Received November 17, 2017, accepted December 17, 2017, date of publication December 27, 2017,
date of current version February 28, 2018.

Digital Object Identifier 10.1109/ACCESS.2017.2787696

Interactive Temporal Recurrent Convolution Network for Traffic Prediction in Data Centers

XIAOFENG CAO[1], YUHUA ZHONG[2], YUN ZHOU[1], JIANG WANG[1], CHENG ZHU[1], AND WEIMING ZHANG[1]

[1]Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha 410073, China
[2]College of Mechatronics and Automation, National University of Defense Technology, Changsha 410073, China

Corresponding author: Yun Zhou (zhouyun@nudt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 71571186, Grant 61703416, and Grant 71471176 and in part by the MOE Research Center for Online Education under Grant 2017YB119.

ABSTRACT Accurately predicting future service traffic would be of great help for load balancing and resource allocation, which plays a key role in guaranteeing the quality of service (QoS) in cloud computing. With the rapid development of data center, the large-scale network traffic prediction requires more suitable methods to deal with the complex properties (e.g., high-dimension, long-range dependence, non-linearity, and so on). However, due to the limitations of traditional methods (e.g., strong theoretical assumptions and simple implementation), few research works could predict the large-scale network traffic efficiently and accurately. More importantly, most of the studies took only the temporal features but without the services' communications into consideration, which may weaken the QoS of applications in the data center. To this end, we applied the gated recurrent unit (GRU) model and the interactive temporal recurrent convolution network (ITRCN) to single-service traffic prediction and interactive network traffic prediction, respectively. Especially, ITRCN takes the communications between services as a whole and directly predicts the interactive traffic in large-scale network. Within the ITRCN model, the convolution neural network (CNN) part learns network traffic as images to capture the network-wide services' correlations, and the GRU part learns the temporal features to help the interactive network traffic prediction. We conducted comprehensive experiments based on the Yahoo! data sets, and the results show that the proposed novel method outperforms the conventional GRU and CNN method by an improvement of 14.3% and 13.0% in root mean square error, respectively.

INDEX TERMS Network traffic prediction, interactive traffic representation, interactive temporal recurrent convolution network, gated recurrent unit, convolution neural network.

I. INTRODUCTION

As cloud computing continues to grow in size and complexity, many different services (e.g., Email, Video, etc.) compete for the shared network and thus suffer the significant unpredictability of performance. Service traffic prediction is an essential need to make better route choices and resource allocation, which results in the performance guarantee and the QoS (Quality of Service) improvement.

More importantly, rather than predicting the single service traffic (temporal), understanding the communication pattern of the entire service space can capture real traffic interaction behaviors among different services (interactive), which may help guide resource managers use more efficient load balancing strategies, and therefore make better use of computing resources.

However, for a network with thousands of servers, capturing the temporal and interactive traffic features of any service is very difficult. Specifically, from the perspective of temporal analysis, a strong correlation usually exists among traffic series, where previous traffic conditions likely have a large impact on future service traffic, which is called Long Range Dependency (LRD) in traffic prediction [1]–[3]. Meanwhile, from the perspective of interactive analysis, some network services often communicate with each other, while others seldom send messages to peers.

In brief, the large-scale traffic prediction needs more representational capabilities for prediction models, such as the capability to extract the LRD in network traffic sequences, the capability to deal with the huge amount high-dimensional data, and the capability to deal with high-computational situation of the non-linear data.

Unfortunately, traditional network traffic prediction methods cannot provide those capabilities due to their strong theoretical assumptions, simple implementation, etc. They cannot adapt to the changing traffic data and once the data do not meet the assumptions, the prediction accuracy could be very poor [4], [5]. Thus, traffic prediction methods have been gradually shifting from traditional statistical models to Computational Intelligence (CI) approaches due to their strong representational capability of dealing with non-linearity of input data. Although CI approaches exhibit a superior capability of modeling non-linear time series problem in an effective fashion, we still have several issues to be addressed before use the CI approach, such as the difficulties in dealing with high-dimensional data and LRD property.

To fill this gap, we construct a novel end-to-end model called Interactive Temporal Recurrent Convolution Network (ITRCN), which is an image-based approach configured mainly by deep learning architectures of Convolution Neural Networks (CNNs) and Gated Recurrent Units (GRUs) Neural Network. Inspired by the motion prediction in the domain of computer vision [6], we represent the service traffic as images, and take network-wide communications between different services as an entirety to predict the future traffic directly. Within the ITRCN model, CNNs are utilized to mine the interactive features among all service pairs in the entire network that adopts layers with convolution filters to extract local features through sliding windows, whereas GRUs are employed to capture the temporal features of service traffic sequences.

To implement this method, we are faced with two major challenges. The first challenge is the transformation from network-wide traffic matrices to a collection of images. Since the network traffic of communications between each pair of services can be formulated as traffic matrices, it can be further converted into one-channel images. The more salient imaged features will be more easily to be captured by the prediction model, resulting in higher prediction accuracy [7]. To construct salient imaged features, we calculate the pair correlation of each service and put the strong correlation services pair on the neighborhood area of traffic matrix, finally transform matrices into one-channel images.

The second challenge is how to build a model that can extract both the temporal and interactive features from the sequence of images. ITRCNs combine CNNs and GRUs to solve the interactive network traffic prediction problem. The ITRCN model converts the network traffic between different services into images, and employs the CNN part to extract global interactive features. Then, these output vectors are put into the GRU part to learn the temporal features. Consequently, the ITRCN model can make an effective prediction of the complicated interactions among all services traffic.

The contributions of this paper can be summarized as follows:

(a) So far, to our best of knowledge, this is the first work that employs the image-based method to network traffic prediction in large-scale data centers. It is different from existing deep learning methods in the network prediction that only take the time series data as model inputs, but ignore the correlation between different traffic sequences.

(b) We construct a novel end-to-end model, namely ITRCN, which combines CNNs and GRUs deep learning approaches. The temporal evolutions and interactive correlations are both considered in this hybrid model. More specifically, we use the entire transformed image as the input and can directly estimate the future demands in large-scale network.

(c) We conduct a comprehensive experiment on real-world network traffic traces of Yahoo! data centers. For the non-interactive network traffic, GRU made higher accuracy in most services than other prevailing methods. For the interactive network traffic, the result showed that our model outperforms the CNN and GRU method by 14.3% and 13.0% in RMSE respectively.

The rest of the paper is organized as follows. Section II gives an overview of the related work. Section III models the problem and describes how to convert network matrices into images. The model is shown in Section IV and the data description as well as extended data analysis are shown in Section V. In Section VI, we present the experiment results and comparisons. Finally, we conclude this paper and outline future works in Section VII.

II. RELATED WORK

Many works indicated that the network traffic always carry the properties of LRD and self-correlation [2], [8], [9]. Unlike these two properties exhibiting in all the trace, non-linearity is also an important feature to most of network traffic in the complex network environment nowadays [10]. These characteristics make it difficult to achieve an accurate prediction for network traffic, which has attracted numerous attentions from researchers for a long time, and many works of them were presented.

Traditional network prediction methods, such as the Auto Regressive (AR) [11], the Auto Regressive Moving Average (ARMA) [12] and the Auto Regressive Integrated Moving Average (ARIMA) [12] are most common models applied to network traffic prediction problems, of which models can capture inherent correlations and the short range dependence, via the moving average and the autoregression respectively. KuanHoong *et al.* [13] applied the ARMA model to the prediction of six days real data from bit torrent application point to point network. Sadek and Khotanzad [14] predicted the high speed network traffic of different applications by the K-factor ARMA model. Yu *et al.* [15] applied the ARIMA model to predict the mobile network data from Heilongjiang province in China. Due to the difficulty of extracting LRD features, these models cannot achieve a high accuracy in prediction. Therefore, the Fractional ARIMA (FARIMA) model, which has the ability to capture such dependence, is introduced in many works, such as the Wireless network traffic and video network traffic prediction [16], [17]. However, all the methods listed above are based on the oversimplified theoretical assumption (e.g., linearity) with the simple implementation, and therefore cannot capture features of non-linear traffic data effectively in large-scale networks.

Hence, some researchers gradually shifted their attentions to the models which are more flexible and suitable for the complex non-linear data. Support Vector Machine (SVM) is one of the most popular prediction models. Bermolen and Rossi [18] explored the use of the SVM model in network link load forecasts. The results showed that SVM model exhibits good robustness and flexibility in network prediction, but consumes long time and large computer memory when dealing with the big data, thus it might not suitable for the large-scale network. Neural Network model is another widely used model in the network traffic prediction because of its advantages of the generalization capability, the strong forecasting capability, the ability of processing the high dimensional and big amount data. Cortez *et al.* [19] used the ANN to forecast the amount of traffic in the TCP/IP based network. Katris and Daskalaki [20] also applied the ANN to predict 9 different network traffic datasets. Due to the simplicity, the ANN achieved lower accuracy than deep learning methods, thus many researchers applied more advanced and deeper models. Park [5] employed the BiLinear Recurrent Neural Network (BLRNN) to the real-world Ethernet network traffic data set, and the experimental result showed a high accuracy. Although deep learning models present complex architectures and achieve better results than other prevailing methods, they only focus on the temporal correlation prediction of independent network traffic flows, ignoring the interactive correlations or communications between different flows, and still hard to deal with the co-existence of LRD and non-linearity.

With the development of Internet, the prediction on the certain network traffic or simple links is unable to provide enough useful information for the network management. While the network traffic matrix prediction is able to solve this problem because traffic matrices reflect the values of traffic flows between different sources and destinations in networks, which can provide rich information and global view of how the traffic flow transmit in the networks. Vardi [21] firstly proposed tomography methods to traffic matrix predictions, which modeled each traffic flow as the independent and identically-distributed Poisson model and applied the EM algorithm to solve this problem. Soule *et al.* [22] proposed a linear dynamic system, which applied the Kalman filter to predict the traffic matrices in networks, and evaluated by the real network data from Tier-1 ISP. Unfortunately, due to the the complex statistical inference process in the prediction, these models are unable to deal with the big data in large-scale networks. In order to overcome the complicated computation, the Long Short-Term Memory (LSTM) structure was introduced to the traffic matrices prediction where Azzouni and Pujolle proposed a LSTM end-to-end method [4], which outperformed the traditional prediction methods and achieved a higher accuracy. However, due to the aforementioned complex properties of the network traffic and limited methods on matrix predictions, the previous researches still cannot comprehensively capture temporal correlations and interactive correlations at the same time.

III. PROBLEM FORMULATION

A. PROBLEM DESCRIPTION

We assume there are multiple data centers, each of which connects to several other Internet service providers (ISPs) to reach its clients through border routers. These data centers can provide clients multiple services, including Email, Messenger, News, Music, Video, etc. Different services are distributed across different data centers and can be identified by the ports of transport layer. The network traffic volumes of each service always change over time (temporal), and different types of services are also likely to interact with each other (interactive) [23].

For temporal features, the traffic of each service is viewed as one-dimensional time series. Suppose that the history traffic is $x = (x_t, x_{t-1}, x_{t-2}, \ldots, x_{t-n})$, where n is the time lag before current time t. To predict the next moment traffic x_{t+1}, a natural way is to find a map function f between the history traffic and future traffic,

$$x_{t+1} = f(x_t, x_{t-1}, x_{t-2}, \ldots, x_{t-n}) \tag{1}$$

Due to the complicated characteristics of the network traffic, we choose the GRU model, which is one of the most powerful deep learning model for the time series prediction, to solve self-correlation, LRD and non-linearity in this paper. [24].

For interactive features, services are likely to communicate with others. And these traffic of communications are strongly correlated [23]. Moreover, within the data center, different ports are corresponding to different services. Thus the congestion, caused by the service communication, in one port not only affects its own service but also may propagate to other far-side ports' services. To capture the communication of each service (port) pair, we use the interactive traffic matrix to represent the exchanged traffic from the resource port (indexed by the rows) to the destination port (indexed by the columns). The matrix at a certain time is mathematically denoted as:

$$M(t) = \begin{bmatrix} m_{11} & m_{12} & \cdots & m_{1n} \\ m_{21} & m_{22} & \cdots & m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \cdots & m_{nn} \end{bmatrix} \tag{2}$$

where n is the number of ports served for the services. Motivated by the video prediction and motion prediction in the computer vision [6], we model the interactive traffic matrix of each time interval as the image of each frame in the video or motion, and then use it as the input of CNN model for prediction [25]. Details about how to transform the interactive traffic matrices into images is discussed in the next section.

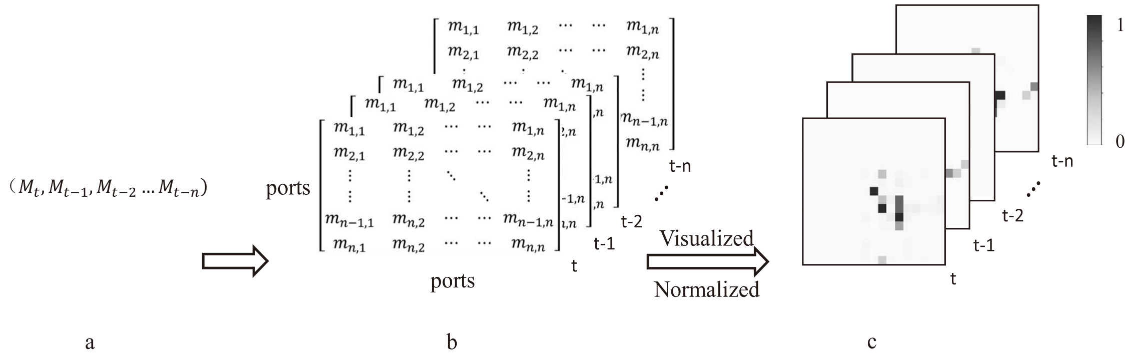

$(M_t, M_{t-1}, M_{t-2} \dots M_{t-n})$
ports
ports
$m_{1,1}$ $m_{1,2}$ ⋯ ⋯ $m_{1,n}$
$m_{2,1}$ $m_{2,2}$ ⋯ ⋯ $m_{2,n}$
$m_{n-1,1}$ $m_{n,2}$ ⋯ ⋯ $m_{n-1,n}$
$m_{n,1}$ $m_{n,2}$ ⋯ ⋯ $m_{n,n}$
t
t-1
t-2
t-n
Visualized
Normalized
1
0
a
b
c

FIGURE 1. Traffic-image conversion. Interactive traffic between services could be formulated as traffic matrix (shown in the subfigure b). Traffic matrix can be visualized into images and further scaled to [0,1] (shown in the subfigure c), where non-interactive traffic is set to white.

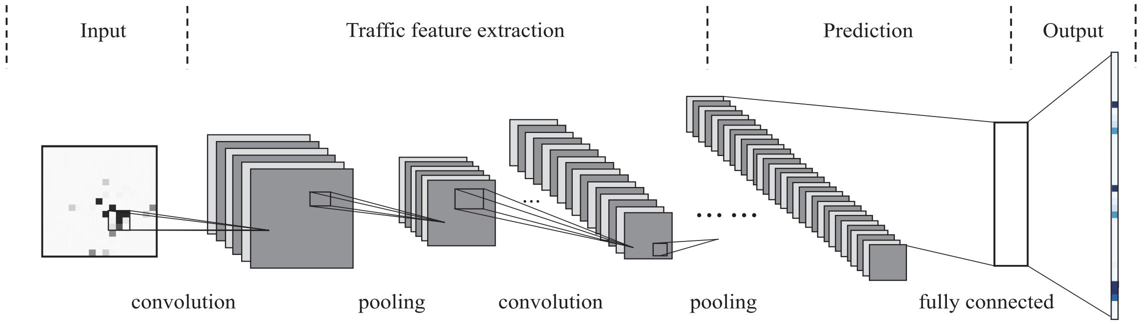


FIGURE 2. The typical architecture of CNN with convolution layers, pooling layers and full-connected layer.

B. CONVERTING TRAFFIC MATRICES TO IMAGES

The interactive traffic matrices can be converted into the one-channel images, which are valued by the network traffic data of different services. Fig. 1 illustrates the converting process between the raw service traffic, the traffic matrix and the final image. Firstly, the raw network traffic of different services is transformed into the interactive traffic matrices, each of whose element represents the traffic value exchanged between certain services. Secondly, the values of the matrices are normalized and scaled to (0, 1). Finally, we serve matrices as images, where the color of non-interactive traffic is set to white and the highest value is set to the black color (as shown in Fig. 1c).

Distinct features and strong spatial correlations can be seen equally as the simple features in the images, which could be easily and accurately extracted by the shallow layers of the CNNs [26]. In order to enhance the model's performance in feature extracting, it is necessary to calculate the correlation coefficient between the network traffic of different services' port pairs in advance, and put the strong correlated services' port pairs in the neighborhood area of the traffic matrix to construct the salient features [27]. More details will be discussed in the experiment.

IV. METHODOLOGY

Our novel ITRCN model is configured mainly by deep learning architectures of CNN and GRU, which converts the network traffic matrices of each time interval into the collection of one-channel images, and takes the global network traffic as an entirety to extract the temporal and interactive correlations simultaneously.

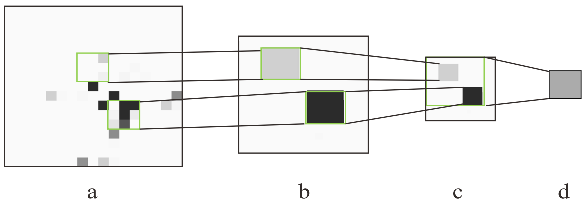


FIGURE 3. An illustration of interactive traffic features captured by CNN model.

A. CNN FOR CAPTURING INTERACTIVE FEATURES

Though services in data center can be categorized into different groups, some of them are strongly correlated due to communication behaviors. To capture the interactive features among services, we construct CNNs to extract features of traffic images. Fig. 2 shows the structure of CNNs with four parts and three kinds of layers. The four parts are Input, Feature extraction, Prediction, and Output part.

Fig. 3 shows the process of extracting image features in network traffic images. The CNN model first learns the local salient correlations as the lower features, and abstracts these features together to the higher level (as shown in subfigures a and b). With the number of model layers increases, the global correlations will be extracted and higher features can be learned (as shown in subfigures c and d).

The core part of the CNN model is the feature extracting layer combining convolution and pooling layers. Convolution layer has many convolution filters, all of which can extract and group the lower features into higher and more abstract network traffic features. The process can be described as:

$$o_l^k = \sigma\Big(\sum_{j=1}^{c_{(l-1)}} (W_l^{jk} o_{(l-1)}^j + b_l^k)\Big) \tag{3}$$

where the input, output, weights and additive bias of the l^{th} convolution layer are denoted as o_{l-1}^j, o_l^k, W_l^{kj} and b_l^k respectively, and j, k are the indices of the convolutional filters, c_{l-1} is the number of convolutional filters in $(l-1)^{th}$ feature extracting layer, and σ is the activation function.

Pooling layers are used to downsample and aggregate the data in the neighborhood region, so as to reduce the scale of the network structure and extract the most salient features. The function can be written as:

$$O_l^k = \sigma(\beta_l^k down(o_{(l-1)}^k) + b_l^k) \tag{4}$$

where β is the multiplicative bias. $down(\cdot)$ represents the downsampling function. We use the Max Pooling operation as the downsampling function, which defines a spatial neighborhood (2x2 in this paper) that slides on the feature map with stride 2, and take the largest element from each neighborhood region. Fig. 4 shows an example of Max Pooling operation on a feature map.

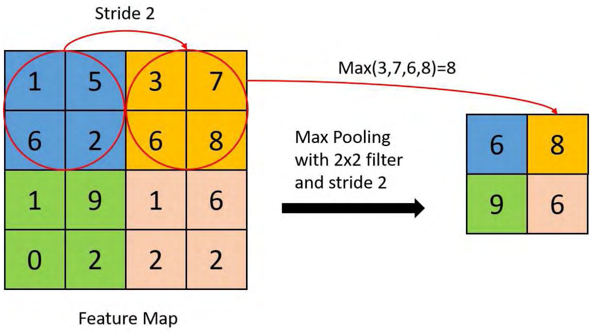


FIGURE 4. An illustration of Max Pooling operation.

Finally the features extracted by the pooling layers and convolution layers are transformed into one dimension vectors by the fully-connected layer, and taken as the input of GRUs. These features captured by CNNs are significant for prediction on interactive pattern.

B. GRU FOR CAPTURING TEMPORAL DEPENDENCY

Services in data centers have distinct temporal dependencies, and the earlier traffic even have a long-term impact on the current traffic. In fact, effectively extracting temporal features is the key to improve the prediction accuracy. One of the most successful methods for capturing temporal dependency is the GRU [24].

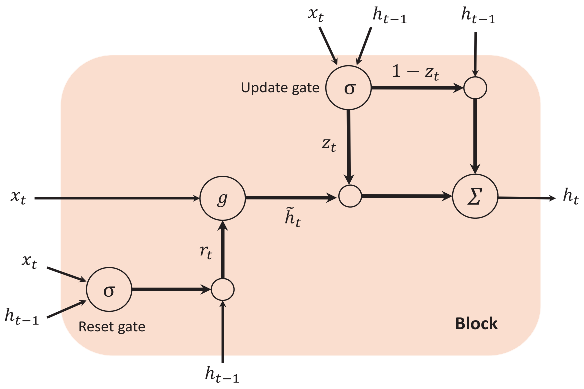


FIGURE 5. The typical architecture of GRU with two gates: reset gate and update gate.

The GRU is a special type of the RNN, where recurrent units can adaptively capture the dependencies of different time scales. Compared with the traditional RNN, the GRU avoids vanishing and exploding gradients problems, which also has been shown to be more effective than LSTM [28], [29]. Fig. 5 shows the structure of the GRU with two gating units, namely, the update gate z_t and the reset gate r_t. These gating units are used to modulate flows of information inside the units. The steps of the GRU are formulated as:

$$\begin{aligned} r_t &= \sigma(W_r x_t + U_r h_{t-1} + b_r) \\ z_t &= \sigma(W_z x_t + U_z h_{t-1} + b_z) \\ \tilde{h_t} &= g(W_h x_t + U_h(r_t \odot ht - 1) + b_h) \\ h_t &= (1 - z_t)h_{t-1} + z_t \tilde{h_t} \end{aligned} \tag{5}$$

where r_t, z_t, $\tilde{h_t}$ and h_t are the state of reset gate, the state of update gate, updating activation, and hidding activation at time interval t respectively, W, U and b denote the weights of input, the weights of hidden units and the bias respectively, $\odot$ represents the element-wise multiplication, σ represents the Sigmoid function [30], and g represents the Rectified Linear Units (ReLu) function [31], [32].

C. ITRCN MODEL FOR INTERACTIVE NETWORK TRAFFIC PREDICTION

In this work, we present a hybrid network model named Interactive Temporal Recurrent Convolutional Networks (ITRCN), which has strong capability of extracting features on interactive and temporal patterns of the network traffic.

As shown in Fig. 6, features are passed in the order of the CNN layers, the fully-connected layer, and the GRU layers in the model. During the training process, the CNNs capture the interactive features, which are stacked by two CNN feature extracting layers.

The output of the CNNs can be obtained from the last extracting layer, and concatenated into a dense vector.

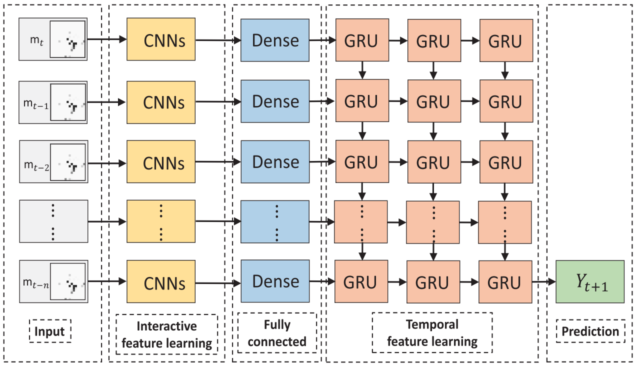


FIGURE 6. The framework of the proposed ITRCN model.

These processes can be written as follows:

$$\begin{aligned} O_L^k &= pool(\sigma(\sum_{j=1}^{c_{(L-1)}}(W_L^{jk}O_{(L-1)}^j + b_L^k))) \\ O_L^{flatten} &= flatten([O_L^1, O_L^2, \cdots, O_L^{c_2}]) \end{aligned} \tag{6}$$

where L is the last layer of the CNNs, the $pool(\cdot)$ represents the pooling operation, and the $flatten(\cdot)$ stands for the concatenated operation mentioned above.

And then, the vector is transformed into the input of GRU layers through the fully-connected layer. The process can be described as:

$$O_{cnn}^{t+1} = W_F O_L^{flatten} + b_F \tag{7}$$

where O_{cnn}^{t+1} is the output of the CNNs for time interval $t+1$. W_F and b_F are the weights and bias of the fully-connected layer.

The three-layer GRU part takes the output of fully-connected layer, and extracts the temporal features from the network traffic vector that derived from the fully-connected layer. Denote the GRU procedure as gru, the output of the e^{th} GRU layer can be written as:

$$O_e = gru_e(O_{cnn}^{t+1}) \tag{8}$$

thus the overall output of the ITRCN model is:

$$\hat{y}_{t+1} = gru_G(O_{cnn}^{t+1}) \tag{9}$$

where G is the last layer of the GRUs.

Our ITRCN model takes traffic images of previous five time intervals as the inputs, which can be written as:

$$M_t = (m_t, m_{t-1}, m_{t-2}, m_{t-3}, m_{t-4}) \tag{10}$$

where m_t is the network traffic image of time interval t. The ITRCN model can extract the features from these historical traffic and make an accurate one-step ahead prediction for the interactive network traffic.

D. TRAINING METHOD FOR ITRCN

The mean squared error (MSE) is employed as the cost function C in our work, which can measure the effectiveness of the model training process by calculating the distances between the predictions and the ground truth. The cost function is defined as follow:

$$MSE = C = \frac{1}{n}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{11}$$

where i is the prediction time interval.

The set of the training parameters (weights and bias) can be grouped as:

$$\Omega = (W_L^{jk}, W_r, W_z, W_h, W_F, U_r, U_z, U_h, b_r, b_z, b_h, b_L^k, b_F) \tag{12}$$

where these parameters are dynamically learned from the historical traffic data by the model. The goal of the training is to minimize the value of MSE and find the corresponding optimal set Ω.

The input-output relationship between each layer in the neural network model can be simplified as the following formula:

$$o^l = \sigma(w^l o^{l-1} + b^l) \tag{13}$$

where o^l, w^l, b^l are the output, weights, bias of the l^{th} layer respectively. And we denote z^l as the *weighted input* to the l^{th} layer, which can be defined as:

$$z^l = w^l o^{l-1} + b^l \tag{14}$$

We apply the Adaptive Moment Estimation (Adam) optimizer to cost optimization, which learns the optimal variables and minimize the cost in each layer [33]. The processes are written as:

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1)\nabla C_t \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2)\nabla C_t^2 \\ \hat{m}_t &= \frac{m_t}{1-\beta_1^t} \\ \hat{m}_t &= \frac{v_t}{1-\beta_2^t} \\ \hat{\Omega}_{t+1} &= \hat{\Omega}_t - \frac{\eta}{\sqrt{\hat{v}_t}+\epsilon}\hat{m}_t \end{aligned} \tag{15}$$

where β_1 and β_2 are the decay rates, ϵ is a constant, v_t and m_t are the exponentially decaying average of past squared gradient and the exponentially decaying average of past gradient respectively, $\hat{v}_t$ and $\hat{m}_t$ are the corrected estimates for v_t and m_t, ∇C_t is the gradient vector of the cost function C, $(\cdot)_t$ stands for the variable at t^{th} iteration, and η is the learning rate.

In order to compute the gradient of the cost function C to each layer, we employ the standard backpropagation method to our training in this work. The gradient vector can be written as:

$$\nabla C = (\frac{\partial C}{\partial v_1}, \frac{\partial C}{\partial v_2}, \cdots, \frac{\partial C}{\partial v_n})^T \tag{16}$$

where $v = v_1, v_2, \cdots, v_n$ represents the variables (weights and bias) of the C, and T denotes the transposing operation. Then, the intermediate quantity *error* δ is used to calculated the gradient, and the procedure can be described as follows:

$$\begin{aligned}\delta^l &= \nabla_o C \odot \sigma^{'}(z^l)\\ \delta^l &= ((w^{l+1})^T \delta^{l+1}) \odot \sigma^{'}(z^l)\\ \frac{\partial C}{\partial b^l} &= \delta^l\\ \frac{\partial C}{\partial w^l} &= o^{l-1}\delta^l \end{aligned} \tag{17}$$

where the δ^l is the *error* of the l^{th} layer. $\nabla_a C$ is defined as a vector whose components are the partial derivatives $\frac{\partial C}{\partial o_j^l}$.

After repeatedly apply these chain rules mentioned above, the model could learn all the optimal parameters and achieve good performance on the traffic prediction. The optimization result can be described as:

$$\hat{\Omega} = \arg\min_{\Omega} \frac{1}{n} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2 \tag{18}$$

V. DATA DESCRIPTION

A. DATASET

In this paper, we use traces from Yahoo! production workloads to test the effectiveness of different models. The traces come from five Yahoo! data centers, which are located at Dallas (DAX), Washington DC (DCP), Palo Alto (PAO), Hong Kong (HK) and the United Kingdom (UK). Most of the core services, such as Web, Email, Messenger, etc., are provided in DAX, DCP and PAO. Chen *et al.* [23] divided the traffic into two categories: 1) D2C, that the traffic between servers and clients, 2) D2D, that the traffic between different Yahoo! servers at different locations.

Our study is based on D2C Netflow datasets collected at the border router of DAX. It contains an one-day period data from April 30th, 2008, 8:00 am to May 1st, 2008, 8:00 am, which includes both the inbound and outbound traffic. Each record in the Netflow data includes: a) timestamp, b) source and destination IP addresses, c) transport layer port numbers, d) source and destination interface on the router, e) IP protocol, f) number of bytes and packets exchanged. By developing a pre-analysis on the D2C Netflow traffic data in this paper, we aim to figure out how the single service traffic and the interactive traffic change over time in the D2C data of DAX.

B. STATISTIC TRAFFIC OVERVIEW

Statistic traffic is the aggregation of raw services' D2C traffic. The statistic traffic values are defined as the amount of bytes transmitting during one minute. Then we totally get 1440 time intervals with the values of each minute in the 24 hours period. Table 1 presents a part of the transformed data with four basic attributes.

To have a first look at overall variation trend of traffic during one day, we aggregate the D2C traffic, which contains all the services traffic. As shown in Fig. 7, the total traffic changes quickly and has an obvious down trend during the time interval 800 to 1100, followed by an up trend during 1100 to 1440.

TABLE 1. A part of transformed data with basic four attributes.

Minutes	Source Port	Destination Port	Sum Traffic(bytes)
19	5050	80	52
19	5100	80	48
19	5100	5000	53
19	11999	25	1500
20	25	1935	40
20	53	53	45317
20	80	1935	36380
20	443	1935	171

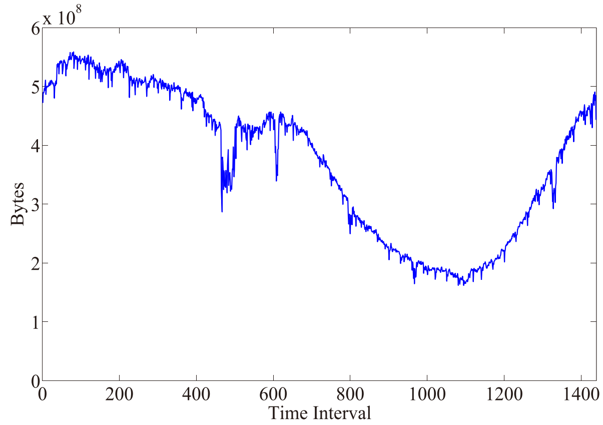


FIGURE 7. The changing bytes of the total network traffic over time interval 0 to 1440.

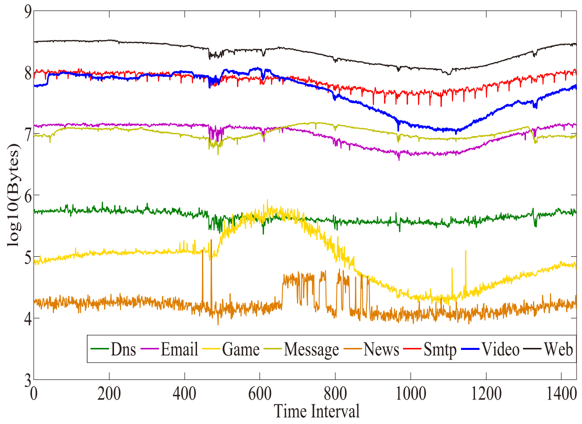


FIGURE 8. The network traffic of eight different services over time interval 0 to 1440.

C. SERVICE TRAFFIC ANALYSIS

The total traffic is the sum of D2C traffic in the data center, which could be divided into 8 types of different services, including DNS, Email, Game, Message, News, Video, SMTP and Web. The traffic of different services show significant fluctuations at different times. As shown in Fig. 8, some of services have long and slightly trends, such as Web,

TABLE 2. The result of exploratory analysis for different services network traffic data.

Service	Normality Jarque-Bera	Randomness	Stationarity ADF test	Autocorrelation Ljung-Box	Hurst exponent	White test	Suggested structure
DNS	52.51(<0.05)	32.69(<0.05)	1.59(>0.05)	1166.10(<0.05)	0.160	99.77(<0.05)	non-linear
Email	168.17	-35.64	-1.08	1395.80	0.141	681.27	non-linear
Game	1227.90	-34.89	3.97(<0.05)	1323.00	0.067	47.99	non-linear
Mess	48.86	-35.80	-0.63	1396.90	0.301	20.54	non-linear
News	230280	-18.10	-8.70	538.87	0.161	2.23(>0.05)	linear
SMTP	98.93	-33.23	-1.26	1335.10	0.282	538.71	non-linear
Video	134.39	-37.34	-0.77	1433.50	0.408	26.10	non-linear
Web	111.43	-36.55	-0.65	1432.20	0.370	583.03	non-linear

Mess, Email and DNS, while others present regular fluctuations (e.g., SMTP). However, the other network traffic of services, including Game, Video and News services, show violent and irregular fluctuations in the certain period. Especially, News has an explosive high traffic value from time interval 650 to 900.

1) STATISTICAL TEST

For each service type, we conduct a statistical test for the normality, randomness, stationarity, and autocorrelation. Table 2 displays the results and the corresponding P-values from different tests. It can be seen that all services are deviated from normality, indicating that the traditional prediction method (e.g., ARIMA), which relys on the assumption for normal distribution error, are insufficient. Further, each service can be seen as non-random, which means that it can be predicted and described explicitly by suitable models. Based on the Augmented Dickey-Fuller (ADF) test and the Ljung-Box test, we conclude that most services are non-stationary and auto-correlated, except Game is stationary. Importantly, the autocorrelation means models with dependency structure are needed for this kind of traffic.

Lastly, we employ the Hurst exponent test and the White test to check the LRD and the non-linearity of services' traffic. It can be observed that all the Hurst exponent is between 0 and 0.5, which means the anti-persistence and indicates that high values are more likely to be followed by low values and vice versa. On the other hand, only News accepts linearity, while the other services tend to show the non-linear feature. Analysed by the exploratory and statistical test, it is a strong suggestion that prediction approaches in Section III are highly suitable for all service types.

2) INTERACTION ANALYSIS

As some of services are more likely to communicate with each other, we developed an analysis about the changing rate of services and the interactive network traffic matrix. First, the changing rate of eight different services at each minute (60s) were calculated orderly (DNS, Email, Game, Messenger, News, Smtp, Video, Web) by the following equation:

$$R_t^q = |T_q(t+\Delta) - T_q(t)|/|T_q(t)| \tag{19}$$

where R_t^q is the changing rate of the q^{th} service network traffic. $T_q(t)$ and $T_q(t+\Delta)$ are the network traffic values at time interval t and $t+\Delta$ respectively. The$|\cdot|$ denotes the absolute value function.

Then we calculate the changing rate of the interactive network traffic matrices at each minute as the equation below:

$$R_t = |M(t+\Delta) - M(t)|/|M(t)| \tag{20}$$

where $M(t)$ and $M(t+\Delta)$ are the interactive network traffic matrices at t and $t+\Delta$. The details of the matrix have been presented in Section III. The numerator and the denominator can be written as:

$$\begin{aligned} |M(t+\Delta) - M(t)| &= \sum_{I=1}^{n}\sum_{J=1}^{n}|m_{IJ}^{t+\Delta} - m_{IJ}^{t}| \\ |M(t)| &= \sum_{I=1}^{n}\sum_{J=1}^{n}|m_{IJ}^{t}| \end{aligned} \tag{21}$$

where I and J are the numbers of rows and columns in the matrices. n is the number of ports, which is equal to 17. m_{IJ}^{t} is the element in the matrix $M(t)$.

As shown in Fig. 9 and 10, the changing rate of the interaction is compared to the services' changing rate. It can be seen from the figure that the region from time interval 0 to 400 was surrounded by the red square, where the changing rates of services were less than 0.25 in most of time, while the changing rate of News was slightly higher, but the average rate was still only 0.18. Compared to the different services', the changing rate of interaction remain a high level with an average rate 0.37, which is larger than 0.4 in most of the time.

We can conclude that even when the single service traffic remains the same (changing rate relatively small region), the interactive traffic still changes appreciably. If only the non-interactive service traffic is considered in the resource allocating, ignoring the service interaction, the integrity of some applications or services in data center may be cut apart. Thus, it is necessary to predict the overall network traffic of the interaction between different services.

VI. EXPERIMENTS

The experiment of this study includes two parts: a) prediction for traffic of single service, and b) prediction for interactive

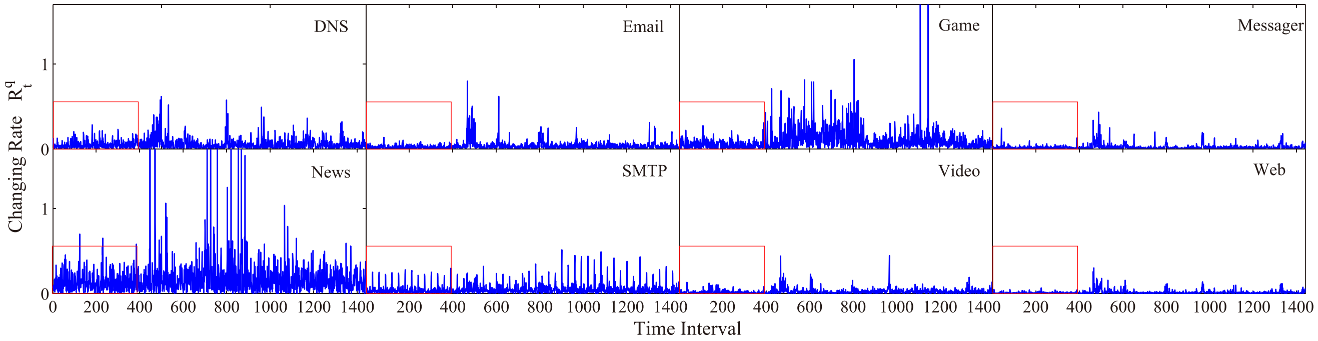


FIGURE 9. The changing rate of eight different services' traffic over time interval 0 to 1440.

TABLE 3. The RMSE results of the single service network traffic prediction.

(KB)	ARIMA (p, d, q)	SVM (c, g)	SRNN	GRU
DNS	30.82 (2, 1, 1)	46.19 (800, 0.01)	35.55	**30.18**
Email	601.89 (2, 1, 1)	707.75 (100, 0.01)	608.70	**582.17**
Game	43.85 (3, 1, 1)	51.79 (1000, 1)	43.06	**41.76**
Mess.	324.14 (2, 1, 1)	377.65 (100, 0.1)	328.47	**318.63**
News	**5.67** (3, 1, 1)	7.66 (100, 0.01)	9.10	7.13
SMTP	4673.30 (2, 1, 1)	5738.89 (30, 0.1)	4689.11	**4431.53**
Video	2612.54 (2, 0, 1)	3505.40 (300, 0.001)	2606.91	**2535.57**
Web	7696.97 (3, 1, 1)	7069.14 (500, 0.001)	5875.45	**5822.36**

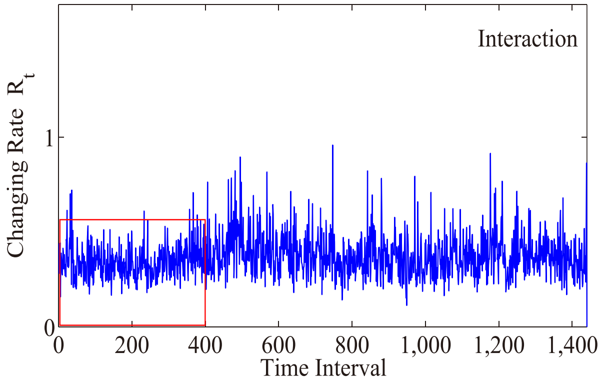


FIGURE 10. The changing rate of interactive traffic between different services over time interval 0 to 1440.

traffic between different services. More details are described as follows.

A. SINGLE SERVICE PREDICTION

The single service prediction aims to make the one-step prediction for the certain service network traffic in data centers.

1) TEMPORAL DATA GENERATION

In this work, the input of single service prediction is the one-dimension traffic data from eight services in Yahoo! data center. Due to the traffic characteristics of day and night are significantly different, we divided the traffic data of each service into two parts: a) April 30th, 2008, 8:00 am to April 30th, 2008, 8:00 pm and b) April 30th, 2008, 8:00 pm to May 1st, 2008, 8:00 am. Further, the first 80% of each part is used as training set for features learning, and the last 20% is testing set for model test.

2) EXPERIMENTAL SETUP

In this experiment, we evaluate our temporal prediction part GRU, and compare it with other time series prediction methods, including the ARIMA, SVM, and simple RNN (SRNN). The GRU is trained based on the optimizer Adam [33], which has been proven to work well. The learning rate is set to 0.01, the number of hidden units is set to 140, and the loss function is the Mean Squared Error (MSE). The dropout layer is used to prevent over-fitting. All parameters of the model are depended on numerous experiments to find an optimal structure. The parameter setting of the SRNN is exactly the same as the GRU, whereas details of other two models (the ARIMA and the SVM) are described as following.

For the ARIMA model, the optimal auto regressive parameter p, difference parameter d and moving average parameter q are determined according to the best Akaike Information Criterion (AIC) value [34]. For the SVM model, the kernel function is the Radial Basis Function (RBF), the optimal cost parameter c and the width parameter g are determined on numerous experiments. The parameters of these two models that resulted after applying the fitting process on the training set of each service are displayed in Table 3.

3) METRICS

To measure the accuracy of different prediction models, we calculate Root Mean Square Errors (RMSE), which can be formulated as:

$$RMSE = \sqrt{\frac{1}{n_s}\sum_{s=1}^{n_s}(\hat{D}_s - D_s)^2} \tag{22}$$

where n_s is the prediction steps, $\hat{D}_s$ and D_s denote the predictive traffic and real traffic at s^{th} prediction time interval respectively. In our experiment, n_s is 144, which indicates that we test the traffic data with 144 time labels for both two testing sets. We tested the models separately and calculated the average RMSE of these two parts as the final prediction results.

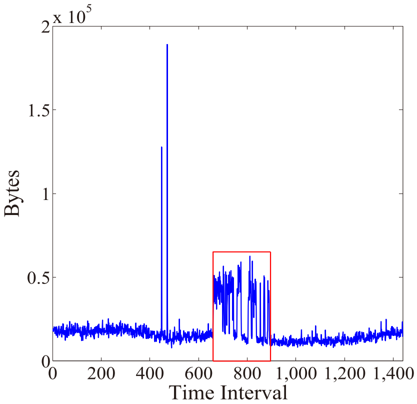


FIGURE 11. **Traffic characteristics for News service over time interval 0 to 1440.**

4) PERFORMANCE COMPARISON

As shown in Table 3, we can observe that the GRU yields the best accurate results for most traffic prediction in terms of RMSE, except for a little lower accuracy than the ARIMA in News. The SVM model shows the poorest prediction performance, whereas the ARIMA model outperforms the SRNN on the most services traffic prediction except for Game and Web. One possible reason is that the traffic dataset we used only contains an one-day period data, which significantly limits the ability of the SRNN to learn day-cycled behaviors. Compared with the GRU, the average RMSE values for the other models increased by 13.1%. One thing to be noted is that both the GRU and the SRNN perform poorer than the ARIMA on News prediction. Fig. 11 depicts the News traffic characters from time table 1 to 1440. We can observe that there are large fluctuation ranges in the time interval [660, 780] and [800, 900], which display explosive features compared to the rest of the day. Due to the internal mechanism, such as the autoregressive function, integrated function, moving average function, the ARIMA makes better predictions for these explosive traffic. However, if we have enough data and the traffic can be valued in a long time range (e.g., a week, a month or years), it's more likely to show a cyclical pattern instead of explosive characteristic. Thus, the GRU and the RNN can achieve better performances. We also leave the explosive short-term prediction for further works.

B. INTERACTIVE NETWORK TRAFFIC PREDICTION

Interactive network traffic prediction aims to make the one-step prediction for the services' communications in the data centers.

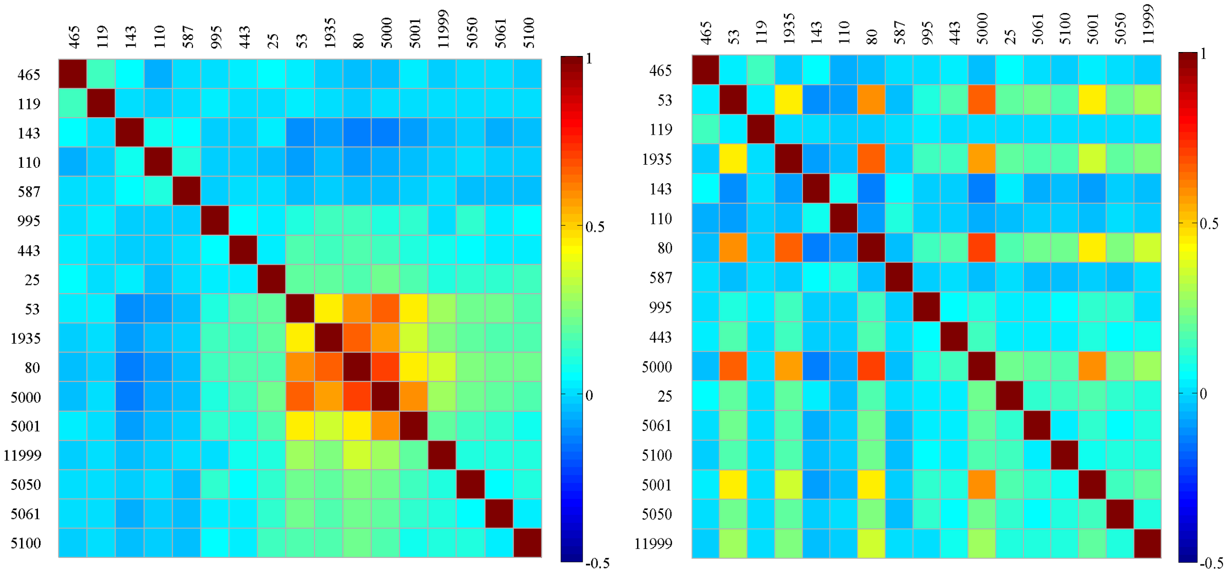
TABLE 4. **The correspondence between service types and ports.**

Service Type	Port Numbers
Email	110,143,465,587,995
Messenger	5000,5001,5050,5061,5100
Web	80,443
DNS	53
SMTP	25
News	119
Video	1935
Game	11999

1) INTERACTIVE NETWORK TRAFFIC IMAGE GENERATION

As shown in the work [23], there are 17 popular server ports observed in Yahoo! trace. All the services can be categorized into 8 groups. The mapping of each service group and its corresponding ports are listed in Table 4.

As described in Section III, we transfer the interactive traffic of these ports into network traffic matrices. As image features become more salient, the accuracy of prediction model will be improved. One simple way to construct distinctive images is to put the traffic with strong correlation together in the traffic matrix. Motivated by this, we get a clear view by calculating the correlation of each service. Fig. 12 (a) shows the correlation between each pair of services in the Yahoo! Dallas data center. As shown in this figure, service port numbers are listed on both x-axis and y-axis. The colored square corresponding to the correlation between different services that served by the certain ports. It can be seen that there is one significant cluster from port 995 to port 5100, including Email, Web, SMTP, DNS, Video, Messenger and Game, while the other ports do not show significant clustering features. Thus we could arrange the service ports in the traffic matrix with the order of (465, 119, 143, 110, 587, 995, 443, 25, 53, 1935, 80, 5000, 5001, 11999, 5050, 5061, 5100) and finally convert the matrices into images. In order to compare with the input images of orderly arranged ports, we also generated the images with randomly arranged ports and the correlation is shown in Fig. 12 (b). There are 1440 interactive traffic images for each time interval in our study. The first 80% of the images are used as training set, and the other are used as testing set. The details of the performance comparisons are showed later.



(a)

(b)

FIGURE 12. The cross-correlation between each pair of services. The ports with strong correlation are put together in (a). The ports are randomly arranged in (b).

TABLE 5. The parameters of ITRCN model.

Layer	Name	Units	Size
0	Input	1	(17,17)
1	Convolution	16	(3,3)
	Max-pooling	16	(2,2)
	Activation	Relu	–
2	Convolution	32	(3,3)
	Max-pooling	32	(2,2)
	Activation	Relu	–
3	Flatten	–	–
4	Fully connected	–	60
5	GRU 1	–	16
	Activation	Relu	–
	Dropout	–	0.2
	Batch normalization	–	–
6	GRU 2	–	16
	Activation	Relu	–
	Dropout	–	0.2
	Batch normalization	–	–
7	GRU 3	–	16
	Activation	Relu	–

2) EXPERIMENTAL SETUP

The parameters and details of our ITRCN model are given in Table 5. The model is trained on Adam optimizer [33]. It is worth noting that the time lag is set to 5, which indicates that the interactive traffic states of the previous five time intervals are used to predict the interactive traffic state one time interval ahead. During the training process, the learning rate was set to 0.001.

We also compare our ITRCN model with CNNs and GRUs individually. For CNNs, we test the three-hidden-layer structure with 256, 128, 64 units. For GRUs, we test the three-hidden-layer structure with 32, 32, 289 units. To the similar way of the ITRCN, we select these parameters through a large number of tests. All of these models are trained on Adam optimizer [33].

3) METRICS

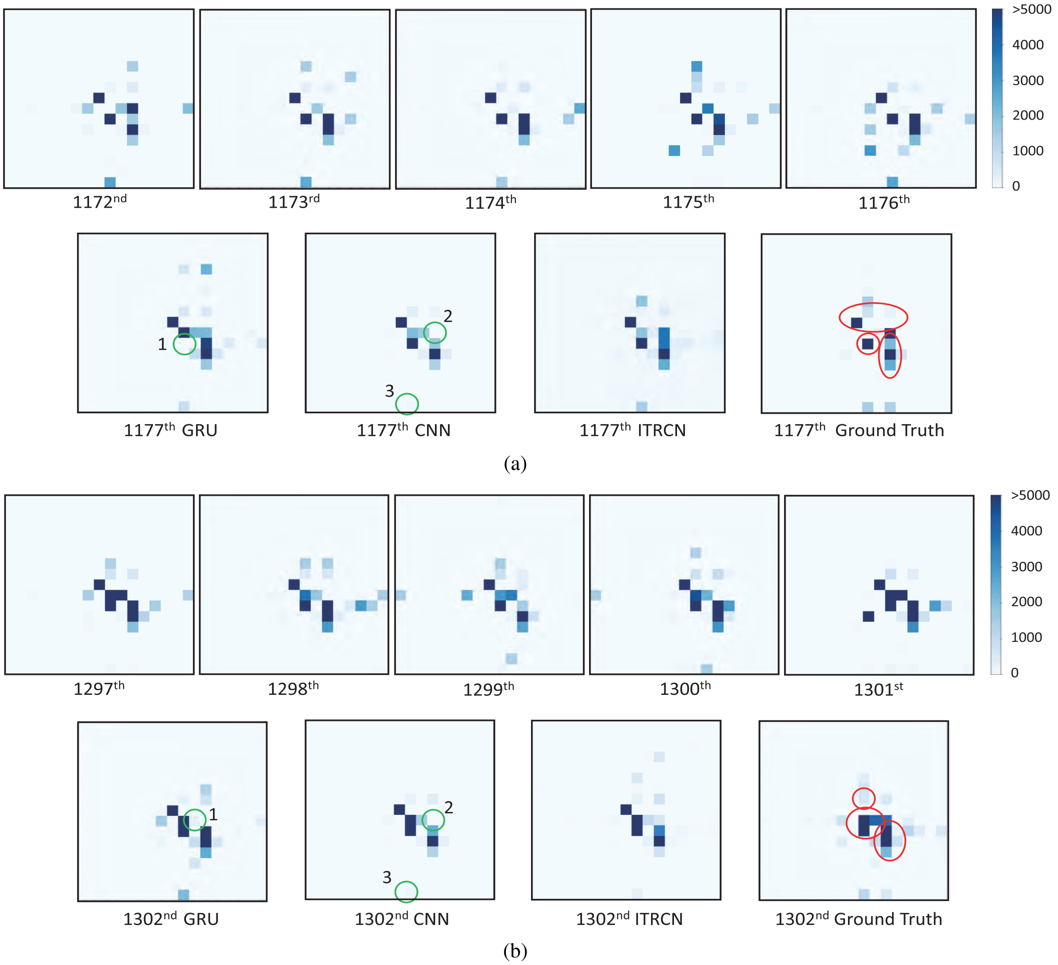
Same as the former experiments, the RMSE is used to measure the accuracy of different methods. In this experiment, the RMSE is defined as follows:

$$RMSE = \sqrt{\frac{1}{m \cdot d}\sum_{I=1}^{d}\sum_{K=1}^{m}(\hat{C}_{KI} - C_{KI})^2} \tag{23}$$

where m and d are the total order number of ports, and both of them is equal to 17. I and K are the order of ports. $\hat{C}_{KI}$ and C_{KI} are the interactive value between the K^{th} port and I^{th} port of prediction and truth respectively.

4) PERFORMANCE COMPARISON OF DIFFERENT MODELS

To give an insight for prediction accuracy of different methods, the RMSE is calculated and displayed in Table 6. It can be seen that the ITCRN made the most accurate prediction for interactive network traffic with the RMSE value of 1080. One possible reason is that the ITRCN takes both the interactive features and temporal features into considerations, which enhances the ability of the model to predict the communication among the network traffic. Compared with the ITRCN structure, the RMSE values of these two models increased by 14.3% and 13.0% respectively. Thus it can be concluded



(a)

(b)

FIGURE 13. The images of interactive traffic prediction. (a) is the prediction images for 1177th time interval and the true network interaction images from time interval 1172 to 1177. (b) is the prediction images for 1302nd time interval and the true network interaction images from time interval 1297 to 1302.

TABLE 6. The RMSE comparison among three models.

	GRU(3-layers)	CNN(3-layers)	ITRCN
RMSE	1234	1220	1080

that both interactive and temporal features are important to the interactive traffic prediction.

In order to explain the results more clearly, we show the prediction results of different models and true network interaction images at 1177^{th} and 1302^{nd} time interval respectively in Fig.13, as well as the ground truth of previous five time intervals are displayed. We set the non-communication to white color and communication value larger than 5000 to deep blue in the color bar, so as to more details can be displayed in the images. From these images, we can category the interactive features into temporal patterns and relatively inherent patterns, which mean the interactive pairs that sometimes appeared and always appeared in the true network interaction images respectively.

As shown in the Fig. 13 (a), we depict the relatively inherent patterns on the network interaction image of time interval 1177 by the red circle, and compared ground truth with the results of the ITRCN, CNNs and GRUs. We find that GRUs lost the relatively inherent patterns as the green circle 1 displayed, and CNNs lost the temporal patterns as the green circle 2 and 3 shown. As for the Fig. 13 (b), we can make the similar conclusion as the green circle 1 to 3 shows in the prediction results of GRUs and CNNs models.

These experimental results show that neither can the GRUs capture all the correlations between different services and inherent characteristics of the interactive network traffic, nor can the CNNs effectively extract the temporal features. Meanwhile, the ITRCN can catch both the temporal and interactive features, thus achieve a fairly good performance.

5) PERFORMANCE COMPARISON OF DIFFERENT PORTS' ARRANGEMENT

In order to verify the effect of the orderly arrangement in the traffic matrix, we put the images of orderly arranged ports and images of randomly arranged ports into the ITRCN model respectively. The prediction results of different inputs are

TABLE 7. The RMSE comparison among two kinds of inputs.

	Orderly arrangement	Randomly arrangement
RMSE	1080	1126

presented in Table 7. Compared with orderly arrangement, the RMSE value of randomly arrangement is improved by 4.2%, which indicates that the method of gathering ports with strong correlations fits CNNs feature extracting mechanism well. This is because CNNs can accurately extract the image features with strong spatial correlations during the training process. However, randomly arrangement makes the features scattered in the image, thus the correlation between the adjacent pixels is weak, which make it difficult for models to achieve accurate prediction.

VII. CONCLUSION

In this paper, we predicted the network resources for both the non-interactive network traffic and interactive network traffic. For non-interactive network traffic, we compared the GRU model with other prevailing methods. The GRU model made better predictions in the experiment by an average accuracy improvement of 13.1% in RMSE. For interactive side, we presented a novel end-to-end model, named ITRCN, which transformed the interactive network traffic into images and applied CNNs to capture the interactive features in the network traffic of data center. And the GRU model was used to extract the temporal features. The ITRCN model took advantages from both of the CNN and the GRU model, which outperformed the CNN and the GRU model in the experiment of Yahoo! data center network traffic by an average accuracy improvement of 13.6% in RMSE, showing a strong ability in extracting temporal features and interactive features of network traffic data.

For the future works, we will test our hybrid ITRCN model on more network traffic datasets. And we will also try to figure out how the variance of the days influences the effectiveness of our model. To sum up, this work provides a new insight into researches of the network traffic prediction, and can greatly help solve resource allocation problems in the data center.

REFERENCES

[1] M. E. Crovella and A. Bestavros, "Self-similarity in world wide Web traffic: Evidence and possible causes," *IEEE/ACM Trans. Netw.*, vol. 5, no. 6, pp. 835–846, Dec. 1997.

[2] J. Beran, R. Sherman, M. S. Taqqu, and W. Willinger, "Long-range dependence in variable-bit-rate video traffic," *IEEE Trans. Commun.*, vol. 43, no. 2, pp. 1566–1579, Feb. 1995.

[3] J. Beran, *Long Memory Processes—Probabilistic Properties and Statistical Methods*. Berlin, Germany: Springer, 2013.

[4] A. Azzouni and G. Pujolle. (2017). "A long short-term memory recurrent neural network framework for network traffic matrix prediction." [Online]. Available: https://arxiv.org/abs/1705.05690

[5] D.-C. Park, "Structure optimization of bilinear recurrent neural networks and its application to Ethernet network traffic prediction," *Inf. Sci.*, vol. 237, no. 13, pp. 18–28, 2013.

[6] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh, "Action-conditional video prediction using deep networks in Atari games," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2863–2871.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[8] J. Beran, R. Sherman, M. Taqqu, and W. Willinger, "Variable-bit-rate video traffic and long-range dependence," *IEEE Trans. Commun.*, vol. 43, nos. 2–4, pp. 1566–1579, Feb. 1995.

[9] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," *IEEE/ACM Trans. Netw.*, vol. 2, no. 1, pp. 1–15, Feb. 1994.

[10] C. Katris and S. Daskalaki, "Comparing forecasting approaches for Internet traffic," *Expert Syst. Appl.*, vol. 42, no. 21, pp. 8172–8183, 2015.

[11] G. U. Yule, "On a method of investigating periodicities in disturbed series, with special reference to Wolfer's sunspot numbers," *Philos. Trans. Roy. Soc. London A, Containing Papers Math. Phys. Character*, vol. 226, pp. 267–298, Apr. 1927.

[12] G. E. P. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *J. Amer. statist. Assoc.*, vol. 65, no. 332, pp. 1509–1526, 1970.

[13] P. KuanHoong, I. K. Tan, and C. YikKeong, "Bittorrent network traffic forecasting with ARMA," *Int. J. Comput. Netw. Commun.*, vol. 4, no. 4, p. 143, 2012.

[14] N. Sadek and A. Khotanzad, "Multi-scale high-speed network traffic prediction using k-factor gegenbauer ARMA model," in *Proc. IEEE Int. Conf. Commun.*, vol. 4. Jun. 2004, pp. 2148–2152.

[15] Y. Yu, J. Wang, M. Song, and J. Song, "Network traffic prediction and result analysis based on seasonal ARIMA and correlation coefficient," in *Proc. Int. Conf. Intell. Syst. Design Eng. Appl. (ISDEA)*, vol. 1. Oct. 2010, pp. 980–983.

[16] G. Chiruvolu and R. Sankar, "An approach towards resource management and transportation of VBR video traffic," in *Proc. IEEE Int. Conf. Commun. (ICC)*, vol. 1. Jun. 1997, pp. 550–554.

[17] S. Gowrishankar and P. S. Satyanarayana, "A time series modeling and prediction of wireless network traffic," *Int. J. Interactive Mobile Technol.*, vol. 3, no. 1, pp. 53–62, 2009.

[18] P. Bermolen and D. Rossi, "Support vector regression for link load prediction," *Comput. Netw.*, vol. 53, no. 2, pp. 191–201, 2009.

[19] P. Cortez, M. Rio, M. Rocha, and P. Sousa, "Multi-scale internet traffic forecasting using neural networks and time series methods," *Expert Syst.*, vol. 29, no. 2, pp. 143–155, 2012.

[20] C. Katris and S. Daskalaki, "Prediction of Internet traffic using time series and neural networks," in *Proc. Int. Work-Conf. Time Ser. Anal. (ITISE)*, vol. 1. 2014, pp. 594–605.

[21] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," *J. Amer. Statist. Assoc.*, vol. 91, no. 433, pp. 365–377, 1996.

[22] A. Soule, K. Salamatian, A. Nucci, and N. Taft, "Traffic matrix tracking using Kalman Filters," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 33, no. 3, pp. 24–31, 2005.

[23] Y. Chen, S. Jain, V. K. Adhikari, Z.-L. Zhang, and K. Xu, "A first look at inter-data center traffic characteristics via Yahoo! datasets," in *Proc. INFOCOM*, Apr. 2011, pp. 1620–1628.

[24] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *Proc. 4th Conf. Chin. Assoc. Autom. Acad. Annu. (YAC)*, 2016, pp. 324–328.

[25] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[26] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time-series," in *The Handbook of Brain Theory and Neural Networks*, M. A. Arbib, Ed. Cambridge, MA, USA: MIT Press, 1995, pp. 255–258.

[27] S. Wang, L. He, B. Cao, C.-T. Lu, P. Yu, and A. Ragin, "Structural deep brain network mining," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 475–484.

[28] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. (2014). "On the properties of neural machine translation: Encoder-decoder approaches." [Online]. Available: https://arxiv.org/abs/1409.1259

[29] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. (2014). "Empirical evaluation of gated recurrent neural networks on sequence modeling." [Online]. Available: https://arxiv.org/abs/1412.3555

[30] J. Han and C. Moraga, "The influence of the Sigmoid function parameters on the speed of backpropagation learning," *Natural Artif. Neural Comput.*, vol. 930, pp. 195–201, Jun. 1995.

[31] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 315–323.

[32] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 807–814.

[33] D. P. Kingma and J. Ba. (2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980

[34] H. Akaike, "A new look at the statistical model identification," *IEEE Trans. Autom. Control*, vol. 19, no. 6, pp. 716–723, Dec. 1974.

XIAOFENG CAO received the B.S. and M.S. degrees in mechanical engineering from the National University of Defense Technology (NUDT), China, in 2014 and 2016, respectively.

He is currently pursuing the Ph.D. degree with the Science and Technology on Information Systems Engineering Laboratory, NUDT. His current research interests include network data mining and cyber security.

YUHUA ZHONG received the B.S. and M.S. degrees in control science and engineering from the National University of Defense Technology, China, in 2014 and 2016, respectively.

His current research interests include pattern recognition and lidar recognition.

YUN ZHOU received the Ph.D. degree from the Risk and Information Management Research Group, Queen Marry University of London, U.K., in 2016.

He is currently a Lecturer with the Science and Technology on Information Systems Engineering Laboratory, NUDT. His current research interests include data mining, Bayesian network learning, and cyber security reasoning.

JIANG WANG received the B.S. degree in information system engineering and the M.S. degree in management science and engineering from the National University of Defense Technology, China, in 2011 and 2013, respectively.

He is currently pursuing the Ph.D. degree with the Science and Technology on Information Systems Engineering Laboratory, NUDT. His current research interests include spatio-temporal data mining and trajectory mining.

CHENG ZHU received the Ph.D. degree in management science and engineering from the National University of Defense Technology (NUDT), China, in 2004.

He is currently a Professor with the Science and Technology on Information Systems Engineering Laboratory, NUDT. His current research interest is network data mining and cyber security.

WEIMING ZHANG received the Ph.D. degree in management science and engineering from the National University of Defense Technology (NUDT), China, in 2001.

He is currently the Chief of the Science and Technology on Information Systems Engineering Laboratory, NUDT. His current research interest is command and control organizations.