An Evaluation of Network Service Monitoring Method Using User Traffic Information

Kei TAKESHITA
NTT Network Service Systems Laboratories
NTT Corporation
Tokyo, Japan
kei.takeshita.ud@hco.ntt.co.jp

Yuji SOEJIMA
NTT Network Service Systems Laboratories
NTT Corporation
Tokyo, Japan
yuji.soejima.pc@hco.ntt.co.jp

***Abstract*—In many cases, telecommunications carriers maintain their communication networks by monitoring the normality of equipment, which they assume represents normality of services. Since telecommunications carriers provide a network service to users, they want to directly manage and operate the service itself. However, a service monitoring method cannot be provided on a large scale due to the problems of system size and high cost. In recent years, with the spread of telemetry, individual users' traffic information can be obtained from routers at low cost, which means the service usage status of individual users can be monitored. On the other hand, it is difficult to know whether a service has failed or is simply not being used when no traffic is observed. To overcome this problem, we propose and evaluate a service normality check method by using time series prediction in this paper.**

***Index Terms*—Service monitoring, Service failure, Machine learning, Telemetry**

I. Introduction

Telecommunication carriers usually monitor the normality of devices for their network maintenance operation. Carriers usually use SNMP (Simple Network Management Protocol) to monitor the normality of network devices such as the health of each module and the resource values of the devices (CPU, memory, session, etc.). However, carriers should preferably monitor the normality of the network service itself. For example, carriers collect and monitor the number of active users or time taken to establish a session. For example, a carrier collects and monitors the number of users of the network, the usage time, or the time taken to establish a session and then analyzes those data to detect abnormalities. [1]

Such service monitoring is widespread for systems composed of general-purpose servers such as web services but has not been used in network devices. This is because various kinds of monitoring software enabling service monitoring can be installed on a general-purpose server. Web service provides can also develop a monitoring program by themselves. Another reason is users are aggregated to their server located at some datacenters, so the scale of the system is usually smaller than that for a network service provider. On the other hand, network service providers have difficulty introducing service monitoring since most network devices have vendor-specific OSs (Operating Systems), so the providers cannot introduce the monitoring function by themselves. In addition, it is necessary to analyze where the user's traffic passes through a wide area network and how the service is provided. Integrating devices of different generations and vendors to create a service monitoring system is not easy. Therefore, many carriers only monitor the normality of device health by SNMP individually.

In addition, there are problems in device monitoring by SNMP. When setting the monitoring conditions, it is necessary to evaluate the impact of equipment failure on the user and set an appropriate monitoring level. For that purpose, the developer of a network service should list all patterns of failure and correctly evaluate the impact of each failure on the network service. However, since a network consists of multiple layers and each layer has functions such as redundancy that affect services, it is difficult to accurately evaluate impact and set an appropriate severity of alarm. Therefore, some alarms are set as low severity alerts for impact. In such a case, some failures are not properly notified to maintenance personnel at the NOC (Network Operation Center). As a result, this leads to silent failures (i.e., the service is affected without the maintenance personnel being made aware of the failure).

Furthermore, alarm monitoring might be overlooked when many alarms occur due to device operation, such as during troubleshooting. For example, unrecovered parts of modules might be overlooked. In such a case, the same state as that of the silent failure will occur. Once in a silent failure state, maintenance personnel cannot find failure in

[1] As in this example, we use "service monitoring" in this paper to refer to the method of monitoring the usage of a network rather than the normality of a device.

the framework of device monitoring [2] and will instead be reported by affected users, which will cause a prolonged service impact.

Although there are various methods to avoid silent failures, which are described in detail in Sec.II, there is no definitive technology for large-scale monitoring.

In recent years, new management technology for network devices has been developed that reviews the inefficiency of SNMP so that a network operator can acquire large amounts of data at a lighter weight. Standardizations include RFC8641 [1] and Streaming Telemetry [2], which are generally called telemetry, and the recent OSs of network equipment vendors support telemetry. As mentioned earlier, telemetry is a lightweight protocol that can reduce the load on the device when acquiring data. Therefore, a network operator can acquire the service status such as the traffic volume of each user, which cannot be accrued in SNMP due to the load. As a result, the usage status of the user can be understood, and we believe that services can be effectively monitored by monitoring the usage status.

However, when the traffic volume of a certain user is zero, it is impossible to distinguish whether the service is simply not used or cannot be used for some reason even if the user wants to use it. Some monitoring services have a mechanism in which a device that operates 24 hours a day is set up at a base, and if that device cannot be communicated with, the service can be judged to have failed. However, it is difficult to have a highly reliable device that operates 24 hours a day for every individual user or SMB (Small and Medium-sized Business. Therefore, in this paper, to distinguish between the case where a service is not used and the case where it cannot be used, we use time series prediction from the history of usage status. We propose a method for suspecting service failure when there is no traffic flow and evaluate it in a simulation.

This paper is organized as follows: In Section II, we present the related work. In Section III, we present the proposed method. We show the evaluation result in Section IV. We present our conclusions in Section V.

II. related work

A. Related studies about network service monitoring

In this subsection, we introduce the technology for monitoring network services.

- Flow and packet analysis
  Flow technology [3] and DPI (Deep Packet Inspection) technology are methods of monitoring the communication contents of the user. These technologies enable monitoring of not only the total usage volume but also quality indicators such as QoS (Quality of Service) or QoE (Quality of Experience) [4]. However, since these technologies have a large load on the network equipment, the network device greatly reduces the number of subscribers. There is also a technique to branch the signal with network TAP and introduce a dedicated device. In this case, the cost of the dedicated device also increases.
- Monitoring line
  This is a method of monitoring the communication status by preparing the same line as the user in the NOC and simulating the user's communication on that line. This technique is often introduced in networks that provide services such as video where not only communication but also network service quality is important. However, since this method only monitors one line of the network as long as it simulates the user, it is suitable for service monitoring of devices that have many users such as gateways with other networks. However, to monitor a large number of devices such as PE (Provider Edge) routers, a line must be drawn from each device, which is difficult to apply.
- Cooperation with call center
  In this method, the contents of inquiries to a call center are analyzed, and when the number of inquiries about failures increases, the analysis is performed in the inquiry area and the like to find equipment failures. However, this is a measure to be taken after a user's report is received, and prolongation of failure is unavoidable.
- Utilization of external information
  Network service anomalies can be found by analyzing the posts on SNS (Social Networking Services) when there are many negative posts about a company's network. This method requires that the network is large enough to generate posts on SNS and that the posts are numerous enough for anomalies to be detected statistically. Therefore, this method is only applied during a large-scale failure on a fairly large-scale network [5].

B. Related research about traffic prediction

This section introduces related research on the traffic prediction technology of networks, which is the technical viewpoint of this research.

Techniques for predicting network traffic have long been studied for, and a survey is summarized by Boutaba et al. [6], who roughly classified traffic prediction techniques into statistical prediction models and prediction models based on supervised machine learning. Statistical prediction models, such as the ARIMA (Auto Regressive Integrated Moving Average) model [7] and Holt-Winters model [8], define the structures that are characteristic of

[2]In SNMP, an alarm is often sent when the state is changed, and in many cases, an alarm is not issued periodically even if an abnormal state occurs.

the time series (such as autoregressive components, trend components, and seasonality) and learn the parameters for each components. The prediction model by machine learning analyze the data into multidimensional and non-linear without considering prior knowledge like time series characteristics and implicitly learns the characteristics related to the time series fluctuations to create a model for prediction.

There are various methods for machine learning, but according to Boutaba et al. [6], there are many reports that the machine learning model has higher accuracy than the statistical prediction model. Bermolen and Rossi [9] compared machine learning-based SVR (Support Vector Regression) and AR (Autoregressive model) for predicting the traffic flow of network devices and found that SVR was better.

In recent years, LSTM (Long Short-Term Memory) [10], [11] has been widely used as a machine learning model able to better process time series data. LSTM further improves the RNN (Recurrent Neural Network) that applies a neural network so that time series data can be used. The RNN allows the neuron to recursively input its own output into itself, thereby continuing to have the influence of the previous input data. Since the RNN has its own recursive input, it can theoretically have long-term data. However, in reality, as data is weighted in the learning process, the data disappears or is excessively affected. LSTM solves this problem by outputting past data without weighting only when necessary. LSTM has high accuracy and is a representative algorithm used in various time series data processing such as machine translation, marketing, and finance.

III. PROPOSED METHOD

Figure.1 shows the outline of the proposed method. In this system, it is assumed that all user traffic stored in the device for service monitoring is collected at all times by using technology such as the telemetry. In addition, it is assumed that system can continuously acquire the traffic volume for each user even if a certain user's subscribed network device was changed due to some reason such as a redundancy by an accommodation management system.

Whether or not the network service of user A is restored is determined by whether or not the traffic of user A can be observed on the network device side. If the traffic from user A can be observed, it can be judged that the service has been restored, but if the traffic cannot be observed, as described above, it cannot be distinguished that the failure recovery is not successful or just they are not using it.

Therefore, we construct a method to estimate future traffic volume by using the framework of machine learning based on the traffic information collected for each user. When a failure occurs, the traffic volume estimated by

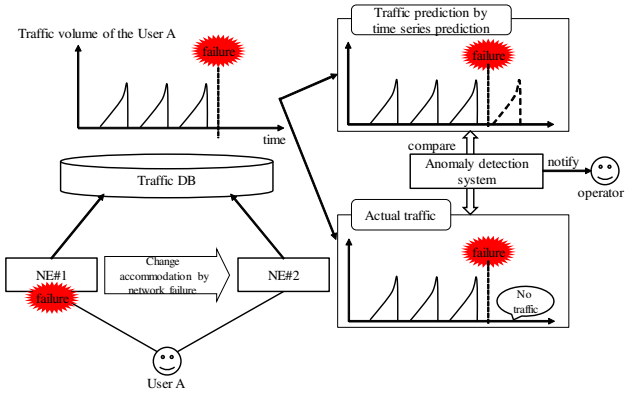

Fig. 1. overview of the method

the time series model from the traffic volume of the user before the failure and the actual traffic volume after the failure are compared for the users affected by the failure, and the users with actual traffic is smaller than prediction are suspected of failure. Therefore, users who do not generate traffic according to the prediction are excluded from the target, so as to avoid erroneously detecting a failure when the traffic volume is zero. The requirement for the detecting failure of the network service is to be set within a few minutes in this paper. This is because the definition of an accident to be reported to the Ministry of Internal Affairs and Communications is 1 hour [12], we set a few minutes for the detection phase.

We explain the methods of preprocessing, learning, and fault extraction.

First, Algorithm 1 shows the pseudocode of preprocessing and learning. In this method, the model is trained by using time-series data in which the traffic volume in 1 minute units for each user. We set the time unit to one minute, because we balanced the length so that the traffic prediction could be stable to some extent while satisfying the condition that the time requirement for failure detection was within a few minutes.

In general, telemetry allows to acquire data with a time granularity of less than a minute, we took the average value of ten second unit and then took the maximum value from those values as a statistical process to make the data of one minute or less into one minute unit. The reason for setting the maximum value is that we expected that the characteristics of the application such as voice and data transfer would be better. The prediction method uses LSTM, an ARIMA model that can perform time series prediction.

Next, Algorithm 2 shows the pseudocode for fault extraction. The input of this system is the detection by the device monitoring system

This system extracts users who are accommodated in

Algorithm 1 Process for creating prediction model

precondition, input: $u_i \in U(i = 0, n)$: an useru_i in user group U, $t_{u_i,j}(j = 0, m)$: traffic volume of $u_i$in time$j$[second]

output: $f(t_{u_i})$: prediction model of u_i

```
for i = 0 ... n do
    j = 0, k = 0
    while j < m do
        d_{u_i}, k = max{t_{u_i,j}, j = j, j+1, ..., j+59}
        j = j + 60, k = k + 1
    learning f(d_{u_i})
```

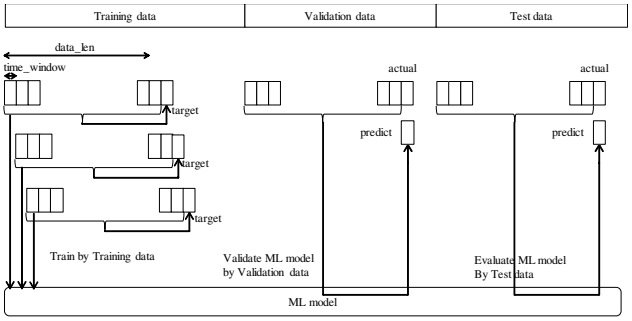

Fig. 2. Image of training data

Algorithm 2 Process for extracting silent failure

precondition, input: $Failureoccurredattimet.j = t, \ldots, t{+}60[minutes], thisalgorithmexplainsattimej = t + l$.

output: alarm at failure monitoring system

```
SuspiciousCount = 0, PredictedCount = 0
for i = 0 ... n do
    if l = 0 then
        p_{u_i,l} = f(d_{u_i,t-x}, ..., d_{u_i,t})
    else
        p_{u_i,l} = f(d_{u_i,t-x+l}, ..., d_{u_i,t}, p_{u_i,0}, ..., p_{u_i,l})
    if p_{u_i,l} > 0 then
        PredictedCount+ = 1
        if p_{u_i,l} > t_{u_i,l} × 2 then
            SuspiciousCount+ = 1
if SuspiciousCount/Predictedcount > threshold then
    send alarm to alarm monitoring system
else
    do nothing
```

the object to be monitored, such as devices and interfaces that have undergone an abnormality and have undergone accommodation changes or repairs. For each user, based on the traffic data of the time before the failure, the time series model is used to predict the traffic for a certain period from the occurrence of the failure, and the predicted value and the measured value are compared.

Since this method predicts the traffic of individual users and depends largely on the behavioral characteristics of individual users, it is difficult to predict it with high accuracy compared to the traffic that aggregates links and the like. Therefore, a certain number of users are aggregated by device or interface unit, and if the suspected users are above a certain amount, it is determined that the failure is suspected. In this paper, if the user*SuspiciousCount* whose actual value is less than half of the predicted value exceeds a certain *threshold* among the number of users *PredictedCount* whose predicted value is not zero, it is judged as failure suspect.

IV. Evaluation

A. Evaluation conditions

We used public Internet traffic data to evaluate the proposed method [13]. There are three days of evaluation data, and we used the first two days as learning data and the last day as test data.

An image of the training data is shown in Figure.2. We trained the model by dividing the traffic data by a certain time_window and generating data with a data length of data_length by shifting by 1 time_window on the basis of the evaluation data. To prevent over-learning, we abort learning when the prediction performance of the prediction model using the verification data stops improving, even if the fixed number of times of learning has not ended (early end). The verification data used was 10% of the evaluation data.

Then, we evaluated the performance of the proposed method at detecting failures by using test data. For the purpose of evaluation, it is assumed that a failure occurs at random times in the test data, and the accommodated users are dispersed and saved in another four interfaces. At that time, it is assumed that one interface is not restored normally and the traffic of the user assigned to it is measured as zero. Under this condition, while varying the number of users in Table I, we evaluated whether the failure event could be detected, whether false detection occurred in other interfaces, and how long the failure took to be detected.

In addition, to understand the relationship between model complexity and performance in the LSTM and ARIMA models, the number of hidden layers in LSTM and the calculation range of ARIMA were varied as shown in Table I to evaluate the prediction performance. Note that we used the settings in bold if there are no annotations in other evaluations.

Note that the LSTM algorithm uses the method in Gers et al. [11], the neurons are fully connected, and the activation function is ReLU (Rectified Linear Unit). In the learning of the model, we performed dropout for 50% of

the neurons and applied early termination with a learning rate of 0.01 and an upper limit of 20. The batch size for batch learning was 300 and the threshold was 0.5.

TABLE I
PARAMETER

Parameter	Value
Number of subscribers (n)	20, 60, 100, **200**
Number of hidden layers of LSTM	**1**, 2, 3, 4
Autocorrelation of ARIMA	**1**, 3, 5
Moving average of ARIMA	**1**, 3, 5

We used detection accuracy and time to detection as evaluation indices. As an index of detection accuracy, the detection rate is the rate at which a failure can be correctly extracted for one interface that did not recover normally in Table II below (TP / (TP + FN)). In addition, the false detection rate is defined as the ratio of failures to the three interfaces that have been restored normally (FP / (FP + TN)). Since this system aims to increase the fault detection rate as much as possible while suppressing false detections, both the detection rate and false detection rate indices must be sufficiently high.

TABLE II
ACCURACY

		Actual results	
		failure	not failure
predicted as	failure	TP	FP
	not failure	FN	TN

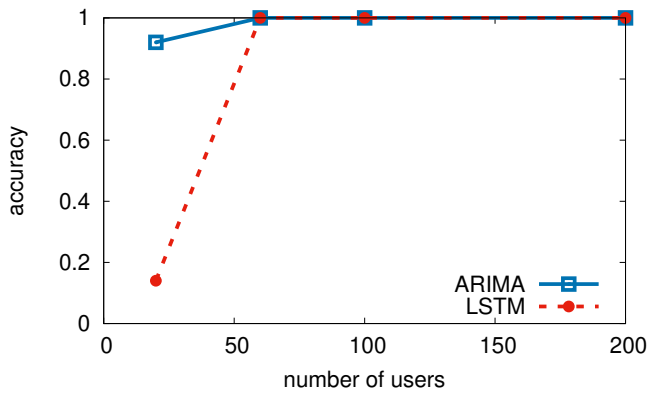

Fig. 3. Detection rate of failure

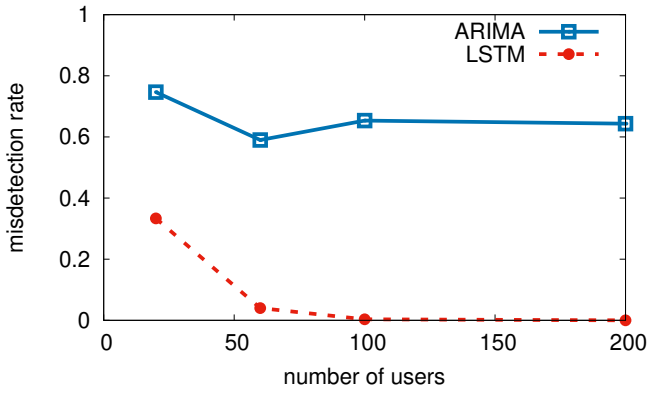

Fig. 4. False detection rate of failure

B. Evaluation result 1: failure detection

The detection rate and false detection rate for each method of ARIMA and LSTM are shown in Figures 3 and 4, respectively. These graphs show precision and accuracy for ARIMA and LSTM for 1000 failure events. In either method, both the detection rate and the false detection rate improve as the number of users increases, and after the number of users exceeds 100 (the number of users per interface is about 25), almost all faults can be extracted. Regarding the detection rate, ARIMA had higher detection accuracy even with a small number of users. On the other hand, the false positive rate was suppressed lower in the LSTM model, and was almost 0% when the number of users exceeded 100. This is assumed to be because ARIMA produces a higher predicted traffic volume than LSTM, and the number of suspected users (*PredictedCount*) tends to increase, so a failure has a higher possibility of being determined.

Next, Figure 5 shows the time required to detect a fault condition. The time taken to detect a failure state is the average of the time [minutes] required to determine this failure for the first time by executing this method every minute. The confidence intervals in the figure were calculated at 95% on both sides. Although the number of users was small, the failure state detection was delayed due to the failure of prediction, and as a result, the average value was also large. On the other hand, as the number of users increased, failures could be detected immediately in almost all patterns, and the goal of this use case, which was detection within a few minutes, was satisfied.

C. Evaluation result 2: parameter tuning for each model

Next, we show the results of parameter tuning for each method. In this evaluation, we clarify whether the accuracy improves as the structure is deepened in each of the LSTM and ARIMA models. As shown in Table I, the evaluation was performed by changing the layer depth of the neural network for LSTM and the range MA (q) of the moving average for the ARIMA model. We used RMSE (Root Mean Squared Error), which is the error of the test data, as the evaluation index of accuracy. Note that each is normalized by the results of the parameters in bold in Table I.

Figure 6 shows the results of varying the parameters for LSTM. According to this, the error function did not

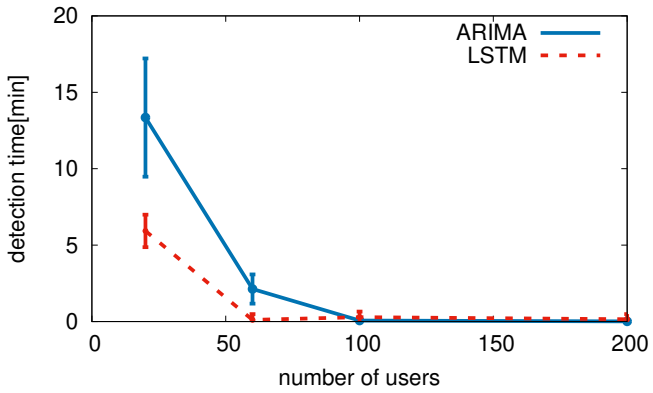

Fig. 5. Time taken to detect a failure

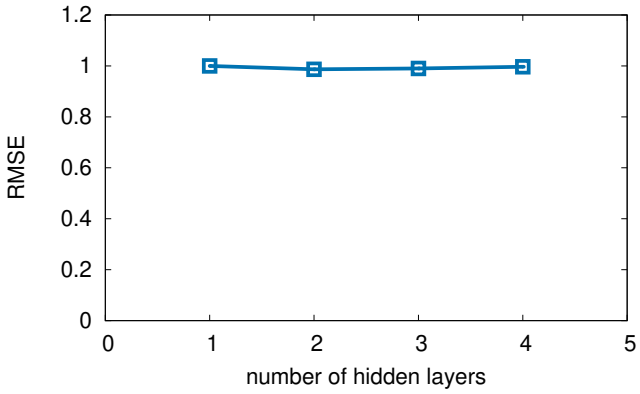

Fig. 6. LSTM

improve even if the layers were deepened or the number of neurons was increased. This is because there is no deep feature that can be extracted from a simple one-dimensional time series model, and only simple autoregressiveness or periodicity is extracted. Therefore, to further improve the accuracy, new features need to be introduced

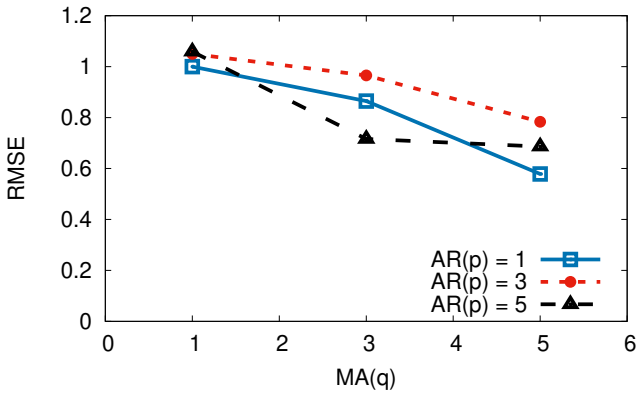

Fig. 7. ARIMA

such as application information and user attributes. Figure 7 shows the results of varying the parameters for ARIMA. It was found that the error function does not improve even if AR (p) is increased, but the accuracy improves as MA (q) is increased. Our future task is to clarify the relationship between parameters and accuracy and improve the prediction accuracy.

V. Conclusion and future work

In this paper, we proposed a method to judge the normality of a network by monitoring the user's communication status in addition to the maintenance work of the existing device alarm-based communication network. We also showed its effectiveness in a simulation. In the future, we plan to conduct a detailed study to improve the traffic prediction technology. In addition, since it is easily assumed that user stops using the network once failure has occurred, it is necessary to examine whether failures can be successfully extracted in such conditions. Therefore, we think that it is necessary to evaluate by actual data.

References

[1] "Subscription to YANG Notifications for Datastore Updates." [Online]. Available: https://rfc-editor.org/rfc/rfc8641.txt

[2] "Streaming telemetry." [Online]. Available: http://www.openconfig.net/project/telemetry/

[3] "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information." [Online]. Available: https://rfc-editor.org/rfc/rfc70111.txt

[4] Y. Chen, K. Wu, and Q. Zhang, ""from qos to qoe: A tutorial on video quality assessment"," IEEE Communications Surveys Tutorials, vol. 17, no. 2, pp. 1126–1165, Secondquarter 2015.

[5] K. Takeshita, M. Yokota, and K. Nishimatsu, "Early network failure detection system by analyzing twitter data," in 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), May 2015, pp. 279–286.

[6] R. Boutaba, M. A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, and O. M. Caicedo, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," Journal of Internet Services and Applications, vol. 9, no. 1, p. 16, 2018.

[7] G. E. P. Box and G. Jenkins, Time Series Analysis, Forecasting and Control. USA: Holden-Day, Inc., 1990.

[8] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," International Journal of Forecasting, vol. 20, no. 1, pp. 5 – 10, 2004.

[9] P. Bermolen and D. Rossi, "Support vector regression for link load prediction," in 2008 4th International Telecommunication Networking Workshop on QoS in Multiservice IP Networks, Feb 2008, pp. 268–273.

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

[11] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," Neural Computation, vol. 12, pp. 2451–2471, 1999.

[12] "Reports of serious accident from Ministry of Internal Affairs and Communications(in Japanese)." [Online]. Available: https://www.soumu.go.jp/menu_seisaku/ictseisaku/net_anzen/jiko/judai.html

[13] J. S. Rojas, Á. R. Gallón, and J. C. Corrales, "Consumption behavior analysis of over the top services: Incremental learning or traditional methods?" IEEE Access, vol. 7, pp. 136 581–136 591, 2019.