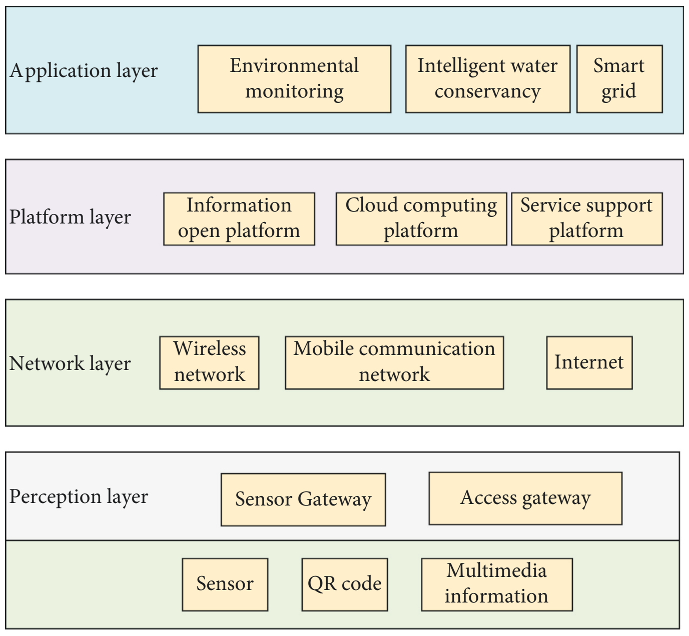
$$U_t, V_t = \gamma\left(x_t, h_{t-1}\right). \tag{7}$$

In equation (7), x_t represents the current time step's input and h_{t-1} refers to the hidden state of the previous time step. Besides, γ denotes a convolution operation between x_t and h_{t-1} and U_t, V_t stands for the convolution output.

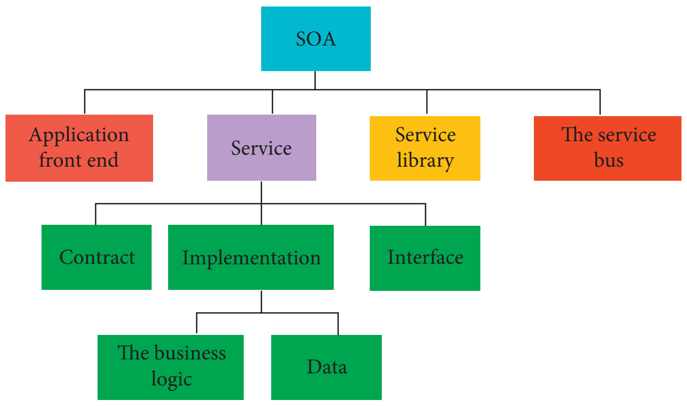
The traditional convolutional LSTM directly uses the image of the current time step or the output of the upper-layer convolutional RNN as the input of the present time step. In contrast, the spatial adaptive convolution LSTM reported here obtains the topological link between the output of the adaptive convolution layer and the input through the γ convolution operation before inputting the image. Then, it uses the topological link to spatially transform the current input to make it align with information in the hidden state for accurate memory preservation and image sequence prediction.

Aiming at the multivariate correlation characteristics of movie sequence data, this paper proposes an improved multivariate LSTM model to predict movie time series. The network structure of the model includes three layers: input layer, hidden layer, and output layer. The input layer controls the format of the input data. The hidden layer is a structure containing several LSTM units, and the error is reduced until convergence by repeated iteration and adjustment of weights. The output layer restores the results to the original data format. The input layer converts the preprocessed time-series data into data that can be used for supervised learning. T time steps are selected as intervals, the data of T time steps before each moment are taken as the input of this moment, and the corresponding sample value of this moment is taken as the target output. The data are divided into input sets and corresponding output sets. This paper integrates multitype sensor data into 3-dimensional data: [sample value, time step, feature] to make the input data contain multivariate. The input is then entered into the hidden layer with a time step as the unique index.

This paper examines the cross-correlation between Google search volume for financial keywords and movie box office. Multifractal features including the coronavirus disease 2019 (COVID-19) period and before the COVID-19 outbreak are discussed. The dynamic evolution of cross-correlation and the impact of COVID-19 on the cross-correlation between the film industry and financial market volatility are observed through the sliding window method. Using the MF-DCCA method, this paper reveals the cross-correlation between two emotion proxies: Google Trends' fear index and Twitter's daily happiness index.



(a)



(b)

FIGURE 4: Structure of SoA ((a) overall architecture; (b) detail architecture).

Earlier researchers defined the Internet of Things (IoT) as a network connecting everything to the Internet to solve the connection between people and things and items and things. The study summarizes the IoT as an intelligent network consisting of perception, network, and application layers to connect people and things or things and things. The essence of IoT is to complete the interaction and connection between people and things or items and things. People or things can control things through the network, and the corresponding things can receive control signals and generate related actions. At present, the functions of the IoT are constantly expanding and updating, evolving from simple object positioning, tracking, and evolution to applying various sensors and intelligent information processing technologies. Like traditional IoT, the polymorphic IoT also has three layers: perception layer, network layer, and application layer. However, it can collect more comprehensive information than traditional IoT. Besides, its network layers also become more complicated when processing data. In polymorphic IoT, the primary vital technologies include radio frequency identification (RFID) and wireless network sensor technology. This paper believes IoT is an extension based on the Internet with the same foundation. The user end of the IoT can expand to anyone and objects [19, 20]. Figure 4 displays the service-oriented architecture (SoA) structure of IoT.

SoA in IoT is necessary for service users and providers. SoA guarantees the interoperability between heterogeneous devices in several ways. Figure 4 provides a common SoA consisting of four layers with different functions. (1) Data perception layer: the perception layer integrates with valid hardware targets to perceive the state of things. The perception layer contains terminals collecting data from IoT, including RFID tags, intelligent phone terminals, wireless sensors, and Bluetooth. (2) Network layer: it is the infrastructure supported by wireless or wired links. The network layer in IoT links everything and enables them to perceive their ambience. Things can share data with interconnected objects through the network layer, essential for intelligent event processing and management in IoT. In addition, the network layer can gather data from current IT infrastructure and then transfer the data to high-level decision-making units, complex services. (3) Service Layer: the service layer is the services required to manage and create applications or users. The service layer needs the support of middleware technology, a crucial initiator of IoT applications and services. Middleware technology offers a cost-efficient system that can reuse hardware and software platforms. (4) Interface layer: the interface layer is composed of the interactive modes with the user or application. In IoT, there are many devices provided by different vendors. Thus, the interface layer does not always adhere to the same standard. Through the previous introduction to the basic architecture of the IoT, the TSD of IoT analyzed here principally comes from the data perception layer. This paper concludes the following characteristics of the TSD of IoT from analyzing the devices with multitudes of data perception layers. (1) Massive: in such a vast IoT architecture, the terminals generate and transmit data all the time. (2) High frequency: these IoT sensing terminals have a very high sensing frequency to monitor and sense the surrounding environment at all times. (3) Heterogeneity: since data streams come from many spatially distributed data sources, IoT data are of different kinds. Also, since mobile devices in the perception layer support multiple sensors, they have different data types. (4) Time dependence: it has a strong temporal correlation. IoT sensor data collected by devices placed in specific locations are time stamped.

2.2. Multifractal Analysis Method of Time Series. With the deepening of research on nonlinear theory, researchers have put forward many methods for multifractal analysis of time series. In the 1990s, Shen et al. proposed the trended fluctuation analysis method. Gil et al. put forward the multifractal detrended fluctuation analysis (MF-DFA) method based on Shen et al., which could easily calculate the main parameters of time series [21, 22]. The conventional Euclidean geometry cannot describe complicated atypical objects in nature. Correspondingly, the concept of fractal was born in the 1960s, indicating the erose objects. The contour lines of the fractal are represented by anomalous fractal curves. There is only a descriptive definition of a fractal without a mathematical description [23, 24]. The generally represented fractal has five features. (1) The fractal is a small-scale complicated and delicate configuration. (2) Conventional geometric expression cannot describe it in a direct manner. (3) It has statistical self-similarity and approximate self-similarity. (4) The fractal dimension is more than the topological dimension in some cases. (5) Recursive methods can produce plain fractal pictures. The categorical attributes can be discussed from two aspects based on the classification characteristics. On the one hand, it is essential to judge whether there is a "scale space." The same natural phenomenon has several scale-free intervals under specific conditions according to the relevant research data. The fractal characteristics may differ in different scale intervals. On the other hand, fractal inlay in nature is the opposite of that in mathematics. The following fractals in the present work belong to the mathematical sense. Generally speaking, the rationale and practical use of fractal are studied in a theoretical perfect space, named the fractal space [25, 26]. The q fluctuation function $F_q(s)$ in the multifractal detrended fluctuation analysis algorithm can be written as.

$$F_q(s) = \left\{ \frac{1}{2N_s} \sum_{v=1}^{2N_s} \left[F^2(s,v) \right] \frac{q}{2} \right\}^{1/q}. \tag{8}$$

In equation (8), q is any real number that is not zero; $F_q(s)$ increases with the growth of s in a power-law relationship. For each s, there is a corresponding function value $F_q(s)$.

The detrended cross-correlation analysis method analyzes the power-law long-term interrelation among different nonstationary time series. For two time series, $F2(s)$ and the sum scale s obey the power-law relationship $F2(s) \sim s\lambda$ in double logarithmic coordinates. λ means the long-term inter-relation scale coefficient. $\lambda > 0.5$ signifies the positive long-term inter-relation between the two series and vice versa [27, 28]. The DCCA method can be extended to study the long-term inter-relation between two unsteady signals. It is assumed that there are two time series denoted as $\{x(t)\}$ and $\{y(t)\}$, where $i = 1, -2, 3 \ldots, M$. The mean values of the two time series are 0, and the time series is composed of $M_s = (M/s)$ blocks with a nonoverlapping size of s. The $\mathscr{V}_{th}$ block $[l_v + 1, l_v + s]$ can be written as equations (9) and (10).

$$X_v(k) = \sum_{j=1}^{k} x(l_v + j), \tag{9}$$

$$Y_v(k) = \sum_{j=1}^{k} y(l_v + j),\ k = 1, \ldots, s. \tag{10}$$

The local tendencies of $\{Y_v(k)\}$ and $\{X_v(k)\}$ are expressed by

$\{\widetilde{Y}_v(k)\}$ and $\{\widetilde{X}_v(k)\}$. Equation (11) illustrates the trend covariance of each block.

$$F_v(s) = \frac{1}{s} \sum_{k=1}^{s} \left[X_v(k) - X_v(k) \right] \left[Y_v(k) - \widetilde{Y}_v(k) \right]. \tag{11}$$

When $q = 0$, the q_{th} detrended covariance can be written as.

$$F_{xy}(q,s) = \left(\frac{1}{m} \sum_{v=1}^{m} F_v(s)^{q/2} \right)^{1/q}. \tag{12}$$

When $q = 0$, the q_{th} detrended covariance can be expressed as.

$$F_{xy}(0,s) = \exp\left(\frac{1}{2m} \sum_{v=1}^{m} \ln F_v(s) \right). \tag{13}$$

Equation (19) can be transformed into.

$$F_{xy}(q,s) \sim s^{h_{xy}(q)}. \tag{14}$$

To sum up, the calculation process of the DCCA method can be divided into five steps. First, two time series are defined as $\{x_t\}$ and $\{y_t\}$, $(t = 1, 2 \ldots, N)$ with time length of N.

(1) Two new time series are established on the basis of $\{x_t\}$ and $\{y_t\}$, $(t = 1, 2 \ldots, N)$, as shown in.

$$\begin{aligned} X(t) &= \sum_{i=1}^{t} [x_i - \langle x \rangle], \\ Y(t) &= \sum_{i=1}^{t} [y_i - \langle y \rangle]. \end{aligned} \tag{15}$$

In equation (15), $\langle x \rangle$ and $\langle y \rangle$ represent the average value of $\{x_t\}$ and $\{y_t\}$, $(t = 1, 2 \ldots, N)$.

(2) $\mathrm{Y}(t)$ and $\mathrm{X}(t)$ are separated into subtime series without crossing with the number of $N_s = [N/S]$.

(3) The local trends $X_v(i)$ and $Y_v(i)$ of the time series are evaluated by the least square method. Then, the downward trend covariance F^2 is obtained, as shown in.

$$\begin{aligned} F^2(s,v) = \frac{1}{s} \sum_{i=1}^{s} &\left| Yvs - Ys + Yi - Y_v(i) \right| \\ &\cdot \left| Xvs - Xs + Xi - X_v(i) \right|. \end{aligned} \tag{16}$$

For each subtime series $v(v = 1, 2 \ldots, N_s)$, there is the relationship shown in.

$$\begin{aligned} F^2(s,v) = \frac{1}{s} \sum_{i=1}^{3} &\left| YN - Yv\mathrm{S} + YN_s s + Yi - Y_v(i) \right| \\ &\cdot \left| XN - Xvs + XN_s s + Xi - X_v(i) \right|. \end{aligned} \tag{17}$$

(4) The downward trend covariance function of the whole time series is obtained by taking the average value of the downward trend covariance of the subtime series with the number of $2N_s$, as presented in.

$$\mathrm{FF}^2_{DCCADCCA} 2^2(s) = \frac{1}{2N_s} \sum_{v=1}^{2N_s} F^2(s,v). \tag{18}$$

(5) There is a power-law correlation between the downtrend covariance function $F_{DC\,CA}(s)$ and the time scale s, as shown in.

$$\mathrm{F}_{DCCA}(s) \propto s^h. \tag{19}$$

In equation (19), h means the thurst index.

The MF-DFA method's calculation process is like the detrended fluctuation analysis approach, except for Step 4. Equation (20) demonstrates Step 4 of MF-DFA.

Equation (20) manifests the decreasing trend variance function with q order of time series when $q = 0$.

Table 1: Settings of the hardware and software environment.

Hardware		Software	
CPU	Inter(R) core(TM) I5–7200U CPU@ 2.5 GHz	Database	MySQL
Operating system	Window10 64 bit ubuntu	Development framework	TensorFlow

$$F(q,s) = \left\{ \frac{1}{2N_s} \sum_{v=1}^{2N_s} \left[F^2(s,v) \right]^{q/2} \right\}^{1/q}. \tag{20}$$

When $q = 0$, there is a relationship as shown in.

$$F(q,s) = exp\left\{ \frac{1}{4N_s} \sum_{v=1}^{2N_s} \ln\left[F^2(s,v) \right] \right\}. \tag{21}$$

3. Experimental Environment and Experimental Data set Configuration

Table 1 lists the hardware and software environment in this experiment.

Mixed National Institute of Standards and Technology (MNIST) is a picture data set of handwritten numbers. The data set was organized by the National Institute of Standards and Technology in the United States, counting a total of 250 pictures of handwritten numbers by different people. It is also a basic data set in the field of computer vision. Many projects related to computer vision conduct basic experiments on this data set.

The experimental data set used here is the MNIST data set. There are 60,000 pictures in the MNIST data set. This paper uses 80% of the pictures as the training set and the rest as the test set. In this paper, the data in the training set are composed of 40,000 image sequences with a length of 25 frames, and the pictures in the test set are composed of 20,000 image sequences with a length of 25 frames. This paper randomly selects 3 numbers from 0 to 9 and then selects the corresponding digital pictures from the MNIST data set. The material rotation angle and zoom factor are set to generate 20 frames of video. This paper also uses the public movie data set MovieLens for training. The data set contains three parts: user data, movie data, and user rating data. The user data include user identification (ID), user gender, user occupation, user age, and the code of the user's region. Since the code of the location region can only represent the region to which the user belongs, the scope is very wide and cannot be representative, so this attribute is eliminated. Movie data contain movie ID, movie name, and movie genre. It should be noted that the movie name field is a combination of the movie name and the release time. There are a total of 18 movie genres. A movie may belong to more than one movie genre. User rating data are composed of user ID, movie ID, rating, and timestamp.

In the model's training process, the Adam algorithm is used to optimize this model. The data set constructed via the

Table 2: Test results of the common user function and the system administrator function.

		Number of tests	Number of failures	Success times	Pass or not
General user function test	Information update	100	0	100	Adopt
	Watch movies	100	1	99	Adopt
	Movie evaluation	100	0	100	Adopt
	Delete comments	100	0	100	Adopt
	Verification code acquisition	100	1	99	Adopt
System administrator function test	Information management	100	0	100	Adopt
	Search for movies	100	1	99	Adopt
	Update movie	100	0	100	Adopt
	Add movie	100	0	100	Adopt
	Delete movie	100	1	99	Adopt
	Log movie	100	0	100	Adopt

*the experiments in Table 2 are similar to the experimental settings in reference [29].

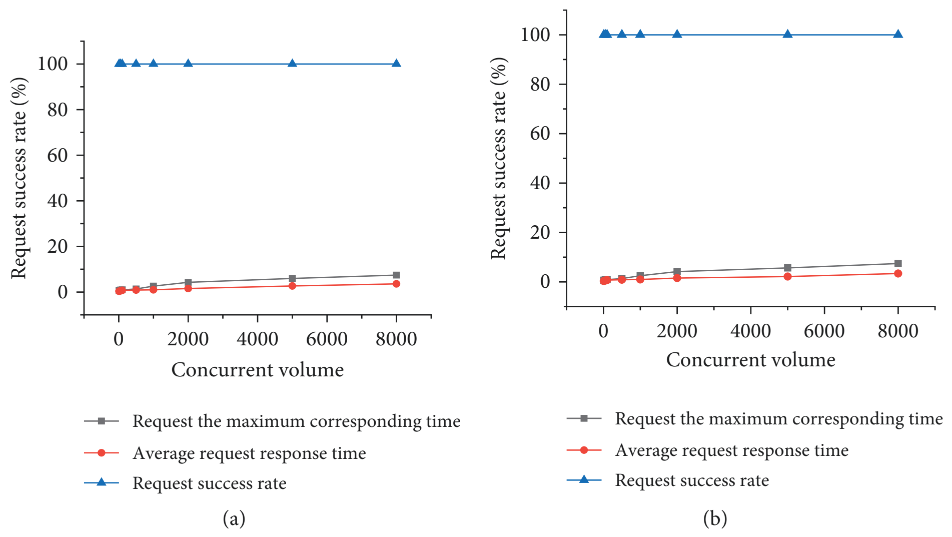


Figure 5: System pressure test ((a) 1st test; (b) 2nd test).

MNIST database is used to train the network. The network is set to iterate 1,500 times. Control variables are used to verify the effectiveness of the optimized algorithm after all iterations. Batch size will affect two aspects of the network: the optimization performance and the convergence speed of the model and the memory size of the GPU. When the value of batch size is enormous, the convergence speed of the model becomes fast, but the increase of accuracy is reduced to some extent. After many experiments, this paper set the batch size to 8. This experiment tests the three layers adaptive convolution LSTM. The number of convolution kernels in each threshold layer from bottom to top is 64, 192, and 192, respectively.

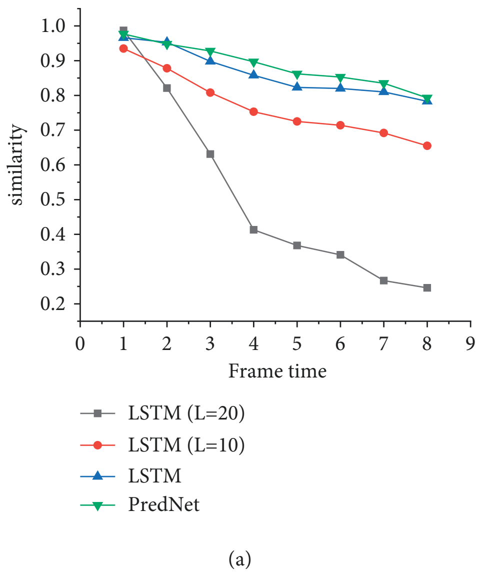
4. Algorithm Test and Simulation

4.1. System Test. Table 2 summarizes the test results of the common user function and the system administrator function.

As can be seen from Table 2, when the system performs the function test of ordinary users, all the subfunction tests of the system pass. When the system performs the function test of the system administrator, all the subfunction tests of the system pass. To sum up, the system has good stability, and the functions of this part meet the analysis requirements.

Figure 5 presents the pressure test results of the system, in which the abscissa refers to the number of users, and the ordinate refers to the worst-case response time.

The difference between the two tests in Figure 5 is that the environment for the first test is relatively simple, and the environment for the second test is complex. The system stress test results indicate that the system's response time rises with increased user concurrency. When the user concurrency is 10, the worst-case response time of the request is 0.634 s, the mean response time of the proposal is 0.342 s, and the request success rate is 100%. When the user concurrency is 500, the worst-case response time of the request is 1.342 s, the mean response time of the proposal is 0.875 s, and the request success rate is 100%. When the user concurrency is 5000, the worst-case response time of the request is 5.965 s, the mean response time of the request is 2.645 s, and the request success rate is 100%. When the user concurrency is 8000, the worst-case response time of the



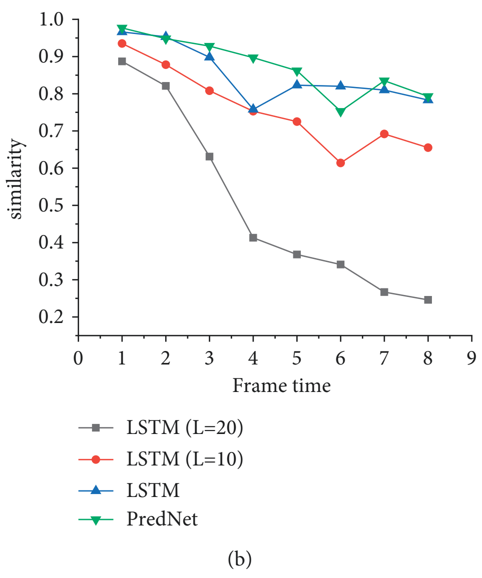
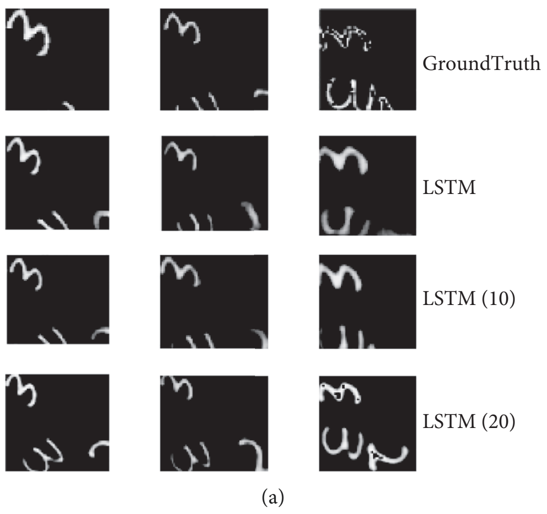


Figure 6: Frame by frame structural similarity evaluation results of video prediction on MNIST database ((a) the learning rate is 0.05; (b) the learning rate is 0.005).



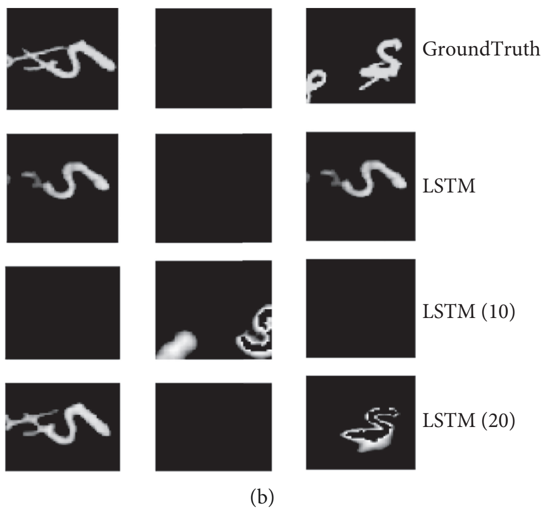


Figure 7: MNIST experimental results ((a) test on the number “3”; (b) test on the number “5”).

request is 6.233 s, the mean response time of the request is 3.174 s, and the request success rate is 100%. On the whole, although the system response time rises slightly with the increasing in users, the system can respond within the range that users can tolerate, so the performance of the system is good.

4.2. *Frame-by-Frame Structure Similarity Evaluation of MNIST Video Prediction.*

Figure 6 reveals the frame-by-frame structure similarity evaluation results of MNIST video prediction, in which the abscissa is the frame time, and the ordinate is the similarity.

According to Figure 6, when the prediction scheme is PredNet, the similarity of PredNet is 0.978 under one frame; the similarity of the PredNet network is 0.206 under nine frames. In contrast, when the prediction scheme is LSTM, the similarity of the LSTM network is 0.935 under one frame; the similarity of the LSTM network is 0.551 under nine frames. When the prediction scheme is LSTM ($l = 10$), the similarity of the LSTM (10) network is 0.966 under one frame; the similarity of the LSTM (10) network is 0.707 under nine frames. When the prediction scheme is LSTM ($l = 20$), the similarity of the LSTM ($l = 20$) network is 0.977 under one frame; the similarity of the LSTM ($l = 20$) network is 0.727 under nine frames. To sum up, because PredNet lacks the Ground Truth structure to calculate the error, the similarity drops rapidly from the second frame of prediction. On the contrary, the performance of the model reported here is excellent.

Figure 7 signifies the experimental results of MNIST. The first column displays the experimental results of predicting the 2nd frame in the video sequence. The second column presents the experimental results of predicting the 10th frame in the video sequence. The third column bespeaks the prediction results of predicting the 15th frame in the video sequence.

Figure 7 suggests that when the traditional LSTM network processes an image with a complex shape, the definition of the image decreases. Still, the adaptive LSTM networks with ten links and 20 links can reasonably predict the difficult changes of the image. Compared with the adaptive LSTM network with 20 links, the adaptive LSTM network with ten links can better maintain the definition of the image, and it can better predict the clarity of the image.

5. Conclusions

Based on the IoT technology, an analysis system of TSD from a movie is constructed based on RNN and the MF-DCCA method. In optimizing RNN, a spatial transformation layer is added to its network structure to explicitly learn spatial-temporal changes' characteristic. The performance of the model based on cyclic neural network and MF-DCCA method is evaluated by the prediction results of handwritten video clips. Experiments show that in some cases, designing a single module to explicitly learn some features can endow the network with brilliant generalization capability. In the experiment, when the user concurrency is 5,000, the worst-case response time is 5.965, the mean response time is 2.645, and the request success rate is 100%. Adopting LSTM ($L = 20$) as the prediction scheme, the similarity of the LSTM ($L = 20$) network is 0.977 under one frame, and it is 0.727 under nine frames. Although the optimized LSTM network has considerable performance improvement compared with the traditional convolution LSTM and a strong ability to capture complex spatial-temporal variation characteristics and is more competent for the task of pixel-level video prediction. Still, it has some shortcomings. The data set used in this experiment has a small volume and simple construction. In addition, the network structure also has some room for refinement. The future work will introduce the attention mechanism into the network to enhance the prediction accuracy.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

Acknowledgments

This work was supported by major projects of National Social Science Foundation of China (21&ZD325).

References

[1] C. W. Chen, S. P. Tseng, and T. W. Kuan, "Outpatient text classification using attention-based bidirectional LSTM for robot-assisted servicing in hospital," *Information*, vol. 11, no. 2, p. 106, 2020.

[2] A. Dagliati, N. Geifman, N. Peek, J. H. Holmes, and A. Sacchi, "sing topological data analysis and pseudo time series to infer temporal phenotypes from electronic health records," *Artificial Intelligence in Medicine*, vol. 108, Article ID 101930, 2020.

[3] T. Noumi, S. Inoue, H. Fujita, K. Sadamitsu, and H. Sakaguchi, "Epitope prediction of antigen protein using attention-based LSTM network," *Journal of Information Processing*, vol. 29, pp. 321–327, 2021.

[4] G. Yang and B. Lee, "Utilizing topic-based similar commit information and CNN-LSTM algorithm for bug localization," *Symmetry*, vol. 13, no. 3, p. 406, 2021.

[5] M. Qiao and Z. Cheng, "A novel long- and short-term memory network with time series data analysis capabilities," *Mathematical Problems in Engineering*, vol. 13, no. 2, p. 11, 2020.

[6] S. Majumdar and A. K. Laha, "Clustering and classification of time series using topological data analysis with applications to finance," *Expert Systems with Applications*, vol. 162, no. 1, Article ID 113868, 2020.

[7] W. Zhu and F. Xiao, "Improvement of time series data fusion based on evidence theory and DEMATEL," *IEEE Access*, vol. 71 page, 2019.

[8] Y. Geng and X. Luo, "Cost-sensitive convolutional neural networks for imbalanced time series classification," *Intelligent Data Analysis*, vol. 23, no. 2, pp. 357–370, 2019.

[9] Y. Hu, C. Ji, Q. Zhang, L. Chen, P. Zhan, and X. Li, "A novel multi-resolution representation for time series sensor data analysis," *Soft Computing*, vol. 24, no. 14, pp. 10535–10560, 2020.

[10] X. Le, T. M. Tran, and H. T. Nguyen, "An improvement of SAX representation for time series by using complexity invariance," *Intelligent Data Analysis*, vol. 24, no. 3, pp. 625–641, 2020.

[11] H. Li, "Time works well: dynamic time warping based on time weighting for time series data mining," *Information Sciences*, vol. 547, pp. 592–608, 2021.

[12] K. Hees, S. Nayak, and P. Straka, "Statistical inference for inter-arrival times of extreme events in bursty time series," *Computational Statistics & Data Analysis*, vol. 155, Article ID 107096, 2021.

[13] R. Rezvani, P. Barnaghi, and S. Enshaeifar, "A new pattern representation method for time-series data," *IEEE T KNOWL DATA EN*, vol. 15, p. 10, 2019.

[14] M. Khayati, P. Cudré-Mauroux, and M. H. Bhlen, "Scalable recovery of missing blocks in time series with high and low cross-correlations," *Vine J Inf Knowl Man*, vol. 62, no. 3, p. 23, 2020.

[15] P. Zarbakhsh and H. Demirel, "4D facial expression recognition using multimodal time series analysis of geometric landmark-based deformations," *Knowledge and Information Systems*, vol. 62, pp. 2257–2280, 2020.

[16] W. Xu, H. Peng, X. Tian, and X. Peng, "DBN based SD-ARX model for nonlinear time series prediction and analysis," *VISUAL COMPUT*, vol. 36, pp. 951–965, 2020.

[17] D. C. Nascimento, B. Pimentel, R. Souza, J. P. Leite, and F. Louzada, "Dynamic time series smoothing for symbolic

interval data applied to neuroscience," *INFORM SCIENCES*, vol. 517, no. 1, p. 21, 2019.

[18] R. Nock, N. Polouliakh, F. Nielsen, K. Oka, and H. Connell, "A Geometric Clustering Tool (AGCT) to robustly unravel the inner cluster structures of time-series gene expressions," *PLoS One*, vol. 15, no. 7, Article ID e0233755, 2020.

[19] J. Li, H. Izakian, W. Pedrycz, and I. Jamal, "Clustering-based anomaly detection in multivariate time series data," *Applied Soft Computing*, vol. 100, no. 4, Article ID 106919, 2020.

[20] G. H. Putri, M. N. Read, I. Koprinska, D. Singh, U. Röhm, and T. M. Ashhurst, "ChronoClust: density-based clustering and cluster tracking in high-dimensional time-series data," *Knowledge-Based Systems*, vol. 174, no. 6, pp. 9–26, 2019.

[21] Z. Shen, Y. Zhang, J. Lu, J. Xu, and G. Xiao, "A novel time series forecasting model with deep learning," *Neurocomputing*, vol. 3, no. 1, p. 96, 2019.

[22] A. Gil, M. Quartulli, I. G. Olaizola, and B. Sierra, "Learning optimal time series combination and pre-processing by smart joins," *Applied Sciences*, vol. 10, no. 18, p. 6346, 2020.

[23] Z. Sun, Q. Peng, X. Mou, Y. Wang, and T. Han, "An artificial intelligence-based real-time monitoring framework for time series," *Journal of Intelligent and Fuzzy Systems*, vol. 40, no. 6, pp. 10401–10415, 2021.

[24] C. Luo, N. Zhang, and X. Wang, "Time series prediction based on intuitionistic fuzzy cognitive map," *Soft Computing*, vol. 24, no. 9, pp. 6835–6850, 2020.

[25] L. Hu, T. Huang, and J. You, "Two-step estimation of time-varying additive model for locally stationary time series," *Computational Statistics & Data Analysis*, vol. 130, pp. 94–110, 2019.

[26] B. Chakraborty, "A proposal for classification of multisensor time series data based on time delay embedding," *International Journal on Smart Sensing and Intelligent Systems*, vol. 7, no. 5, pp. 1–5, 2020.

[27] L. Jin, "Robust tests for time series comparison based on Laplace periodograms," *Computational Statistics & Data Analysis*, vol. 160, no. 5, Article ID 107223, 2021.

[28] W. Sulandari, S. Subanar, M. H. Lee, and P. C. Rodrigues, "Time series forecasting using singular spectrum analysis, fuzzy systems and neural networks," *MethodsX*, vol. 7, Article ID 101015, 2020.

[29] R. Kumar, P. Kumar, and Y. Kumar, "Time series data prediction using IoT and machine learning technique," *Procedia Computer Science*, vol. 167, pp. 373–381, 2020.