Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 6896579, 15 pages
https://doi.org/10.1155/2020/6896579

Research Article

Capsules TCN Network for Urban Computing and Intelligence in Urban Traffic Prediction

Dazhou Li[1], **Chuan Lin**[2], **Wei Gao**[1], **Zeying Chen**[1], **Zeshen Wang**[3] and **Guangqi Liu**[4,5]

[1]*College of Computer Science and Technology, Shenyang University of Chemical Technology, Shenyang 110016, China*
[2]*Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Dalian University of Technology, Dalian 116024, China*
[3]*School of Computer Science and Engineering, Northeastern University, Shenyang 110819, China*
[4]*Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China*
[5]*University of Chinese Academy of Sciences, Beijing 100049, China*

Correspondence should be addressed to Chuan Lin; chuanlin1988@gmail.com

Received 15 January 2020; Revised 16 February 2020; Accepted 25 February 2020; Published 4 June 2020

Guest Editor: Wei Wang

Predicting urban traffic is of great importance to smart city systems and public security; however, it is a very challenging task because of several dynamic and complex factors, such as patterns of urban geographical location, weather, seasons, and holidays. To tackle these challenges, we are stimulated by the deep-learning method proposed to unlock the power of knowledge from urban computing and proposed a deep-learning model based on neural network, entitled Capsules TCN Network, to predict the traffic flow in local areas of the city at once. Capsules TCN Network employs a Capsules Network and Temporal Convolutional Network as the basic unit to learn the spatial dependence, time dependence, and external factors of traffic flow prediction. In specific, we consider some particular scenarios that require accurate traffic flow prediction (e.g., smart transportation, business circle analysis, and traffic flow assessment) and propose a GAN-based superresolution reconstruction model. Extensive experiments were conducted based on real-world datasets to demonstrate the superiority of Capsules TCN Network beyond several state-of-the-art methods. Compared with HA, ARIMA, RNN, and LSTM classic methods, respectively, the method proposed in the paper achieved better results in the experimental verification.

1. Introduction

Empowered by Internet of Things (IoTs) technologies and advanced algorithms that can collect and handle massive traffic datasets, urban computing and intelligence can make more informed decisions and create feedback loops between actual traffic situation and management department in the urban environment [1]. It can bridge the gaps between ubiquitous sensing, intelligent computing, cooperative communication, and big data management technologies to create novel solutions which can improve urban traffic environments, quality of life, and smart city systems [2]. In these urban computing methods, the huge datasets used by the scientists are all from various sources, such as geographic information, taxi GPS, and online weather web sites [3].

Urban traffic prediction has become a challenging urgent task for the development of a smart urban city, as it can afford visions for urban planning and traffic administration to improve the performance of urban transportation, as well as provide warnings for public security emergency message as timely [4]. Moreover, urban traffic prediction has been an important research issue with highly social shock [5]. When some emergences happen such as traffic accidents, an earthquake, tornado, and national holiday, urban traffic prediction becomes the top priority for authority (e.g., law enforcement) and traffic management operators (e.g., bus/ferry/subway) to protect people's safety and keep the work of social infrastructures [6]. Particularly for an enormous population city such as New York and London, the urban traffic is very heavy, which commonly leads to

more probability for different traffic collisions and accident situations [6].

To meet this challenge, we are with the purpose of deriving the urban traffic prediction from period, trend, geospatial, and external influences and generate an accurate prediction for the urban traffic in the next time window, which is considered to be an available way to dispose the urban computing. We propose a neural network-based method called Capsules TCN Network based on collected big traffic mobility data and two deep-learning architecture TCN and Capsules Network. For real time, we also proposed a further improvement method for spatial-temporal data processing to achieve supervision of urban area vehicle density.

2. Related Work

Traffic flow prediction has been considered as a key functional component of intelligent transportation systems. Meanwhile, artificial intelligence technology is rapidly growing and the fifth-generation communication technology is approaching [7–15]. Massive traffic data are being continuously collected through all kinds of sources, some of which can be treated and utilized as streaming data for understanding and predicting urban traffic [6]. All these stimulate us to take new efforts and achieve new success on this social issue by using such streaming mobility data and advanced artificial intelligence technologies [6].

The evolution of traffic flow can be considered to be a spatiotemporal process. As early as the 1970s, the autoregressive integrated moving average (ARIMA) model was used to predict the short-term traffic flow of expressways [16]. Traffic flow prediction based on a time series method is a widely used traffic flow prediction technology. Levin and Tsao applied Box-Jenkins time series analysis to predict highway traffic flow and found that the ARIMA (0, 1, 1) model was useful in the prediction of the most statistically significant [17]. Hamed et al. used the ARIMA model to predict the traffic volume of urban arterial roads [18]. In order to improve the prediction accuracy, many variants of ARIMA were proposed, such as Kohonen-ARIMA [19], subset ARIMA [20], ARIMAX [21], space-time ARIMA [22], and seasonal ARIMA [23]. In addition to ARIMA-type time series models, other types of time series models are also used for traffic flow prediction [24].

On account of the random and nonlinear nature of traffic flow, nonparametric methods have received widespread attention in the field of traffic flow prediction. Davis and Nihan used the KNN method for short-term traffic prediction on expressways [25]. Chang et al. proposed a dynamic multi-interval traffic forecasting model based on KNN nonparametric regression [26]. Faouzi developed an autoregressive function with a smooth kernel function for short-term traffic flow prediction, in which a function estimation technique was applied [27]. Sun et al. used a local linear regression model for short-term traffic prediction [28]. A traffic flow prediction method based on Bayesian network was also proposed [29]. It proposed an online learning weighted support vector regression (SVR) for short-term traffic flow prediction. Various artificial neural network models for predicting traffic flow have been established [30–32]. The MA, ES, and ARIMA models are used to obtain three related time series, which are the basis of the nature in the aggregation phase [33]. Zargari et al. developed different linear programming, multilayer perceptron, and fuzzy logic models to estimate 5- and 30-minute traffic flows [34]. Cetin and Comert combined the ARIMA model with the expectation maximization and cumulative sum algorithm [35].

Yao et al. proposed to combine the principal component analysis method with SVR and select urban multisection data to establish a road network short-term prediction model that took into account the relationship between time and space of multiple sections [36]. Li et al. used the wavelet decomposition and wavelet reconstruction of the traffic flow sequence data and then the use of Kalman filtering for dynamic data prediction [37]; Sun et al. proposed the application of the gray system theory to intersection traffic volume prediction [38]. Xiong et al. combined traditional linear models with artificial intelligence prediction models and proposed a short-term traffic flow prediction method based on artificial neural networks and Kalman filtering [39].

This article is divided into 6 sections: The first section describes the research background, significance, and purpose of the traffic forecast of urban vehicle traffic. The second section introduces the current situation and the structure of this article. The third section models the traffic forecast in urban areas and introduces the structure of Capsules TCN Network, which has two main technologies: Capsules Network and Temporal Convolutional Network. At the same time, the Capsules TCN Network model results are superresolution reconstructed to obtain a regional traffic flow forecast map with higher accuracy. The fourth section introduces the dataset used in the experiments and the data preprocesses, the experimental criteria, and the comparative baselines. Moreover, in the experimental environment, platform construction is introduced and the experimental results are demonstrated and analyzed. The fifth section summarizes the whole research.

3. Analytical Model of Regional Traffic

3.1. Regional Flow Prediction Problem. In urban areas, the indicator of vehicle flow can be used to indicate the vehicle flow in an area. This indicator can well reflect the traffic, population density, and public safety of a region. This article predicts two types of vehicle group traffic: inflow and outflow, as shown in Figure 1(a). Inflow refers to the total volume of vehicles entering a certain area from other areas within a given time interval. Outflow represents the total flow of vehicles leaving the area in a given time interval. Both types of traffic are used to indicate the movement patterns of vehicle traffic in urban areas. Understanding them can be of great help in risk assessment and traffic management. Inflow and outflow can be measured by the number of cars driving near the road, the number of cars driving on public transportation systems (e.g., subway and buses), the number of taxis, or all available data. Figure 1(b) shows an example of using the GPS trajectory of a rental car to measure the amount of traffic. The results show that the inflow

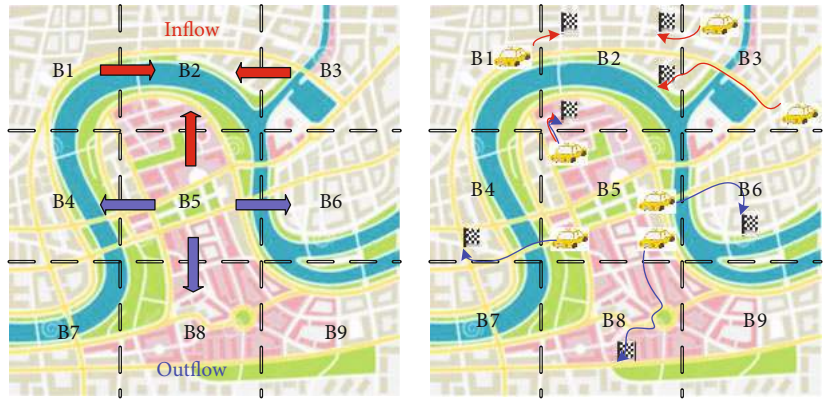


Figure 1: (a) Inflow and outflow. (b) Using the GPS trajectory of a rental car to measure the amount of traffic.

in area B2 is 4 and the outflow in B5 is also 4. Obviously, predicting traffic flow can be regarded as a spatial-temporal prediction problem.

There are three complex factors in the spatial-temporal prediction problem:

3.1.1. Space dependence. As shown in Figure 1(a), the inflow in the B2 area is affected by the outflow in its vicinity (such as B5). Similarly, the outflow of B5 will affect the inflow of other regions (such as B2). The inflow of the B2 region will affect its own outflow. Urban traffic flow may be even affected by distant areas. For example, people who live far away from the office always take the car or taxi to work, which means that the outflow of long-distance residential areas directly affects the inflow of office areas.

3.1.2. Time dependence. The change of the traffic flow in any area is generally continuous from the perspective of time. It means the traffic flow at the next moment and the traffic flow at the previous moment have the strongest correlation. With the increase of the time interval, the correlation of traffic flow will gradually decrease. Figure 2 shows the time-varying curves of the traffic flow in a typical residential area and a typical working area from our dataset. It can be seen that both curves are relatively smooth, reflecting the continuous change characteristics described above. At the same time, it can be seen from Figure 2 that the change curve of the traffic flow in the living area is different from the change curve of the traffic flow in the working area, which reflects the regional differences.

Different regions have different numbers of population densities. Residential areas are suitable for living and resting. In a residential area, each person has a larger unit space that is more suitable for living and resting. Therefore, the lower the population density of a residential area, the better the residential area. In the work area, the closer the workers are, the more convenient the communication is and the work is more efficient. Therefore, the population density in the work area is much larger than that in residential areas. Different population densities determine different needs for public transportation. It can be seen from Figure 2 that although the trend of the number of taxis in the residential

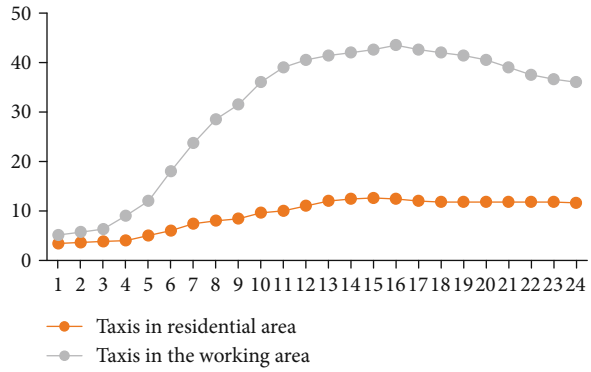


Figure 2: Time continuity and difference of the number of taxis in the residential area and the working area.

area and the working area is basically the same over time, there are obvious differences in the magnitude of the two.

As shown in Figure 3, whether it is a change in the traffic flow in the work area or a change in the traffic flow in the residential area, there are obvious characteristics of periodic changes. To further complicate matters, this periodicity will also be different under different time scales. When you observe in days, you can see the daily fluctuations of vehicles from morning to night. When you observe in weeks, you can see fluctuations of vehicles from work. If you look at the unit of year, you can see the impact of the climate and holidays on the traffic flow in the four seasons.

This paper divides time dependence into period and tendency. Period: traffic during the morning rush hour is similar on consecutive working days. The morning rush hour usually occurs from 8 AM to 10 AM, and the evening rush hour is usually from 17 to 21 PM, repeated every 24 hours. Tendency: there is a cyclical difference between traffic between a working day and a nonworking day, with a time interval of one week.

3.1.3. External factors. Some external factors, such as weather conditions and holidays, can drastically change traffic flow in different areas of the city. As shown in Figure 4, a rainstorm affects the speed of traffic on the road and further

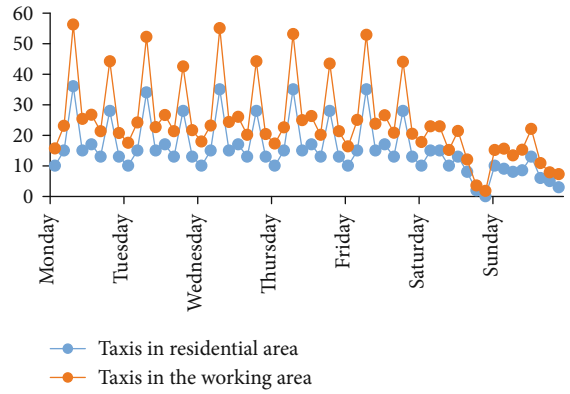


Figure 3: Time periodicity of the number of taxis in the residential area and the working area.

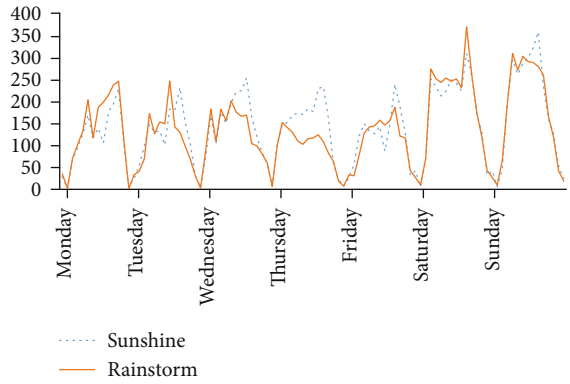


Figure 4: Impact of extreme weather on the number of taxis in the working area.

changes the area's traffic volume. Figure 5 shows the impact of holidays on a regional traffic.

There are many ways to divide the city area. According to the function, it can be divided into working area, residential area, mixed area, etc. It can also be divided according to the structure of the urban road network, and the city can be divided into main roads by using the map division method. The division method is introduced in the paper. We use grids to divide cities according to latitude and longitude. As shown in Figure 6(a), j and k represent the number of rows and columns in the area, respectively. In actual life, the values of j and k can be adjusted according to different city sizes and different application scenarios. In this paper, the scenario is divided into 16×16 grids.

Let R be the trajectory set of the ith time interval. For the grid (j, k) located in the jth row and the kth column, the inflow and outflow at the time interval i are defined as

$$\alpha_i^{j,k} = \sum_R \text{Count}(L_{i-1} \notin (j,k) \text{ and } L_i \in (j,k)), \quad (1)$$

$$\beta_i^{j,k} = \sum_R \text{Count}(L_i \in (j,k) \text{ and } L_{i+1} \notin (j,k)). \quad (2)$$

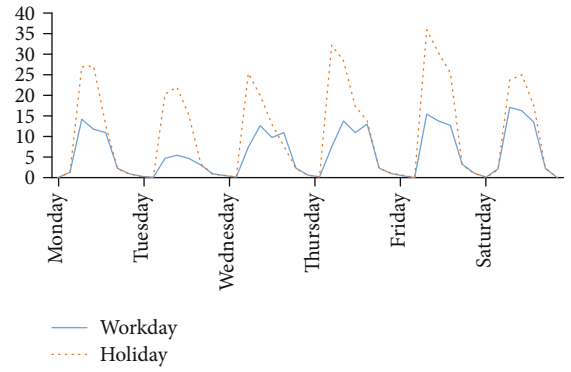


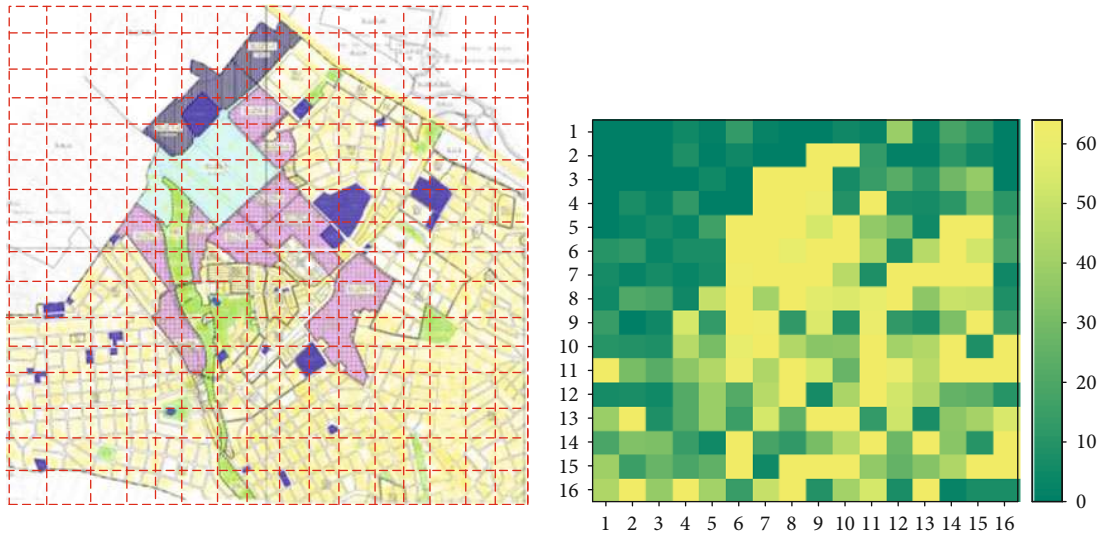
Figure 5: Impact of social event on the number of taxis in the working area.

Among them, L_i is the trajectory of all the vehicles in R at ith time interval. Here, the trajectory is determined according to the GPS coordinates in the dataset and the grids divided by the map. In the ith time interval, inflow of 16×16 grids in the entire area can be represented by a matrix composed of $\alpha_i^{j,k}$ as shown Figure 6(b). The traffic prediction problem is transformed into known historical data $\alpha_i^{j,k}$ and $\beta_i^{j,k}$ to predict $\alpha_{i+1}^{j,k}$ and $\beta_{i+1}^{j,k}$ in the next moment.

3.2. Algorithm Model of Capsules TCN. Both recurrent neural networks (RNN) and long-term short-term memory (LSTM) are capable of learning remote time dependence. However, if RNN or LSTM is used to simulate time periods and trends, it requires very long input sequences, which make the entire training process very complicated. According to the knowledge of space-time domain, only a few previous key frames will affect the next key frame. Therefore, we use time period, tendency, and geographic space to select key frames for modeling. Figure 7 shows the architecture of Capsules TCN Network proposed in the paper. It consists of four primary parts, which model time period, tendency, geospatial, and external influences.

As shown Figure 7, first, the methods introduced in formulas (1) and (2) are used to convert the inflow and outflow of the entire city at each time interval into a 2-channel matrix. The spaced 2-channel stream matrix in each time segment is sent to the first two parts, respectively, and the same network structure of proposed Capsules TCN Network is used for modeling. This structure also captures the spatial dependence between nearby and distant areas. They are provided to the same neural network structure in the external factors. The output of the four parts is fused in the way of fully convolutional networks. Finally, the result is mapped to the range [-1, 1] by the Sigmoid function, which produces faster convergence than the standard logic function during the backpropagation learning process. The entire neural network structure consists of two important methods: Capsules Network [40] (CapsulesNet) and Temporal Convolutional Network [41] (TCN).

3.2.1. CapsulesNet in Capsules TCN Network. Sabour et al. published a paper on Google Brain entitled "Dynamic



(a) (b)

Figure 6: (a) Grid division of urban areas. (b) Inflow matrix in the ith time slice of the urban areas in the (a).

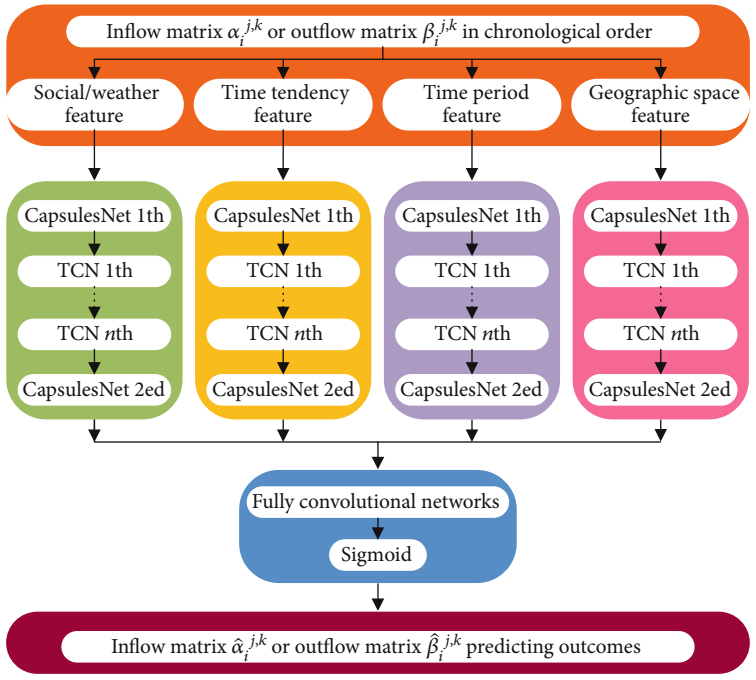


Figure 7: Capsules TCN architecture.

routing between capsules" [40]. We use the ideas from the reference when designing our Capsules TCN Network. Figure 8 shows the architecture of Capsules network (CapsulesNet). CapsulesNet, like ordinary neural networks, consists of many layers. The lowest capsule layer is called the primary capsule layer: each capsule unit in them receives a region of a matrix as input and detects the presence and posture of a specific object, and higher layers can detect larger and more complex objects.

Capsules are a group of neurons whose input and output vectors represent instantiation parameters of a specific entity type (that is, the probability of certain objects, conceptual entities, etc. appearing and certain attributes). The capsules at the same level use the transformation matrix to predict the instantiation parameters of higher-level capsules. When multiple predictions are consistent (this paper uses dynamic routing to make predictions consistent), higher-level capsules become active. The activation of the neurons in the capsule represents the various properties of the specific entities present in the matrix. These properties can include many different parameters, such as pose (position, size, and orientation), deformation, speed, reflectivity, color, texture, and more.

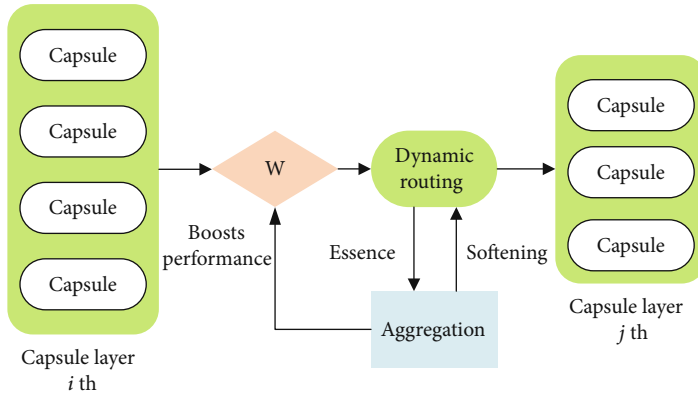


Figure 8: Capsule network architecture.

The length of the input-output vector represents the probability of an entity appearing, so its value must be between 0 and 1. To achieve this compression and complete capsule level activation, Sabour et al. used a nonlinear function called "squashing." This nonlinear function ensures that the length of the short vector can be shortened to almost zero, and the length of the long vector is compressed to close to but not more than 1 [40]. Here is the expression for this nonlinear function [40]:

$$V_j = \frac{\|S_j\|^2}{1+\|S_j\|^2}\frac{S_j}{\|S_j\|}, \tag{3}$$

where V_j is the output vector of capsule j, which S_j is the weighted sum of the vector output by all capsules in the previous layer to capsule j in the current layer, which S_j is simply the input vector of capsule j. The nonlinear function can be divided into two parts [40], namely,

$$\frac{\|S_j\|^2}{1+\|S_j\|^2}, \quad \frac{S_j}{\|S_j\|}, \tag{4}$$

the first part is the scaling of the input vector S_j, and the second part is the unit vector of the input vector S_j. This nonlinear function not only retains the direction of the input vector but also compresses the length of the input vector to the interval [0,1). When S_j is zero, V_j can take 0, and when S_j is infinity, V_j approaches 1 infinitely. This nonlinear function can be seen as a kind of compression and reallocation of the vector length, so it can also be seen as a way to "activate" the output vector after the input vector.

Then, as mentioned above, the input vector of capsule is equivalent to the scalar input of a classic neural network, and the calculation of this vector is equivalent to the way of propagation and connection between two layers of capsules. The calculation of the input vector is divided into two phases, namely, linear combination and routing. This process can be expressed by the following formula [40]:

$$S_j = \sum_i c_{ij}\hat{u}_{j|i}, \hat{u}_{j|i} = W_{ij}u_i, \tag{5}$$

where $\hat{u}_{j|i}$ is a linear combination of u_i, which can be seen as a general neuron in the previous layer outputs with different strengths to a neuron in the next layer [40]. Just that capsule has a set of neurons (to generate a vector) at each node compared to a general neural network, which $\hat{u}_{j|i}$ means that the output vector of the ith capsule in the previous layer is multiplied by the corresponding weight vector (W_{ij} representing a vector). The resulting prediction vector $\hat{u}_{j|i}$ can also be understood as the strength of connecting to the jth capsule in the latter layer if the previous layer is the ith capsule.

After $\hat{u}_{j|i}$ decision is made, routing needs to be used for the second stage of allocation to calculate S_j in the output nodes. This process involves iterative updates c_{ij} using dynamic routing. We can get the S_j of the next layer of capsule through routing and then put S_j into the "squashing" nonlinear function to get the output of the next layer. The entire capsule layer and the process of propagation between them have been completed.

Coupling coefficient c_{ij} is updated and determined iteratively by a dynamic routing process. The sum of the coupling coefficients between capsule i and all capsules in the next level is 1. In addition, c_{ij} is determined by "routing softmax," and b_{ij} in the softmax function is initialized to 0. The softmax of c_{ij} is calculated as [40]

$$c_{ij} = \frac{\exp\left(b_{ij}\right)}{\sum_k \exp(b_{ik})}. \tag{6}$$

b_{ij} depends on the position and type of the two capsules but does not depend on the current input matrix. The consistency between the current output V_j of each capsule j in the subsequent hierarchy can be measured. The prediction vector of the previous capsule i iteratively update the

coupling coefficient with the consistency of the measurement. This paper simply measures this consistency by the inner product as

$$a_{ij} = V_j \cdot \widehat{u}_{j|i}. \tag{7}$$

This part also involves using routing to update the coupling coefficient [40]. The routing process is the update process. It calculates the product of V_j, and $\widehat{u}_{j|i}$ updates b_{ij} by adding it to the original b_{ij} and then uses softmax (b_{ij}, j) to update c_{ij}. When the output V_j is new, it can be updated c_{ij} iteratively, so that the parameters are updated directly by calculating the consistency of the input and output without back propagation.

For all capsule i and capsule j, initialize b_{ij} to equal to zero. The routing algorithm is very easy to converge; basically, it can have a good effect in 3 iterations. c_{ij} is updated through consistent routing. It does not need to be updated according to the loss function, but other convolution parameters and W_{ij} in the entire network need to be updated according to the loss function. In general, these parameters can be updated directly for the loss function using standard back propagation. The expression of this loss function is [40]

$$L_c = T_c \max(0, m^+ - \|v_c\|)^2 + \lambda(1 - T_c) \max(0, \|v_c\| - m^-)^2, \tag{8}$$

where c is the classification category, T_c is the indication function of classification (c exists as 1, and c does not exist as 0), m^+ is the upper boundary, and m^- is the lower boundary. In addition, v_c modulus is the L_2 distance of the vector.

3.2.2. TCN in Capsules TCN Network. TCN has better performance than a baseline recursive architecture in a wide range of sequence modeling tasks. Because these tasks include various benchmarks that are often used to evaluate recurrent network designs, it shows that the recent success of convolutional architectures in applications such as sequence processing is not limited to these areas [41].

TCN is based on two principles: the network produces an output of the same length as the input, and it cannot leak from the future to the past. To complete the first point, TCN uses a one-dimensional full convolutional network architecture, where each hidden layer is the same length as the input layer, and a zero-padding length (kernel size-1) is added to keep the subsequent layers from the previous layers. To achieve the second point, TCN uses causal convolution, and the output at time t is only transformed with elements from current time and earlier layers from the previous layer. It can be found by careful observation that TCN = 1D FCN causal convolution.

The major difference between TCN convolution and ordinary 1D convolution is the use of dilated convolutions. The higher the level, the larger the convolution window, and the more "holes" in the convolution window. More formally, for a 1D sequence input $X \in R^n$ and a filter $f : \{0, \cdots, k-1\} \to R$, the dilated convolution operation F on element s of the sequence is defined as

$$F(s) = (x * df)(s) \sum_{i=0}^{k-1} f(i) * x_s - d * i, \tag{9}$$

where d is the expansion factor, k is the size of the filter, and $x_s - d * i$ represents the past direction [28]. Therefore, expansion is equivalent to introducing a fixed step between every two adjacent filter faucets.

A primitive timing sequence convolution is just able to run back over at a point in time with size linear in the depth of timing sequence of the network. It makes a challenge to put in the mentioned causal convolution for time series, in which a longer history is critical. To acquire an exponentially large receptive field, a good part of the solution is dilated convolution. As illustrated in formula (9), d is the expansion factor. When $d = 1$, the expansion convolution is reduced to regular convolution. In order to figure a broad range of inputs, a larger dilation can be applied at the top level of the output. This ensures that there is a wider scale that expand the receptive field of a convolution within the effective history, meanwhile also extending for a long effective history using deep networks.

Every two such convolutional layers and identity mapping are encapsulated into a residual module (the residual module here is different from ResNet). The residual module contains RelU function, and a fully convolutional layer is used instead of a fully connected layer in the last few layers, as shown in Figure 9(a).

Generally, when using expanded convolution, we will increase d exponentially as the depth of the network increases. When the expansion factor is 1, as shown in Figure 9(b), the expansion convolution degenerates into causal convolution with a receptive field of 2. When the expansion factor is 2, the convolution kernel of the expanded convolution becomes 4. The final output contains all input information. By controlling the expansion factor, the size of the convolution kernel is increased to achieve the purpose of increasing the receptive field.

There are two disadvantages in large-scale neural networks: (1) it is too time-consuming; (2) it is easy to be overfitting. The dropout layer prevents overfitting of the network. Dropout is the process of training the network during deep learning. First, a part of the neural network units is temporarily dropped from the network with a certain probability, which is equivalent to finding a more streamlined network from the original network.

3.3. Superresolution Matrix of Inflow and Outflow Based on GAN. Unlike traditional time series prediction, the result of urban traffic flow prediction is a matrix rather than a simple value. When a high-resolution prediction result is needed, for example, the city is divided into 32×32. Through the two neural network models of the Capsules Network and Temporal Convolutional Network within a minute time level to obtain the final predicted 32×32 time results, it cannot be achieved by the hardware conditions at this stage. Therefore,

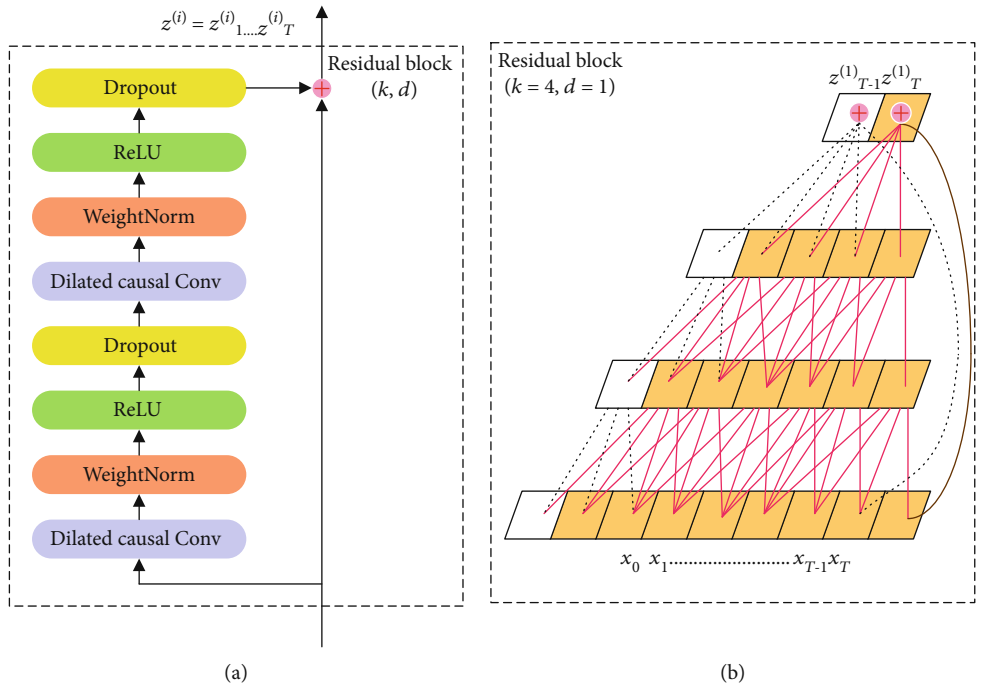


Figure 9: (a) TCN residual block in proposed architectural. (b) TCN residual connection in proposed architectural.

by reducing the resolution (matrix dimension) of the input data, it can achieve an exponential time-saving effect. Using a GAN-based superresolution reconstruction model is reasonable to reconstruct the high-resolution prediction results. Although the accuracy of the prediction result is sacrificed, it can obtain a minute-level high-resolution prediction result under the available hardware conditions. Under the urban traffic command and public safety guarantee scenarios, it is vital to obtain near-accurate results faster, which can provide better support for decision makers to make timely and effective judgments.

Because the amount of data is huge and the calculation is complicated, only the vehicle scene prediction of the experimental scene city in the 16×16 grid is calculated. However, in actual life applications, the experimental scene city is divided into 16×16 grids which is not inadequate. Dividing the city into a finer-grained grid is undoubtedly the solution to this problem. However, the traffic flow at the next interval cannot be predicted in time for more data to be computed.

The superresolution reconstruction reconstructs the city traffic flow a 16×16 experimental scene and obtains a 32×32 traffic flow prediction result. When we want to get better 32×32 fine results, the input data that needs to be processed increases by 4 times, and the overall calculation volume will also increase exponentially. We directly predict 32×32 results based on the predicted 16×16 results based on Generative Adversarial Network (GAN). The overall structure and workflow of traffic superresolution reconstruction of GAN are shown in Figure 10.

Figure 10 shows the structure of the superresolution reconstruction process based on GAN. The 16×16 experimental scene of urban vehicle traffic is used as a low-resolution matrix sequence after convolution layers to form a set of arranged matrixes. This set of matrixes output a 32×32 high-resolution matrix after passing through the GAN.

The process of inputting a convolution layer of a low-resolution matrix is based on the input of a low-resolution matrix of a frame and then convolving the matrix. The training process of the convolutional layer network is the optimization process of the parameters. The spatial transformation can be expressed as

$$I'_{t+k} = T_{\theta_i}(I_{t+k}). \tag{10}$$

The matrix I'_{t+k} represents the high-resolution matrix obtained by transforming $T_{\theta_i}(I_{t+k})$, and the transformation is $T(.)$ [42]. Regarding the loss function of the convolutional layer network, we utilize a regularization method to express it. The optimal parameter estimation process can be expressed as [42]

$$\theta_i^* = \arg_{\theta_i} \min \left\| I_t - I'_{t+k} \right\|_2^2 + \lambda \left\| Q I'_{t+k} \right\|_2^2, \tag{11}$$

among them, θ_i^* represents the parameters of the optimization estimation, λ is a regularization parameter, and Q is a Laplacian. Differentiate the parameters θ_i^* on the right side of formula (11), and make the differentiated result equal to 0. Use the fastest gradient descent method to iteratively solve the equation until the error is less than a preset threshold. The output parameter θ_i^* is the estimated optimal parameter.

The weight representation of the reconstruction network refers to defining a weight for each input low-resolution matrix, then performing weight representation on the input

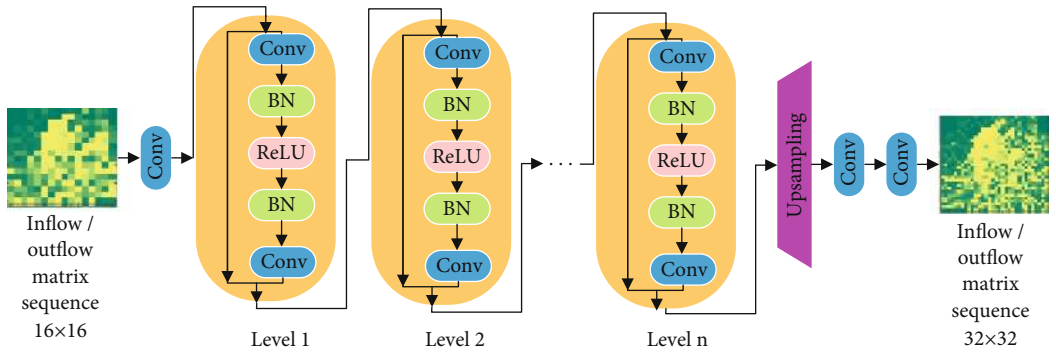


Figure 10: The architecture of superresolution matrix of inflow and outflow based on GAN.

low-resolution matrix to obtain a frame of high-frequency detail information. We add a convolution layer before the generative adversarial reconstruction network to complete the weight representation of the low-resolution matrix after the convolution layer. The mathematical expression of the weight representation can be expressed as [42]

$$X(m,n) = \sum_{k=0}^{K-1} \omega_k(m,n) I_{t+k}(m,n), \tag{12}$$

where $\omega_k(m,n)$ represents the weight value corresponding to the matrix block of the low-resolution matrix sequence. Generally, the same weight is defined for the matrix block. K represents the number of input low-resolution matrixes, (m,n) represents the serial number corresponding to the matrix block, and $(m \in 0, \rightarrow, M-1\,; n \in 0, \rightarrow, N-1)$.

Table 1: Table of dataset.

Dataset	Taxi GPS
Type of data	Taxi GPS track data
Time span	2018/6/10-2019/10/4
Time interval	30 minutes
Map grid size	(16, 16)
Track data	
Number of taxis	5000+
Available time interval	31,724
External factor data	
Holiday	26
The weather	12 types
Temperature	[-14.2, 38.6]
Wind speed	[0, 31.7]

4. Numerical Evaluation and Discussion

4.1. Experimental Data and Preprocessing. In the experimental verification part, the urban taxi dataset (taxi GPS) of the experimental scenario is used, and the data is shown in Table 1.

This article uses the reserve method: (1) the dataset is divided into two disjoint parts, one is the training set and the other is test set; (2) keep the data distribution roughly consistent, similar to stratified sampling; (3) in this paper, the amount of data for one year is used as the training set, and the amount of data for 4 months is used as the validation set. The amount of training set data should account for 75%.

We mainly use historical taxi traffic data prediction to refer to the forecast of rental vehicle traffic data at the future moment. The experiment selects the urban taxi GPS track data from the experimental scenes from June 10, 2018, to June 10, 2019, as the training set, and the remaining data as the test set. In order to facilitate the display and calculation of the results, we select the period from 8:00 to 10:00 AM for analysis.

The grid is divided into 16×16 grids, as shown in Figure 6(a). The GPS trajectory of the taxi is then mapped to the grid area, and a grid area map is developed, as shown in Figure 6(b). The grids represent regions, and the line segments connect the two regions (connected by taxi in this article). The area map actually combines data from the road network and taxi trajectory.

In Keras, learnable parameters are initialized with a uniform distribution with default parameters. The convolution of CapsulesNet *1*st and all TCNs uses 32 filters of size 3×3, and CapsulesNet *2*ed uses the convolution of 2 filters of size 3×3. Each Capsules TCN Network unit consists of 4 TCNs and 2 CapsulesNets. Table 2 for details, there are five additional hyperparameters in Capsules TCN Network.

In our superresolution reconstruction experiment, the 16×16 grid map is also an input as a low-frame image, and a 32×32 grid map is obtained through calculation by a GAN-based traffic prediction network. The magnification of the reconstruction experimental resolution is 2×2. The initial learning rate is set to 10^{-4}, and with each 10,000 iterations, the learning rate drops by 5%. In order to balance the convergence and training time of the network, the maximum number of iterations for superresolution reconstruction is set to 106.

4.2. Experimental Environment and Evaluation Criteria. The experimental verification in our research mainly runs on

Table 2: Table of detail in Capsules TCN architecture.

Network layer	Size	Social/weather feature	Time tendency feature	Time period feature	Geographic space feature
CapsulesNet *1*st	16 × 16	3 × 3, 32	3 × 3, 32	3 × 3, 32	3 × 3, 32
TCN *1*st	16 × 16	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$
TCN *2*ed	16 × 16	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$
TCN *3*rd	16 × 16	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$
TCN *4*th	16 × 16	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 32 \\ 3\times3, 32 \end{bmatrix} \times 2$
CapsulesNet *2*ed	16 × 16	3 × 3, 32	3 × 3, 32	3 × 3, 32	3 × 3, 32

the GPU server, and its detailed information is shown in Table 3.

We use Root Mean Square Error (RMSE) to evaluate the model [43].

$$\mathrm{RMSE} = \sqrt{\frac{1}{Z}\sum_{i}(x_i - x\wedge_i)^2}, \tag{13}$$

where x is the real value and $\hat{x}$ is the corresponding predicted value; Z is the number of all available true values. The RMSE is used to measure the deviation between the observed value and the true value, which is more suitable in this experiment.

Furthermore, in order to measure the quality of the superresolution reconstruction algorithm, evaluation indicators need to be used. The requirements for reconstruction results are different in different application scenarios, so the evaluation standards used are also different. Evaluation methods are generally divided into two categories, one is subjective evaluation and the other is objective evaluation. In objective evaluation, the two most commonly used evaluation indicators are Peak Signal-to-Noise Ratio (PSNR) [44] and Structural Similarity (SSIM) [45].

The specific calculation formula of PSNR [44] is described as follows:

$$\begin{aligned} \mathrm{PSNR} &= 10\log_{10}\frac{\left(2^l - 1\right)^2}{\mathrm{MSE}}, \\ \mathrm{MSE} &= \frac{1}{mn}\sum_{x=1}^{m}\sum_{y=1}^{n}[f\wedge(x,y) - f(x,y)]^2. \end{aligned} \tag{14}$$

MSE is the mean square error, $f(x, y)$ represents the reference matrix. In the experiment, 32×32 grids represent the matrix. It can be known from the formula that when the PSNR of the matrix to be evaluated is larger, the reconstruction result is better.

Table 3: Table of detail in experimental environment.

OS	Windows7
Memory	32 GB
CPU	Intel Core i5-6500 3.20 GHz4 cores
GPU	Nvdia 1080Ti∗2
Software	
CUDA ver.	8.0
CUDNN ver.	8.0
Keras ver.	1.1.1
TensorFlow ver.	1.1.0

The specific calculation formula of SSIM [45] is given as follows:

$$\mathrm{SSIM} = \frac{\left(2\mu_f\mu_{\hat{f}} + C_1\right)\left(2\sigma_{f,\hat{f}} + C_2\right)}{\left(\mu_f^2 + \mu_{\hat{f}}^2 + C_1\right)\left(\sigma_f^2 + \sigma_{\hat{f}}^2 + C_2\right)}, \tag{15}$$

where μ_f is the average value of the reference matrix, $\mu_{\hat{f}}$ represents the average value of the matrix to be evaluated, σ_f is the variance of the reference matrix, and $\sigma_{\hat{f}}$ is the variance of the matrix to be evaluated.

4.3. Effect of Hyperparameters on Experimental Results. The number of CapsulesNet has an effect on the taxi GPS dataset experiments, as shown in Figure 11(a). The network depth also greatly affects the experimental results. As shown in Figure 11(b), the number of TCN increases; the RMSE of the model fluctuates. It indicates that the network is not the deeper the better, because it captures not only close-space dependencies but also far-space dependencies. When the network is very deep (such as when the number is 15), training becomes very difficult. Based on the above comparison, the number of CapsulesNet is two, and the number of TCNs is set to four.

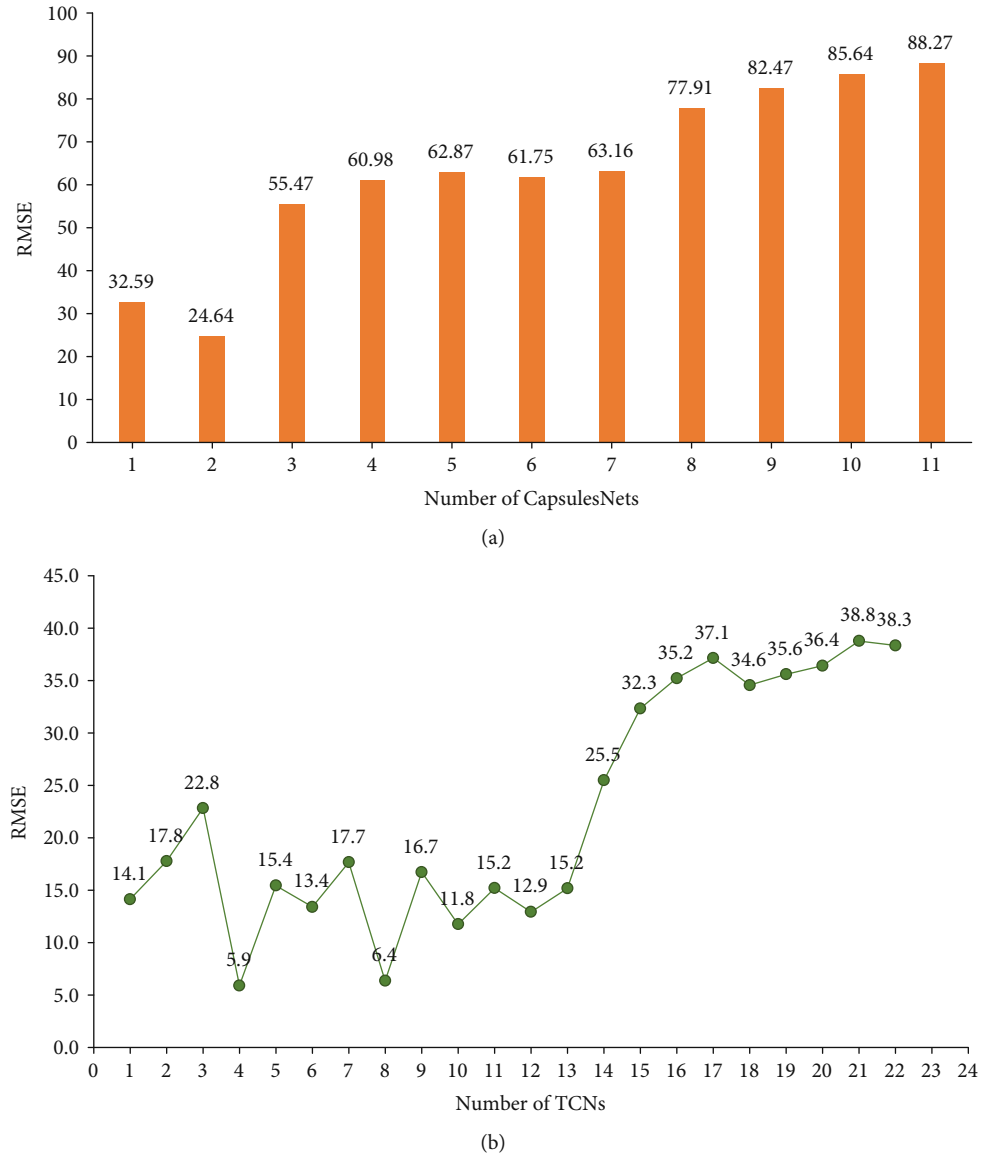


Figure 11: (a) Effect of the number of CapsulesNets on experimental results. (b) Effect of the number of TCNs on experimental results.

4.4. Experimental Results and Analysis. By comparing with historical average (HA), autoregressive integrated moving average (ARIMA), recurrent neural network (RNN), and long-term short-term memory (LSTM) network, the validity of the Capsules TCN Network model for urban area traffic flow prediction is verified.

HA predict the inflow and outflow of people based on the historical average of inflow and outflow at the same time and area in the past. For example, to predict the inflow of a region from 10:00 to 10:30 AM this Thursday morning, calculate the average of the inflow from 10:00 to 10:30 AM every Thursday morning in this region.

ARIMA is a well-known model for understanding and predicting future values in a time series. In the traditional linear model, the autoregressive integrated moving average model has been widely used in passenger flow prediction. It is a general formula for autoregressive (AR) models, integral (I) models, or moving average (MA) models.

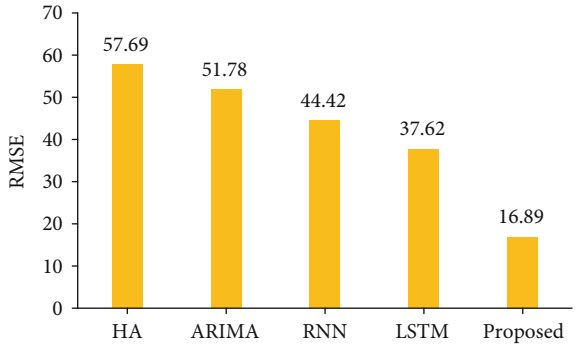


Figure 12: Comparison of the proposed algorithm with traditional existing algorithms.

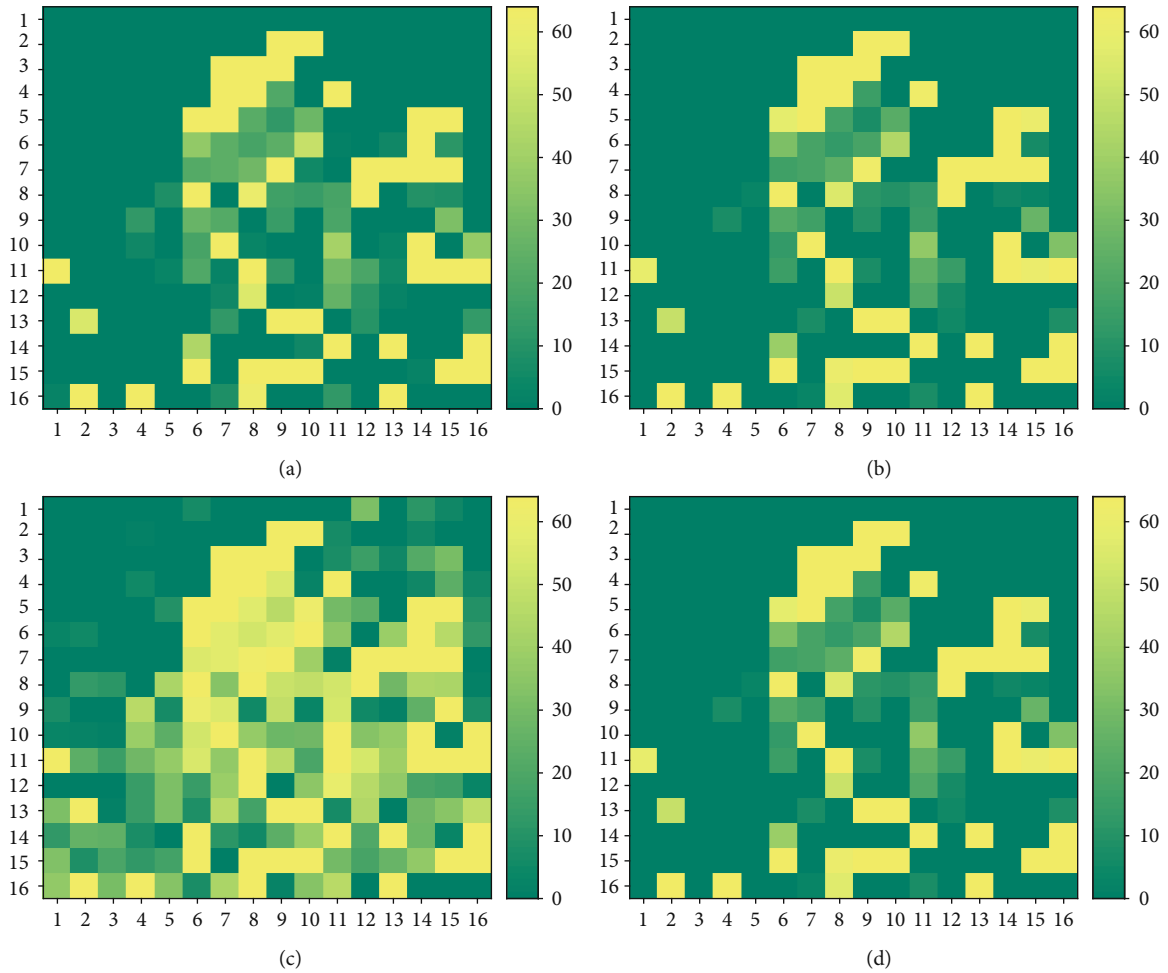


Figure 13: The prediction results of the algorithm in 4 different times (a) PM 8:00. (b) PM 8:30. (c) PM 9:00. (d) PM 10:00.

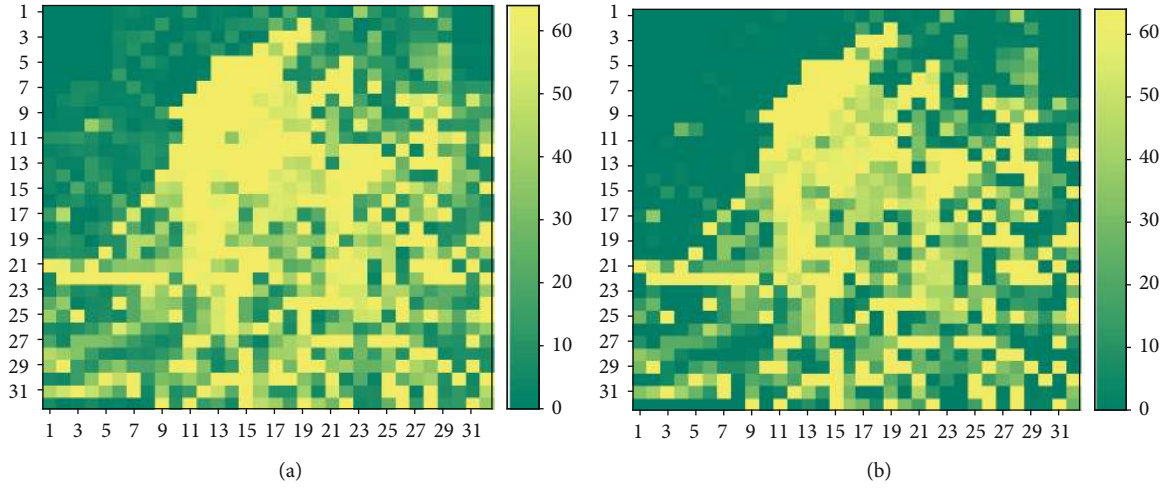


Figure 14: (a) Results of 32×32 matrix based on superresolution GAN reconstruction according to 16×16 matrix of Figure 6(b). (b) Real 32×32 matrix based on input data according to 16×16 matrix of Figure 6(b).

RNN is a deep-learning model that captures time sequence dependencies. Formally, RNNs can train sequences of any length [46].

LSTM is a special RNN that can learn long-term time dependencies [47].

We compare the RMSE between the Capsules TCN Network and the true value and then compare it with other prediction models to verify the validity of Capsules TCN Network. The results are shown in Figure 12. According to the comparison results, it can be seen that the proposed Capsules TCN Network has smaller RMSE. It has higher prediction accuracy than other models, indicating the effectiveness of the proposed Capsules TCN Network for traffic prediction tasks.

Figure 13 shows the spatial-temporal distribution of taxi traffic during the morning rush hour at 8:30-10:00 AM on September 30, 2018. From the results in Figure 13, it can be seen that the proposed Capsules TCN Network better grasps the spatiotemporal characteristics of the changes in taxi traffic and makes predictions with sufficient accuracy.

The experimental results of superresolution matrix of inflow and outflow based on GAN are also demonstrated. Figure 14(a) is the result of superresolution reconstruction based on GAN, and Figure 14(b) is the real value, when the input low-resolution matrix is used from Figure 6(b). According to experiments, it can be seen subjectively that GAN-based superresolution reconstruction has achieved good reconstruction results and is close to the real value visually. The objective assessment is as follows: PSNR is 33.844 and SSIM is 0.93. We also obtain the PSNR and SSIM of superresolution matrix of inflow and outflow based on GAN in 64×64 and 128×128, respectively. In 64×64 scene, PSNR is 28.94 and SSIM is 0.88. In 128×128 scene, PSNR is 22.75 and SSIM is 0.79.

5. Conclusions

Traffic forecasting has been a core issue in transportation planning and management, and it has also been a major issue in urban computing. The prediction of traffic volume can help the development of urban traffic safety, and traffic flow will be more order. We propose a method based on the Capsules Network and Temporal Convolutional Network to predict traffic flow in local areas of the city. This method is called Capsules TCN Network. The Capsules TCN Network model can learn the spatial dependence, time dependence, and external factors of traffic flow prediction. We evaluated the GPS track data of urban taxis in the experimental scenarios and verified that the model has a good applicability in vehicle traffic prediction. Because the accuracy of regional traffic flow is different in different scenarios, we propose a GAN-based superresolution reconstruction model of traffic flow to improve the accuracy of Capsules TCN Network model results. The experimental results show that the GAN-based traffic superresolution reconstruction model not only has a better subjective visual effect but also has more prominent objective evaluation indicators.

Data Availability

The dataset used in this article is from a commercial company. If you need the dataset used in this study, you can send a usage request to centaureacyanus@foxmail.com. After being authorized by the company, the dataset will be transmitted to the applicant in the form of an email attachment.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This project is supported by The Science and Technology Funds from Liaoning Education Department (No. LQ2017 008), the Doctoral Research Startup Fund Project of Liaoning Province (No. 2016011968), and the China Postdoctoral Science Foundation (No. 2019M661096).

References

[1] W. Wang, J. Chen, J. Wang, J. Chen, and Z. Gong, "Geography-aware inductive matrix completion for personalized Point-of-Interest recommendation in smart cities," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4361–4370, 2020.

[2] W. Wang, J. Chen, J. Wang, J. Chen, J. Liu, and Z. Gong, "Trust-enhanced collaborative filtering for personalized point of interests recommendation," *IEEE Transactions on Industrial Informatics*, p. 1, 2019.

[3] J. Liu, T. Li, P. Xie, S. Du, F. Teng, and X. Yang, "Urban big data fusion based on deep learning: an overview," *Information Fusion*, vol. 53, pp. 123–133, 2020.

[4] Z. Pan, Y. Liang, W. Wang, Y. Yu, Y. Zheng, and J. Zhang, "Urban traffic prediction from spatio-temporal data using deep meta learning," in *KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1720–1730, Anchorage, AK, USA, July 2019.

[5] R. Jiang, X. Song, D. Huang et al., "DeepUrbanEvent: a system for predicting citywide crowd dynamics at big events," in *KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2114–2122, Anchorage, AK, USA, July 2019.

[6] P. Xie, T. Li, J. Liu, S. Du, X. Yang, and J. Zhang, "Urban flow prediction from spatiotemporal data using machine learning: a survey," *Information Fusion*, vol. 59, pp. 1–12, 2020.

[7] R. Li, Z. Zhao, X. Zhou et al., "Intelligent 5G: when cellular networks meet artificial intelligence," *IEEE Wireless Communications*, vol. 24, no. 5, pp. 175–183, 2017.

[8] J. Pérez-Romero, O. Sallent, R. Ferrús, and R. Agustí, "Artificial intelligence-based 5G network capacity planning and operation," in *2015 International Symposium on Wireless Communication Systems (ISWCS)*, pp. 246–250, Brussels, Belgium, August 2015.

[9] Y. Bi, C. Lin, H. Zhou, P. Yang, X. Shen, and H. Zhao, "Time-constrained big data transfer for SDN-enabled smart city," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 44–50, 2017.

[10] D. Zou, S. Li, X. Kong, H. Ouyang, and Z. Li, "Solving the combined heat and power economic dispatch problems by an improved genetic algorithm and a new constraint handling strategy," *Applied Energy*, vol. 237, pp. 646–670, 2019.

[11] X. Shen, D. Zou, N. Duan, and Q. Zhang, "An efficient fitness-based differential evolution algorithm and a constraint handling technique for dynamic economic emission dispatch," *Energy*, vol. 186, p. 115801, 2019.

[12] M. Yao, M. Sohul, V. Marojevic, and J. H. Reed, "Artificial intelligence defined 5G radio access networks," *IEEE Communications Magazine*, vol. 57, no. 3, pp. 14–20, 2019.

[13] C. Lin, Y. Bi, H. Zhao, Z. Liu, S. Jia, and J. Zhu, "DTE-SDN: a dynamic traffic engineering engine for delay-sensitive transfer," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 5240–5253, 2018.

[14] Y. Fu, S. Wang, C. X. Wang, X. Hong, and S. McLaughlin, "Artificial intelligence to manage network traffic of 5G wireless networks," *IEEE Network*, vol. 32, no. 6, pp. 58–64, 2018.

[15] D. Zou, S. Li, X. Kong, H. Ouyang, and Z. Li, "Solving the dynamic economic dispatch by a memory-based global differential evolution and a repair technique of constraint handling," *Energy*, vol. 147, pp. 59–80, 2018.

[16] M. S. Ahmed and A. R. Cook, *Analysis of freeway traffic time-series data by using Box-Jenkins techniques (No. 722)*, Transportation Research Board, 1979.

[17] M. Levin and Y. D. Tsao, "On forecasting freeway occupancies and volumes (abridgment)," *Transportation Research Record*, no. 773, pp. 47–49, 1980.

[18] M. M. Hamed, H. R. Al-Masaeid, and Z. M. B. Said, "Short-term prediction of traffic volume in urban arterials," *Transportation Engineering*, vol. 121, no. 3, pp. 249–254, 1995.

[19] M. Van Der Voort, M. Dougherty, and S. Watson, "Combining Kohonen maps with ARIMA time series models to forecast traffic flow," *Transportation Research Part C: Emerging Technologies*, vol. 4, no. 5, pp. 307–318, 1996.

[20] S. Lee and D. B. Fambro, "Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting," *Transportation Research Record*, vol. 1678, no. 1, pp. 179–188, 1999.

[21] B. M. Williams, "Multivariate vehicular traffic flow prediction: evaluation of ARIMAX modeling," *Transportation Research Record*, vol. 1776, no. 1, pp. 194–200, 2001.

[22] Y. Kamarianakis and P. Prastacos, "Forecasting traffic flow conditions in an urban network: comparison of multivariate and univariate approaches," *Transportation Research Record*, vol. 1857, no. 1, pp. 74–84, 2003.

[23] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: theoretical basis and empirical results," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[24] B. Ghosh, B. Basu, and M. O'Mahony, "Multivariate short-term traffic flow forecasting using time-series analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 2, pp. 246–254, 2009.

[25] G. A. Davis and N. L. Nihan, "Nonparametric regression and short-term freeway traffic forecasting," *Journal of Transportation Engineering*, vol. 117, no. 2, pp. 178–188, 1991.

[26] H. Chang, Y. Lee, B. Yoon, and S. Baek, "Dynamic near-term traffic flow prediction: system-oriented approach based on past experiences," *IET Intelligent Transport Systems*, vol. 6, no. 3, pp. 292–305, 2012.

[27] N. E. El Faouzi, "Nonparametric traffic flow prediction using kernel estimator," in *Transportation and traffic theory: proceedings of the 13th International Symposium on Transportation and Traffic Theory*, Lyon, France, July 1996.

[28] H. Sun, H. X. Liu, H. Xiao, R. R. He, and B. Ran, "Use of local linear regression model for short-term traffic forecasting," *Transportation Research Record*, vol. 1836, no. 1, pp. 143–150, 2003.

[29] Y. S. Jeong, Y. J. Byon, M. M. Castro-Neto, and S. M. Easa, "Supervised weighting-online learning algorithm for short-term traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1700–1707, 2013.

[30] K. Y. Chan, T. S. Dillon, J. Singh, and E. Chang, "Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and Levenberg–Marquardt algorithm," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 644–654, 2012.

[31] K. Kumar, M. Parida, and V. K. Katiyar, "Short term traffic flow prediction for a non urban highway using artificial neural network," *Procedia-Social and Behavioral Sciences*, vol. 104, pp. 755–764, 2013.

[32] W. Zheng, D. H. Lee, and Q. Shi, "Short-term freeway traffic flow prediction: Bayesian combined neural network approach," *Journal of Transportation Engineering*, vol. 132, no. 2, pp. 114–121, 2006.

[33] M.-C. Tan, S. C. Wong, J.-M. Xu, Z.-R. Guan, and P. Zhang, "An aggregation approach to short-term traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 1, pp. 60–69, 2009.

[34] S. A. Zargari, S. Z. Siabil, A. H. Alavi, and A. H. Gandomi, "A computational intelligence-based approach for short-term traffic flow prediction," *Expert Systems*, vol. 29, no. 2, pp. 124–142, 2012.

[35] M. Cetin and G. Comert, "Short-term traffic flow prediction with regime switching models," *Transportation Research Record*, vol. 1965, no. 1, pp. 23–31, 2006.

[36] Y. Zhisheng, S. Chunfu, X. Zhihua, and Y. Hao, "Short-term traffic flow prediction of road networks based on principal component analysis and support vector machines," *Journal of Jilin University*, vol. 38, no. 1, pp. 48–52, 2008.

[37] L. Cunjun, R. Yang, and Z. Jiashu, "Traffic flow forecasting method based on wavelet analysis," *Computer Application*, vol. 23, no. 12, pp. 7-8, 2003.

[38] S. Yan, C. Senfa, and Z. Zhenguo, "Application of grey system theory to traffic flow prediction at detector-free intersections," *Journal of Southeast University: Natural Science Edition*, vol. 32, no. 2, pp. 256–258, 2002.

[39] X. Weiqing, Y. Xiaobo, J. Shouxu, and L. Zhijun, "Short-term traffic flow prediction based on BP neural network and fuzzy inference system," *Intelligent Computers and Applications*, vol. 5, no. 2, pp. 43–46, 2015.

[40] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in neural information processing systems*, pp. 3856–3866, Curran Associates, Inc, 2017.

[41] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, http://arxiv.org/abs/1803.01271.

[42] S. Lian, H. Zhou, and Y. Sun, "FG-SRGAN: a feature-guided super-resolution generative adversarial network for unpaired image super-resolution," in *Advances in Neural Networks* –

ISNN 2019. ISNN 2019. Lecture Notes in Computer Science, vol 11554, H. Lu, H. Tang, and Z. Wang, Eds., Springer, Cham, 2019.

[43] T. Chai and R. R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)?–arguments against avoiding RMSE in the literature," *Geoscientific Model Development*, vol. 7, no. 3, pp. 1247–1250, 2014.

[44] A. Hore and D. Ziou, "Image quality metrics: PSNR vs. SSIM," in *2010 20th International Conference on Pattern Recognition*, pp. 2366–2369, Istanbul, Turkey, August 2010.

[45] S. S. Channappayya, A. C. Bovik, and R. W. Heath Jr., "Rate bounds on SSIM index of quantized images," *IEEE Transactions on Image Processing*, vol. 17, no. 9, pp. 1624–1639, 2008.

[46] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning. Book in preparation for MIT Press*, 2016, http://www.deeplearningbook.org.

[47] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.