Migration and Energy Aware Network Traffic Prediction Method Based on LSTM in NFV Environment

Ying Hu[1], Liang Zhu[1*], Jianwei Zhang[2], Zengyu Cai[1], and Jihui Han[1]
[1] College of Computer and Communication Engineering, Zhengzhou University of Light Industry
Zhengzhou, Henan 450000 China
[e-mail:lzhu@zzuli.edu.cn]
[2] College of Software engineering, Zhengzhou University of Light Industry
Zhengzhou, Henan 450000 China
[*]Corresponding author: Liang Zhu

Received March 21, 2022; revised December 29, 2022; accepted February 22, 2023; published March 31, 2023

Abstract

The network function virtualization (NFV) uses virtualization technology to separate software from hardware. One of the most important challenges of NFV is the resource management of virtual network functions (VNFs). According to the dynamic nature of NFV, the resource allocation of VNFs must be changed to adapt to the variations of incoming network traffic. However, the significant delay may be happened because of the reallocation of resources. In order to balance the performance between delay and quality of service, this paper firstly made a compromise between VNF migration and energy consumption. Then, the long short-term memory (LSTM) was utilized to forecast network traffic. Also, the asymmetric loss function for LSTM (LO-LSTM) was proposed to increase the predicted value to a certain extent. Finally, an experiment was conducted to evaluate the performance of LO-LSTM. The results demonstrated that the proposed LO-LSTM can not only reduce migration times, but also make the energy consumption increment within an acceptable range.

Keywords: Network Function Virtualization, Virtual Network Function Migration, Network Traffic Prediction, Long Short-Term Memory

1. Introduction

The traditional deployment mode for network services is based on dedicated hardware. However, it has problems such as the high cost of replacing network devices, long deployment cycles of new network services and low resource utilization rate. The rapid development of network services has considerably increased the problems existing on the traditional deployment mode [1]. The emergence of network function virtualization (NFV) solves the problems existing on the current network service mode. The network services are software-based, and the software is deployed on the universal hardware. Users can flexibly and efficiently rent services on demand. Here, sub-service functions run as software, and the software in turn runs on general hardware devices. Virtual network functions (VNFs) are embedded in the software. This new approach to service function chain (SFC) deployment faces many new challenges with respect to resource allocation and reconstruction [2][3][4][5]. Herein, we focus on network traffic prediction in an NFV environment to guarantee a low frequency of passive migration and sacrifice some energy consumption.

Frequent migration in an NFV environment occurs by ignoring network traffic changes when the flow rate of network traffic is large. However, a small flow rate of network traffic can lead to unnecessary energy consumption due to low node usage. The usual solution is to divide each day into multiple time slices. It obtains the resource demand according to the peak network traffic in each time slice, and actively performs migration before each time slice starts [6][7][8][9]. One fundamental problem is to predict the peak network traffic for the next time slice [10]. If the prediction value of network traffic is higher than the real value, there will be excess energy consumption; otherwise, it may lead to passive migration within the time slice. To reduce the unpredictable passive migration within the time slice, we amplify the prediction value of the network traffic to a certain extent. To reduce energy consumption, the amplification coefficient of the predicted value should not be too high. Therefore, how to predict the network traffic for the next time slice becomes one crucial problem in the NFV environment.

For this paper, the main contributions are as follows:

(1) We believe that dynamic scaling needs to be considered in NFV to meet Quality of Service. According to the characteristics of network, dynamic and adaptive techniques are needed to improve the performance of resource management methods. Before allocating network traffic, we need to estimate network traffic and set the estimated value to a larger size to reduce migration times and ensure service quality.

(2) According to the above reasons, we design a network traffic prediction method called LO-LSTM. It elastically amplifies network traffic to reduce the passive migration and sacrifice the energy consumption as little as possible.

(3) We have implemented LO-LSTM and presented extensive evaluations. Experiments with the changing network loads show that LO-LSTM substantially reduces passive migration, and energy consumption can be sacrificed according to pre-set weight parameters to optimize the overall goal.

2. Related Research

Network traffic trends have been characterized by time series [11][12][13][14]. Network traffic prediction uses time series for modelling and analysis. The network traffic prediction model can be divided into stationary and non-stationary categories.

The stationary model can be divided into two types, i.e. the short correlation and long correlation models. Short correlation models include the Poisson model, Markov model [15] and auto-regressive moving average model (ARMA model). Long correlation models include the fractional difference auto-regressive moving average model (FARMA model). The above stationary traffic model lays a solid foundation for network traffic prediction. However, owing to the sudden and uneven network traffic distribution, accurately describing the characteristics of network traffic changes using a single stationary model is difficult.

With the development of machine learning (ML) technologies, boosting, support vector machine (SVM) and other ML methods have gradually been applied to network traffic prediction. Furthermore, with the development of neural networks researchers frequently use back propagation and Elman neural networks to predict network traffic. Experimental results show that these methods effectively improve the accuracy of network traffic prediction compared with traditional methods. Azzouni et al. proposed a LSTM recurrent neural network model [16], which considers the time correlation between network traffic and historical data to predict network traffic. The above research laid a solid foundation for network traffic prediction. However, all these methods work in the traditional network environment. When applied to the NFV environment, the problems of VNF migration caused by unpredictable network traffic still arise.

Recently, network traffic and cloud resource prediction methodologies have been proposed in the NFV environment. Tang et al. [17] proposed a network traffic prediction method for scaling resources in NFV environment using network traffic modelling with an ARMA process. The predicted network traffic values were obtained by minimizing the loss. Some solutions are based on time series forecasting with LSTM recurring neural networks [18], predicting host load in cloud infrastructures. The study by Rahman [19] was based on ML classification procedures, and the classification objective was to minimize operational cost and QoS degradation. Vincenzo Eramo et al. [20] proposed a solution for NFV environments resource orchestration in which the different values of the over-provisioning and under-provisioning cost was considered. Liu et al. [21] proposed a complete stateful scaling system that efficiently reduces flow-level latency and achieves near-optimal resource usage to deal with time-varying loads in NFV environment. Mahsa Moradi et al. [22] analyzed and compared three algorithms of machine learning.

The main contribution of our work is to propose a network traffic prediction technique. The technique is aware of the difficulty in accurately predicting traffic, therefore it employs high network traffic prediction values to reduce passive migration. We minimize the probability of insufficient resource allocation by increasing the predicted traffic value to a certain extent. This objective is achieved by minimizing an asymmetric loss function, and the loss function is characterized by considering the penalties for over-provisioning and under-provisioning and making the loss value increase faster for under-provisioning.

3. Problem Statement

3.1 Problem Description

In the NFV environment, users request resources based on their requirements, service providers construct service function chains (SFCs) for users based on their requirements and the substrate network status according to predefined rules and policies. Each sub-service requires certain resources. The dynamic change in network traffic produces a constant change in resource demands [23][24][25]. An increase in network traffic increases the resource demand for SFC. When the substrate network cannot carry the network load, VNFs migrate to ensure QoS. Similarly, a decrease in network traffic decreases resource demands. When the resource utilization of the physical node is low, service providers migrate VNFs and shut down nodes for energy saving.

VNFs migrate passively when the available resources cannot feed the VNFs or the end-to-end delay exceeds the delay boundary, leading to unpredictable network status changes and end-to-end delays. We should predict network traffic in advance and actively migrate VNF to reduce the passive migration.

3.2 Network Model

1. Substrate Network

We represent the substrate network as a graph (N_e, L_e) including NFV enabled nodes N_e and physical links L_e. Each physical node $n_i^e \in N_e$ has properties such as processing capacity $CPU(n_i^e)$. Each physical link $l_i^e = (n_i^e, n_{i+1}^e)$ ($l_i^e \in L_e$, $n_i^e \in N_e$ and $n_{i+1}^e \in N_e$) has attributes such as link bandwidth $BW(l_i^e)$.

2. SFC Request

A set of SFC requests is defined as R_q. Each SFC t ($t \in R_q$) can communicate through the service path. Each SFC t has a source node $i(t)$, a destination node $o(t)$, and a sequence of VNF nodes $N_v(t)$. The end-to-end delay of SFC t cannot exceed the delay boundary $\delta(t)$. SFC t has a set of virtual links $L_v(t)$, including the virtual link between the source node $i(t)$ and the first VNF, the virtual link between the previous VNF and the subsequent VNF, and the virtual link between the last VNF and the destination node $o(t)$.

3. Resource Demand

Network traffic r^{real} changes dynamically with time. The greater the network traffic, the greater the resource requirements [23][24][25].

The CPU requirement is approximately linearly correlated with network traffic [23] [24] [25]. Define CPU demand as (1):

$$\mathrm{Re}\,qCPU^{real}(n_i^v) = r^{ratio} \cdot \mathrm{Re}\,qCPU(n_i^v) \tag{1}$$

Where $\mathrm{Re}\,qCPU^{real}(n_i^v)$ is the CPU requirement of VNF n_i^v under the current traffic; $\mathrm{Re}\,qCPU(n_i^v)$ is the CPU requirement of VNF n_i^v under the baseline traffic; r^{ratio} is the amplification factor.

Define r^{ratio} as follows:

$$r^{ratio} = \frac{r^{real}}{r_0} \tag{2}$$

Where r_0 is the baseline traffic, we set the baseline traffic based on initial resource requirement; r^{real} is the current network traffic.

Bandwidth requirement $\mathrm{Re}\,qBW^{real}(t)$ of SFC t is approximately linearly correlated with amplification factor r^{ratio} [23][24][25]:

$$\mathrm{Re}\,qBW^{real}(t) = r^{ratio} \cdot \mathrm{Re}\,qBW(t) \tag{3}$$

Where $\mathrm{Re}\,qBW(t)$ is the bandwidth requirement under the baseline traffic.

4. Latency

End-to-end latency is also related to network traffic, and the larger the network traffic load, the longer the latency [26][27]. The latency is approximately linearly related to the network traffic amplification factor.

5. Energy Consumption

Both nodes and links consume energy. As the energy consumption of link is relatively small, we only consider the energy consumption of node [28]. Node energy consumption is approximately linearly correlated with CPU utilization [29].

$$PW(n_i^e) = p_{\min} + (p_{\max} - p_{\min}) \cdot CPUutil(n_i^e) \tag{4}$$

Where $p_{\min}$ is the minimum node energy consumption, and $p_{\max}$ is the peak value of node energy consumption, $CPUutil(n_i^e)$ is the CPU utilization of node n_i^e.

3.3 Optimization Objective

The migration should satisfy constraints 1~4, and take energy-saving, passive migration reducing, and migration failure reducing as optimization objectives. Define the general objective as follows:

$$Minimize\ \left(\eta_1 \cdot \Delta PW\_running(T_q) + \eta_2 \cdot NF\_migrate(T_q) + \eta_3 \cdot Fail\_migrate(T_q)\right) \tag{5}$$

Where η_1, η_2, and η_3 are constant coefficients, $\Delta PW\_running(T_q)$ is the difference between the assigned and required energy in time slice T_q.

The number of passive migration $NF\_migrate(T_q)$ is shown in (6):

$$NF\_migrate(T_q) = \sum_{t \in R_q} \sum_{n_i^v \in N_v(t)} \sum_{n_j^e \in N_e} \sum_{n_k^e \in N_e} Y(T_q, t) \cdot Z(t, n_i^v, n_j^e, n_k^e) \tag{6}$$

Where $Y(T_q, t)$ is 1 when SFC t migrates within time slice T_q, otherwise, it is 0. $Z(t, n_i^v, n_j^e, n_k^e)$ is 1 when VNF n_i^v in SFC t migrates from physical node n_j^e to physical node n_k^e; otherwise, it is 0.

The number of passive migration failures $Fail\_migrate(T_q)$ is shown in (7):

$$Fail\_migrate(T_q) = \sum_{t \in R_q} Y(T_q, t) \cdot F(T_q, t) \tag{7}$$

Where $F(T_q, t)$ is 1 when the migration fails within time slice T_q; otherwise, it is 0.

All constraints are described as follows:

Constraint 1: each VNF n_i^v in SFC t is only embedded on one physical node.

$$\begin{gathered} \forall t \in R_q, \forall n_i^v \in N_v(t) \\ \sum_{n_j^e \in N_e} X(t, n_i^v, n_j^e) = 1 \end{gathered} \tag{8}$$

Where $X(t, n_j^v, n_i^e)$ is 1 when VNF n_j^v in SFC t is mapped to physical node n_i^e; Otherwise, $X(t, n_j^v, n_i^e)$ is 0.

Constraint 2: the CPU usage cannot exceed its capacity.

$$\begin{gathered}\forall n_j^e \in N_e \\ \sum_{t \in R_q} \sum_{n_i^v \in N_v(t)} \operatorname{Re} qCPU(n_i^v) \cdot X(t, n_i^v, n_j^e) \le CPU(n_j^e)\end{gathered} \tag{9}$$

Where $CPU(n_j^e)$ is the CPU capacity of physical node n_j^e.

Constraint 3: the bandwidth usage cannot exceed its capacity.

$$\begin{gathered}\forall l_j^e \in L_e \\ \sum_{t \in R_q} \sum_{l_i^v \in L_v(t)} \operatorname{Re} qBW(t) \cdot Z(t, l_i^v, l_j^e) \le BW(l_j^e)\end{gathered} \tag{10}$$

Where $BW(l_j^e)$ is the bandwidth capacity of physical link l_j^e; $\operatorname{Re} qBW(t)$ is the bandwidth demand for SFC t; $Z(t, l_i^v, l_j^e)$ is 1 when virtual link l_i^v in SFC t is mapped to physical link l_j^e; otherwise, it is 0.

Constraint 4: the end to end latency of each SFC cannot exceed the latency boundary $\delta(t)$.

$$\begin{gathered}\forall t \in R_q \\ D(t) \le \delta(t)\end{gathered} \tag{11}$$

Where $D(t)$ is the end to end delay of SFC t.

4. Network Traffic Prediction

We divide one day into many time slices, and predict the peak network traffic for the next time slice. To reduce the probability of passive migration within the time slice, we amplify the prediction value. In addition, to reduce energy consumption, the amplification value should not be too large. To learn the changing rules of network traffic and the interdependence between adjacent time slices, we propose a network traffic prediction method based on LSTM.

4.1 LSTM Model

Set up L intervals and perform active VNF migration when time slices alternate. The neural network architecture based on LSTM is shown in **Fig. 1**.

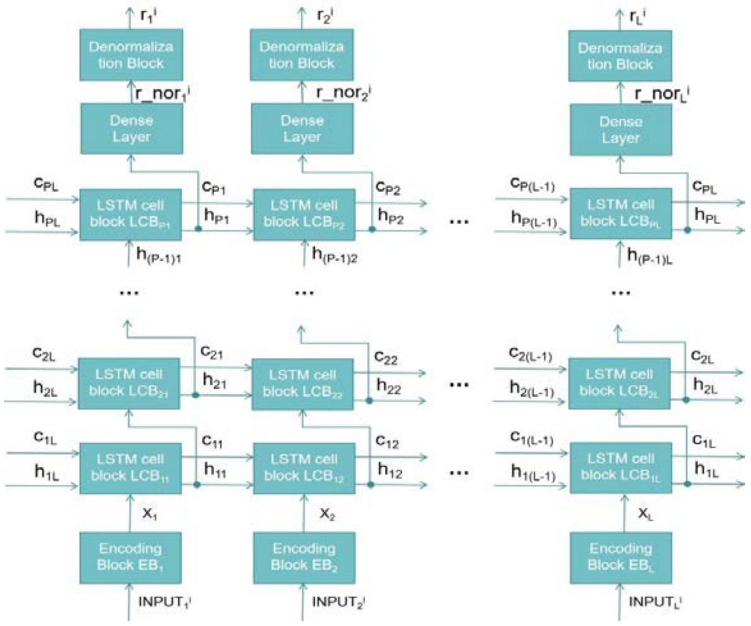


Fig. 1. Neural network architecture based on LSTM

Each part is described as follows:

i) Input

Each input $INPUT_q^i$ includes four components:

- the actual peak value of network traffic r_q^{i-1} in time slice T_q on day D_{i-1}
- the actual peak value of network traffic r_q^{i-2} in time slice T_q on day D_{i-2}
- the actual peak value of network traffic r_q^{i-7} in time slice T_q on day D_{i-7}
- the actual peak value of network traffic r_{q-1}^{i} in time slice T_{q-1} on day D_i

ii) Encoding Block

The encoding block can standardize the input data, as shown in (12):

$$r\_nor_q^i = \frac{r_q^i - mean}{std} \tag{12}$$

Where std is the standard deviation, $mean$ is the average value.

iii) Dense Layer

The output dimension of LSTM is the same as LSTM units. We use the dense layer to convert the output dimension into the required size.

iv) Denormalization Block

This module performs denormalization as shown in (13):

$$r_q^i = r\_nor_q^i \cdot std + mean \tag{13}$$

v) Output

The output value is the peak value of network traffic r_q^i in time slice T_q on day D_i.

vi) LSTM layers

The more layers, the stronger the learning ability. However, if the LSTM has too many layers, it is easy to over-fit. Generally, the neural network has no more than three layers.

4.2 Optimization of Loss Function

We optimize the loss function for LSTM, which is called LO-LSTM. The optimization objectives include energy-saving, reducing passive migration and migration failure. We should take these factors into account to calculate the loss.

1. Calculation of Loss

When the predicted value is higher than the real value, the loss value is only related to energy consumption. Otherwise, it is only associated with migration frequency and mapping failures. Therefore, we calculate loss as follows:

$$\begin{aligned} loss_q^i &= \eta_1 \cdot \Delta PW\_running(T_q) \cdot I(op_q - r_q) \\ &+ (\eta_2 \cdot NF\_migrate(T_q) + \eta_3 \cdot Fail\_migrate(T_q)) \cdot I(r_q - op_q) \end{aligned} \tag{14}$$

Where $I(x) = 0$ when $x < 0$; and $I(x) = 1$ when $x > 0$.

2. Loss of Energy Consumption

$\Delta PW\_running(T_q)$ is the difference between predicted value and actual value of energy consumption in time slice T_q, and we use the predicted value to reallocate resources. It can be calculated as follows:

$$\begin{aligned}
&\Delta PW\_running(T_q) \\
&= PW\_assign(T_q) - PW\_request(T_q) \\
&= (t_1 - t_0) \cdot \sum_{n_i^e \in N_e} PW(n_i^e) - (t_1 - t_0) \cdot \sum_{n_i^e \in N_e} PW'(n_i^e) \\
&= (t_1 - t_0) \cdot \left(\sum_{n_i^e \in N_e} (p_{\min} + (p_{\max} - p_{\min}) \cdot CPUutil(n_i^e)) - \sum_{n_i^e \in N_e} (p_{\min} + (p_{\max} - p_{\min}) \cdot CPUutil'(n_i^e)) \right) \\
&= (t_1 - t_0) \cdot (p_{\max} - p_{\min}) \cdot \left(\sum_{n_i^e \in N_e} CPUutil(n_i^e) - \sum_{n_i^e \in N_e} CPUutil'(n_i^e) \right) \\
&= (t_1 - t_0) \cdot (p_{\max} - p_{\min}) \cdot \left(\sum_{n_i^e \in N_e} \frac{CPU^{assign}(n_i^e)}{CPU(n_i^e)} - \sum_{n_i^e \in N_e} \frac{CPU^{request}(n_i^e)}{CPU(n_i^e)} \right) \\
&= (t_1 - t_0) \cdot (p_{\max} - p_{\min}) \cdot \left(\sum_{n_i^e \in N_e} \frac{(CPU^{assign}(n_i^e) - CPU^{request}(n_i^e))}{CPU(n_i^e)} \right) \\
&= (t_1 - t_0) \cdot (p_{\max} - p_{\min}) \\
&\cdot \left(\sum_{n_i^e \in N_e} \frac{\left(\sum_{t \in R_q} \sum_{n_j^v \in N_v(t)} \operatorname{Re}qCPU^{assign}(n_j^v) \cdot X(t, n_j^v, n_i^e) - \sum_{t \in R_q} \sum_{n_j^v \in N_v(t)} \operatorname{Re}qCPU^{request}(n_j^v) \cdot X(t, n_j^v, n_i^e) \right)}{CPU(n_i^e)} \right) \\
&= (t_1 - t_0) \cdot (p_{\max} - p_{\min}) \\
&\cdot \left(\sum_{n_i^e \in N_e} \frac{\left(\sum_{t \in R_q} \sum_{n_j^v \in N_v(t)} ratio\_assign_q^i \cdot \operatorname{Re}qCPU(n_j^v) \cdot X(t, n_j^v, n_i^e) - \sum_{t \in R_q} \sum_{n_j^v \in N_v(t)} ratio\_real_q^i \cdot \operatorname{Re}qCPU(n_j^v) \cdot X(t, n_j^v, n_i^e) \right)}{CPU(n_i^e)} \right) \\
&= (t_1 - t_0) \cdot (p_{\max} - p_{\min}) \cdot \left(\sum_{n_i^e \in N_e} \frac{\sum_{t \in R_q} \sum_{n_j^v \in N_v(t)} X(t, n_j^v, n_i^e) \cdot \operatorname{Re}qCPU(n_j^v) \cdot (ratio\_assign_q^i - ratio\_real_q^i)}{CPU(n_i^e)} \right) \\
&= (t_1 - t_0) \cdot (p_{\max} - p_{\min}) \cdot (ratio\_assign_q^i - ratio\_real_q^i) \cdot \left(\sum_{n_i^e \in N_e} \frac{\sum_{t \in R_q} \sum_{n_j^v \in N_v(t)} X(t, n_j^v, n_i^e) \cdot \operatorname{Re}qCPU(n_j^v)}{CPU(n_i^e)} \right)
\end{aligned} \tag{15}$$

Where t_0 is the start of time slice T_q, and t_1 is the end of time slice T_q; $CPUutil\left(n_i^e\right)$ is the CPU utilization of physical node n_i^e under predicted network traffic; $CPUutil'\left(n_i^e\right)$ is the CPU utilization of physical node n_i^e under actual network traffic; $CPU^{assign}\left(n_i^e\right)$ is the total CPU allocation on physical node n_i^e; $CPU^{request}\left(n_i^e\right)$ is the total CPU requirement on physical node n_i^e; $\mathrm{Re}qCPU^{assign}\left(n_j^v\right)$ is the CPU allocation for VNF n_j^v under forecasting network traffic; $\mathrm{Re}qCPU^{request}\left(n_j^v\right)$ is the CPU requirement of VNF n_j^v under actual network traffic.

$ratio\_assign_q^i$ is the amplification factor according to predicted network traffic :

$$ratio\_assign_q^i = \frac{r\_assign_q^i}{r_0} \tag{16}$$

Where $r\_assign_q^i$ is the predicted peak value of network traffic in time slice T_q of day D_i.

$ratio\_real_q^i$ is the amplification factor according to actual network traffic:

$$ratio\_real_q^i = \frac{r\_real_q^i}{r_0} \tag{17}$$

Where $r\_real_q^i$ is the actual peak value of network traffic in time slice T_q of day D_i.

Further, calculate $\Delta PW\_running\left(T_q\right)$ as follows:

$$\begin{aligned}
&\Delta PW\_running\left(T_q\right)\\
&= \alpha_1 \cdot \left(t_1 - t_0\right) \cdot \left(p_{\max} - p_{\min}\right) \cdot \left(\frac{\left(r\_assign_q^i - r\_real_q^i\right)}{r_0}\right) \cdot \left(\sum_{n_i^e \in N_e} \frac{\sum_{t \in R_q} \sum_{n_j^v \in N_v(t)} X\left(t, n_j^v, n_i^e\right) \cdot \mathrm{Re}qCPU\left(n_j^v\right)}{CPU\left(n_i^e\right)}\right)\\
&= \alpha_1 \cdot \left(t_1 - t_0\right) \cdot \left(p_{\max} - p_{\min}\right) \cdot \frac{1}{r_0} \cdot \left(\sum_{n_i^e \in N_e} \frac{\sum_{t \in R_q} \sum_{n_j^v \in N_v(t)} X\left(t, n_j^v, n_i^e\right) \cdot \mathrm{Re}qCPU\left(n_j^v\right)}{CPU\left(n_i^e\right)}\right) \cdot \left(r\_assign_q^i - r\_real_q^i\right)
\end{aligned} \tag{18}$$

According to (18), $\Delta PW\_running\left(T_q\right)$ and $\left(r\_assign_q^i - r\_real_q^i\right)$ are linearly dependent.

3. Loss of Migration

If the allocated resource is lower than the required resource, there will be more passive migration within the time slice. The greater the gap, the higher the frequency of passive migration. The growth rate (slope of the curve) of migration may be more prominent than the growth rate of energy consumption. Therefore, we define the loss of migration as shown in (19):

$$\eta_2 \cdot NF\_migrate(T_q) + \eta_3 \cdot Fail\_migrate(T_q) = \theta \cdot \left(r\_assign_q^i - r\_real_q^i\right)^2 \tag{19}$$

Where θ is a constant coefficient.

In summary, we define the loss as follows:

$$\begin{aligned} loss_q^i &= \gamma \cdot \left(r^{assign\_\max} - r^{real\_\max}\right) \cdot I\left(op_q - r_q\right) \\ &+ \theta \cdot \left(r^{real\_\max} - r^{assign\_\max}\right)^2 \cdot I\left(r_q - op_q\right) \end{aligned} \tag{20}$$

Where γ is constant coefficient, and the loss function is in time slice T_q on day D_i.

4. Loss Function

We use mean square error (MSE) as the loss function:

$$loss\_function = \frac{1}{M \times L} \cdot \sum_{i=1}^{M} \sum_{q=1}^{L} \left(loss_q^i\right)^2 \tag{21}$$

Where M is the total number of days, and L is the total number of time slices in one day.

4.3 Algorithm Description

The training and the prediction process is shown below:

Table 1. Network traffic prediction algorithm based on LO-LSTM

```
Inputs：LSTM parameters, substrate network on which many SFCs have been embedded, benchmark value of network traffic, historical data of network traffic, and other parameters
outputs：peak value of network traffic within the next time slice
1. Divide the data set into train data, validation data, and test data
2. Build the LSTM model
3. Initialize the parameters of the LSTM model
4. For $i = 0,1,\ldots,N^{ite}-1$ do /*training*/
5.    For each train data $\left(x_i^{train}, y_i^{train}\right)$ do
6.        Fit LSTM model for $x^*$ with train data $x_i^{train}$
7.        Evaluate the loss value using $x^*$ and $y_i^{train}$ according to (21)
8.        Use the loss value to propagate back
9.        Use the optimizer to step through and update parameters in LSTM model
```

```
10.    For each validation data $\left(x_i^{validation}, y_i^{validation}\right)$ do
11.        Fit the LSTM model for $x^*$ with validation data $x_i^{validation}$
12.        Evaluate the loss value using $x^*$ and $y_i^{validation}$ according to (21)
13.        If the loss value is less than the historical values
14.            Save this LSTM network model
15. Fetch the optimal LSTM model /*predicting*/
16. For each test data $\left(x_i^{test}, y_i^{test}\right)$ do
17.    Fit the optimal LSTM model for $x^*$ with test data $x_i^{test}$
18.    Evaluate the loss value using $x^*$ and $y_i^{test}$ according to (21)
```

In terms of complexity from bottom to top, considering N^{ite} as the number of iterations, and N^{train} as the number of train data, the complexity of the train process is in the order of $O\left(N^{ite} \cdot N^{train}\right)$. Considering $N^{validation}$ as the number of validation data, the complexity of the validation process is in the order of $O\left(N^{ite} \cdot N^{validation}\right)$. Considering N^{test} as the number of test data, the complexity of the predicting process is in the order of $O\left(N^{test}\right)$.

5. Experimental Simulation and Analysis

5.1 Experimental Environment Setting

We performed the experiment in Python, created the LSTM neural network in Pytorch, and conducted it on a computer with Intel Core I5-8250U 1.8 ghz CPU and 8 GB memory. The star topology network structure is consistent with [30], containing ten servers, one switch, and ten links. The properties of each server node and physical link are shown in **Table 2** below.

Table 2. Host and link Properties

Host Properties Host ID	1	2	3	4	5	6	7	8	9	10
Node CPUs	10	9	8	7	10	9	9	9	8	9
Link BW	1000	1000	500	400	500	600	700	800	900	600
Link Latency	30	50	10	50	50	50	50	50	50	50

We set up eight different types of VNF in the experiment, set resource requests for each VNF according to [30], and showed the VNF resource demand in Table 3. We randomly select many VNFs from the eight types of VNF to generate an SFC request. We generated six SFC requests in the experiment. To test the performance of the algorithm under different network loads, we changed the length of the SFC requests. The length of the time slice is one hour in the experiment.

Table 3. VNF Properties

VNFD Properties / VNFD ID	1	2	3	4	5	6	7	8
CPUs Required	4	3	3	2	2	2	1	1
BW Required	10	8	6	2	2	2	2	2
Processing Latency	10	8	6	2	2	2	2	2

In (20), γ and θ are constant coefficients. To increase the punishment for passive migration, they are set to 0.2 and 0.8 respectively. In (5), η_1, η_2, and η_3 are constant coefficients. To increase the penalty for passive migration, they are set to 0.2, 0.3, and 0.5. In (4), we set $p_{\min}$ to 0.2, and set $p_{\max}$ to 0.8.

We adopted the Adam Optimizer algorithm for optimizing the iterative updating method of weight parameters. The main parameters of LSTM are shown in Table 4.

Table 4. Main parameters of LSTM

Hyper parameter	Value
Learning Rate	0.0001
Batch Size	8
LSTM Layers	2
LSTM Hidden Size	50
Embedding Size	10
Training Count	1000
Input Size	4
Output Size	1
Discard Rate	0.2
Time Slice Count for One Day	24

5.2 Contrast Algorithm

To evaluate the effectiveness of the method, we selected the original LSTM and the ABCNN-LSTM [21] as the comparison algorithms. Both of them compute loss with the difference between predicted value of network traffic and actual value of network traffic, without considering the operation of the NFV network.

5.3 Evaluation Index

The related methods are compared in the following five aspects:

① predicted value of network traffic;

② loss according to (21);

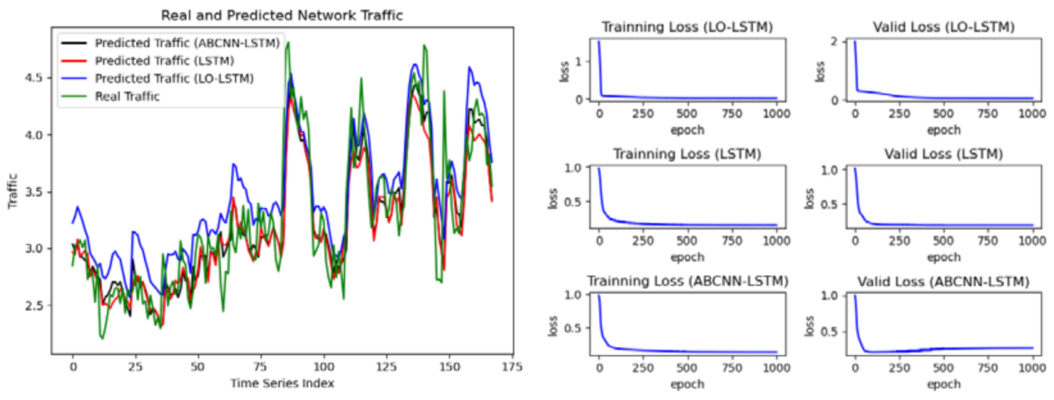
③ energy consumption increment. If the predicted value of network traffic is higher than the actual value, set the energy consumption increment to $\Delta PW\_running(T_q)$; otherwise, set it to 0.

④ number of passive migration according to (6);

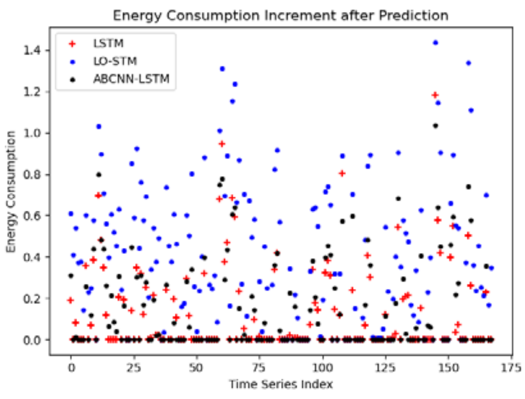
⑤ number of migration failures according to (7).

5.4 Analysis of Experimental Results

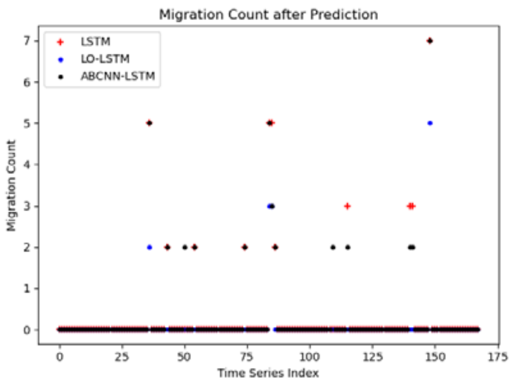
We adjust the network load by changing the length of SFC, and compare three groups of experiments: ① the length of SFC is six, and the results are shown in **Fig. 2**; ② the length of SFC is three, and the results are shown in **Fig. 3**; ③ the length of SFC is a random value between two and six, and the results are given in **Fig. 4**.



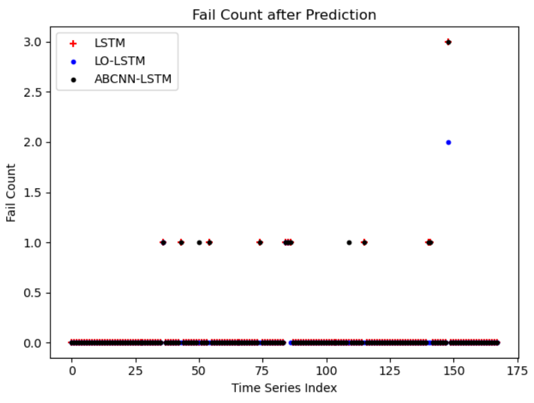
(a) (b)



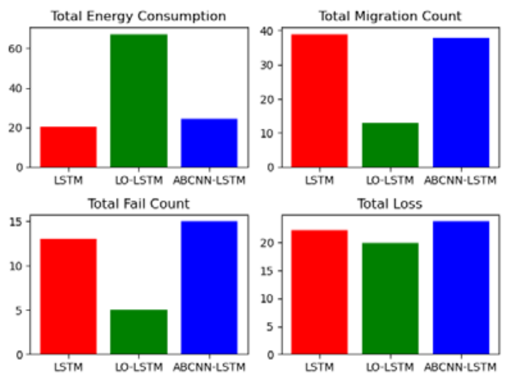
(c)



(d)

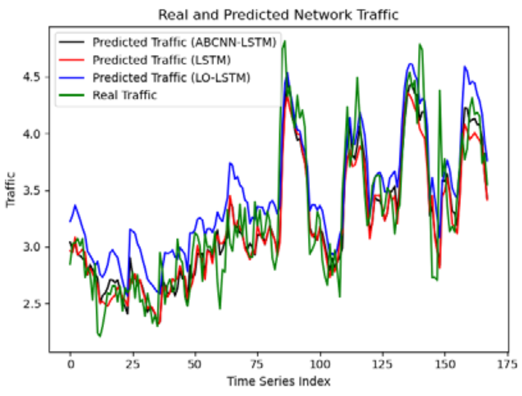


(e)

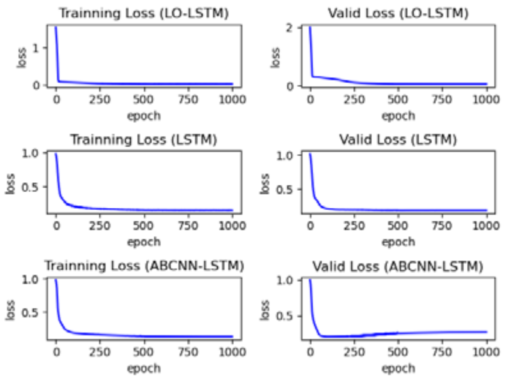


(f)

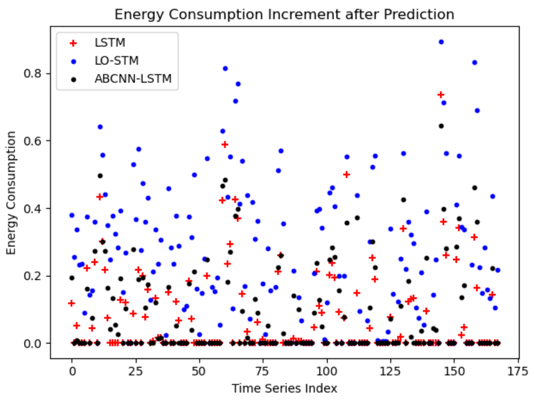
Fig. 2. Experiment results when the length of SFC is six



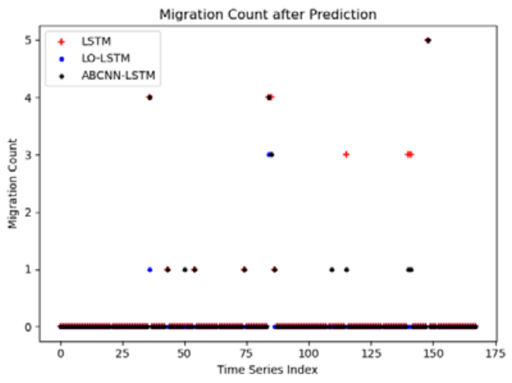
(a)



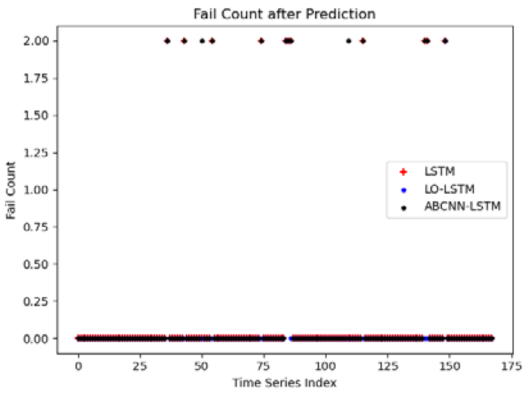
(b)



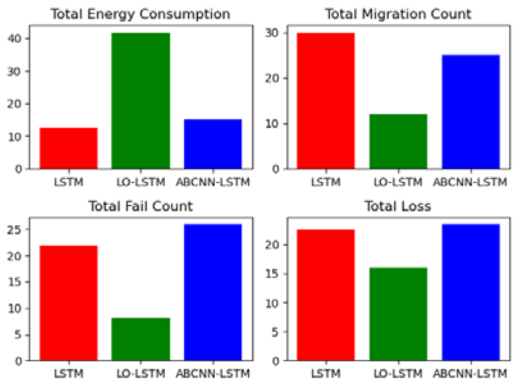
(c)



(d)

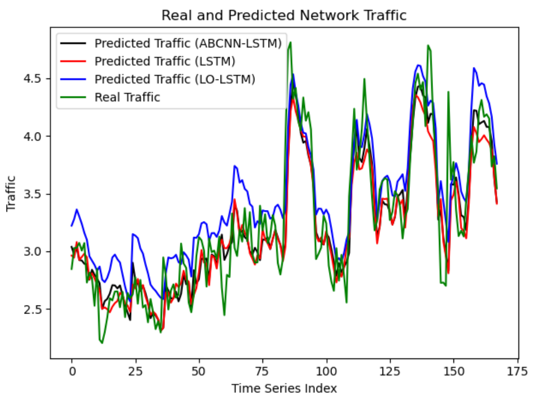


(e)

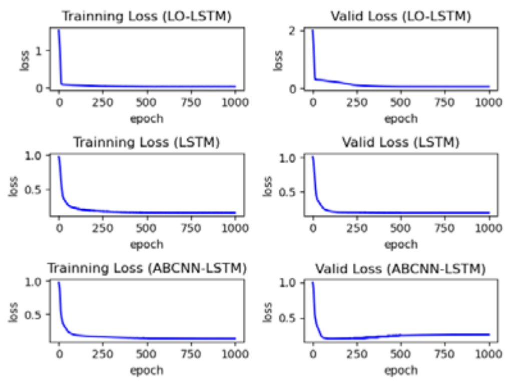


(f)

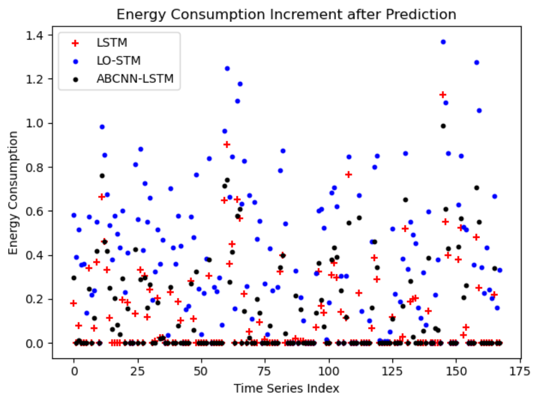
Fig. 3. Experiment results when the length of SFC is three



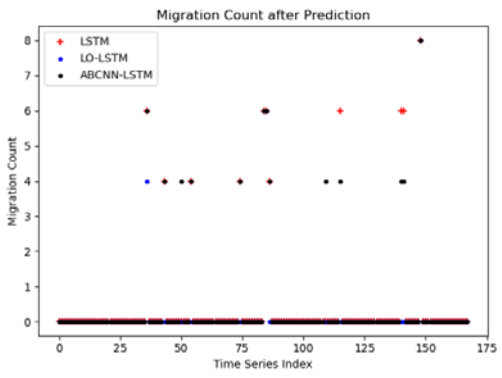
(a)



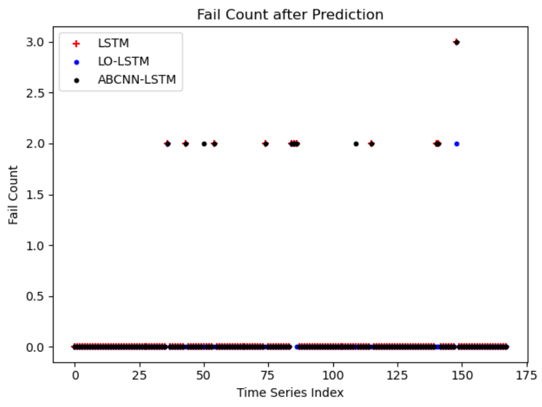
(b)



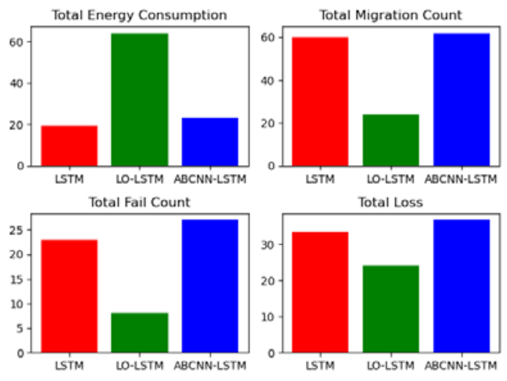
(c)



(d)



(e)



(f)

Fig. 4. Experiment results when the length of SFC is between two and six

As shown in **Fig. 2** (a), **Fig. 3** (a), and **Fig. 4** (a), predicted value of network traffic by the LO-LSTM algorithm is higher than other algorithms in general. The LO-LSTM algorithm computes loss function based on three aspects, including energy consumption, migration frequency, and migration failure, and it tends to amplify the predicted value in order to reduce migration.

As shown in **Fig. 2** (b), **Fig. 3** (b), and **Fig. 4** (b), the value of the loss function drops rapidly during training and validating. Therefore, the three algorithms all have great convergence performance.

As shown in **Fig. 2** (c), **Fig. 3** (c), and **Fig. 4** (c), the LO-LSTM algorithm has a higher energy consumption increment. The predicted value of the LO-LSTM is higher than other algorithms. Therefore, the energy consumption increment of the LO-LSTM algorithm is higher than other algorithms.

As shown in **Fig. 2** (d), **Fig. 3** (d), and **Fig. 4** (d), the LO-LSTM algorithm has a lower passive migration frequency. The predicted value of network traffic is higher than the actual value, and it is less likely to occur passive migration within the time slice.

As shown in **Fig. 2** (e), **Fig. 3** (e), and **Fig. 4** (e), the LO-LSTM algorithm has fewer passive migration failures within the time slice. The passive migration is less, which makes the migration failures fewer.

Fig. 2 (f), **Fig. 3** (f), and **Fig. 4** (f) show the weighted sum of all indices. The indices include energy consumption increment, the number of passive migration, the number of migration failures. We combine the multiple factors according to (5). As shown in the figures, the LO-LSTM algorithm performs better under different network loads.

6. Conclusion

We introduced LO-LSTM, an efficient network traffic prediction method to achieve migration reducing. We added factors including energy consumption and migration into the loss function of the LSTM model. We implemented LO-LSTM and evaluated it. Our experiment results show that LO-LSTM significantly achieves efficient passive migration reducing for VNFs. This paper provides an effective solution of network traffic prediction for migration reducing in NFV environment, which can give better quality of service.

In future research, we will try to solve the VNF online migration problem in the NFV environment, and reduce passive migration and save energy as far as possible.

Acknowledgement

The authors thank the editor and the anonymous reviewers for helpful comments and suggestions. The work was supported by the National Natural Science Foundation (NSFC) under grant (No. 61902361), Henan Provincial Department of Science and Technology (No. 222102210025, No. 222102210233).

References

[1] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck and R. Boutaba, "Network function virtualization: state-of-the-art and research challenges," *IEEE Commun. Surv. Tutor.*, vol. 18, no. 1, pp. 236-262, First Quarter 2016. Article (CrossRef Link)

[2] Fangyu Zhang, Hancheng Lu, Fengqian Guo, Zhoujia Gu, "Traffic Prediction Based VNF Migration with Temporal Convolutional Network," in *Proc. of 2021 IEEE Global Communications Conference (GLOBECOM)*, Madrid, Spain, December 2021. Article (CrossRef Link)

[3] Song Yang, Fan Li, Stojan Trajanovski, and Xiaoming Fu, "Traffic routing in stochastic network function virtualization networks," *J. Netw. Comput. Appl.*, vol. 169, pp. 1-11, November 2020. Article (CrossRef Link)

[4] Mirna Awad, Nadjia Kara, and Claes Edstrom, "SLO-aware dynamic self-adaptation of resources," *Future Gener. Comp. Sy.*, vol. 133, pp. 266-280, Augest 2022. Article (CrossRef Link)

[5] TianZhang He, Adel N. Toosi, and Rajkumar Buyya, "SLA-aware multiple migration planning and scheduling in SDN-NFV-enabled clouds," *J. Syst. Software*, vol. 176, June 2021. Article (CrossRef Link)

[6] Seyeon Jeong, Nguyen Van Tu, Jae-Hyoung Yoo and James Won-Ki Hong, "Proactive Live Migration for Virtual Network Functions using Machine Learning," in *Proc. of Int. Conf. Netw. Serv. Manag.: Smart Manag. Future Networks Serv. (CNSM)*, Izmir, Turkey, October 2021. Article (CrossRef Link)

[7] Fengsheng Wei, Shuang Qin, Gang Feng, Yao Sun, Jian Wang and Ying-Chang Liang, "Hybrid Model-Data Driven Network Slice Reconfiguration by Exploiting Prediction Interval and Robust Optimization," *IEEE Trans. Netw. Serv. Manage.*, vol. 19, No. 2, pp. 1426-1441, June 2022. Article (CrossRef Link)

[8] Morteza Golkarifard, Carla Fabiana Chiasserini, Francesco Malandrino and Ali Movaghar, "Dynamic VNF placement, resource allocation and traffic routing in 5G," *Comput. Networks*, vol. 188, April 2021. Article (CrossRef Link)

[9] Laaziz Lahlou, Nadjia kara and Claes Edstrom, "DAVINCI: online and Dynamic Adaptation of evolvable virtual Network services over Cloud Infrastructures," *Future Gener. Comput. Syst.*, vol. 127, pp. 396-408, February 2022. Article (CrossRef Link)

[10] Francisco Carpio, Wolfgang Bziuk and Admela Jukan, "Scaling migrations and replications of Virtual Network Functions based on network traffic forecasting," *Comput. Networks*, vol. 203, February 2022. Article (CrossRef Link)

[11] Runlong Xia, Yuantao Chen and Binbin Ren, "Improved Anti-Occlusion Object Tracking Algorithm using Unscented Rauch-Tung-Striebel Smoother and Kernel Correlation Filter," *J. King Saud. Univ.-Com.*, vol. 34, No.8, pp. 6008-6018, September 2022. Article (CrossRef Link)

[12] Jianming Zhang, Wenjun Feng, Tingyu Yuan, Jin Wang and Arun Kumar Sangaiah, "SCSTCF: Spatial-Channel Selection and Temporal Regularized Correlation Filters for visual tracking," *Appl. Soft Comput.*, vol. 118, March 2022. Article (CrossRef Link)

[13] Yuantao Chen, Linwu Liu, Volachith Phonevilay, Ke Gu, Runlong Xia, Jingbo Xie, Qian Zhang and Kai Yang, "Image super-resolution reconstruction based on feature map attention mechanism," *Appl. Intell.*, vol. 51, no. 7, pp. 4367-4380, 2021. Article (CrossRef Link)

[14] Jianming Zhang, Xin Zou, Lidan Kuang, Jin Wang, R. Simon Sherratt and Xiaofeng Yu, "CCTSDB 2021: A more comprehensive traffic sign detection benchmark. Human-centric Computing and Information Sciences," *Hum.-centric Comput. Inf. Sci.*, vol. 12, 2022. Article (CrossRef Link)

[15] H. Heffes and D. Lucantoni, "A Markov modulated characterization of packetized voice and data traffic and related satistical multiplexer performance," *IEEE J. Select. Areas Commun.*, vol. 4, pp. 856-868, September 1986. Article (CrossRef Link)

[16] A. Azzouni and G. Pujolle, "A Long Short-Term Memory Recurrent Neural Network Framework for Network Traffic Matrix Prediction," *arXiv*, 2017. Article (CrossRef Link)

[17] H. Tang, D. Zhou, D. Chen, "Dynamic network function instance scaling based on traffic forecasting and VNF placement in operator data centers," *IEEE Trans.Parallel Distrib. Syst.*, vol. 30, pp. 530-543, 2019. Article (CrossRef Link)

[18] H.N. Kim, D. Lee, S. Jeong, H. Choix, J. Yoo and J. Won-Ki Hong, "Machine learning-based method for prediction of virtual network function resource demands," *in Proc. of IEEE Conf. Netw. Softwarization: Unleashing Power Netw. Softwarization (NetSoft)*, Paris, France, June 2019. Article (CrossRef Link)

[19] S. Rahman, T. Ahmed, M. Huynh, M. Tornatore and B. Mukherjee, "Auto-scaling VNFs using machine learning to improve QoS and reduce cost," in *Proc. of IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, United states, May 2018. Article (CrossRef Link)

[20] V. Eramo, F. G. Lavacca, T. Catena and P. J. P. Salazar, "Application of a long short term memory neural predictor with asymmetric loss function for the resource allocation in nfv network architectures," *Comput. Networks*, vol. 193, July 2021. Article (CrossRef Link)

[21] Libin Liu, Hong Xu, Zhixiong Niu, Jingzong Li, Wei Zhang, Peng Wang, Jiamin Li, Chun Jason Xue and Cong Wang, "ScaleFlux: Efficient Stateful Scaling in NFV," *IEEE Trans. Parallel Distributed Syst.*, vol. 33, no. 12, pp. 4801-4817, 2022. Article (CrossRef Link)

[22] Mahsa Moradi, Mahmood Ahmadi and Rojia Nikbazm, "Comparison of Machine Learning Techniques for VNF Resource Requirements Prediction in NFV," *J. Netw. Syst. Manag.*, vol. 30, no. 1, 2022. Article (CrossRef Link)

[23] S. Gebert, R. Pries, D. Schlosser and K. Heck, "Internet Access Traffic Measurement and Analysis," in *Proc. of TMA 2012*, Vienna, Austria, pp. 29-42, 2012. Article (CrossRef Link)

[24] G. Maier, A. Feldmann, V. Paxson and M. Allman, "On dominant characteristics of residential broadband internet traffic," in *Proc. of ACM SIGCOMM Internet Meas. Conf. (IMC 2009)*, Chicago, IL, United states, pp. 90-102, November 2009. Article(CrossRef Link)

[25] M. Z. Shafiq, L. Ji, A. X. Liu and J. Wang, "Characterizing and modeling internet traffic dynamics of cellular devices," in *Proc. of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, pp. 305-316, 2011. Article(CrossRef Link)
[26] J. Pei, P. Hong, M. Pan, J. Liu and J. Zhou, "Optimal VNF Placement via Deep Reinforcement Learning in SDN/NFV-Enabled Networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 263-278, February 2020. Article(CrossRef Link)
[27] A. Gember-Jacobson, R. Viswanathan, C. Prakash, R. Grandl, J. Khalid, S. Das and A. Akella, "OpenNF: Enabling Innovation in Network Function Control," in *Proc. of the SIGCOMM Chicago 2014 and the Best of the Co-located Workshops*, vol. 44, no. 4, pp. 163-174, Chicago, IL, United states, August 2014. Article(CrossRef Link)
[28] K. Yang, H. Zhang and P. Hong, "Energy-aware service function placement for service function chaining in data centers," in *Proc. of IEEE Glob. Commun. Conf.(GLOBECOM 2016)*,Washington, DC, United States, December 2016. Article(CrossRef Link)
[29] J.A. Aroca, A. Chatzipapas, A.F. Anta and V. Mancuso, "A Measurement-Based Characterization of the Energy Consumption in Data Center Servers," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 12, pp. 2863-2877, December 2015. Article(CrossRef Link)
[30] R. Solozabal, J. Ceberio, A. Sanchoyerto, L. Zabala, B. Blanco and F. Liberal, "Virtual Network Function Placement Optimization With Deep Reinforcement Learning," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 2, pp. 292-303, February 2020. Article(CrossRef Link)

Ying Hu received the B.S. degree and the Ph.D. degree in Computer Science from Zhengzhou University, Zhengzhou, China, in 2003 and 2016, respectively. She is currently a Lecturer, Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou, China. Her current research interests include network virtualization and network function virtualization.

Liang Zhu received Ph.D. degree in computer science and technology from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in Octo-ber 2017. He is currently a lecturer with the Institute of Computer and Communication Engineering at Zhengzhou University of Light Industry, Henan, China. His current re-search interests include mobile social networks, personalized service recommendation, and privacy preserving.

Jianwei Zhang received his Ph.D. degree in computer application technology from PLA Information Engineering University in 2010. He is a professor at Zhengzhou University of Light Industry. His research interests include video object tracking and network security.

Zengyu Cai received his master degree in computer application technology from Northeast Normal University, Changchun, China, in 2006. He is an associate professor at Zhengzhou University of Light Industry. His research interests include computer vision, plan recognition and information security.

Jihui Han received the B.S. degree in Physics from Zhengzhou University, Zhengzhou, China, the M.S. degree and the Ph.D. degree in Theoretical Physics from Central China Normal University, Wuhan, China, in 2011, 2014, and 2017, respectively. He is currently a Lecturer, Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou, China. His current research interests include Complex Systems, Econophysics, Graph Representation Learning, Knowledge Graph.