A Deep Learning Approach to VNF Resource Prediction using Correlation between VNFs

Hee-Gon Kim*, Se-Yeon Jeong*, Do-Young Lee *, Heeyoul Choi§, Jae-Hyung Yoo¶, James Won-Ki Hong*

*Computer Science and Engineering, Pohang University of Science and Technology, Pohang, South Korea
{sinjint, dylee90, jsy0906, jwkhong}@postech.ac.kr

§Handong Global University, Pohang, South Korea
hchoi@handong.edu

¶Graduate School of Information Technology, Pohang University of Science and Technology, Pohang, South Korea
styoo@postech.ac.kr

***Abstract*—Software-Defined Networking (SDN) and Network Function Virtualization (NFV) greatly facilitate network service management. Specifically, these new network paradigms help manage the network environment dynamically and cost-efficiently. Virtual Network Function (VNF) and Service Function Chaining (SFC) are important aspects of the NFV environment. In terms of NFV management, resource demand of VNFs can be predicted at a future time to handle Quality of Service (QoS) and resource allocation problems efficiently. Hence, researchers study and build a management system where machine-learning-based predictions of VNF information are used to handle auto-scaling, deployment and migration of VNFs. In addition, in recent studies, these systems have involved SFC to obtain useful information, not just a lone VNF. However, not many of studies explain clearly how chaining dependency among VNFs in a SFC can be used to predict future resource demand of a VNF. In this paper, we introduce VNF resource prediction machine learning model that maximizes the benefits of using SFC. Then, we compare several machine learning models and analyze how SFC data can help predict resource usage patterns of VNFs. We also show benefits of Attention model to improve prediction accuracy and convergence time through experiments.**

***Index Terms*—Virtual Network Function, Service Function Chaining, Resource Prediction, Deep Learning**

I. Introduction

Traditional networks consist of physical middleboxes which are Firewall, Load Balancer, Intrusion Detection System (IDS), and other essential network functions. As the middleboxes are hardware, they make a significant time loss and lead additional operational cost when administrator needs to control them. However, with the introduction of Network Function Virtualization (NFV), these problems are solved. NFV virtualizes the existing middleboxes to Virtual Network Function (VNF)s which are the software function. VNF provides flexibility to network managements.

Service Function Chain (SFC) is one of the important concepts in NFV. SFC comprises an ordered chain of several VNFs and, conceptually, SFC is a unit that provides a network service to users. While a network service (SFC) is being provided to users, VNFs use computing and network resources in the infrastructure (e.g., Data center) and receive traffic from other VNFs. Thus, each VNF is impacted by another VNFs that belong the same SFC, and this characteristics are important information for intelligent network managements.

In intelligent network managements, administrator prevents the network problems by predicting the errors and reconfiguring environments. However, as networks environments can be changed, administrator also needs to predict network environment changing. In NFV environment, VNFs conditions are predicted by network manager and SFCs become the important information to predict VNF conditions.

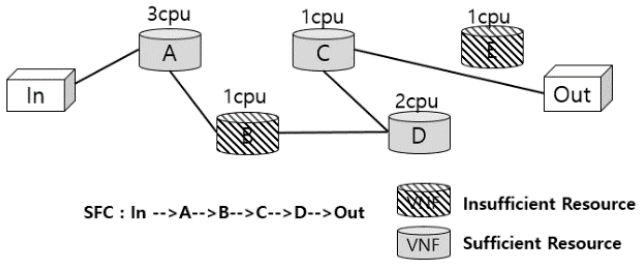

Fig. 1: Service Function Chaining

The Fig. 1 shows the example of SFC. In Fig. 1, there are five VNFs (A, B, C, D, E) with different resources. SFC consists of four VNFs as (A→ B→ C→ D) and VNF E is not chained. Fig. 1 also shows resource conditions for each VNF. VNFs A, C, D have sufficient resources, while B has insufficient resources. For intelligent network managements, administrator should predict that each VNFs' resource conditions. Network administrator can use historical resource usage data and extract meaningful patterns in the resources. However, there are challenges in this method. For example, as shown in Fig. 1, VNF B does not work properly due to resource deficiency. As one VNF fails can lead to trash traffic or breaks connection to other VNFs, we cannot know that this

978-1-5386-9376-6/19/$31.00 ©2019 IEEE

monitored resource pattern is appropriate when another VNFs does not work well or all VNFs have no problems. Thus, the VNF prediction should base on the SFC units and it also could improves the prediction accuracy comparing the methods that use one VNF history information only.

In this paper, we use SFC data to predict VNF resource demand. We introduce and compare the several neural network models in ability to leverage SFC data satisfactorily, and look their performance. Also, as VNFs and SFCs has different characteristics, we analyze which machine learning models is good for specific VNFs and SFCs. The main contribution of our study is that analysis about how SFC data affects VNF resource prediction.

II. Related Work and Background

A. Related Work

The network environment changes dynamically and prediction is difficulty. Also, the larger the network size, the harder it is to understand. NFV can help the administrator to manage the network, but due to the complexity above, it does not completely alleviate the need for professional management personnel and its related, elevated cost. Thus, recently machine learning is drawing attention from researchers regarding intelligent network management technology.

These papers [3] [4] [5] [6] [7] [8] are examples of network management using machine learning. Reinforcement learning [9] is used in [3] that introduces VNF resource scaling using reinforcement learning. This paper sets up the VNF resource conditions, actions and rewards. The authors defined actions for resource scaling in and out and rewards based on network delay and resources usage percentages. [4] shows the results on VNF auto-scaling using Random Tree, J48, PEPTree, Random Forest, Decision Table, Multi-Layer Perceptron, Bayesian Network. In addition, this work suggests two algorithms, QoS-prioritized ML and Cost-prioritized ML.

These papers [5] [6] [7] [8] use machine learning techniques, as well as SFC data, not just VNF data. [5] suggests an NFV workflow using MDP. The author sets up a state for MDP as NFV is allocated to a specific cloud. The action is the NFV allocation to a specific cloud. The transition probability is calculated using Bayesian algorithms as inference method.

[6] focuses on SFC creation problems. The paper introduces a problem in which VNFs must be selected to form an SFC with several VNFs of the same type. The authors use a deep belief network (DBNs) to set the initial parameter values and perform learning with back propagation for fine-tuning. They use a DBN for unsupervised learning only using the current SFC processing stage, namely the current VNF, and generate the next VNF. After the DBN, fine-tuning is conducted, using labeling data that includes the current VNF and the correct next VNF. If the data is sufficient and labeling is correct, then a DBN is not required and a Recurrent Neural Network (RNN) is sufficient to improve performance.

[7] suggests random selection algorithms. The objective of this paper is to optimize service function chain placement in a dynamic cloud environment. The authors suggest that random selection algorithms are useful compared with integer linear programming in terms of convergence time. In addition, using support vector regression (SVR) to make predictions on cloud resources and using this prediction can help the optimization significantly in a dynamic environment.

[8] predicts resource usage for each VNF and uses the result to reduce delays and costs. The authors suggest using GNN models; here graphically adjacent VNFs are used to predict a target VNF. In addition, as in [5] [6] [7], historical resources data is being used. However, a GNN model is a type of FNN that is not adequate for learning sequence data, in contrast with RNNs. In addition, similarly to [5] [6] [7], the data used here is time sequence data, not SFC sequence data. SFC data was just embedded, and it is stacked, not built in sequences.

In this paper, we suggest using SFC sequence data and embedding historical data. This data configuration could help the model to understand SFC and achieve satisfactory prediction accuracy. In addition, we highlight interdependences between VNFs using Attention [2] models. We can confidently point out that the relationship between VNFs has not been accurately ascertained yet using machine learning. Our approach to using SFC sequence data is also unique.

B. Background

1) Target-Dependent LSTM: In this paper, we use a deep neural network (DNN) model and use labeling data for supervised learning. A neural network (NN) is one of the most commonly used machine learning models aiming to mimic the neural system for learning. An NN has several layers, with input data being transferred to the higher layers and modified by several parameters to obtain a result. A DNN is a term used for an NN that has many layers (Fig. 2). We use a modified long short term memory (LSTM) model [10], a type of RNN. An RNN (Fig. 2) is an NN model that is effective with sequence data. RNN does not only transfer data to the higher layers but also to layers at the same height. This structure allows RNN layers to get two inputs as current and past data. RNN may have many structures based on the input and output data vectors, and we make use of many of the RNN structures.

LSTM is an improvement on RNN and it has an input, output and forget gate inside each RNN cell. These gates help to solve the gradient vanishing problem [10]. In this paper, we use several LSTM models and point to those with good accuracy. We use several different LSTM models to predict the VNF resource usage, compare how effective models are in learning SFC data and identify the drivers for relative performance. The LSTM models used are LSTM, target dependent LSTM (TD-LSTM) [1] and bi-directional LSTM [11]. Fig. 3 and Fig. 4 show a TD-LSTM and a bi-directional LSTM. The TD-LSTM has two layers, a left layer and a right layer. This model does not use the input data directly, but instead uses it separately. As in Fig 3, the input data is separate based on target data. One important point is that the input direction of the separated data must be opposite to the right layer. Bi-directional LSTM does not separate input

data, but instead inverts the data and inputs it again as per Fig. 4. The rationale for using these LSTM variants is to prevent unbalanced learning of data and improving learning performance for individual VNFs while using global SFC data.

2) Attention: Fig. 5 shows the Attention model [2]. The Attention model learns the outputs of a layer that are more valuable and linked to results. The output of each layer is multiplied by the alpha parameter and go activation functions. After learning, the models derive the most optimized alpha parameters. We use this Attention model to improve the learning process. Our model can learn what VNF data is important for prediction for a target VNF and help visualize the relationship between VNFs.

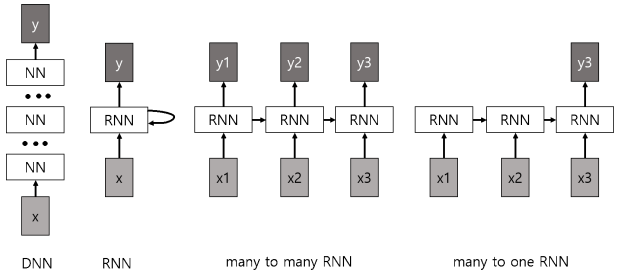

Fig. 2: Deep Neural Network and Recurrent Neural Network

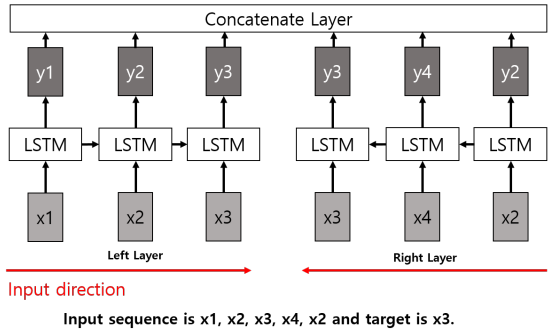

Fig. 3: Target-Dependent LSTM

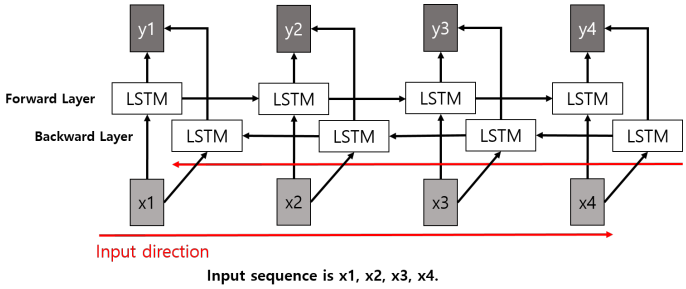

Fig. 4: bi-directional LSTM

III. OpenStack Environment

We first describe the evaluation testbed based on OpenStack. For the deployment of the OpenStack Pike release, we used

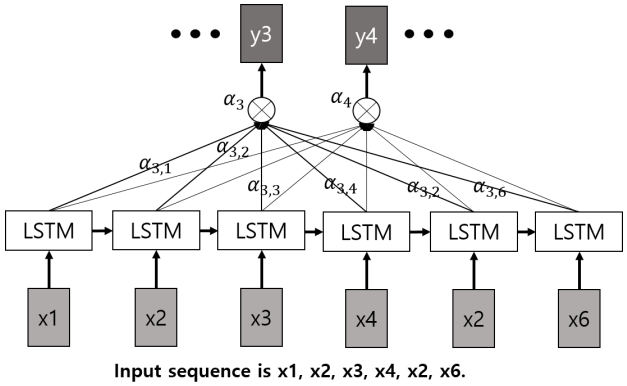

Fig. 5: Attention based LSTM

two servers (Intel Xeon X5650 2.67 GHz with 6 cores and a 1-Gbps Ethernet card) as an OpenStack controller node and a compute node, respectively.

For the practical evaluation and verification of our approach, we deployed two types of SFCs in the OpenStack testbed by chaining different types of VNFs with different chain lengths. Each VNF, as a dedicated network function, ran within a Virtual Machine (VM) hosted in the compute node. We used the OpenStack networking-sfc component that supports OpenStack APIs to build an SFC path by chaining the network interfaces of the VMs, SFC-target traffic classification and their redirection to the SFC path, based on flow rules for virtual switches in the compute node (Fig. 6).

SFC comprised a sequential chain of five separate VNFs: A firewall, a NAT, an IDS, a DPI and a load balancer. Detailed descriptions with information on the VNFs used can be found in Table I. We used apache ab, a webserver stress tool, to generate traffic passing through each SFC. After passing through the related VNFs in order, the traffic generated arrived at each load balancer VNF where the huge amount of HTTP requests are distributed into the destination web servers in a round-robin manner. CPU utilization data was collected every 5 seconds during 12 hours from the composing VNFs. We also used collectd as a resource monitoring agent on each VNF.

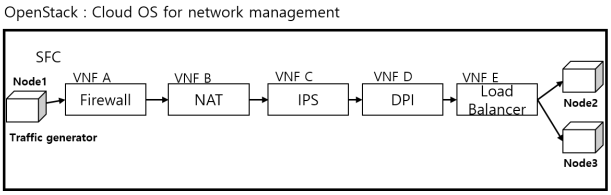

Fig. 6: SFC environment

IV. Problem Description

A. Problem Definition

Each VNF is connected to and from an SFC, and we define what VNFs are used and the composition of the SFC in terms of VNFs. In Fig. 6, VNF A, B, C, D, and E form SFC. SFC receives traffic from Node1 and sends traffic to

TABLE I: Overall information of used VNFs

Functions	Opensource	vCPU (core)	vRAM (GB)
Firewall	iptables	1	2
NAT	iptables	1	2
IPS	Suricata	2	4
DPI	ntopng	2	4
Load Balancer	haproxy	1	2

Node2 and Node3. For this problem, we assume that each link between VNFs has sufficient capacity and only consider VNF resources.

We predict the resource usage for each VNF and use the data from the SFC to which the VNF belongs. For example, learning VNF A involves VNF A, VNF B, VNF C, VNF D, and VNF E as they all belong to the SFC. Each VNF data point includes CPU, memory, disk, and process number. Because these types of resource usage are related, much resource information could be used to predict one resource usage. However, to simplify, we only use and predict CPU data and usage, respectively.

B. Data Design

As we conduct supervised learning, we use a labeled data set as per Fig. 7. The data structure contains three categories of data, target, VNF, and labeling. The target refers to the VNF identification number for resource prediction. The VNF contains a lot of resource information on the target VNF and every other VNF belonging to the same SFC. Resource information for the VNF consists of the historical resource usage data, and each data point is represented as $C_{x,y}$. $C_{x,y}$, namely, the resource usage for VNF x at time $T - 5 * y$. For example, $C_{3,6}$ is the resource data for VNF C at time T-30. As we collect data for 12 hours, the time T can range from 30 seconds to 12 hours. Resource data for the VNF could be CPU, disk, memory, or other system resource usage such as process number and OS load; however, for simplicity, we only use CPU data. We use 5-fold cross validation for training and test. This method alleviates overfitting issues linked to the data distribution.

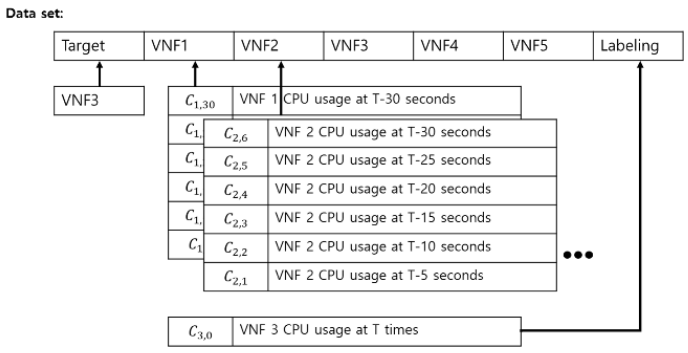

Fig. 7: Detail Input Data

V. Machine learning Model

We implement two data sets, SFC and History sequence data focusing on data direction. These data sets are shown in Fig. 8. SFC sequence data has VNF's historical data in vertically and SFC sequence data in horizontally. History data has SFC sequence data in vertically, historical data in horizontally. All existing studies leverage History sequence data, while we leverage SFC sequence data. Using this data, we can learn the relationship between VNF and provide satisfactory results.

We use SFC sequence data and compare four machine learning models an LSTM, a bi-directional LSTM, bi-directional LSTM+ and a CAT-LSTM(Content & Aspect Embedded Attentive Target Dependent Long Short Term Memory). We use CAT-LSTM from the our previous research, and this model consists of aspect embedding, a target dependent LSTM and Attention as shown in Fig. 9. The LSTM and bi-directional LSTM are already mentioned in Fig. 3 and Fig. 4. In this paper, we apply attention and aspect embedding for all LSTM models, so each model structures are quite similar to that in Fig. 9, with only one difference, namely, whether the LSTM is separated, not separated, or bi-directional. The bi-directional LSTM+ is a model that we make a change of bi-directional LSTM. While Bi-directional LSTM learns attention after concatenating the results of forward and backward LSTM cells, the bi-directional LSTM+ separately learns the forward and backward LSTM attention and concatenates results as CAT-LSTM do.

As we use Attention with the three models, many-to-many structures are used across the board, with each cell having a distinct output. The output size is decided by the LSTM hidden layer number and it is represented as n in $H_{x,n}$ in Fig. 9. All LSTM models reach their own result value for Attention and concatenate with the output of their last cell. If the LSTM structure is separated as for a CAT-LSTM, additional concatenation is required to yield single output values. Our optimization function is the root mean square error (RMSE). The models were learned in 60 iterations with the learning rate of 0.01.

VI. Evaluation

TABLE II: RMSE Comparison of SFC and historical sequence

	LSTM(SFC)	LSTM(History)
VNF A	8.81	13.47
VNF B	6.94	11.45
VNF C	8.22	6.78
VNF D	6.51	7.72
VNF E	7.18	12.30
ALL	7.10	29.53

First, we evaluate SFC sequence data set and historical sequence data set for prediction of all VNFs or individual VNFs. The results is showed in Table II. The first column of Table II lists prediction targets. If the target is one specific VNF, the model only predicted one specific VNF. If the target is ALL, the model predicted all VNF simultaneously.

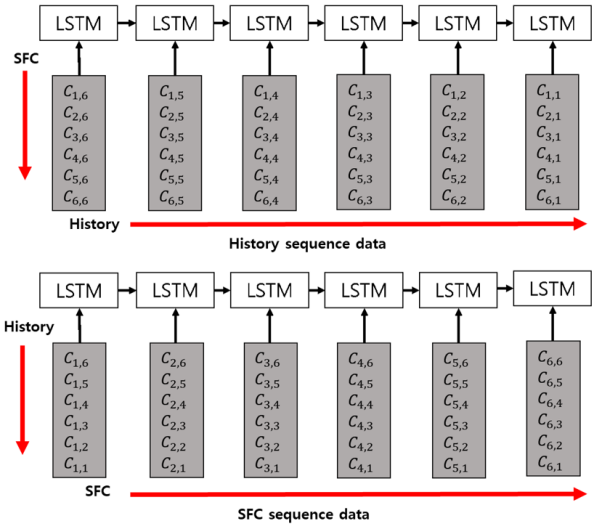

Fig. 8: Comparison of models that leverage historical sequence data and SFC sequence data

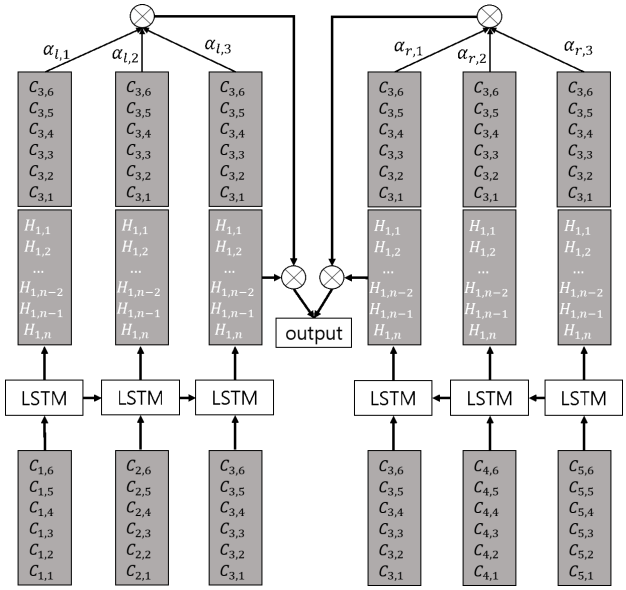

Fig. 9: CAT-LSTM

TABLE III: RMSE Comparison of Attention models

	CAT-LSTM	LSTM	bi-directional LSTM	bi-directional LSTM+
VNF A	8.64	8.81	8.65	8.51
VNF B	7.13	6.94	6.95	6.86
VNF C	6.83	8.22	10.74	8.12
VNF D	7.48	6.51	6.60	6.17
VNF E	7.17	7.18	6.95	6.93
ALL	7.05	7.10	7.11	7.03

In general, learning SFC sequence data has low RMSE loss than learning VNF historical sequence data. Also, when model predicts VNF simultaneously, SFC sequence data provides much good performances comparing historical sequence data.

TABLE IV: RMSE Comparison of Attention and non-Attention models

	CAT-LSTM (attention)	CAT-LSTM (no attention)	LSTM (attention)	LSTM (no attention)
30 iter	8.41	16.26	8.81	10.79
60 iter	7.05	11.94	7.10	7.96
100 iter	6.93	10.21	7.09	7.63
300 iter	6.41	8.96	6.51	7.08
500 iter	6.18	8.62	6.24	6.96

This result means that historical sequence data cannot provide enough information for model to learn each VNFs' characteristic differently. However,the SFC sequence data set help the model learn each SFC data differently when predicts all VNF simultaneously. If we can predict all VNF simultaneously, we can learn the all VNFs feature and it makes no need to additional learning time when some VNFs are newly deployed.

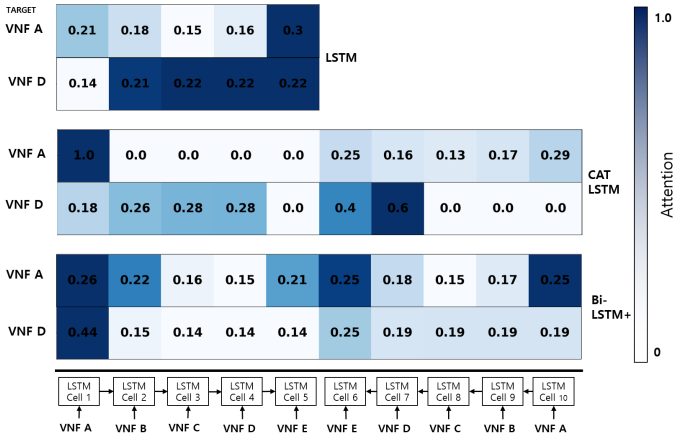

Fig. 10: Attention scores

Secondly, we compares models that use the attention, and result is shown in Table III. Most of all cases, CAT-LSTM has higher or similar accuracy than others. However, for some of the prediction results, bi-directional LSTM and bi-directional LSTM+ provided high accuracy than CAT-LSTM. The result reason can observe on Fig 10. We shows the attention score for models in Fig 10. LSTM cells' outputs are used to learn Attention with learning parameters and this parameters scored which outputs are useful to predict VNF resources. If some VNF resource information is related on the prediction target VNF resources then, the learning parameter get high scores. Fig 10 describe the scores as the dark paint.

As shown in Fig 3 and Fig 4, CAT-LSTM has Right and Left LSTM, and Bi-LSTM+ has Forward and Backward LSTM, so Attention scores ten LSTM cells. As we said in Section 2, each LSTM directly get one VNF data and also get another VNF data from an adjacent LSTM cell. Thus, each LSTM has several VNF information, and last LSTM cell has all VNF information. Basic LSTM model usually uses this cell's output directly to predict VNF information. However, Attention learns all cell's outputs and gives more attentions about which data

is important to learning. For example, in Fig 10, LSTM cell 3 has less scores for predicting VNF A resource, and LSTM cell 1 and 5 have higher scores. So, we can observe that VNF E information can help predict VNF A. Comparing scores with CAT-LSTM and bi-directional LSTM+ , we can figure out that the attention scores are different for models. As comparing the result of prediction VNF A and VNF D, CAT-LSTM more pay attentions to target VNF and bi-directional LSTM+ does more SFC. The CAT-LSTM separates input data to left and right layer based on target data. It means that each layer only has a part of information. Thus, the attention learns on small data and more focuses on target data. If target data is located in first or last LSTM cell then, left or right layer has only the target data. Thus, CAT-LSTM have to more focus on target data. In other hands, bi-directional LSTM+ just duplicates input data for forward and backward layer. Both layer has full input data and attention more focus on SFC data. Thus, the attention learning parameters have different attention scores, and the prediction accuracy is different. This is the reason that the accuracy difference CAT-LSTM and Bi-directional LSTM in Table III. We also additionally get the VNF characteristic whether VNF is dependent to SFC or not comparing the model accuracy. If one VNF has a higher accuracy from CAT-LSTM than bi-directional LSTM, we know that this VNF is less dependent to SFC. Also, if we know that some VNF is highly dependent SFC, then we can use bi-directional LSTM if we have to choose the prediction model.

We compares attention model and non-attention model of CAT-LSTM and LSTM respectively in Table IV Because attention model has a benefit of learning more important data, the models using attention generally have fast convergence speed and high prediction accuracy. In our comparison, the models with attention shown better performance and fast convergence speed.

Fig. 11 and Fig. 12 show how CAT-LSTM predicts the resources of VNF A and D. VNF A and D have different resource usage patterns. but in both cases, model learns each cases. However, we also find that CAT-LSTM does not catch the oscillations for VNF D comparing VNF A.

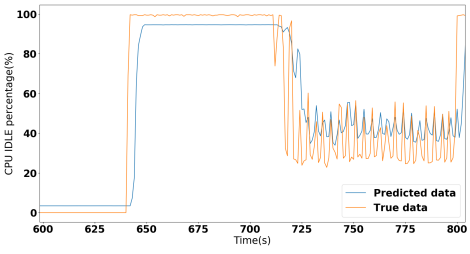

Fig. 11: VNF A CPU IDLE prediction using CAT-LSTM

VII. Conclusion

In this paper, we apply machine learning to VNF resource prediction. We use SFC sequence data to predict VNF resources and prove the effectiveness of SFC data. Also, we compare several machine learning models and evaluate their performance. This paper is the first paper to show how each VNF data affects to other VNFs in machine learning models to the best of our knowledge. In future work, we will study various machine learning approaches including reinforcement learning for intelligent management on complex networks. Also, as we figure out the effectiveness of SFC data, we will research why individual VNF data can help predict other VNFs.

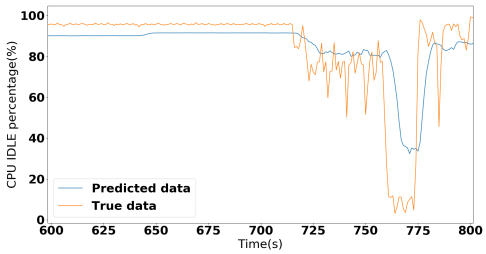

Fig. 12: VNF D CPU IDLE prediction using CAT-LSTM

VIII. Acknowledgment

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2018-0-00749, Development of virtual network management technology based on artificial intelligence) and (No. 2017-0-00195, Development of Core Technologies for Programmable Switch in Multi-Service Networks).

References

[1] Tang, Duyu, et al. "Effective LSTMs for target-dependent sentiment classification." arXiv preprint arXiv:1512.01100 (2015).
[2] Vaswani, Ashish, et al. "Attention is all you need." Advances in Neural Information Processing Systems. 2017.
[3] Mijumbi, Rashid, et al. "Design and evaluation of learning algorithms for dynamic resource management in virtual networks." 2014 IEEE Network Operations and Management Symposium (NOMS). IEEE, 2014.
[4] Rahman, Sabidur, et al. "Auto-Scaling VNFs Using Machine Learning to Improve QoS and Reduce Cost." 2018 IEEE International Conference on Communications (ICC). IEEE, 2018.
[5] Shi, Runyu, et al. "MDP and machine learning-based cost-optimization of dynamic resource allocation for network function virtualization." 2015 IEEE International Conference on Services Computing. IEEE, 2015.
[6] Pei, Jianing, Peilin Hong, and Defang Li. "Virtual Network Function Selection and Chaining Based on Deep Learning in SDN and NFV-Enabled Networks." 2018 IEEE International Conference on Communications Workshops (ICC Workshops). IEEE, 2018.
[7] Gupta, Lav, et al. "COLAP: a predictive framework for service function chain placement in a multi-cloud environment." 2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2017.
[8] Mijumbi, Rashid, et al. "A connectionist approach to dynamic resource management for virtualised network functions." 2016 12th International Conference on Network and Service Management (CNSM). IEEE, 2016.
[9] Sutton, Richard S., and Andrew G. Barto. "Reinforcement learning: An introduction." (2011).
[10] Hochreiter, Sepp, and Jrgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.
[11] Graves, Alex, and Jrgen Schmidhuber. "Framewise phoneme classification with bidirectional LSTM networks." Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.. Vol. 4. IEEE, 2005.