Predicting VNF Deployment Decisions under Dynamically Changing Network Conditions

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

*Computer Science and Engineering, Pohang University of Science and Technology, Pohang, South Korea
{stasl,sinjint,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
jhyoo78@postech.ac.kr

***Abstract*—In addition to providing network operators with benefits in terms of flexibility and cost efficiency, softwarization paradigms like SDN and NFV are key enablers for the concept of Service Function Chaining (SFC). The corresponding networks need to support a wide range of services and applications with highly heterogeneous requirements that change dynamically during the network's lifetime. Hence, efficient management and operation of such networks requires a high degree of automation that is paired with fast and proactive decisions in order to cope with these phenomena.**

In particular, determining the optimal number of VNF instances that is required for accommodating current and upcoming demands is a crucial task that also affects subsequent management decisions. To enable fast and proactive decisions in this context, we propose a machine learning-based approach that uses recent monitoring data to predict whether to adapt the current number of VNF instances of a given type. Furthermore, we present a methodology for generating labeled training data that reflects temporal dynamics and heterogeneous demands of real world networks. We demonstrate the feasibility of the approach using two different network topologies that represent WAN and mobile edge computing use cases, respectively. Additionally, we investigate how well the models generalize among networks and provide guidelines regarding the prediction horizon, i.e., how far ahead predictions can be performed in a reliable manner.

***Index Terms*—NFV, SFC, Deployment, Placement, Machine Learning, Management, Orchestration.**

I. Introduction

Network softwarization paradigms such as SDN and NFV promise operators numerous improvements regarding the scalability, flexibility, as well as resource and therefore cost efficiency of their networks. In the case of SDN-enabled networks, the logically centralized control and the separation of control and data planes pave the way for programmable networks [1]. With NFV [2], dedicated hardware middleboxes such as firewalls and load balancers are replaced with software instances that run on commodity servers, enabling adaptability to fluctuations in terms of network- as well as service-level requirements. Finally, specific services can be realized by routing packets through sequences of the resulting Virtualized Network Functions (VNFs) in the context of Service Function Chaining (SFC). However, successful management and operation of current and future networks and services requires addressing several research challenges in order to maximize the aforementioned benefits.

In particular, the complexity, heterogeneity, and temporal dynamics of use cases in the networking domain call for a high degree of automation as well as fast decision making. These aspects are crucial to achieve self-driving networks [3], [4] that can autonomously adapt to dynamic conditions. In the context of NFV and SFC, the following tasks are particularly relevant:

- Making proactive *VNF deployment decisions*, i.e., deciding whether to increase, decrease, or keep the current number of VNF instances in order to meet performance requirements of current and upcoming demands.
- Optimizing *VNF placement and chaining*, i.e., determining VNF locations and routing SFC demands through their requested set of VNFs while minimizing costs and maximizing QoS-/QoE-levels.

Since the decision *whether* to instantiate a new VNF instance is the foundation for the subsequent placement and chaining processes, it can have a significant performance impact on the entire network. Hence, in this manuscript, we focus on predicting VNF deployment actions. In order to cope with the large amount of monitoring data that is collected under heterogeneous conditions, originates from various sources, and is stored at different levels of granularity, we design a prediction mechanism that is based on machine learning (ML).

Alongside our prediction mechanism, we present a methodology for generating realistic data traces under different network conditions and training models before deploying them in a production network. In addition to evaluations regarding the accuracy of our prediction mechanism under dynamic conditions using real world topologies that represent a WAN and a mobile edge computing scenario, we provide several guidelines for network operators. First, we assess the importance of features that are used by the prediction mechanism in order to identify relevant metrics that should be collected by a monitoring system. Additionally, we conduct a parameter study regarding the prediction horizon which provides quantitative insights into the trade-offs regarding accuracy

978-3-903176-24-9

when making long-term predictions. Finally, we shed light into the generalizability of the models by training them on one topology and testing their performance on another.

This article extends our previous work [5] in several directions. Firstly, we consider a catalog of realistic SFCs rather than using random permutations of VNFs. Secondly, we improve the prediction performance of our models by integrating additional features, e.g., those related to the arrival process of individual SFCs. Thirdly, evaluations regarding the impact of the prediction horizon, the performance in the context of another network topology, and investigations w.r.t. model generalizability provide deeper insights into the feasibility and applicability of our proposed methodology.

The remainder of this manuscript is structured as follows. In Section II, we provide an overview of related work in the areas of VNF deployment, placement, and chaining, as well as relevant studies that focus on applying machine learning techniques to network-related problems. We introduce the problem statement and outline the main steps of our proposed methodology in Section III. After presenting evaluation results in Section IV, we conclude the paper and discuss directions for future work in Section V.

II. Related Work

In this section, we cover three main research areas that are relevant to our study. On the one hand, these include the application of machine learning to network-related problems in general and to NFV management and orchestration problems in particular. On the other hand, approaches for solving VNF placement and chaining problems are discussed.

A. Machine Learning in the Networking Context

The continuously increasing heterogeneity of use cases in current and future communication networks as well as the heterogeneity of deployment options for NFV-based solutions [6] leads to a large parameter space that can not be efficiently optimized with traditional methods. Furthermore, the diversity of emerging applications calls for a high degree of flexibility, adaptability, and automation [7]. Due to the successful application of machine learning in a wide range of domains as well as the widespread availability of ready-to-use frameworks and libraries, numerous machine learning-based approaches have been proposed to accelerate and automate decision making in the networking domain [8], [9]. The problems addressed by these approaches range from general networking problems such as heavy hitter detection [10] to specific problems that arise in softwarized networks [11], [12]. However, several of these works point out that results heavily depend on the data set and parameters that are used, and therefore are hard to reproduce and verify. Hence, in addition to proposing a methodology that covers the entire process from data set generation to model training and evaluation, we also publish the source code in order to foster reproducibility and allow extensions.

B. Machine Learning in Virtualized Networks

In the particular context of virtualized networks, machine learning approaches have been used for admission control in the scenario of Virtual Network Embedding (VNE) [13] as well as resource assignment [14]. In contrast to the VNE problem, we consider the use case of NFV and SFC where a single VNF instance can be shared among multiple requests. Furthermore, we predict what actions should be taken to accommodate future demands rather than making decisions for demands as they arrive. An approach that shares more similarity with ours has been proposed in [15], where resource requirements of VNFs in a virtualized IP Multimedia Subsystem (vIMS) are predicted and used as an input to a threshold-based resource allocation mechanism. However, the authors consider only a limited degree of heterogeneity: on the one hand, temporal dynamics are represented by two alternating arrival rates whereas we employ a continuous, time-varying arrival rate. On the other hand, only two types of requests (audio / video calls) that use a similar set of vIMS VNFs are used, as opposed to our scenarios which feature different chains of different lengths. Machine learning techniques have also been applied to various demand, resource, and performance prediction tasks in NFV-based networks, e.g., CPU usage prediction of VNFs [16]–[18] as well as prediction of service metrics in cloud environments [19]. In order to improve the accuracy of our proposed decision making approach, the output of such mechanisms could be integrated in the form of additional features.

C. VNF Placement and Chaining

Numerous works deal with the optimization problem of placing and chaining VNFs [20]. Most approaches are based either on Integer Linear Programming (ILP) and are capable of returning optimal solutions but are feasible only for small networks, or on heuristics that do not have optimality guarantees but produce results significantly faster [21], [22]. A recent study also shows the applicability of reinforcement learning techniques to the VNF placement problem [23]. However, while the proposed optimization schemes can operate offline or online, the corresponding works do not cover prediction or proactive decisions that are crucial for fast admission of newly arriving requests. In order to address this aspect, we use ILP-based solutions to generate ground-truth labels for temporally dynamic request traces and use them to train supervised learners that can predict VNF deployment decisions.

III. Problem Statement and Proposed Methodology

In this section, we first introduce the problem of predicting VNF deployment actions and then outline our proposed methodology for obtaining a machine learning-based model that is capable of making such predictions.

The general setting of our prediction problem is similar to that of the VNF placement problem [21], [22]. In particular, we consider a physical network that is represented as an undirected graph whose nodes have resources such as CPU

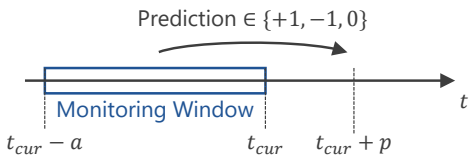

Fig. 1: Overview of the prediction process.

and memory, and whose links have bandwidth capacities. Furthermore, a VNF catalog defines the available types of VNFs that can be deployed alongside their resource consumption and bandwidth capacity per instance. SFC requests arrive in the network and are characterized by their time of arrival, duration, bandwidth demands, and the requested SFC, i.e., a sequence of VNFs that need to be traversed by each packet of the corresponding flow. Finally, we assume that a monitoring system collects various network statistics such as the number of active requests in the network, the arrival rate of SFC requests, the required bandwidth per VNF type, as well as the number of active instances per VNF type.

Given a timestamp t_{cur}, the VNF type under consideration, and a *prediction horizon* p, the task of predicting a deployment action consists of deciding whether the current number of active VNF instances of that type should be increased $(+1)$, decreased (-1), or kept the same (0) in order to accommodate the set of SFC requests that will be active at time $t_{cur} + p$. To this end, a prediction algorithm gets access to the most recent a seconds of monitoring data, i.e., a *monitoring window* of $[t_{cur} - a, t_{cur}]$. A graphical representation of the prediction process as well as the most relevant variables is provided in Figure 1. In the following, we present the steps for deriving a prediction model for this family of problems.

A. Proposed Methodology

We interpret the outlined prediction problem as a classification problem where the task consists of assigning a class in $\{+1, -1, 0\}$ to a vector of features that are extracted from the monitoring window. In particular, we employ supervised learning - a common method for addressing such classification problems. In this context, an algorithm learns patterns in the relationship between features and class membership from labeled examples.

Hence, one important step consists of generating pairs of feature vectors and class labels that correspond to realistic situations in dynamic NFV / SFC environments. To generate such problem instances, we utilize realistic network topologies, a time-varying SFC request arrival process that is based on real world traffic matrices, five types of VNFs that are combined into SFCs of different lengths, as well as varying flow durations and bandwidth requirements.

In order to obtain labels regarding the optimal number of VNF instances per type, we leverage an ILP-based algorithm from literature that is invoked at each arrival event. Finally, different machine learning algorithms are trained and evaluated w.r.t. their prediction performance. In the following, we provide a detailed description of each step. An implementation of the entire procedure is available on GitHub[1].

1) General Configuration: In the first step, general aspects of the network environment are configured. These include the network topology graph, the traffic matrix, the VNF catalog, as well as parameters that control the overall network load and the composition of VNF chains.

For the results that are presented in this work, we use two network topologies that represent a WAN as well as a mobile edge computing (MEC) scenario, respectively. These two scenarios represent two common SFC use cases: the WAN case covers the perspective of a service provider who can instantiate VNFs at different locations of a regional or global cloud provider. In contrast, the MEC scenario is representative of a telco operator who owns the entire infrastructure and can plan at a finer granularity.

In the case of the WAN scenario, we utilize the *Internet2*[2] network with 12 nodes and 15 links. In addition to the topology data, traffic matrices that contain the amount of traffic between each pair of nodes for consecutive 5-minute intervals are available. By aggregating and normalizing the total amount of traffic in the network for each interval, we extract realistic temporal dynamics that capture inter- as well as intra-day phenomena and integrate them into the SFC request arrival process. An exemplary development of the resulting normalized traffic volume over the duration of one week is displayed in Figure 2. As mentioned above, the traffic volume exhibits distinct temporal characteristics such as peaks and valleys during the five work days as well as a lower volume combined with less regular patterns during the weekend. The mean normalized traffic volume over the entire week equals 0.73.

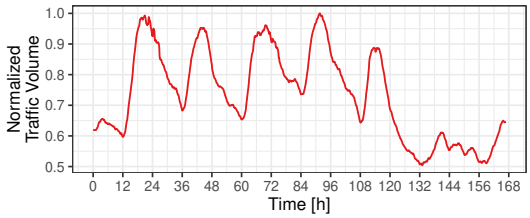

Fig. 2: Traffic matrix data from the Internet2 network is used for integrating temporal dynamics into our SFC request traces.

The topology that is used for the MEC scenario consists of 16 nodes and 15 links and is depicted in Figure 3. It contains four central offices (COs) whose servers are connected to a hierarchical structure of edge, core, and center routers. Additionally, a cloud data center (DC) can be reached via the center router to access additional computational resources in case the capacity at the COs runs low. However, accessing these resources poses a trade-off in terms of latency since the whole network needs to be traversed in order to reach the cloud DC. In order to reduce the run time of the ILP solver, we

[1]https://github.com/dpnm-ni/2019-ni-deployment-prediction
[2]http://www.cs.utexas.edu/%7Eyzhang/research/AbileneTM/

abstract details regarding the internal interconnection structure of servers within the cloud DC and CO servers, and represent them by individual leaf nodes, respectively.

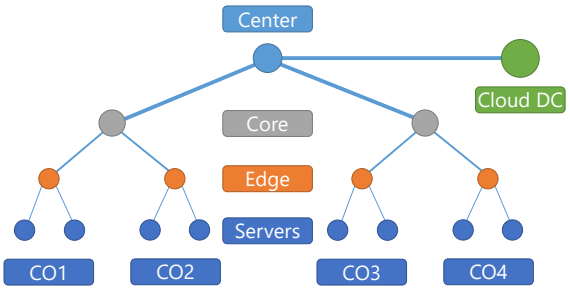

Fig. 3: MEC topology.

For the VNF catalog, we use data that is published in [22] and contains the VNFs *firewall (FW)*, *proxy (P)*, *network address translation (NAT)*, and *intrusion detection system (IDS)* alongside data regarding their CPU resource consumption and resulting bandwidth capacity. Additionally, the VNF catalog contains a *WAN optimizer (WANO)* whose characteristics are extracted from the data sheet of a VNF provider[3].

Furthermore, by defining a peak SFC request arrival rate as well as an average flow duration, it is possible to control the load in terms of the number of simultaneously active requests in the network. Finally, requested VNF chains can be configured w.r.t. the occurrence probability of certain chain lengths as well as their composition, e.g., whether chains should be constructed by randomly permuting the corresponding number of items from the VNF catalog or by using predefined chains from an SFC catalog. In the evaluations that are presented in this manuscript, we use the SFC catalog that is shown in Table I. Requests of each SFC follow a separate arrival process whose peak rate is determined by the desired fraction of the total traffic volume for the corresponding service chain. We use a broad mix of chain lengths, traffic fractions, and occurrence probabilities of individual VNFs for two reasons. On the one hand, this reflects the heterogeneity of services and applications in modern communication networks. On the other hand, such a mix allows us to assess the effects of the aforementioned characteristics on the prediction performance.

TABLE I: SFC catalog.

SFC	Fraction of total traffic
(1) NAT-FW-IDS	60 %
(2) NAT-WANO	20 %
(3) NAT-P	10 %
(4) NAT-FW-IDS-WANO	10 %

2) SFC Request Generation: While the abovementioned peak request arrival rate dictates the average amount of arrivals during the network's lifetime, the variation of these arrivals is affected by two factors. On the one hand, we use the normalized traffic volume to modulate the arrival process, i.e., we set the time-varying arrival rate to be the product of the peak request rate and the normalized traffic volume at a given time. In particular, we first generate SFC request arrivals at the peak rate for the entire trace duration and then remove requests according to a probability that is derived from the normalized traffic volume at the corresponding time. For example, we remove a request whose arrival time equals t_{arr} with probability $(1 - v(t_{arr}))$, where $v(t_{arr})$ refers to the normalized traffic volume at time t_{arr}. On the other hand, the distribution that is employed for generating request interarrival times has an impact on the variability of the arrival process. Furthermore, the overall variability is also affected by the distributions regarding the duration and requested bandwidth.

After this step, we obtain an *SFC request trace* that contains the following information for each SFC request: arrival time, duration, source and destination nodes, requested bandwidth, as well as the requested VNF sequence.

3) Placement Calculation: Given the information in the SFC request trace, we use the ILP-based solver for the placement problem that is proposed in [22] to calculate the optimal number and placement of VNF instances after each arrival. To this end, we determine the set of active requests at the corresponding time and transform them into an instance of the placement problem. Afterwards, we extract the information that is relevant for our prediction task, i.e., the number of instances per VNF type. The placement algorithm optimizes different OPEX aspects such as deployment, energy, and forwarding costs as well as resource fragmentation.

4) Training Data Generation: By combining the SFC request trace with the solver results, we can generate the labeled training data for our supervised learning algorithms. Given the width of the monitoring window a and the prediction horizon p, we generate one monitoring window for each arrival.

First, various features are extracted from the monitoring window. On the one hand, these include statistics that are independent of the VNF type like the number of active requests as well as the request arrival and departure rate in the monitoring window. On the other hand, we also track features that are specific to the VNF type under consideration, e.g., the total requested bandwidth for that type as well as its bandwidth utilization. The latter is derived by normalizing the requested bandwidth with the total capacity of instantiated VNFs. Furthermore, we extract the maximum amount of available bandwidth for each VNF type among all nodes in order to assess the fragmentation of remaining capacity. Finally, temporal features such as the time since the last arrival and departure per VNF type and SFC are captured as well. After feature extraction, the deployment action for each VNF type is determined by computing the sign of the difference between the number of VNF instances at time $t_{arr} + p$ and t_{arr}.

5) Model Training & Evaluation: For model training, we leverage the H_2O [24] machine learning framework that offers several state-of-the-art supervised learning algorithms such as XGBoost, Gradient Boosting Machine (GBM), Deep Random

[3]https://www.riverbed.com/document/datasheets/steelhead-family-specsheet19.pdf, CCX PERF-TIER4 (500 Mbps, 16 vCPUs).

Forest (DRF), Extremely Randomized Trees (XRT), and Neural Networks (NN). Furthermore, it is possible to extract key performance indicators (KPIs) such as the accuracy and mean per class error as well as information on feature importance. For example, the latter is represented by the frequency at which a certain feature is used for splitting a decision tree and to which extent this split reduces the resulting classification error. If not stated otherwise, we use a ratio of 75 % / 25 % to divide our data set into training and testing data, and report KPIs w.r.t. the latter when evaluating model performance.

We deliberately chose this straightforward approach towards supervised learning in order to achieve the following goals:

- Model explainability and assessment of feature importance to derive operational guidelines.
- Investigation of the general applicability of the proposed methodology to the problem.
- Avoidance of early overspecialization to allow model generalizability to similar problems in softwarized networks.

Parameter Values: In order to investigate the feasibility of the models for predicting VNF deployment decisions, we evaluate their prediction performance in the WAN and MEC scenarios using the respective network topologies. Additional parameters that are used in our experiments are summarized in Table II.

TABLE II: Parameters that are used in the evaluation.

Parameter	Value(s)
Peak SFC request arrival rate	1 per 30 sec
Average flow duration	1,000 sec
Distribution and coeff. of variation of interarrival times and flow durations	normal (0.25)
Distribution of requested bandwidth	uniform(70 Mbps ... 120 Mbps)
Function chains and probabilities	cf. Table I
Source / destination nodes	Internet2: random pairs MEC: random pairs of COs
Width of monitoring window a	600 sec
Prediction horizon p	60 sec

We use a combination of peak arrival rate and average flow duration that leads to enough fluctuation in terms of the number of VNF instances without overburdening the network. While the distributions of interarrival times and flow durations can affect prediction performance, our evaluations have shown that this impact is limited. Hence, we only present results that were obtained using normally distributed SFC request arrivals. By considering variation in terms of the requested bandwidth, effects regarding the fragmentation of allocated resources are integrated into the evaluation as well. Finally, since VNF chains are formed between random pairs of nodes (Internet2) or central offices (MEC) and have varying lengths, the ability of the prediction model to cope with such stochastic aspects can be investigated. We use a monitoring window of 10 min and a prediction horizon that covers an average of two arrivals at the peak rate.

For each of the two topologies that are used to represent the WAN and MEC scenarios, we generate SFC request traces from which we extract 20,000 labeled feature vectors that are used for model training. In order to account for the fact that the per-class occurrence probabilities are not always balanced and therefore in order to prevent classifications that favor the majority class, we additionally set per-observation weights so that the total weight per class in the data set is equal.

IV. Performance Evaluation

In this section, we present evaluation results regarding our two scenarios. In order to identify a suitable machine learning algorithm for our prediction task, we first train several learners on our data. Afterwards, we fine-tune hyper parameters of the most promising algorithm and report detailed prediction results of the final model. These include the confusion matrix as well as information regarding feature importance. In the second part, we perform a parameter study w.r.t. the prediction horizon in order to quantify the trade-offs between the capability of making long-term predictions and their accuracy. Finally, we analyze how well the models generalize by evaluating the MEC prediction performance of a model that has been trained in the WAN scenario and vice versa.

All evaluations have been performed on a server that has a total of 64 GB of RAM and is equipped with an Intel Xeon Silver 4114 CPU with 10 cores and two NVIDIA Quadro P5000 GPUs. For the detailed analysis regarding model choice and hyper parameters that is presented in Section IV-A, we exemplarily report results regarding the firewall VNF and the Internet2 topology using a prediction horizon of 60 seconds. Subsequent evaluations show aggregated results and cover all combinations of VNFs and topologies.

A. Model Choice and Hyper Parameter Tuning

In order to assess the feasibility of different algorithms for our prediction task, we use the *AutoML*[4] function of the H_2O framework. This function allows training multiple models from each of the major families of machine learning algorithms that are compatible with a given data set. In our case, these are XGBoost, GBM, DRF, XRT, and NN. Using a total training time of 10 hours, the model that performed best w.r.t. the mean per class accuracy was based on XGBoost and achieved a mean per class accuracy of 75.7 %. While some models such as GBM and XRT achieved accuracy values within less than 1 % of this value, we decided to choose XGBoost since this algorithm also had less fluctuations w.r.t. the accuracy among different configurations.

The XGBoost algorithm has numerous hyper parameters[5] that can be tuned in order to control trade-offs regarding over- and underfitting. Since an exhaustive evaluation of all parameter combinations is not feasible due to the large parameter space, we perform a *random grid search*. In this context, we train over 500 models using hyper parameter combinations from a set of candidates that are summarized in Table III.

While some parameters like the number of trees did not have a significant impact on the prediction performance, changes to

[4] http://docs.h2o.ai/h2o/latest-stable/h2o-docs/automl.html

[5] http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/xgboost.html

TABLE III: Tested XGBoost hyper parameters and values.

Parameter	Values	Chosen Value
Number of trees	$\{80, 100, \ldots, 200\}$	140
Maximum depth	$\{4, 6, 8, 10\}$	8
Minimum child weight	$\{1, 2, \ldots, 10\}$	6
Learning rate	$\{0.05, 0.1, \ldots, 0.4\}$	0.05
Row / column sample rates	$\{0.6, 0.8, 1\}$	0.6
L2 regularization (λ)	$\{0.8, 1\}$	0.8
L1 regularization (α)	$\{0, 0.01\}$	0

sampling and learning rate as well as the L2 regularization parameter improved the models' tendency to overfit the training data. The final model achieves a mean per class accuracy of 77.8 % on the testing data, i.e., an accuracy improvement of 2.1 % when compared to the model obtained in the first step. The chosen hyper parameter values are reported in the table.

1) Confusion Matrix: For a detailed analysis of classification performance, we provide the confusion matrix of the final model in Table IV. For each combination of actual and predicted class label, it displays the number of items in the testing set that were classified accordingly. While the majority of predictions are accurate, a significant number of misclassifications can be observed at the boundary of the "0" class, i.e., cases of "+1" and "-1" getting predicted as "0" and vice versa. This phenomenon can be explained by the fact that while the classes are encoded in a categorical fashion, they are actually derived from numerical values, i.e., the difference in the number of VNF instances. Hence, errors are more likely to happen between classes that are numerically close. However, due to the stochastic and time-varying properties of the SFC request arrival process, errors of the type "+1 misclassified as -1" - while rarely - can also be observed.

TABLE IV: Confusion matrix for the best-performing predictor of firewall-related actions in the Internet2 topology.

		Predicted			**Total**	**Recall**
		+1	*0*	*-1*		
Actual	*+1*	**832**	248	34	1,114	0.75
	0	478	**1,871**	174	2,523	0.74
	-1	2	110	**776**	888	0.87
	Total	1,312	2,229	984		
	Precision	0.63	0.84	0.79		

2) Feature Importance: In order to provide network operators with guidelines regarding the choice of metrics that should be collected by a monitoring system, we list the features that are most important for predicting VNF deployment actions in Table V. These features are involved in the majority of decision tree splits and help reducing the classification error. On the one hand, features that are directly related to the VNF for which actions are predicted dominate this list and are shown in the top part. These features are mostly related to the global relationship between the total requested bandwidth for the particular VNF type and the available capacity that is provided by instantiated VNFs, e.g., utilization and remaining capacity. On the other hand, local features such as the mean available firewall capacity per node help determining the likelihood of being able to accommodate additional requests without instantiating more VNFs. Finally, features that are related to the arrival process and the type-independent load situation help anticipating and adapting to temporal dynamics. Although the firewall VNF is part of two SFCs, information regarding the departure time of SFC1 has a higher importance due to the fact that it is the chain with the highest relative traffic volume and therefore arrival rate (cf. Table I).

TABLE V: Feature importance when predicting deployment actions for the firewall VNF. Features in the upper part are firewall-specific whereas the ones in the bottom part reflect global and SFC-level characteristics.

Feature	Relative importance
Remaining firewall bandwidth capacity	1.00
Firewall bandwidth utilization	0.74
Mean remaining firewall capacity per node	0.36
Time since last arrival	0.35
Time since last departure (SFC1)	0.26
Mean arrival rate	0.12

3) Performance w.r.t. other VNFs and MEC Scenario: Having discussed in-depth results for one particular scenario in the previous subsection, we provide an overview of the prediction performance that is achieved in the case of the MEC topology as well as all five VNFs that are part of our study in this section. On the y-axis, Figure 4 displays the mean per class accuracy that is achieved when training XGBoost models for different VNFs that are listed on the x-axis. Differently colored bars represent the two topologies.

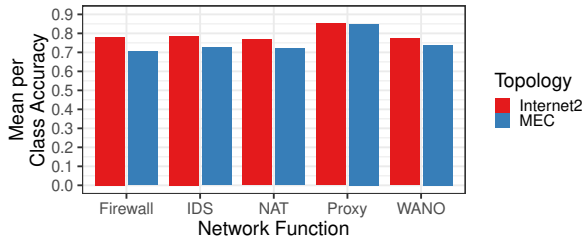

Fig. 4: Prediction performance w.r.t. different VNFs in the WAN and MEC scenarios. Prediction horizon $p = 60s$.

There are two main observations. First, the prediction performance regarding all VNFs is similar for a given topology. The proxy VNF is the only exception to this phenomenon, which can be explained by several of its characteristics: the proxy VNF is part of only one SFC and this SFC amounts to only 10 % of the total traffic volume. Hence, a lower number of instantiated proxies is sufficient for serving the corresponding traffic and the lower arrival rate reduces fluctuation w.r.t. this number, resulting in a higher degree of predictability. Second, the accuracy that is achieved by models that are trained to make predictions in the MEC scenario is consistently lower

than of those for the WAN scenario. This can be explained by the fact that the MEC scenario has several characteristics that increase the difficulty of the prediction task. On the one hand, there is a larger fluctuation in the number of VNF instances. Since the placement optimizer tends to avoid long detours for SFC requests, VNFs are placed so that the servers at the central offices of source and destination nodes host all required VNFs in a chain. On the other hand, the availability of the cloud data center (CDC) can lead to a state transition where VNF instances have to be placed at the CDC due to the limited capacity of the central offices. During this state transition, migrations and relocation can occur, which, in turn, further increase the fluctuation in the network. Nonetheless, a mean per class accuracy of over 70 % is achieved for all VNFs. In order to improve the performance further, we plan to integrate additional topology-, request-, and node-level features that help identifying the outlined state transitions and node relationships.

B. Impact of the Prediction Horizon

The models discussed in the previous section were trained to make predictions with a prediction horizon of $p = 60\,s$. While this value represents a reasonable balance between short- and long-term predictions, some use cases might call for lower values, higher values, or even combined predictions. For instance, short-term predictions could be used to prevent imminent performance degradation by placing additional VNF instances in a greedy fashion while long-term predictions leave enough time for finding an optimal placement for VNF instances before the corresponding requests arrive.

Hence, in order to quantify the relationship between the prediction horizon and the prediction performance, we conduct the following parameter study. For values in the range from 10 to 100 seconds and increments of 10 seconds, we generate corresponding data sets for each parameter value and VNF based on our labeled ground truth from the ILP solver (step 4 of the methodology). In order to obtain statistically significant results, we train five models for each combination of VNF and value of p, using a different seed for splitting the data into training and testing set in each repetition. We report results regarding the Internet2 topology in this section. However, qualitatively analogous results have been obtained in the case of the MEC topology, albeit with an accuracy offset that is caused by the characteristics we discussed previously. Furthermore, it is worth noting that since the VNF placement problem is NP-hard [22], predicting the number of required VNF instances is hard even when all problem parameters are known. Hence, even in the context of a short prediction horizon, a perfect accuracy can not be guaranteed.

Figure 5 contains four plots that correspond to different VNFs under consideration. Since the IDS and firewall VNFs occur in the same set of function chains, our prediction models have similar accuracy levels for these VNFs. Hence, we omit results regarding the IDS VNF in the following. For values of the prediction horizon on the x-axis, the y-axis denotes the mean per class accuracy that is achieved by the corresponding models and error bars represent 95 % confidence intervals that are obtained from multiple repetitions. Analogously to previous observations, the models achieve a similar prediction performance w.r.t. the different VNFs. Again, the proxy VNF poses the only exception since it is significantly easier to predict due to its characteristics, i.e., lower total number of instances and higher SFC request interarrival times. These characteristics also explain the insensitivity towards the investigated range of values for the prediction horizon.

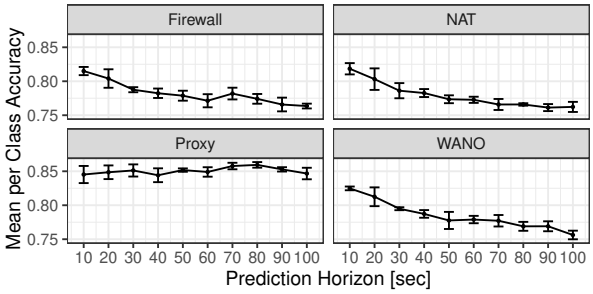

Fig. 5: Impact of the prediction horizon on the accuracy of predicting deployment actions for different VNFs.

In contrast, the mean per class accuracy of models that predict deployment actions for the remaining VNFs drops in an almost linear fashion when the prediction horizon is increased. The accuracy develops from values around 82 % in the case of $p = 10\,s$ to around 75 % when $p = 100\,s$. Hence, even in the context of long-term predictions, reliable statements are possible. However, prior to deploying such prediction mechanisms, network operators should consider the resulting accuracy trade-off.

C. Generalizability of Prediction Models

A desirable feature of prediction models is their capability to generalize well across different problem instances. In contrast to instance-specific models, general models avoid the need for laborious re-training upon each configuration change and can be used directly while also integrating specific characteristics of the new configuration. In order to assess the generalizability of our models and to derive insights that help improving it, we conduct the following experiments. For each VNF, we create three data sets that contain labeled data for the Internet2 topology, the MEC topology, and their union, respectively. For each data set, a prediction model is trained on 75 % of the data. Afterwards, the trained models are individually applied to the three testing sets that are constructed from the remaining 25 % of each data set.

The results of these experiments are depicted in Figure 6. Each subplot represents a VNF while x- and y-axes show training and testing sets, respectively. The color of each cell corresponds to the mean per class accuracy that is achieved by the different models and additionally contains a label with the exact accuracy value.

The highest accuracy values in each subplot tend to be located along the diagonal. This stems from the fact that

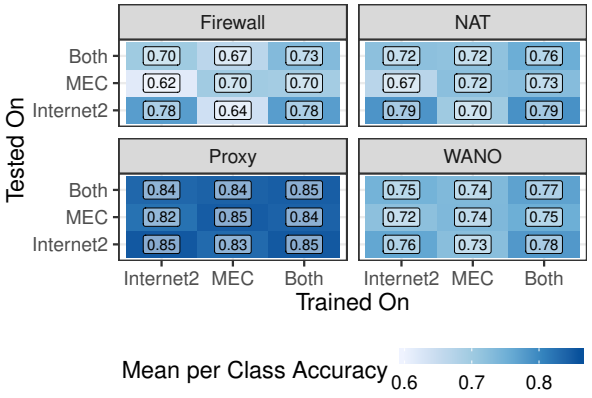

Fig. 6: Cross-prediction accuracy for different VNFs. Prediction horizon $p = 60\,s$.

these values represent cases in which training and testing data originate from the same scenario (*Internet2*, *MEC*) or the same mix of scenarios (*Both*) and therefore, involve no generalization. Additionally, the accuracy levels that are achieved when using the mixed data set are close to the mean of the two accuracy values that are obtained for the individual data sets. This indicates that when model training is performed on the union of data sets, decision thresholds for features are adapted in a way that balances inaccuracies between the data sets. In the case of the WANO VNF, the characteristics in the individual scenarios are similar enough so that the combined model even manages to outperform the individual ones by leveraging the larger amount of training data. The performance regarding the proxy VNF is very stable for all combinations due to the fact that this VNF has the most predictable arrival process and therefore exposes almost identical characteristics in all considered scenarios.

When performing cross-prediction, i.e., training on one data set and testing on another, accuracy decreases of up to 8 % can be observed in comparison to the corresponding values on the diagonals. However, such a drastic drop is only observed in the case of the firewall VNF whereas predictions regarding other VNFs generalize significantly better. Furthermore, the direction "training on MEC, testing on Internet2" results in higher accuracy levels than the reverse. A likely explanation for this phenomenon is that models trained on the MEC scenario learn both general relationships as well as MEC-specific characteristics. Hence, when applied to the Internet2 scenario, the general relationships are sufficient for making good predictions. In contrast, models trained on the Internet2 scenario tend to misclassify situations due to a lack of topology-specific knowledge.

In summary, the current models already generalize well w.r.t. different topologies. As discussed previously, we expect further improvements by adding topology-, node-, and request-level features. Furthermore, training the model on a wider range of topologies prior to applying it to a new one is expected to add stability.

V. Conclusion

Efficient management and operation of current and future softwarized networks requires fast decision making while facing heterogeneous demands in dynamically changing environments. As a step towards networks that can automatically and proactively adapt themselves to such conditions, we propose a machine learning-based approach for predicting VNF deployment decisions in the context of Service Function Chaining (SFC). In order to foster reproducibility, flexible configuration, and extensibility, we propose a methodology that encompasses the entire process from generating realistic SFC request traces to training and evaluating prediction models. In particular, this also includes extracting features that are used for model training and labeling training data with information that is based on optimal VNF placements.

Evaluations featuring dynamic request arrivals in realistic network topologies that represent WAN and mobile edge computing (MEC) scenarios are used to highlight the applicability of the proposed approach. Furthermore, we provide insights into the importance of individual features in order to assist operators with choosing appropriate metrics for monitoring. We also evaluate the impact of the prediction horizon on the classification accuracy to analyze the performance trade-offs that result from making long-term predictions. Finally, we demonstrate the models' capabilities in terms of general applicability across different topologies. Our evaluations show that even without leveraging topology-specific information, models can make reliable predictions without being re-trained.

There are several directions for future work: on the one hand, we plan to add new graph- and request-related features in order to improve the performance when making predictions for individual problem instances as well as for the context of generalizing across different configurations regarding network topology, network size, and traffic mix. On the other hand, aspects of applicability, scalability, and generalizability can be studied by integrating the proposed approach into an actual NFV and SFC testbed which adds challenges w.r.t. the communication with different network entities as well as the timeliness and reliability of monitoring data. Finally, extending the proposed methodology to address related challenges in the area of softwarized networks, and placement problems in particular, could lead to further improvements in terms of network automation.

Acknowledgments

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (2018-0-00749, Development of Virtual Network Management Technology based on Artificial Intelligence).

References

[1] M. Jarschel, T. Zinner, T. Hoßfeld *et al.*, "Interfaces, Attributes, and Use Cases: A Compass for SDN," *IEEE Communications Magazine*, 2014.

[2] B. Yi, X. Wang, K. Li *et al.*, "A Comprehensive Survey of Network Function Virtualization," *Computer Networks*, 2018.

[3] P. Kalmbach, J. Zerwas, P. Babarczi *et al.*, "Empowering Self-Driving Networks," in *Afternoon Workshop on Self-Driving Networks*, 2018.

[4] N. Feamster and J. Rexford, "Why (and How) Networks Should Run Themselves," in *Applied Networking Research Workshop*, 2018.

[5] S. Lange, H.-G. Kim, S.-Y. Jeong, H. Choi, J.-H. Yoo, and J. W.-K. Hong, "Machine Learning-based Prediction of VNF Deployment Decisions in Dynamic Networks," in *Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2019.

[6] L. Linguaglossa, S. Lange, S. Pontarelli *et al.*, "Survey of Performance Acceleration Techniques for Network Function Virtualization," *Proceedings of the IEEE*, 2019.

[7] W. Kellerer, P. Kalmbach, A. Blenk *et al.*, "Adaptable and Data-Driven Softwarized Networks: Review, Opportunities, and Challenges," *Proceedings of the IEEE*, 2019.

[8] R. Boutaba, M. A. Salahuddin, N. Limam *et al.*, "A Comprehensive Survey on Machine Learning for Networking: Evolution, Applications and Research Opportunities," *Journal of Internet Services and Applications*, 2018.

[9] M. Wang, Y. Cui, X. Wang *et al.*, "Machine Learning for Networking: Workflow, Advances and Opportunities," *IEEE Network*, 2018.

[10] V. Sivaraman, S. Narayana, O. Rottenstreich *et al.*, "Heavy-hitter Detection Entirely in the Data Plane," in *Symposium on SDN Research*, 2017.

[11] J. Xie, F. R. Yu, T. Huang *et al.*, "A Survey of Machine Learning Techniques Applied to Software Defined Networking (SDN): Research Issues and Challenges," *IEEE Communications Surveys & Tutorials*, 2018.

[12] A. Blenk, P. Kalmbach, W. Kellerer *et al.*, "o'zapft is: Tap Your Network Algorithm's Big Data!" in *Workshop on Big Data Analytics and Machine Learning for Data Communication Networks*, 2017.

[13] A. Blenk, P. Kalmbach, P. Van Der Smagt *et al.*, "Boost Online Virtual Network Embedding: Using Neural Networks for Admission Control," in *International Conference on Network and Service Management*, 2016.

[14] R. Mijumbi, J.-L. Gorricho, J. Serrat *et al.*, "Design and Evaluation of Learning Algorithms for Dynamic Resource Management in Virtual Networks," in *IEEE Network Operations and Management Symposium (NOMS)*, 2014.

[15] R. Mijumbi, S. Hasija, S. Davy *et al.*, "Topology-aware Prediction of Virtual Network Function Resource Requirements," *IEEE Transactions on Network and Service Management*, 2017.

[16] H. Jmila, M. I. Khedher, and M. A. El Yacoubi, "Estimating VNF Resource Requirements Using Machine Learning Techniques," in *International Conference on Neural Information Processing*, 2017.

[17] A. Mestres, A. Rodriguez-Natal, J. Carner *et al.*, "Knowledge-defined Networking," *ACM SIGCOMM Computer Communication Review*, 2017.

[18] H.-G. Kim, D.-Y. Lee, S.-Y. Jeong *et al.*, "A Machine Learning-based Method for Virtual Network Function Resource Demand Prediction," in *IEEE Conference on Network Softwarization (NetSoft)*, 2019.

[19] F. Moradi, R. Stadler, and A. Johnsson, "Performance Prediction in Dynamic Clouds using Transfer Learning," in *IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2019.

[20] J. G. Herrera and J. F. Botero, "Resource Allocation in NFV: A Comprehensive Survey," *IEEE Transactions on Network and Service Management*, 2016.

[21] S. Lange, A. Grigorjew, T. Zinner *et al.*, "A Multi-objective Heuristic for the Optimization of Virtual Network Function Chain Placement," in *International Teletraffic Congress*, 2017.

[22] M. F. Bari, S. R. Chowdhury, R. Ahmed *et al.*, "On Orchestrating Virtual Network Functions," in *International Conference on Network and Service Management*, 2015.

[23] M. Nakanoya, Y. Sato, and H. Shimonishi, "Environment-Adaptive Sizing and Placement of NFV Service Chains with Accelerated Reinforcement Learning," in *IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2019.

[24] The H2O.ai team, *H2O: Scalable Machine Learning*, 2015. [Online]. Available: http://www.h2o.ai