Dynamic Prediction of Resource Consumption in Virtualized Environments

by

Asma BELLILI

THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
Ph.D.

MONTREAL, 30 JULY 2024

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mrs. Nadjia Kara, Thesis supervisor
Department of Software engineering and IT, École de technologies supérieure

Mrs. Rola Assi, Chair, Board of Examiners
Department of Construction engineering, École de technologie supérieure

Mr. Abdelouahed Gherbi, Member of the Jury
Department of Software engineering and IT, École de technologies supérieure

Mr. Chamseddine Talhi, Member of the Jury
Department of Software engineering and IT, École de technologies supérieure

Mr. Bessam Abdulrazak, External Examiner
Department of Computer engineering, University of Sherbrooke

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON 29 MAY 2024

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

I am deeply grateful for Professor Nadjia Kara, the director of this thesis, for her encouragement, advice and listening. I would also like to express to her all my gratitude for her support and comprehension throughout this thesis. Her experience and her knowledge of the field contributed greatly to the success of my thesis.

I would like to express my sincere thanks to Mrs. Rola Assi, Mr. Bessam Adbulrazak, Mr.Abdelouahed Gherbi, Mr. Chamseddine Talhi for agreeing to evaluate this work.

I would like to offer my special thanks to my family and my friends and to everyone who has supported me and participated directly or indirectly in this work.

Prédiction dynamique de la Consommation des Ressources dans les Environnements Virtualisés

Asma BELLILI

RÉSUMÉ

Les environnements infonuagiques d'aujourd'hui offrent d'énormes capacités d'hébergement de services et d'applications. Dans de tels environnements, l'augmentation exponentielle de la consommation de ressources (ex., de calcul, de stockage) rend leur gestion très difficile. Cette dernière doit respecter différentes contraintes de Qualité de Service (QdS) qui sont négociées dans le cadre de divers contrats de service (Service Level Agreements, SLA) pour différentes applications (ex., application hautement sensible au délai). Ces applications représentées par des chaînes de microservices sont déployées dans ces environnements qui sont dynamiques (ex., variabilité de la disponibilité des ressources, volatilité des charges de trafic). Dans cette thèse, nous nous intéressons à l'adaptation de ressources pour des chaînes de fonctions de services (Service Function Chain -SFCs) composées de plusieurs fonctions de réseau virtualisées (Virtual Network Function -VNF). Les variations élevées de la charge de trafic des applications entraînent une incertitude quant à l'utilisation des ressources. Il devient donc crucial de disposer de mécanismes efficaces basés sur les données pour la gestion automatique des ressources. Ces mécanismes permettent aux systèmes complexes d'anticiper et de réagir efficacement aux fluctuations de ces charges. Ils s'appuient sur des techniques efficaces de prévision de l'utilisation des ressources afin de répondre aux exigences de QdS des applications. Ils doivent anticiper la consommation de différentes ressources (ex., CPU, mémoire) tout en tenant compte de la variabilité. Dans ce contexte, une solution basée sur un modèle de mono-prédiction basé uniquement sur l'historique de la consommation de ressource peut échouer en cas de changement brut de la consomation de resource. Pour permettre au mécanisme de prédiction de s'adapter à la variation des demandes en ressources. Dans cette thèse,nous proposons une nouvelle solution de prédiction qui s'adapte à la variation de la consomation en sélectionnant parmi plusieurs méthodes de prédiction, celle qui est la mieux adaptée à la charge de trafic reçue. Notre solution applique un sélecteur multitâche basé sur la stratégie de méta-learning MT-MLS. Le MT-MLS introduit un nouveau concept en analysant les similitudes de consommation de différentes ressources entre les VNFs d'une SFC (une application). Un mécanisme d'attention est utilisé pour identifier un poid pour chaque VNF sur la base de l'analyse de similarité. Le MT-MLS est conçu comme un classifieur multitâche,qui attribue simultanément et indépendamment à chaque VNF un meilleur prédicteur pour prévoir les besoins en ressources dans une situation de charge de trafic donnée. Le meilleur prédicteur est sélectionné parmi un ensemble de modèles de prédicteurs proposés par notre solution, mais sans s'y limiter. En effet, notre solution peut etre appliquée avec de nombreux predicteurs qui sont capables de fournir une prediction de la consomation future des resources. Parmi les prédicteurs de base proposés par notre solution: un modèle GNN efficace qui utilise les caractéristiques de la topologie d'une SFC et qui a démontré ses performances dans le cas de charges de trafics élevé. Nous avons également développé d'autres modèles, un modèle CNN, un modèle LSTM et un modèle hybride. Ces prédicteurs de base ont été utilisés pour générer des métadonnées afin de développer le MT-MLS. L'analyse et

la comparaison des résultats expérimentaux montrent clairement l'efficacité et la performance de nos solutions.

Mots-clés: gestion des ressources, prédiction des ressources, apprentissage automatique, dépendances entre VNFs, chaîne de fonctions de service, classification multitâche, affinité VNF, GNN

Dynamic Prediction of Resource Consumption in Virtualized Environments

Asma BELLILI

ABSTRACT

Today's cloud environments host various services and applications. In such an environment, the consumption of resources is growing exponentially which makes their management a challenging task especially in a dynamic cloud environment (e.g., resource availability, traffic volatility) and under different Quality of Service (QoS) requirements that are specified in Service-Level-Agreements (SLAs) for various applications (e.g., highly latency-sensitive applications). These applications designed as chains of microservices are deployed in a dynamic cloud environment. In this thesis, we focus on adapting resources for service function chains (SFCs) which consist of a set of Virtual Network functions (VNFs). High traffic load variations of applications lead to uncertainty in resource utilization. Consequently, the need for efficient data-driven mechanisms for automatic resource management becomes paramount. These mechanisms enable distributed systems to anticipate and efficiently respond to resource consumption fluctuations. They rely on accurate prediction techniques to satisfy the resource needs, to meet the QoS requirements for cloud applications and service infrastructures. These mechanisms are requiered to handle with the heterogeneity and the dynamic of resource demands. In this context, a mono-prediction model based on only historical consumption is inefficient for a volatile traffic workload. In this thesis, we propose a new prediction technique for dynamic workload variation. This technique allows the choice from various prediction techniques, the efficient one to trigger. Our solution applies a multi-task selector based on a meta-learning strategy MT-MLS. The MT-MLS introduces a new concept by analyzing similarities of resource consumption between various VNFs of an SFC. Additionally, an attention mechanism is employed to assign a weight to each Virtual Network Function (VNF) through similarity analysis. The MT-MLS is designed as a multi-task classifier, concurrently and independently assigning the most suitable predictor for forecasting multidimensional resource consumption for each Virtual Network Function (VNF) in a given workload. The best predictor is selected among a set of predictor models supported by our solution. Among the predictor models supported by our solution, the solution comprises an efficient Graph Neural Network (GNN) Model that uses SFC topological features demonstrating its performance under high-load traffic. The solution also includes an Long Short Term Memory (LSTM) model, a Convolutional Neural Network (CNN) model, and a hybrid model. These base predictors are used to generate metadata to train the MT-MLS. The performances of each model were compared with MT-MLS performances. The analysis of the various experimental results shows clearly that the proposed solution outperforms the other models.

Keywords: Virtualized network function, deep learning, VNF dependencies, service function chain, cloud computing, multitask classification, meta-learning, VNF similarity, GNN

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Page

INTRODUCTION

0.1 Context and challenges

Recently, there has been significant progress in the field of communication and information technologies because of the adoption of various concepts such as cloud computing and Network Function Virtualization (NFV). The cloud computing principle consists of delivering on-demand virtualized resources (e.g. storage, computing, networking) using the principle of pay-as-you-use. NFV aims to separate network functions from hardware platforms using virtualization technologies, this will alleviate the disadvantages of legacy network architectures since hardware equipment will be shared by multiple applications. This has allowed many industries and enterprises to migrate their services to a cloud computing environment. This migration allows cloud providers to instantiate virtual environments (e.g., Virtual Machines, Containers) on shared hardware equipment. Indeed, cloud providers may increase or decrease the resources allocated to a service by scaling down or up the resources to optimize resource consumption, allow innovative service offerings, and provide a Pay-As-You-Go model while avoiding overprovisioning or underprovisioning of resources. Indeed, the applications are deployed in more scalable environments, which offer great flexibility in managing services and resources. The multiple advantages of NFV such as resource consumption optimization, the ease of VNF (Virtual Network Function) upgrades, more affordable software and hardware maintenance and costs to name a few have made this concept attractive to many service providers (SP). However, guaranteeing these advantages requires efficient mechanisms for optimal and automatic resource management. The optimal management of shared resources is one of the major challenges facing the adoption of NFV and is emphasized with the significant growth of cloud-based applications that lead to increasing demand for computing power and storage. Moreover, in such shared and dynamic environments, it is necessary efficiently and automatically

manage resource consumption while meeting the performance requirements (specified in the Service Level Agreement (SLA)) of the hosted applications.

Indeed, an underestimation of the capacity of the allocated resources could lead to a degradation of service and a violation of the SLA, while an overestimation could lead to wastage of resources and an increase of the operational cost of cloud infrastructures. Therefore, the resource management mechanisms must effectively and efficiently estimate the needed resources according to the resource consumption variations of the hosted applications while meeting their performance requirements.

One of the proposed solutions is to predict the future applications' workloads to estimate the future consumptions of resources for VNF of an SFC Bansal & Kumar (2023). This will allow a resource manager to optimize the resource usage, to reduce costs (e.g., operational and energy costs). Deep learning models (DL) can help in forecasting future usage since they have demonstrated their efficiency in many prediction problems Morid *et al.* (2023). Therefore, training a DL-based prediction model on historical data of resource consumption can be a promising solution to predict the future application workload, which will enable optimal planning of shared resource usage.

0.2 Problematic

Cloud providers claim to provide on-demand service access while respecting the Service Level Agreement. They provide the needed resources to VNF to guarantee the QoS. The common practice is to use scaling methods in order to meet the performance requirements of applications, without anticipating their resource needs. However, the resource consumption is very dynamic. Thus, developing a proactive scaling solution based on future resource consumption prediction enables minimizing the costs resulting from an underestimation or an overestimation of resource consumption. Adopting such prediction techniques is a challenging task since the cloud platforms

host different applications with various workloads and different resource needs (e.g., computing, networking, storage). This complicates the use of a DL model that should remain efficient when the resource consumption change and fluctuate to ensure the quality of services offered to users. Indeed, most DL-based resource prediction methods Saxena *et al.* (2023) suffer from considerable performance degradation in the presence of suden and unusual changes in traffic load. Moreover, DL models are often highly sensitive to the data nature. Also, network entities (such as VNFs) may consume multiple resources that need to be predicted at the same time. Knowing that those entities may be very different in terms of resource consumption behavior, training a DL model that should predict the consumption of multiple resources is a big challenge that may threaten the model's performance.

Therefore, in this thesis, we intend to study the following research question: How to model DL techniques that enable efficient prediction of consumption of multiple resources under variable workloads?

0.2.1 Objectives

The global objective of this thesis is to propose a new proactive and adaptive mechanism that allows to efficiently predict the resources provided in virtualized environments depending on the current resource consumption. We divided our global objective into four main objectives:

- Obj1: Developing a multi-attribute model for resource utilization prediction based on VNFs dependencies using deep learning techniques.
- Obj2: Analyzing and comparing multiple Artificial Neural Networks (ANNs) in terms of prediction efficiency.
- Obj3: Developing a technique that studies the similarity between different VNFs of the SFC, to capture consumption similarities between the VNFs. The similarity analysis will be used to guide the resource consumption prediction of the VNFs.

- Obj4: Developing an automatic selection approach that enables to select the most suitable DL model based on VNFs similarity analysis and conduct a comparative study with existing prediction models. The selection approach allows to enhance the prediction mechanism accuracy.

Upon completion of the objectives, this project will offers a resource consumption prediction mechanism that could be part of the Network Functions Virtualization Management and Network Orchestration (NFV MANO). This infrastructure was developed by a working group with the same name within the specification group for NFV (ISG NFV) from the European Telecommunications Standards Institute (ETSI). It is dedicated to the orchestration an the management of shared resources in a virtualized data center. NVF MANO contains three functional blocks: the VNF manager, the NFV Orchestrator (NFVO), and the virtualized infrastructure manager (VIM). Our mechanism can be integrated in the VIM component. It will allow the VIM to predict different types of resource consumption of a set of network functions. Our objective is to design a mechanism that can be integrated into the VIM to predict future resource consumption (e.g., CPU, memory) for various number of network entities (e.g., VNFs).

0.3 Methodology

In this thesis, we assume that the service function chains are deployed and historical resource consumptions of each VNF are recorded. To achieve our objective, we first establish the problem statement and hypothesis. Our hypothesis is to use resource consumptions of SFCs deployed in close to real-world deployment environments (non-synthetic data). In this project, we use datasets of resource consumptions of SFCs that have been deployed in testbed using two types of virtualized environments (Kubernetes in openstack and Bare-metal virtualized environment) using ÉTS facilities. The test environment and scenarios are described in Chapter 3. Several

test experiments were carried out while considering different traffic loads, to collect various metrics (e.g., CPU, memory).

Figure 0.1 shows the proposed methodology. After a detailed study of the current state of the art in resource prediction methods in virtualized environments, we have stated our research problem and defined our objectives. Based on the state-of-the-art study, we found that LSTM-based models are best suited to prediction problems using time-series data that represents a set of data samples taken at successive equally spaced points in time, in our work, the samples are the resource consumption rates (CPU consumption, Memory consumption, and bandwidth consumption). Further, CNN networks are well known for their performance extraction of relevant features. Therefore, our starting point was the development of a Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) based models for resource consumption prediction. We established, tuned, and modified the architectures of our models iteratively to maximize their performance (e.g., minimize the Mean Square Error) and defined four DL models. In order to enhance the prediction accuracy of the DL architectures, we consider the dependencies between VNFs of an SFC. Thus, we developed a GNN model that involves neighboring dependencies in the learning process. A thorough analysis of the model results enabled us to reveal the pros and cons of each model and to develop a mechanism that takes advantage of each of them. This work represents our first contribution described in our first accepted paper. We devoted the rest of our work to the development and validation of a selection mechanism, which aims to select the best model among a set of predictors based on the received workload. This work has been published in our second paper. We started by formulating the problem as a multi-task learning problem to select the most suitable learning model for a given workload and each VNF of an SFC. To accomplish that, we used the models developed in the first part of this thesis as a part of the meta-learning strategy. To develop our MT-MLS model that selects the best suitable forecasting model to trigger based on the current received workload we followed the following process; 1) we succinctly conducted a state-of-the-art study followed

by a problem formulation; 2) a prediction mechanism was defined; 3) the mechanism has been validated and compared with various baseline resource forecasting models.

To summarize, we present in figure 0.2 the five main steps through which we cover the third and the fourth objectives.

- Step 1: Develop a VNFs neighboring dependencies based forecasting process

  In the third objective, we intended to enhance the efficiency of the prediction mechanism by including new features of the SFC in the learning process. Instead of using only historical consumption, we proposed to use the VNFs neighboring dependencies. Therefore, we developed a graphical neural network for resource consumption prediction that exploits architectural features of an SFC, to identify resource consumption dependencies between neighboring VNFs and consider them in the resource forecasting computation.
- Step 2: Develop DL prediction models

  Since deep learning-based techniques are well known for their adaptation ability to data fluctuations, we developed four DL-based models for resource consumption prediction. We tested and analyzed each model, and we iteratively enhanced its performances by testing different preprocessing techniques and testing different model architectures.
- Step 3: Deep models analysis

  In this stage, we intend to analyze deeply the proposed models to understand their limitations. Thus, we defined multiple workload scenarios, and assessed the efficiency, reliability, and cost (memory and CPU consumption) of each model. Then, we performed an extensive test experiment on each workload scenario. We compare the five DL models in terms of accuracy and in terms of resource and time consumption. The comparison study reveals the weaknesses of each model for some test scenarios. We noticed that the performances of a predictor may degrade significantly for dynamic workloads. We determined that a universal solution is not applicable, as the effectiveness of a DL model is contingent upon the characteristics of the training data. To alleviate this limitation, we propose an adaptive

prediction mechanism that enables a reliable estimate of future resource demands. Therefore, in the third part of this thesis, we intended to answer the following research problem, which DL model should be used to guarantee accurate prediction of resource consumption under specific variable workloads? To answer this question, we added to the research methodology the two following steps.

- Step 4: VNFS similarity analysis

  In this stage, we intended to capture consumption similarities between the different VNFs of an SFC and let the network focus on various consumption profiles. To achieve that, we defined a module that computes the similarity between historical consumption of multiple resources of two different VNFs using statistical methods.

- Step 5: Prediction mechanism

  In this final stage of our strategy, we leveraged the strategies proposed in previous stages of our research work to develop an efficient resource prediction mechanism named MT-MLS, which is an adaptive prediction mechanism based on a meta-learning technique. It uses VNF similarity and DL model selection mechanism to provide efficient forecasting of resource consumption.

0.4 Contribution

Our main contributions are summarized in the following:

- An efficient GNN Model that uses SFC topological features to meet the application performance requirements in high load traffic comparing to the last past consumption.
- A deep analysis of five DL models for resource prediction.
- Resource consumption similarity analysis module for VNFs to enhance resource consumption prediction.
- A mechanism that provides efficient and adaptive resource prediction for VNFs without relying on any pre-existing knowledge. The proposed mechanism can be enhanced by

integrating other types of resource prediction methods such as resource prediction based ARIMA model Calheiros *et al.* (2014), Khandelwal *et al.* (2015).

These contributions have been published in two journals:

1- Bellili, Asma et Kara, Nadjia. 2023. «An efficient adaptive meta-learning model based VNFs affinity for resource prediction optimization in virtualized networks». Journal of Network and Systems Management, vol. 31, nº 2, March 2023.

2- Bellili, Asma et Kara, Nadjia. 2023. «A graphical deep learning technique-based VNF dependencies for multi-resource requirements prediction in virtualized environments». Journal of computing, October 2023.

This thesis is organized as follows: Chapter 1 presents a review of recent related works. Chapter 2 is a part of material published in our first paper Bellili & Kara (2024), the chapter describes the proposed GNN-based VNFs neighboring structure. Chapter 3 presents our Prediction mechanism which is based on our second published work in Bellili & Kara (2023), while Chapter 4 concludes the thesis and highlights future opportunities that this research work opened up.

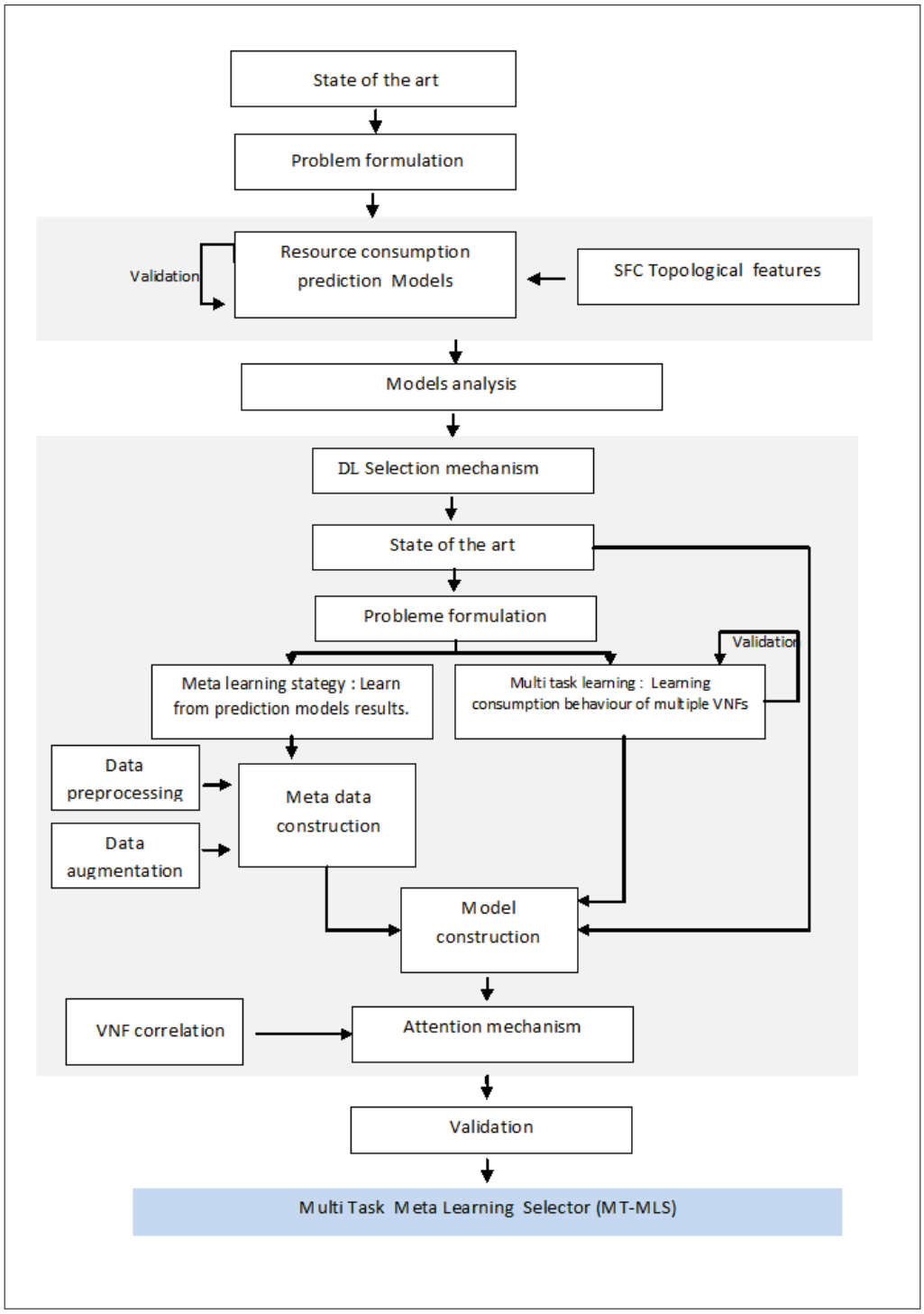

Figure 0.1 Research methodology: development of models for resource consumption prediction using VNF dependencies based on

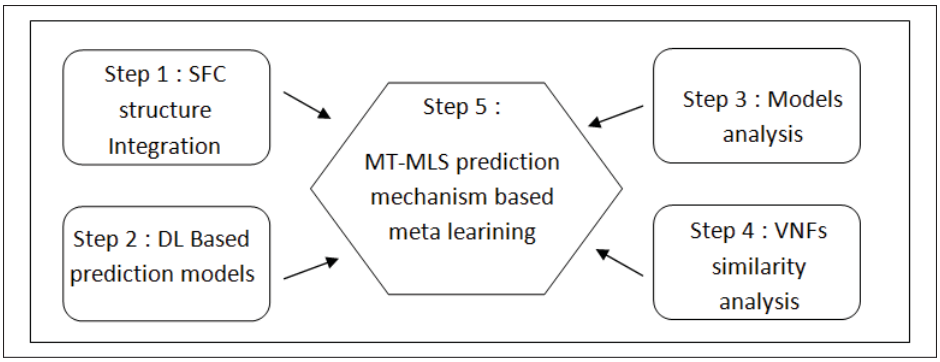

Figure 0.2 Research steps for the development of MT-MLS prediction mechanism based meta learning

CHAPTER 1

STATE OF THE ART

In this chapter, we conduct a literature review of resource requirement prediction in cloud computing. This chapter is structured into three sections. The first section presents a review of resource prediction techniques related to our contributions presented in chapter 3 and classified into classical methods and deep learning based methods. The second section includes a review of three main aspects related to our contributions presented in chapeter 4, manely resource prediction techniques, entities (e.g., VNF) interdepencies and multipredictors-based forcasting techniques. Finally, the fifth section situates our work within the current state-of-the-art and explain the motivation behind the contributions of this thesis.

1.1 Resource prediction

Network Function Virtualization (NFV) has garnered considerable attention from both researchers and the industry in recent years, with a particular focus on resource optimization and cost reduction. This heightened interest has spurred researchers to delve into resource management, specifically emphasizing resource consumption prediction.

Resource consumption forecasting relies on time series analysis, where historical data is analyzed and leveraged to estimate the expected utilization trends of resources. It consists of estimating the amount of a given or a set of given resources for a VNF Sangani *et al.* (2024), Jmila *et al.* (2017), Mijumbi *et al.* (2014a).

Moreover, it helps to identify patterns in resource consumption.Accurately predicting resource consumption enables us to dynamically adjust resources in response to workload variations. However, the prediction in a multidimensional context where multiple types of resource usage are forecasted at the same time poses a significant challenge, particularly when predicting multiple resources simultaneously using the same predictor. Indeed, the resources consumed by VNs are different (e.g., various amounts of CPU, and memory consumption) and may vary over time (dynamic resource consumption) A Vouk (2008). Many surveys Weingärtner *et al.*

(2015), Masdari & Khoshnevis (2019), Anuradha & Sumathi (2014) compare and classify those methods.

In this section we propose to review the most recent works on resource requirement prediction in virtualized networks. we classified the methods into two main categories: classical methods (e.g., ARIMA, Markov Decision Process) and deep learning techniques.

1.1.1 Classical methods

The majority of classical methods consist of statistical approaches such as ARIMA models Sekwatlakwatla & Malele (2024), Calheiros *et al.* (2014) and Markov decision process (MDP) Shi *et al.* (2015).

ARIMA models have been used in many prediction problems such as in predicting glucose and cholesterol levels of a patient Krishnamoorthy *et al.* (2024), and in aviation failure events prediction Zeng *et al.* (2024). Many works was also proposed for workload and resource consumption prediction in virtualized networks. In Bi *et al.* (2024) authors used ARIMA model to forecast workload, after a smoothing step of the time series data using a Savitzky–Golay SG filter, the data is divided into multiple components using wavelet decomposition and passed to an ARIMA model to forecast the workload future trends. Calheiros et al. Calheiros *et al.* (2014) proposed an ARIMA model to predict workload in public clouds characterized by high fluctuations.

Markov decision process was also widely used to forecast the future use of resource Shi *et al.* (2015) Gong *et al.* (2010)Nguyen *et al.* (2013). In these models, resource consumption is segmented into discrete bins of fixed size, and the prediction of future resource utilization is made by estimating the most probable future state.

Other classical methods rely on meta-heuristic algorithms. For instance, Fan-Hsun et al. proposed a genetic algorithm to predict the CPU and the memory utilization of a VM in a data center for VM placement Tseng *et al.* (2017). Many others proposed methods used heuristic

algorithms to tune parameters of a machine learning based prediction model such as in Simaiya *et al.* (2024), authors proposed a hybrid method, in which they propose to use deep learning, Particle Swarm Intelligence and Genetic Algorithm ("DPSO-GA") for workload provisioning. In the first stage, authors employed a hybrid PSO-GA method to fine-tune the hyper-parameters of the DL prediction model, then the prediction model which is a hybrid CNN-LSTM NN is trained to forecast the resource consumption. Also, authors in Jeddi & Sharifian (2019) used two heuristic algorithms: the Water Cycle Algorithm and the Artificial Immune System to optimize parameters of a wavelet neural network (bias and weight) designed to predict future workload demand.

In Jmila *et al.* (2017), the authors employed the support vector regression (SVR) technique to forecast the CPU utilization of a Virtual Network Function (VNF). The SVR was trained using a set of input data consisting of a parameter vector describing the input traffic and CPU consumption. Also, in Nehra & Nagaraju (2022) authors proposed a Support Vector Regression (SVR) approach designed to forecast the future demand of a host by analyzing the historical utilization of multiple resources. Authors proposed a hybrid kernel function combining the radial basis function (RBF) and the polynomial kernel function (PKF). This hybrid kernel is then employed to train the SVR model using the historical utilization data of multiple resources.

In Hsieh *et al.* (2020), the authors presented an approach to the consolidation of virtual machines (VM) to reduce energy consumption while maintaining the QoS. To achieve these goals, they considered the use of current and future resources. Future resource consumption is determined through the Gray-Markov prediction model. The results of their experiment showed a decrease in the number of virtual machine migrations and a reduction in energy consumption.

In Chen & Wang (2020), the authors proposed to deal with the problem of resource consumption prediction in a cloud computing environment by applying an adaptive short-term prediction algorithm based on the ARIMA model. This algorithm primarily uses the principal component analysis (PCA) method and the detection of extreme values to pre-process the data. Subsequently, the prediction method that gives the best precision is selected among those proposed to anticipate

the demand for resources. The authors also considered adjustment techniques for error prediction. The proposed approach outperforms other existing solutions such as ARIMA and the neural network of backpropagation (Back Propagation Neural Network: BPNN) in terms of prediction accuracy.

Several approaches for resource consumption prediction (e.g. CPU usage, memory consumption) are proposed in the literature. They are based on models such as: Kalman filterHu *et al.* (2013), the Kriging method Gambi (2012) and the autoregressive moving average models (ARMA) Iqbal & John (2012) Hoong *et al.* (2012). The results of these approaches are promising, but they remain relative to the contexts of their evaluation and cannot be generalized. These approaches do not include a prediction adjustment and therefore don't perform in the presence of fluctuations and sharp variations in the workloads.

1.1.2 DL based methods

Resource demand prediction in dynamic and virtualized environments such as cloud computing is very challenging. In such a complex and dynamic context, the approaches based on classical methods such as ARIMA models Chen & Wang (2020) are insufficient face to the traffic variation.

Thus, in recent studies, researchers have proposed to predict resource demand through deep learning methods.

In Dubba *et al.* (2024) authors studied the impact of the use of machine learning based prediction model on resource allocation. Thus they designe several predition models based on : Adaptive Boosting algorithm, bagging, extremely randomized trees, histogram based gradient boosting, lightGBM, bayesian network, lasso and Poisson regression models to forecast the future workload, then used the prediction results in the resource allocation test simulation.

In Taha *et al.* (2024) authors introduces a real-time proactive auto-scaler system designed to optimize resource allocation for Virtual Network Functions (VNFs) within a Service Function

Chain (SFC) in a cloud platform. The proposed system relies on a a hybrid MLP-LSTM model to predict the CPU and the memory future utilization.

In Zhang *et al.* (2018a), the authors proposed a deep learning model for predicting the CPU utilization of a virtual machine (VM). The model is constructed using a stacked autoencoder that employs the canonical polyatomic decomposition to decrease the training process's execution time.

In Lu*et al.* (2016) The authors introduce a workload prediction model, termed RVLBPNN (Random Variable Learning Rate Backpropagation Neural Network), designed for energy-efficient Cloud Computing. The proposed model is built upon the Backpropagation Neural Network (BPNN) algorithm.,The authors conducted experiments to evaluate the prediction accuracy and demonstrate that their RVLBPNN improves the prediction accuracy compared to the Naïve Bayes Classifier model and the Hiden Markov Model (HMM). The main drawback of the RVLBPNN is that the network traffic variation affects seriously the RVLBPNN performances.

Long Short Term Memory based methods

LSTM NN is also widely used in resource utilization prediction,

In Yuan *et al.* (2024) authors proposed a model named VSGB that combines LSTM NN, Grid LSTM, Variational mode decomposition and Savitzky Golay to forecast resource consumption in cloud environments. In Jeong *et al.* (2023) authors introduced a Proactive Hybrid Pod Autoscaling model (ProHPA), the model is based on a bidirectional long short-term memory (Bi-LSTM) model developed with an attention mechanism to predict CPU and memory consumption. Imdoukh *et al.* (2020) proposed an auto-scaler architecture based on a prediction model, the model is designed as an LSTM NN to forecast future CPU and memory demands. In Yazdanian & Sharifian (2021) Authors proposed a generator adversial network (GAN) model to forecast the future cloud workload, the proposed GAN is based on LSTM NN used as a generator. The input data are first decomposed into small components using Empirical mode decomposition EMD techniques, then fed to the LSTM NN to forecast each future component,

then the components are combined in the GAN discriminator designed as a 1D Convolutional neural network.

Zhang *et al.* (2020) have introduced an LSTM-based model for an end-to-end online prediction of network traffic. In Ouhame *et al.* (2021), the authors utilized the Long Short-Term Memory (LSTM) model combined with Convolutional Neural Network (CNN) to forecast the future consumption of CPU, memory, and network usage. In Song *et al.* (2018) authors proposed a deep learning model based on the LSTM to predict high-dimensional data of future virtual network function service chain VNF-SC requests Li *et al.* (2018).

In Gupta & Dinesh (2017), the authors proposed to use BLSTM NN, a variant of the LSTM NN that learns by analyzing both backward and forward dependencies in the time series. In Bi *et al.* (2021) authors expanded the BLSTM NN into a model called BG-LSTM by incorporating the GridLSTM proposed by Kalchbrenner *et al.* (2015) to forecast the workload and resource consumption. In Ouhame *et al.* (2021) the authors proposed to use LSTM NN coupled to a convolutional neural network, to predict the future consumption of the CPU, the memory, and the network usage. In Zhu *et al.* (2019), the proposed an LSTM network with an attention mechanism for workload prediction. Several similar research works exist that propose machine learning-based resource consumption prediction methods, but to the best of our knowledge, none of them achieve good accuracy in a dynamic and multidimensional environment (e.g. memory, CPU, bandwidth). Indeed, machine learning-based methods are recognized for their sensitivity to data, and their performance can be influenced by the size and variability of the training data Qiu *et al.* (2016) Mijumbi *et al.* (2014a); Jmila *et al.* (2017). Furthermore, we observe a noticeable difference in terms of performance and accuracy achieved by various ML-based prediction techniques due to the interdependencies between VNFs of an SFC (workload and consumed resources).

Graphical neural network based methods

Graphical neural networks (GNN) Scarselli *et al.* (2008) are recognized for their suitability in communication network problems. Their big learning ability to capture the spatial information of the network topology, and for their generalization capacity they makes them very useful in networking, where the topologies are dynamic, so the architecture is subject to change. GNNs have been largely used in virtualized networks, specifically in SFC placement Xiao *et al.* (2019) VNF placement Wang *et al.* (2021) and VNF management Kim *et al.* (2020) problems. Some works have also been proposed for VNF resource prediction using GNNs. In Li *et al.* (2024) authors proposed the Evolution Graph for Workload Prediction model (EvoGWP), this model is a graph-based evolution learning algorithm designed to forecast long term workload changes. In a first stage, authors proposed to extract shapelets to identify resource usage patterns of workloads, then the shaplets are fed to a a spatio-temporal GNN-based encoder-decoder to forecast future workload.

Jalodia *et al.* (2019) Mijumbi *et al.* (2016a) Mijumbi *et al.* (2017) Moradi *et al.* (2022), where the authors used the theory of GNN to exploit relationships between the different components of the SFC. In Mijumbi *et al.* (2017) and Mijumbi *et al.* (2016a), authors do not take into account the execution time of the model. But in such environments, the flow may be very volatile (sudden changes in traffic load). Thus, the VNF's resource consumption can change drastically. Therefore, a single-trained model might be inefficient in such a dynamic environment. To address this issue, the training is triggered each time there is a need to enhance the prediction accuracy using the latest available traffic data. Therefore, the training time of a prediction model is important to react fast to changes in traffic load.

1.2 Input correlation and entities' interdependencies, and multipredictor-based forcasting techniques

This section describes the e most recent works related to our contributions presented in chapter 4, including Prediction using input feature correlation, works analysiing the entities' interdependencies and works based on algorithm selection.

1.2.1 Prediction using input feature correlation

Some of the recent work has involved correlation coefficient (Pearson correlation, Spearman and Kendall correlation) Gulhane *et al.* (2023), Xu *et al.* (2023). The coefficient of correlation is applied to the input training features. Thus, the feature that does not improve the prediction accuracy is not taken into consideration in the learning process, such as in Antwi *et al.* (2021) authors applied this concept to determine the commodity of future price. Zhang *et al.* (2018b) applied a similar concept to predict the operation status of industrial IoT equipment by analyzing the equipment time series data, they used correlation coefficients to select the most correlated input features that are similar to the observed features in the training process of the proposed LSTM model.

In Singh *et al.* (2024) authors proposed a feature extraction and Clustering methods for CPU future consumption prediction. In the first step authors extract relevant features from the input dataset (CPU consumption of multiple machines), these features are then used in a clustering analysis to group machines with similar behavior together in order to identify similarities among machine's consumption. In the second step, authors developed a generalized version of N-Beats for CPU usage forecasting based on the consumption similarity clusters.

1.2.2 Entities' interdependencies

Numerous studies have showcased the effectiveness of incorporating dependencies between entities during the learning process. such as in Feng *et al.* (2018), the authors consider spatial dependencies between entities to predict network traffic. In Wang *et al.* (2017), authors proposed to apply an auto-encoder to capture both global and local spatial traffic dependencies among neighboring cell towers. Nevertheless, the proposed auto-encoder is limited to the neighboring spatial dependencies by predesigned neighboring areas.

In Dogani *et al.* (2023) authors proposed a hybrid model for workload prediction, the proposed model comprises a statistical analysis of the relationship between the structure of the training data, the different variables and the CPU consumption rate to identify a correlation between

the variables. This analysis is done through a CNN network that extracts features vectors from the input data, then a Gated recurrent units(GRU) with the attention mechanism is proposed to extracts the temporal correlation features.

1.2.3 Multipredictor-based forcasting techniques

Some state-of-the-art research works have proposed to use multiple predictors to forecast future consumption of resources such as in Sekwatlakwatla & Malele (2024), authors proposed to used Autoregressive Integrated Moving Average (ARIMA), Monte Carlo, Extreme gradient boosting regression (XGBoost) to forecast traffic flow.

Authors in Kim *et al.* (2018) use four models : Linear Regression, ARIMA, SVM, and ARMA to forecast the future workload of data centers. The authors use a regression model to determine the coefficient of each prediction model. Using the same principle, Rahmanian *et al.* (2018) proposed to forecast the CPU utilization of Virtual Machines using automata theory allowing to determine the weight of each predictor. However, those hybrid methods use all predictors together to compute the model output of the query workload, which makes them hybrid methods and does not select a predictor depending on the current workload profile.

In Wu *et al.* (2013), authors proposed a feedback control algorithm. The algorithm calculates different resource advantages and cost combinations. The authors proposed to quantify the cost considering two costs of the infrastructure the transition. The infrastructure cost denotes the expense incurred by tenants for renting Virtual Machine resources to host their applications, and the transition cost is evaluated based on the SLA violations during reconfiguration and the duration of reconfiguration. The total cost is then obtained by the summation of the transition and the infrastructure costs. The authors defined the resource advantage as the satisfaction derived for the application remaining in the new configuration while the SLA is respected. Once the advantages and the cost of each combination are evaluated, the algorithm selects the best combination for the highest profit and lowest cost.

In Herbst *et al.* (2017), authors proposed load prediction using artificial neural network,a decision tree, ARIMA model, support vector machine, and Bayesian network for proactive workload prediction. The proposed approach enhances the workload prediction accuracy, however, it is time-consuming. In Markham & Rakes (1998), authors proposed two steps methodology of the ARIMA residual hybrid system. The first step relies on an ARIMA model, to analyze the linear part of the time data. The second step consists of developing an NN model, to model the residuals from the ARIMA model (non-linear component of the time series). The authors motivate their methodology by the fact that the ARIMA model does not capture the nonlinear structure of the time series data, while the residuals of the linear model will contain information about non-linearity. The NN is used to predict the error for the ARIMA model.

In Khandelwal *et al.* (2015), authors applied a two-step methodology using discrete wavelet transform (DWT). The two-stage methodology consists on a decomposition stage and a reconstruction stage. The decomposition stage consists of decomposing the series into filters that identify the highest and the lowest frequency components of the series using DWT. The reconstruction stage consists of reconstructing the low-frequency and the high-frequency components through IDWT (Inverse DWT) Al Wadia & Ismail (2011). After the decomposition-construction stages, the authors applied an ARIMA on the reconstructed high-frequency component, and an ANN on the applied to the corresponding residual along with the low-frequency part. In the end, the combined predictions are derived by summing up the predictions from the two components.

Chen & Wang (2020) proposed an adaptive selection technique that selects one model, either the EEMD-RT-ARIMA, or the EEMD-RT-ARIMA prediction method. The selection process chooses the method having the lowest MAPE value computed on the last observations used for the prediction.

In Zharikov *et al.* (2020), the authors proposed an adaptive workload forecasting approach in cloud data centers by combining several methods, including ARIMA, Linear Regression (LR), Exponential Smoothing (SES), and a set of training data window sizes. This method enhances

prediction accuracy, but it has a high computational cost. Additionally, it is closely tied to linear time series models and cannot be extended to non-linear time series models, limiting its applicability to nonlinear time series datasets.

In Alidoost Alanagh *et al.* (2023) authors proposed an architecture for workload prediction. In the first stage, the workload is classified into different cluster using a classifier trained on sequential statistical characteristics extracted from user's workload. Then, an adaptive model has been developed to select the most suitable algorithm for workload prediction. This model evaluates the statistical characteristics of the workload to choose the best prediction algorithm among from Linear Regression (LR), Support Vector Machine (SVM), and AutoRegressive Integrated Moving Average (ARIMA) to predict the workload.

In contrast to our model, our approach is independent of the base predictor models and can be applied to various sets of predictors that can output consumption predictions of a set of VNFs.

1.3 Motivation and positioning of our research

The proposed approach in this thesis distinguishes itself from the other state oft the art approaches in many aspects, we can mention a few in the following items:

- VNF dependencies:
  Most of the state-of-the-art methods for resource prediction rely only on historical observations of a network entity, St-Onge *et al.* (2023) Taha *et al.* (2024) which limits the efficiency of the models in terms of capturing behavioral consumption changes of the network entities. In this research work, we propose to include graphical aspects of the SFC that enable capturing the change of resource consumption by analyzing the topological relationship between the network entities and involving locale neighborhood features in the learning process of each VNF's resource consumption.
- A selection mechanism: The choice of the right method is a challenging task for prediction mechanisms. This is attributed to the fact that in the real world, a formulation of a dynamic prediction problem is complex, since a single model may be efficient in predicting resource

consumption for specific test scenarios (e.g., non-volatile VNFs' resource consumption) and not for others (e.g., volatile VNF's resource consumption). Typically, a selection mechanism is used to choose one learning model among different ones to achieve the highest prediction accuracy. However, the selected model at the end may not be necessarily the most suitable for future uses. This is due to diverse potential influencing factors, such as model uncertainty and sampling variation. Many empirical studies proved that using several different models and combining them can often improve the prediction accuracy over an individual model. Most of the multi-model forecasting methods are hybrid methods that output a forecasting result by combining multiple prediction models Kim *et al.* (2018), Rahmanian *et al.* (2018). Several models are used at once, and the output forecasting result will correspond to a combination of those model results. In other words, the predictions are always computed by the same predictors. Contrary to our work, the most suitable predictor is selected depending on the current workload feature. To the best of our knowledge, there is no existing research work proposing a technique that combines multiple ML models to automatically select the appropriate one while taking into consideration the workload variation and the interdependencies between VNFs of an SFC.

- VNF consumption similarity: Due to the sparsity and the heterogeneity of VNFs, there are no existing methods that are specifically designed to process directly the resource consumption similarity between VNFs. Consequently, incorporating these discrete factors into the prediction model poses a challenge. In our work, we propose a framework that processes the resource consumption affinities between the different VNFs of the SFC, and take into account those affinities in the prediction process.
- Multidimensional forecasting: Most of the existing machine learning-based prediction methods are univariate methods Farahnakian *et al.* (2016), Chen & Wang (2020), Tseng *et al.* (2018). In a multidimensional resource prediction, using such methods is a time-consuming and inaccurate learning process, since the learning process must be performed for each single resource. For instance, in Qiu *et al.* (2016), the proposed method forecasts only the CPU consumption, and is only applicable for one VM. Besides, many methods in the state-of-the-art are highly related to the VNF nature, which limits the generality of the method

such as in Jmila *et al.* (2017), where the method reported a significant difference in the performance of various VNFs. In this research work, we propose a multivariate prediction model to forecast various resources at a time for different VNFs.

There are still gaps that need to be addressed to enhance the efficieny of resource prediction mechanisms in NFV environments. Therefore in this thesis, we propose a new multi-predictor-based resource forcasting mechanism to cope with the dynamicity of the running workloads. We propose multivariate resource prediction models to forecast CPU, bandwidth and memory needs of multiple VNFs at once. We studied the similarity between the VNFs resource consumption and use them in the prediction process. We also proposed a selection mechanism capable of selecting the most appropriate predictor depending on the current workload.

CHAPTER 2

A GRAPHICAL DEEP LEARNING TECHNIQUE-BASED VNF DEPENDENCIES FOR MULTI-RESOURCE REQUIREMENTS PREDICTION IN VIRTUALIZED ENVIRONMENTS

2.1 Introduction

Internet service providers deliver services utilizing various network functions (NFs) to cater to diverse traffic flows. Previously, these NFs were primarily implemented through specialized middlebox hardware, which leads to high capital expenses (CAPEX) or operating expenses (OPEX) Andrikopoulos *et al.* (2013), especially in case of a new middlebox deployment or network topology change. Recently, the European Telecommunications Standards Institute (ETSI) proposed network function virtualization (NFV), which is a network architecture that enables a software implementation for NFs running on virtual machines (VMs), so that they evolve independently of the hardware. However, NFVs bring a new issue for the management of those VNFs, including their placement and the resources allocated for each of them. The VNF placement was largely studied in many state-of-the-art works Mostafavi *et al.* (2021), while dynamic resource management in NFV is still a challenging task. The time taken for the preparation of a virtual machine is in the order of tens of seconds, even by the best known virtual infrastructure managers (VIMs). Also, in VNs the traffic augments and decreases which leads to a critical fluctuation in the resource consumption of each VNF. Thus, the amount of allocated resources to each VNF cannot rely on threshold-based scaling Salimian *et al.* (2016) because of the provisioning delay. Therefore, there is a crucial need for efficient data-driven mechanisms Amiri & Mohammad-Khanli (2017) Younge *et al.* (2010). These mechanisms enable complex systems to anticipate and efficiently react to workload fluctuations. This requires that the scaling decision must be known in advance. To achieve that, the amount of the future consumed resources must be accurately predicted. This way, the needed resources will be already available when the flow reaches the point of need, and released when they will not be needed. So that, the OPEX and the CAPEX are optimized, and the resource needs are satisfied while meeting the service level objective (SLO) Patel *et al.* (2009).

Several research works have studied resource management da Costa *et al.* (2022) and prediction in virtualized environments Masdari & Khoshnevis (2019) Anuradha & Sumathi (2014). However, most of them focus mainly on the historical consumption of each network entity independently of the other and do not take into consideration the relation between them. Different from these studies, we propose in this chapter to integrate the graph dependencies between VNFs of an SFC in the forecasting process. Also, most studies have analyzed each resource type individually Farahnakian *et al.* (2016) Chen & Wang (2020) Tseng *et al.* (2018). In a multidimensional resource prediction, those methods must be used for each resource type individually, which leads to a critical time prediction, since the whole forecasting process must be repeated for each single resource. Contrary to our GNN model, which forecasts three resource types in one computational time thanks to its output function architecture (see section 2.2.1). In this work, we evaluated the cost entailed by our GNN model and analyzed multiple scenarios where our GNN can outperform others, to identify where our GNN model can be trained considering the time constraint. Also, to demonstrate the generalization ability of our model, we tested the architecture on two different real-world data sets, while different works were achieved on simulated data Mijumbi *et al.* (2017). Also, to reinforce the prediction process, we proposed an augmented GNN that uses an augmented VNF feature vector. The augmented features allow guiding the output function computation by given a signed index of the previous consumption.

In this chapter, we present a graphical deep learning model (GNN) that processes a service function chain as a graph and exploits the neighboring relationship between the nodes. The proposed approach models relevant SFC neighboring information. This information allows the model to identify multidimensional dependencies between the VNFs. Then, compute the future consumption of a node based on the learned dependencies. The motivation behind using GNN is that an SFC represents a directed graph with a pairwise relationship between VNFs and an increase in resource demand for one VNFs may require an increase of resources for its VNF peer. For instance, an increase in resource demand for a Proxy Call Session Control Function (PCSCF) in IP Multimedia Subsystem (IMS) may require an increase of resources in a Serving CSCF node. This, implicitly, generates a dependency between the CPU consumption of both

VNFs. The dependencies might also be considered between different types of resources (e.g., CPU and memory in case of web server and its database). Therefore, the main contributions in this chapter are:

- A GNN model that captures VNF neighboring dependencies and involves them in the resource forecasting process.
- A detailed comparison study with LSTM, CNN, MLP and a Hybrid LSTM models in which extensive experiments were done on two real-world data sets to evaluate the accuracy and the efficiency of each model for several workload scenarios.

The chapter is organized as follows: Section 2.2 describes the proposed GNN model, as well as four other classical models used as baseline. Section 2.3 discusses the main obtained results. The section 2.5 concludes the study.

2.2 Methodology

In this work, we aim to efficiently forecast a set of different resources (such as CPU, memory, and bandwidth) of a set of network entities (such as a VNF). To accomplish that, we use a deep learning Neural network to predict the future multidimensional resource utilization of VNFs constituting a service function chain. We model the topological features of each VNF of the SFC by a feature vector. This feature vector will be used in the learning phase of a feed forward neural network (FNN)in order to learn the dependencies between different VNFs in terms of resource consumption (e.g. CPU, memory). Then, the learned dependencies are used by a second FNN to compute the future resource utilization. This is done using a graphical neural network architecture that gathers the two FNNs and preserves the architectural aspect of the SFC.

In the rest of this section, we present a graphical neural network-based VNF dependencies that exploits the architectural features of an SFC to compute the future resource utilization. Then we will present four other models including a convolutional NN, a simple feed forward neural network, an LSTM NN, and a hybrid model used as the baseline.

2.2.1 Proposed graphical neural network for resource consumption prediction

Our proposed model exploits dependencies between VNFs of the same SFC in the resource prediction process. Indeed, an SFC is a set of connected VNFs in which the traffic flow passes from a VNF to another. This architecture generates a dependency between a VNF and its neighbours in terms of resource consumption. Thus, we adopt a supervised learning GNN model, that aims to forecast multidimensional resource consumption of a set of VNFs. The use of the GNN is motivated by the fact that the node of the graph is processed depending on its neighbour's features. Instead of using a prepossessing step that maps the graphical information to structured data, such as a vector that may result in loos of topological dependencies, such as consumption relationship between neighbour VNFs of one or many resources. Biemann et al. Biemann (2016) have studied the difference between a vector and a graph representation and cited the limitations of vectors compared to graphs, such as semantic neighbourhood and computational limitations.

We model an SFC as a directed graph $G = (N, E)$, where N is the set of nodes and E represents the set of links between nodes. Each node in the graph represents a VNF and a path represents sequential links between VNFs of an SFC. The GNN consists of determining a state S_n for each node N, then determines its output O_n using the state S_n of the same node. The state and the output computation are based on the two following parametric functions named transition and output function, respectively. The transition function expresses the dependence of an VNF_n on its neighbours, and the output function describes how the predicted resource is computed depending on the nodes's states:

$$S_n = \sum_{m \in n*} H_w(F_n, F_m, S_m) \tag{2.1}$$

$$O_n = g_w(S_n, F_n) \tag{2.2}$$

Where F_m and S_m are the features and the state of neighbour m respectively. Therefore, our model includes three principal parts, as shown in figure 2.1. The historical resource consumption is monitored and recorded, as well as the graph of the SFC. After the reprocessing process, the data is fed to the modelling process, which constructs the node features vectors, the node state, and the GNN structure by the graph connectivity. The computed features are given to the transition function FNN to compute the states of the GNN nodes based on connectivity of an SFC. These states are used in the output function, which computes future resource utilization. For each node, the model computes its state and its output. Using the feature of the node and its neighbour's features and a state at time t, the transition function computes the new state at time $t + 1$. The predicted VNF state is then used in addition to its features to predict the output, which represents the resource consumption of the given VNF.

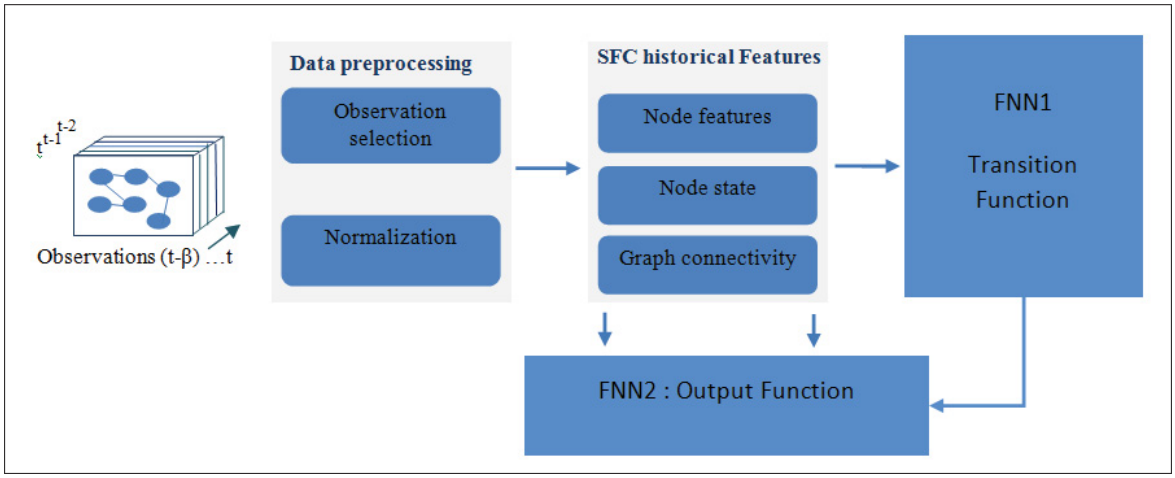

Figure 2.1 Overall model scheme of our GNN-based resource consumption prediction

VNF features

Each VNF is described by an augmented vector containing observations of its multidimensional resource consumption (see figure 2.2), and a consumption evolution index. In our case, we consider three dimensions: CPU utilization, consumed bandwidth and memory. The consumption index is computed using Euclidean distance based on vote majority strategy between the two VNF consumption vectors of the actual iteration and the previous one. i.e. we

compute the Euclidean distance between the normalized [CPU, Memory, Bandwidth] vector at time t and the normalized [CPU, Memory, Bandwidth] vector of the same VNF at time t+1. Then, the resulted distance is signed based on a majority vote, in which we count how many resources has increased their consumption, and how many has decreased. Then, we return the class (increasing/decreasing) with the most votes. If the most class returned is increasing, we encode the resulted distance as a positive feature value, else we encoded it as a negative feature value as a consumption evolution index. We can give an example of a VNF at two consecutive observations, with vectors' consumption equal to [0.225,0.521,-0.237] at a given observation and [0.325,0.746,-0.297]for the next observation. Thus, by computing the Euclidean distance between the two vectors, we obtain a value of 0.246. Then we attribute a positive sign corresponding to the augmentation of the CPU and the memory consumption. Thus, the augmented vector feature will be equal to [0.325,0.746,-0.297,+0.246]. The objective is to train a GNN on historical data of VNF resource consumption while reinforcing the learning by an augmented vector feature, which is used as input data for both transition and output functions.

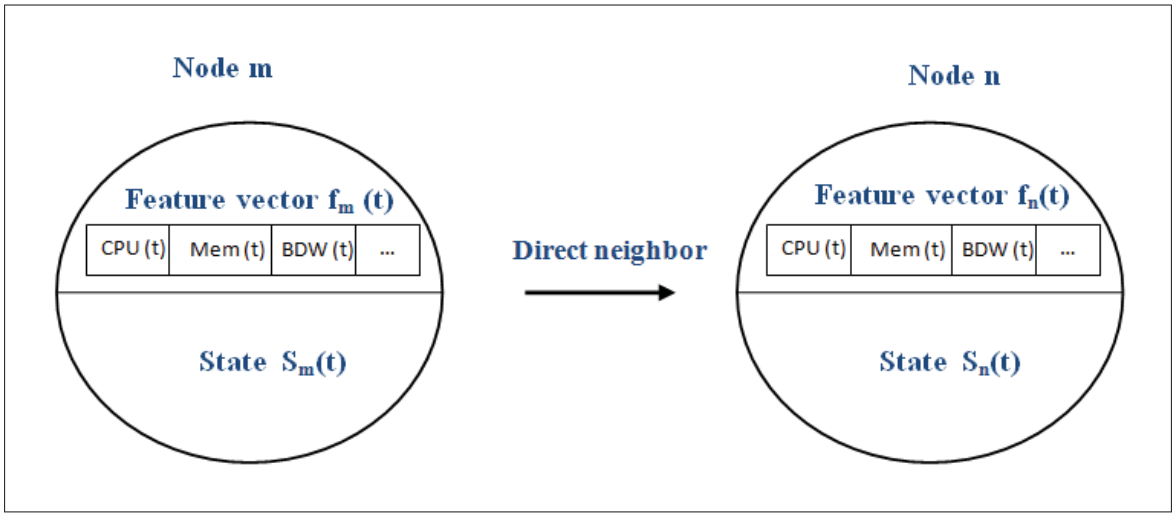

Figure 2.2 VNF features and state

State forecasting

The state is computed using eq. 2.1. In Scarselli *et al.* (2009) the authors demonstrated that the existence and the uniqueness of a solution to eq. 2.1 is guaranteed by Banach fixed-point

theorem. This requires that the global function H_w is a contraction map. If this constraint is satisfied, the state computation is done using an iterative model that stores the current state at time t, and computes the next state at time $t + 1$ when needed. In this work, we propose to use the GNN model cited in Scarselli *et al.* (2009) that models the transition function by a FNN Ramchoun *et al.* (2016) (see figure 2.3) which allows the convergence to a fixed point.

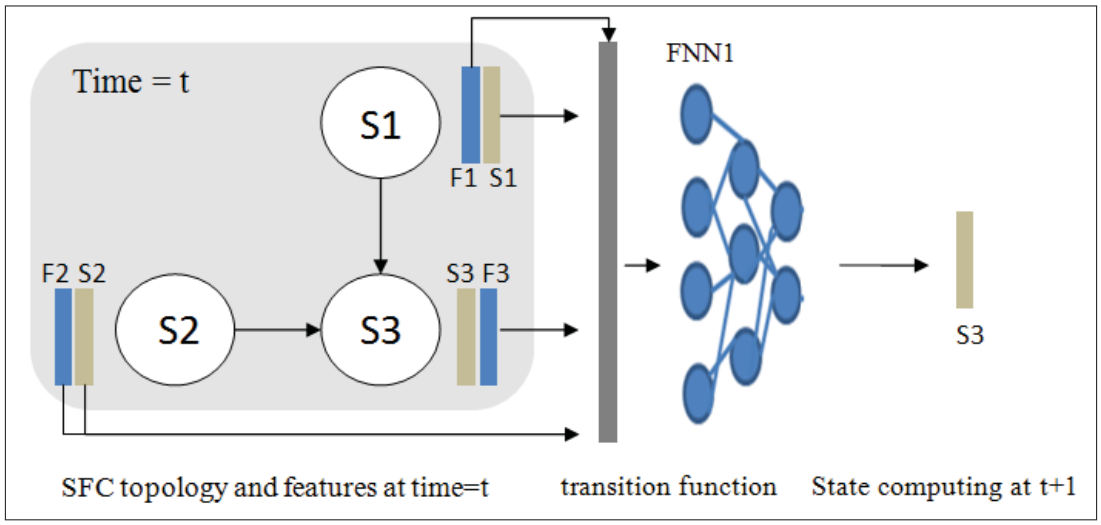

Figure 2.3 Example of state computation process of VNF 'S3' at time=t+1

The iterative state computation consists of passing the precedent states of the node n and its neighbor's states already computed by the transition function, as well as the observed features in the previous iteration. Then, the next state is predicted by the trained FNN. Figure 2.4 illustrates the state forecasting for an SFC composed of n VNF ordered from 1 to n.

Resource prediction

Predicting a VNF resource consumption is performed by computing the output function that takes the node state forecasted by the transition function and the feature node and produces an output corresponding to the future resource utilization of the VNF. This is achieved using the second FNN trained on a given period of time. Figure 2.5 shows the output computation scheme for a single VNF.

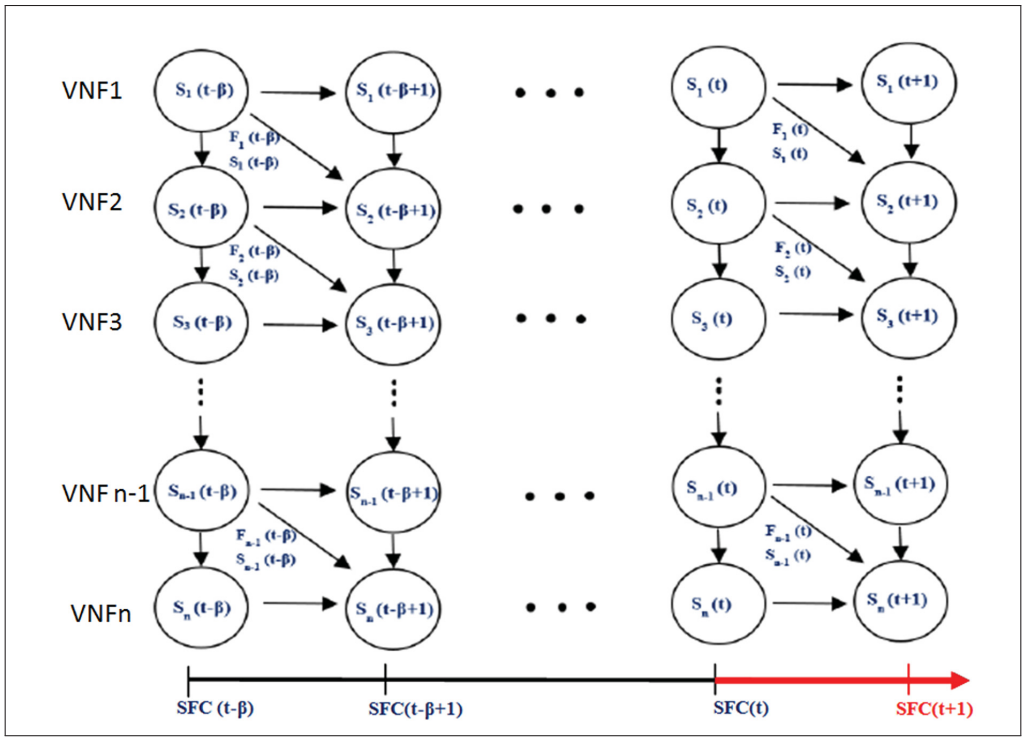

Figure 2.4 Iterative scheme of SFC states prediction

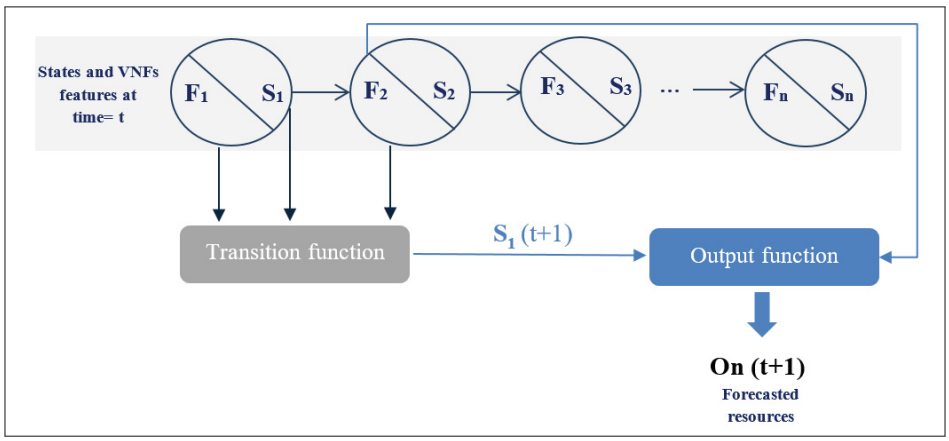

Figure 2.5 Output computation for VNF1

GNN algorithm

The GNN learning algorithm is based on gradient descent strategy and consists of the minimization of a quadratic cost functionScarselli *et al.* (2009), the following steps summarize the algorithm.

1. For each VNF_i state computation:
2. the features and the states vectors at time T of all direct neighbors are concatenated with the feature vector of VNF_i at time T producing the input structure of the transition function.
3. The sates are iteratively updated by the transition function (see figure 2.4) producing the VNF_i state at time T+1.
4. The gradient is computed based on back-propagation through a time algorithm, producing a layered network in which each layer corresponds to a time instant and contains a copy of all units F_w of the network. The units of two consecutive layers are connected following the SFC graph connectivity. The last layer at the time T computes the outputs of the network.
5. Then the weights are updated according to the gradient computed in the second step.

2.2.2 Deep learning techniques for resource consumption prediction

In order to investigate the impact of VNFs' dependencies and the efficiency of the proposed graphical model, we compare our results with four deep learning techniques used for resource consumption prediction.

CNN

We use CNN model Albawi *et al.* (2017) O'Shea & Nash (2015) that learns the variations of resource consumption of a set of VNFs to predict their future consumption. The model is shown in figure 2.6. It consists of five convolutional layers followed by a dropout layer and a max-pooling layer, then three other convolutional layers followed by three fully connected layers. Each convolutional layer is composed of parallel stack channels, where each channel consists of a layer of neurons sharing the same weights, filter size, and biases. The neurons constituting the layers pass the sum of their weighted inputs through a ReLu activation function.

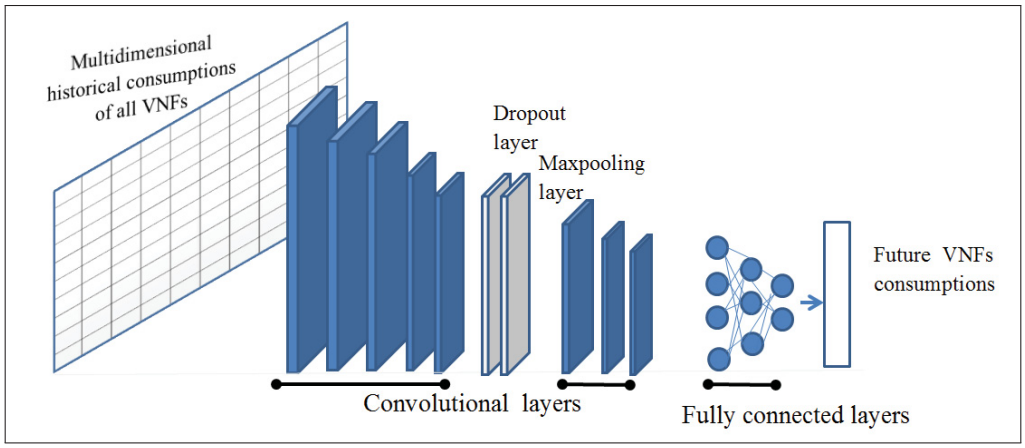

Figure 2.6 CNN resource forecasting model

MLP

In our study, we will compare the performances of GNN and classical MLP models Gardner & Dorling (1998). The MLP model is composed of three layers. The input layer is made of a number of perceptions equal to the number of our consumption feature attributes, and an output layer that returns the predicted resource consumption values, and a hidden layer with a number of neurons equal to the average of the neurons in the input and output layers. Such architectures enable to achieve universal approximations Scarselli & Tsoi (1998).

LSTM

LSTM NN Huang *et al.* (2015) are special cases of RNN Greff *et al.* (2016) proposed to address the problem of exploding gradients or vanishing. The LSTM NN are known to be relatively efficient for time series prediction Lindemann *et al.* (2021). Thus, we established an LSTM NN containing three LSTM layers. An LSTM layer is composed of a set of LSTM units, placed end-to-end to extract relevant features from long sequences of historical resource consumption of VNFs. The LSTM architecture is shown in figure 2.7. The first two layers are followed by a dropout layer, and the third by a fully connected layer. The prediction is then done using a Tanh activation function that predicts the future resource consumption of a set of VNFs. The

establishment (number of LSTM units, number of LSTM layers, number of dance layers, etc.) of the model and the parameter setting will be discussed in the experimental section.

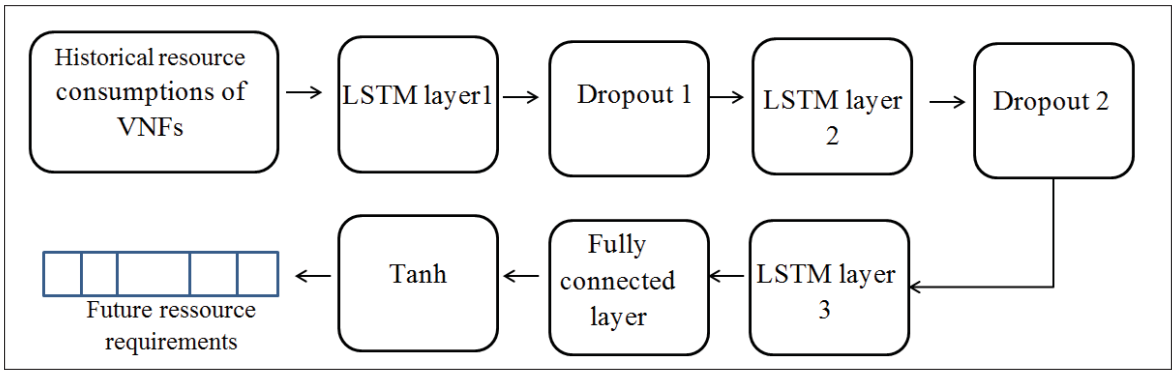

Figure 2.7 LSTM NN model architecture for multidimensional resource consumption prediction

Hybrid MLP-LSTM

This model is a hybrid predictor that combines LSTM and MLP networks. We propose a model that combines the advantages of MLP and LSTM in terms of simplicity and the ability to deal with long sequences. The model principle consists of passing the original data directly after a preprocessing step to an MLP network. This plays the role of a preliminary pretraining that produces discriminant features from the historical resource consumption of the VNFs. The resulted features are then fed to an LSTM NN that learns and predicts the future resource consumption of the VNFs. Figure 2.8 shows the proposed model.

2.3 Test experiments and performance analysis

In this chapter, we are specifically interested in identifying the performance of each model for different scenarios. These scenarios are described in section 2.3.5 . Each scenario defines a specific workload profile. Two datasets are used that represent the resource consumption of two SFCs, IMS and Web. These datasets are described in section 4.2. The five models are trained and tested using these two datasets, each dataset was randomly divided into 70% for the training, and 30 % for the validation and test. The models are trained on the same data, and preprocessed

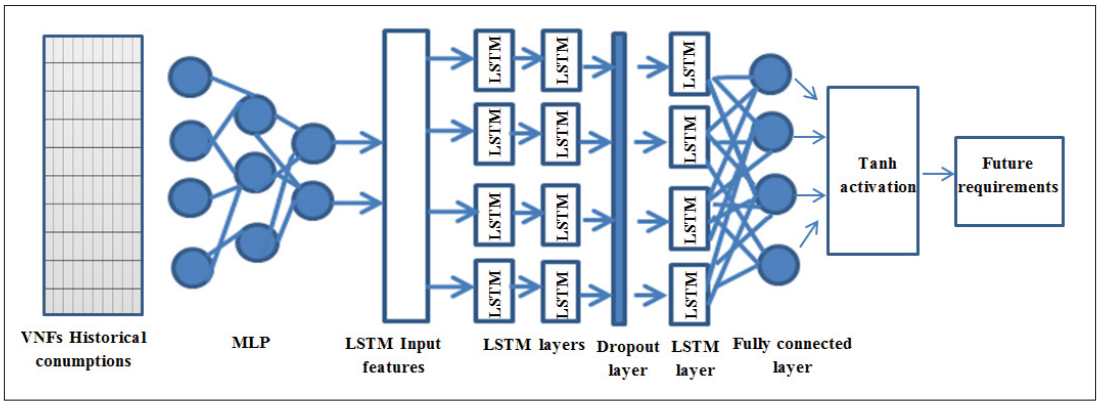

Figure 2.8 Hybrid LSTM MLP model

into input samples of size equal to five consecutive samples recorded every 20 seconds. Each sample contains the CPU, the memory and the bandwidth consumption of each VNFs of the SFC. All models predict three resource values (CPU, bandwidth and memory) as output values for each VNF of an SFC.

2.3.1 Computing

All computing was done on the same machine, the training time, the CPU and memory consumption of each training were recorded and compared. The models have been tested using Anaconda 4.2.0 (64-bit), Python 3.5.2, TensorFlow 0.12.1, Jupyter Notebook 4.3.1., Keras 1.2.1.

2.3.2 Data set description and evaluation metrics

In this work, we use datasets gathered from Clearwater IMS and web SFCs deployed in real-world virtualization environments (openstack-based virtualisation infrastructure). The CPU, memory and bandwidth consumption of each VNF were monitored and recorded every 20 seconds during 16 days. The Clearwater IMS SFC is composed of Bono, Sprout, Homer, Ellis, one instance that includes Astaire, Cassandra, Chronos and one Homestead instance. The web SFC is composed of a web application, three instances of Java APIs, and a database.

Three main metrics are used to evaluate the models: Mean absolute error (MAE) to measure the errors between the predicted and the real resource consumption, the root-mean-square error (RMSE) Chai & Draxler (2014) and the mean squared error (MSE) to measure how close are the estimated values to the real values. We have also computed the explained variance score which measures the dispersion of a given paired observation O'Grady (1982) and the coefficient of determination that measures the amount of variation between the real and the predicted values Ozer (1985), to quantify the degree of relationship between the predicted and the real variables. The mean absolute error (MAE), the mean squared error (MSE), the Root mean square error (RMSE), the explained variance regression score function, and the coefficient of determination are respectively given by equation 3.7-3.10.

$$MSE = (\frac{1}{n}) \sum_{i=1}^{n} \|y_i - x_i\|_2^2 \tag{2.3}$$

$$MAE = (\frac{1}{n}) \sum_{i=1}^{n} \|y_i - x_i\|_1 \tag{2.4}$$

$$RMSE = \sqrt{(\frac{1}{n}) \sum_{i=1}^{n} (y_i - x_i)^2} \tag{2.5}$$

$$R^2 = \frac{\sum_i (\hat{y}_i - \overline{y})^2}{\sum_i (y_i - \overline{y})^2} \tag{2.6}$$

2.3.3 Data preprocessing

To correctly forecast the future resource consumption, the data has been passed throw a preprocessing process that includes the elimination of invalid recorded values and empty values, followed by a normalization function using Min-Max normalization in the range [-1,1]. Then the data was split into input sliding windows such that each window represents a sequence of n historical consumption and the $n + 1$ observation represents the output vector.

2.3.4 Networks structure selection

In order to have a coherent comparison, the structure of each prediction model must be selected, the hyperparameters of the models should be optimized and the configuration of the base models should also be tuned.

To tune the hyperparameters, we have trained each model using the same dataset (IMS and Web datasets). The hyperparameters were tuned, by varying each one of the parameters on 10% of the IMS data set. The best reported scores were adopted in the training process. All tested model configurations were trained with initial learning rates equal to 0.01, with a min-max normalization function in the range [-1,1]. The proposed GNN configuration consists of tuning the GNN specific hyperparameter as well as the transition and the output functions parameters represented by two feed-forward NN. Thus, one of the most important hyperparameters for the GNN is the architecture of the used FNNs. In Scarselli & Tsoi (1998), the authors mentioned that the best architecture of an FNN that often produces universal approximations is a three layered network, with one hidden layer containing a fixed number of neurons to the average of the neurons in the output and the input layers. Thus, we adopted this rule to define the architecture of the GNN functions as well as the architecture of the MLP network. The resulting NNs architectures contain three layers in each network.

Also, to establish the CNN architecture we conducted experiments to analyze the following hyperparameters: number of convolutional layers with and without dropout layers, and the max-pooling vs average pooling layer. We performed several tests with different numbers of convolutional layers. In each time, we tested t the architecture with and without dropout layer. We repeated the same experiments using a max-pooling layer and then an average pooling layer. Table 3.3 shows the result of the MAE obtained from the different test experiments.

Table 2.1 MAE for CNN with different configurations

Number of convolution layers	Without dropout layer	With dropout layer	With average pooling layer	With max-pooling layer
5	0.203	0.209	0.097	0.121
8	0.158	0.019	0,024	0,026
9	0.175	0.011	0,022	0.029

The resulting architecture contains eight convolutional layers, a max-pooling layer and a dropout layer, in addition to the fully connected layer.

For the LSTM NN, we focused on the main specific hyperparameters, which are the number of LSTM layers and the number of LSTM units in each layer. The number of the units were fixed equal in all LSTM layers and is equal to the features in the sliding windows. For the number of LSTM layers, we tested different possible configurations using two, three and four LSTM layers followed by a dropout layer, and selected the optimal architecture based on the reported MAE (table 2.2). The selected architecture contains three LSTM layers followed by a dropout layer and a dense layer in the third LSTM layer that reported the smallest MAE.

Table 2.2 MAE for LSTM NN with different configurations

Number of LSTM layers	**Without dropout layer**	**With dropout layer**
2	0.203	0.184
3	0.134	0,112
4	0.128	0.115

For the hybrid model, we followed the same process as for the MLP network, fixing the number of layers to three and a number of input neurons depending on the input sliding window features. Then, in the same manner as in the LSTM network, we varied the layers structure and selected those reporting the best MAE scores. The resulted architecture is summarized in table 3.5.

Table 2.3 Hybrid LSTM-MLP hyperparameters

Hyperparameter	**value**
Number of LSTM layer	3
Number of dropout layers	1
Number of LSTM units	15
Number of fully connected layers	1
MLP layer number	3

2.3.5 Results and analysis

In this work, we analyze and compare the performance of five neural network models used to predict the resource consumption of VNFs composing an SFC. Our aim is to identify the more suitable model to use for resource prediction. We propose to dissect the problem into different sub categories in order to have a more precise analysis that will lead to a more efficient solution. Thus, we define four different scenarios to identify and analyze the performances of each model for each scenario. For all test experiments, the common parameters of the models were fixed to the same values in order to have a fairly and meaningful comparison. Table 3.2 summarizes the parameters and their values used in this section.

Table 2.4 Common parameters for the models training

Parameter	value
Activation function	Tanh
Learning epochs	100
Normalization	MinMax [0,1]
Learning rate	0.01
Sliding window	5

2.3.5.1 Scenario 1: Global comparison

In this scenario, we aim to evaluate the average performances of the five models using the two datasets gathered from the two SFCs stressed with different workloads. Thus, we evaluated the five models on each dataset separately. For each dataset, we used the entire dataset, which is divided randomly to have 70% of data for training and 30% for the test and validation. The results show that the errors computed using the two datasets are close to one another. For instance, the MAE values for CNN model are equal to 0.054 and 0.066 using IMS and web datasets respectively (table 3.6).

Figure 2.11(a) end (b), show the MAE, MSE, RMSE, variance, and the coefficient of determination reported by each model on the two datasets, note that for clarity purpose, all the metrics on

the test set were grouped and presented as a radar plot. The ideal model will be presented by a circle connecting all metric scores, having the size of the complete plot.

The reported scores show that the CNN and GNN models outperformed the other models for both datasets with respectively MSE value of 0.012 and 0.010 for the IMS data set and 0.042 and 0.058 for the web dataset, while recording the highest cost in resource consumption with 6748 seconds of computational for CNN in the IMS data set which may affect the efficiency of the prediction in terms of time consumption and cost optimization (table 2.5).

Table 2.5 Training time, training CPU and memory consumption recorded by the five models for the IMS dataset

Resource	MLP	LSTM	MLP-LSTM	CNN	GNN
Execution time in seconds	44	3480	5707	6748	153
CPU utilization %	36.7	91.7	92	57.4	41.2
Memory utilization %	50.6	76.1	83	78.4	64.7

2.3.5.2 Scenario 2: Specific comparison

In this scenario, we focussed on the IMS data set. Thus, in the first part of this scenario, we analyze the behaviour of each predictor for each resource (cpu, memory and bandwidth). Indeed, the fluctuation curve of the same VNF between the three resource dimensions can be different between two time intervals. Table 3.6 summarizes the average MAE computed for each resource type and for each model. We can observe from table 3.6 that the LSTM, the MLP and the hybrid LSTM-MLP models show generally good performances in CPU and memory prediction, while the MLP offers a low time complexity. Meanwhile, the GNN and the CNN show the best scores, with the outperformance of the GNN on the others in the bandwidth prediction.

In the second part, we analyze the resource prediction of each single VNF. This is motivated by the fact that at a given time t a given VNF may require intense resource consumption (e.g., CPU) compared to its neighbors, such as for the HSS and the S-CSCF. The experiments demonstrated that the performances of the models are highly related to the processed VNF. For example,

we present in figure 2.9 the predicted CPU versus the real CPU consumed by VNF9 (first line) on 100 consecutive observations. At the same time, we recorded and presented in the second line the predicted CPU versus the consumed CPU of another VNF (VNF10) for the same 100 observations and using the same prediction model (LSTM). Our aim is to highlight the performance fluctuation of a single prediction model between different VNFs, even for the same workload. Therefore, we investigated the results obtained for each VNF comparing to its neighbour. Figure 2.10 presents a performance comparison of two neighboring VNFs using two different prediction models. The predictions are computed on 50 consecutive observations resulted from CNN and the GNN models on two neighbouring VNFs noted VNF9 and VNF10. The results show that the GNN model performances are stable between two neighboring VNFs with a maximum MAE=0.005, while the CNN model graphic shows an important performance deterioration for VNF10 with a maximum MAE value equals to 0.06 compared to VNF9 with a maximum MAE value equals to 0.02.

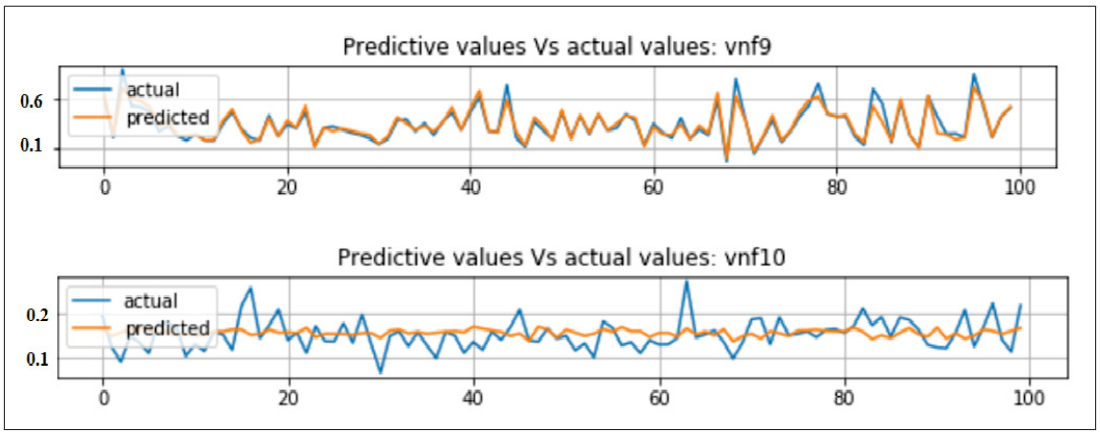

Figure 2.9 Comparison between the predicted CPU rate and the real CPU consumed rate of two VNFs for the LSTM model on 100 observations selected randomly from the test set

2.3.5.3 Scenario 3: Sharp workload fluctuation

In this scenario, we analyze a special case of workload that often occurs in virtualized networks, and is considered one of the main big challenges of resources forecasting and management in a general manner. In this scenario, sharp workload increase and decrease may happen over time,

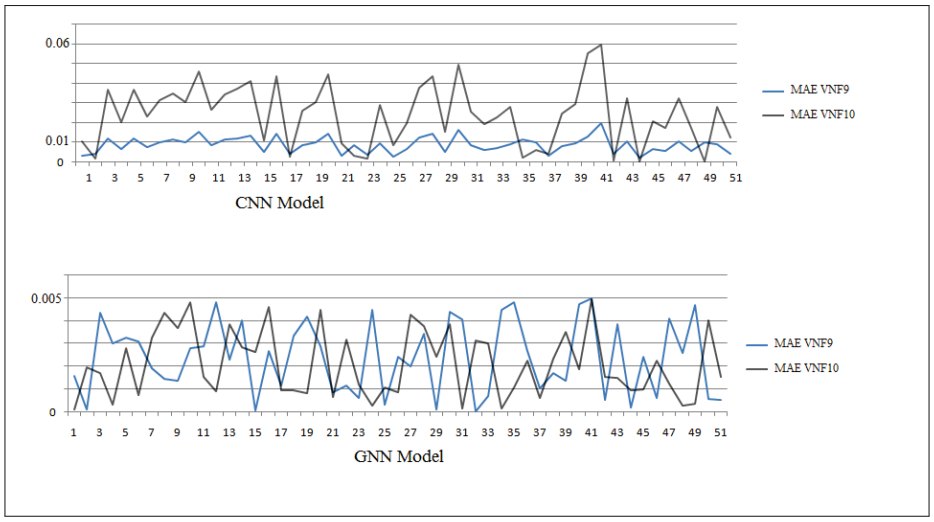

Figure 2.10 MAE errors on 50 consecutive test observations for VNF9 and VNF10 using the CNN and the GNN models

which may cause sudden changes in the consumption of the three resources (CPU, Memory and Bandwidth). To analyze such specific behavior, we divided the test set into two classes and tested the models on only the first class in which we have identified the samples with stable consumption followed by sudden changes in the last observation to be predicted for one or more dimensions.

We reported the average values of the evaluated scores in table 3.6. It is clearly shown that for this scenario the GNN model outperforms the other models with an MAE equal to 0.03 compared to the minimal value of 0.13 recorded for CNN model. We also present in figure2.11.C, the radar plot of the five errors computed for this scenario. The figure shows that the circle connecting the five errors of the GNN is the closest to the size of the complete plot circle. This indicates that the GNN model outperforms the other models for this scenario. Also, compared with the other scenarios presented in figure 2.11.a and in figure 2.11.b, we can notice that the circle shape of the radar plots is irregular, which indicates high error values and thus lower performance.

2.3.5.4 Scenario 4: Stable consumption

In this scenario, we intend to analyze the case where the consumption rate of a VNF is stable in time or with smooth variations. This includes the two cases of VNF with intensive resource consumption. A VNF with intensive resource consumption is the one that requires high and stable amount of at least one resource during a certain period of time. Similarly, a VFN with less intensive resource consumption will require small and stable amount of resources during a certain period of time. We present the obtained results of the five models in table 3.6.

We can notice from table 3.6 that all models show good performances for this scenario, especially for the MLP and the LSTM models where the MAE errors dropped considerably from $0,27$ and $0,17$ to $0,04$ and $0,016$ for the MLP and the LSTM respectively comparing to the high fluctuation scenario, meanwhile by comparing the cost entailed by the models training, we can clearly observe that the simple MLP network outperforms the others in terms of time optimization and consumed resources for the prediction process.

2.4 Discussion

The analysis of different experiments conducted from the different scenarios shows that the performances of each model depend on the resource consumption profiles. Indeed, we can see that in general cases, the CNN and the GNN outperform the other models having the most similar circle to the complete plot circle (see figure 2.11), this occurs when all computed metrics report good scores, and indicates the good quality of the predictor with a stability of the model. The hybrid MLP-LSTM, the MLP and the LSTM models have reported good scores in the MAE, RMSE, and MSE evaluations and poor scores in the coefficient of determination and the explained variance.

Nerveless, we observed that in some cases the prediction process of the CNN and the GNN fails such as in observation number 40 in vnf10 for the CNN model (see figure 2.10) with an MAE equal to 0.061, in addition to the huge execution time of the CNN comparing to the MLP model. Meanwhile, the GNN outperforms the others in the critical case of sharp workload fluctuation

Table 2.6 Reported errors by each model for different scenarios using IMS and Web datasets

Scenario	dataset	Metric	MLP	LSTM	MLP-LSTM	CNN	GNN
scenario 1	IMS	MSE	0,051	0,011	0,023	0,012	0,01
		MAE	0,087	0,057	0,087	0.054	0.057
		RMSE	0,093	0,087	0,065	0,075	0,071
		Explained variance	0,73	0,70	0,72	0,90	0,893
		Coefficient of determination	0,74	0,70	0,72	0,90	0,88
scenario 1	WEB	MSE	0,082	0,073	0,07	0,042	0,058
		MAE	0,085	0,095	0,1	0,066	0,051
		RMSE	0,14	0,123	0,132	0,098	0,086
		Explained variance	0,746	0,765	0,754	0,851	0,848
		Coefficient of determination	0,746	0,759	0,764	0,859	0,851
scenario 2	IMS CPU	MSE	0,048	0,013	0,020	0,011	0,008
		MAE	0,085	0,054	0,084	0,051	0,055
		RMSE	0,091	0,085	0,066	0,074	0,075
		Explained variance	0,75	0,72	0,77	0,92	0,93
		Coefficient of determination	0,74	0,72	0,78	0,91	0,93
scenario 2	IMS Memory	MSE	0,057	0,011	0,028	0,013	0,014
		MAE	0,084	0,0585	0,091	0,054	0,063
		RMSE	0,092	0,0867	0,062	0,076	0,0742
		Explained variance	0,75	0,747	0,728	0,9	0,87
		Coefficient of determination	0,76	0,742	0,730	0,904	0,865
scenario 2	IMS Bandwith	MSE	0,06	0,013	0,02	0,01	0,009
		MAE	0,09	0,0655	0,08	0,056	0,054
		RMSE	0,113	0,093	0,06	0,078	0,068
		Explained variance	0,80	0,73	0,72	0,87	0,89
		Coefficient of determination	0,79	0,73	0,72	0,86	0,89
scenario 3	IMS	MSE	0,27	0,17	0,2	0,11	0,01
		MAE	0,32	0,23	0,27	0,13	0,03
		RMSE	0,36	0,026	0,033	0,19	0,04
		Explained variance	0,50	0,66	0,61	0,79	0,94
		Coefficient of determination	0,51	0,67	0,62	0,79	0,94
scenario 4	IMS	MSE	0,04	0,016	0,034	0,01	0,021
		MAE	0,078	0,063	0,070	0,042	0,068
		RMSE	0,092	0,077	0,087	0,069	0,073
		Explained variance	0,94	0,83	0,87	0,96	0,93
		Coefficient of determination	0,94	0,82	0,86	0,96	0,93

with a MAE equal to 0.03, and the smallest execution time and resource consumption compared to the CNN, the LSTM and the MPL-LSTM models, where the CNN recorded a MAE= 0.26 with 153 seconds for the training time. Indeed, the GNN model has the advantage of the

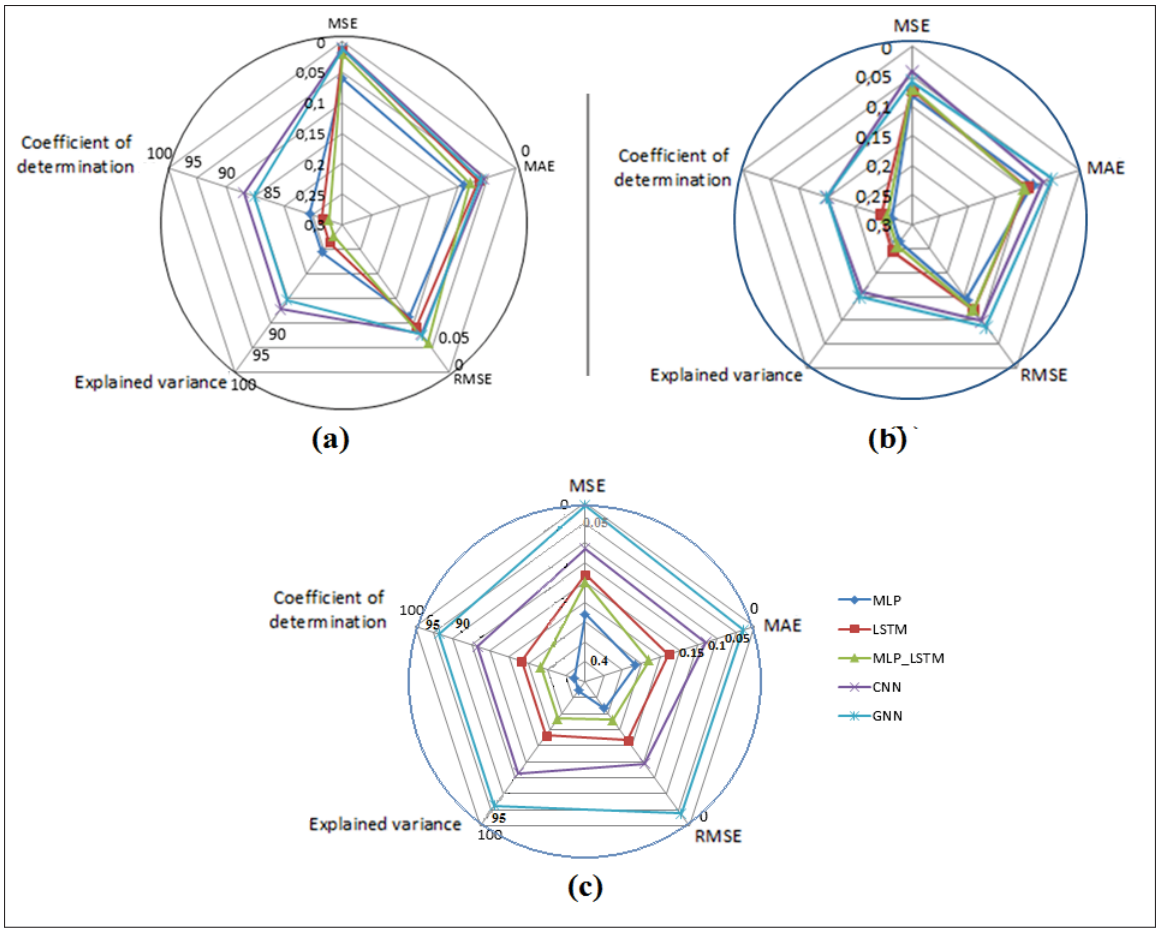

Figure 2.11 Performance evaluation scores on the test set for MLP, LSTM, hybrid MLP-LSTM, CNN and the GNN models. (a) scenario 1 : IMS dataset, (b) scenario 1: Web dataset, (c): scenario 3 case of big fluctuation on the IMS dataset

simplicity and the reduced training time of the MLP, since the training process concerns mainly to train the transition and the output functions implemented in our work by two MLP. Besides, the GNN has the advantage of using the topological dependencies involving the neighbours' features and forecasted states which enhance considerably the performances particularly in high fluctuation cases. This can be explained by the ability of the proposed GNN model to capture the dependencies between VNFs. This fact can also be demonstrated by observing the example of the prediction results of two neighbouring VNFs (see figure 2.10) where the GNN curve on VNF4 shows a better prediction compared to CNN. This can be confirmed by the closest MAE computed values for the two neighboring VNFs in the GNN case which means that adding the

neighboring enables to enhance the prediction of resource consumption of a VNF, contrary to the CNN model in which the MAE error jumped from 0.005 to 0.6 as highest values.

Figure 2.12 shows a performance comparison between the four worklod profiles for the five models. We can notice that in stable workload fluctuation, all models reported good performances for the five metrics, such as the LSTM model that records MAE of 0.063 thanks to its long period of time dependencies tracking ability, with 3480 sec training time, whereas the simple MLP records a MAE equal to 0.078 which is slightly higher than the LSTM error, but it outperforms the others in the coefficient of determination, in the explained variance with a score equal to 0.94 and in terms of time complexity where it records 44 seconds for the training process and 36.7%, 50.6% for respectively the CPU and the memory consumption which is very reduced compared to the other models where the fastest model between the four others (GNN) records 153 seconds for the training and 91.7%, 76.1% for respectively the CPU and the memory consumption. Knowing that in the context of resource management, the training process may be triggered repeatedly, and the training cost is an important quality indicator in a resource prediction forecaster which makes the MLP model the most suitable in such cases. Thereby, the resource forecasting problem requires a prediction mechanism that covers all possible cases of the network flow. This can be done by studying the data and clustering them into similar sets which will be processed by the most appropriate model. The selection of the appropriate prediction model according to the resource consumption profiles will be presented in the next chapter.

2.5 Conclusion

We presented in this chapter an efficient resource consumption forecasting model. The model uses augmented VNF features and exploits the neighboring relationships between VNFs of an SFC, learns consumption dependencies between neighbouring VNFs, and forecasts multi types of resources consumed by several VNFs composing an SFC. The model was evaluated and compared with different machine learning models using five metrics and datasets gathered from real word deployment of SFCs. An extensive experiment on different resource consumption

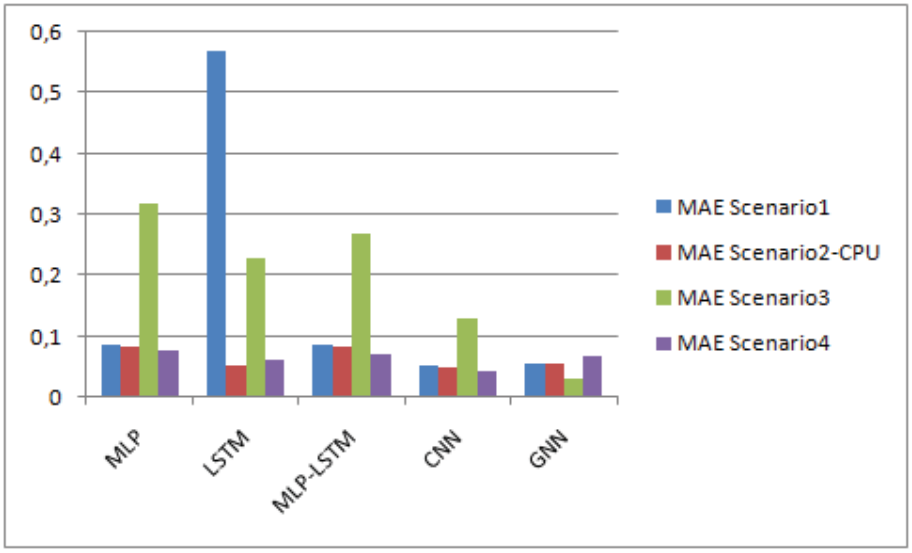

Figure 2.12 MAE error values of the five models depending on the different scenarios

profiles revealed the reliability and the efficiency of the proposed GNN model on high fluctuating workloads, which will allow a resource orchestrator to efficiently manage the future resource utilization under a high workload variation. Also, a deep investigation on the detailed results of five NN models revealed the advantages and the weaknesses of each model in terms of resource consumption forecasting. In the next chapter, we will present a resource prediction mechanism that enables to select the appropriate model based on the resource consumption profiles. The selection mechanism will use the experimentation analysis in order to assign for a given workload in the most appropriate prediction model that will maximize the prediction performances. This way, the selection process will allow handling with the dynamic predictor performance in the different workloads, and thus, accurately forecast the resource consumption in dynamic NFV environments while minimizing the time and space complexity of the prediction model.

CHAPTER 3

AN EFFICIENT ADAPTIVE META LEARNING MODEL BASED VNFS AFFINITY FOR RESOURCE PREDICTION OPTIMIZATION IN VIRTUALIZED NETWORKS

3.1 Introduction

Cloud environment Abohamama *et al.* (2022) is being highly considered for industry and enterprises for the opportunities that it provides, particularly in terms of resource optimization and cost reduction Liu *et al.* (2022). With recent developments in cloud computing Zhang *et al.* (2010) massive unplanned traffic loads are submitted to cloud platforms that lead to uncertainty in resource utilization. Therefore, efficient data-driven mechanisms for automatic resource management becomes crucial Amiri & Mohammad-Khanli (2017) Younge *et al.* (2010). Network Function Virtualization (NFV) concept enables a network soft-warization. However, one of the challenges facing the adoption of this concept is to dynamically manage Virtual Network Functions (VNFs) chained to compose a Service Function Chain (SFC) while meeting the service performance needs. One of the management mechanisms is dynamic allocation of resources for each VNF. While the resource management for SFCs was largely studied Li *et al.* (2023) Mostafavi *et al.* (2021) Thantharate & Beard (2023) dynamic resource management is still a challenging task in dynamic network environments. In such environments, the allocation of resources to each VNF cannot rely on threshold based scaling policies Salimian *et al.* (2016) since they require per-service thresholds setting. Therefore, we propose data driven prediction mechanism to support automatic resource management. These mechanisms enable complex and distributed systems to anticipate and to efficiently react to workload fluctuations. To achieve that, the amount of the future needed resources must be accurately predicted. This way, the needed resources will be already available when the flow reaches the point of need, and released when they are not needed.

Several machine learning techniques have been proposed for resource prediction in virtualized environments, such as LSTMNN Ouhame *et al.* (2021), autoencoder Zhang *et al.* (2018a), online learning algorithms Mijumbi *et al.* (2014a) to name few.

However, most of the existing methods focus mainly on the historical consumption that depends on the current workload scenario, which makes capturing several workload scenarios features by a single learner a challenging task.

In this chapter, we propose a model selector that learns the workload features based on meta learning strategy, and automatically selects the most suitable method to be used among a set of predictors depending on the present workload and based on VNF consumption dependencies captured by a self-attention mechanism. The model allows at once to maximize the accuracy prediction by selecting the most performing algorithm, and to maximize the efficiency using an attention mechanism based VNF dependencies analysis and a multitask learning strategy. The model was tested and compared to the base single predictors using five metrics, the results show the performances of our model and the high accuracy prediction.

3.1.1 Problem analysis

In this section, we present a brief data analysis of resource consumption of VNFs in a cloud infrastructure and discuss the prediction problem complexity. Further, we show the degree of heterogeneity, the uncertainty, and the diversity in resource consumption and their effect on the performances of some resource consumption prediction models. We use a dataset gathered from testbed deployed using the Clearwater IMS solution and Openstack.

Figure 3.1 shows the fluctuation of one hour CPU consumption by a single online service (a VNF) over intervals of 20 seconds in two different periods of the day noted H1 and H2. Each period contains 180 observations recorded in two different hours.

We can see in the figure 3.1.(a) that the CPU consumed fluctuate significantly, representing the uncertainty of the CPU demands that can augment by 0.37 in 20 seconds such as in observation number 176. It can reach 0.98 consumption, while the minimum value is 0.47 with an average consumption of 0.73 per hour. In some infrastructures, such fluctuations require many scaling procedures and may require the creation of several VMs, where each VM creation takes tens of seconds. The CPU consumption changes continuously as well as in Figure 3.1 (b). However, this

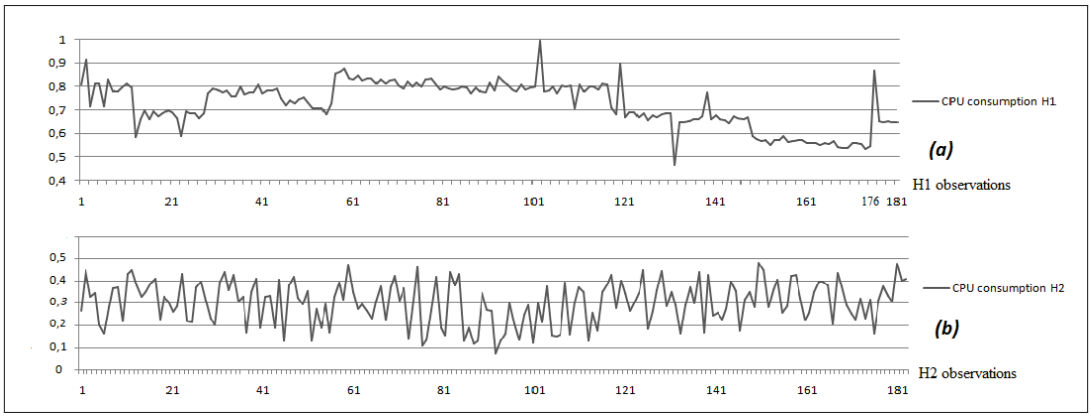

Figure 3.1 Fluctuation of one hour CPU consumption by a single online service (a VNF) over intervals of three minutes in two different period of time, H1 and H2

last shows 0.48 as a maximum value and 0.27 as an average value, which corresponds to only 36 % of the average consumption in H1 of the same VNF. This, demonstrates the non uniformity of the VNF demands. Also, the diversity of the resource type emphasizes the problem complexity, where each VNF uses multiple types of resources such as memory and bandwidth, and each resource type consumption change dynamically. Thus, learning the consumption behavior by a given model is a critical task since the consumption behavior exhibits randomness.

We present in Figure 3.2 the MAE errors of four different deep learning models for resource consumption prediction (section 3.2.3) computed on 33 minutes CPU consumption with a sampling rate of 20 seconds of the same VNF. We can see that the four average errors on the 30 minutes are quite similar and vary between [0, 0.6]. However, the performance of each model on a single point is very different, such that in observation 22, we recorded an MAE value equal to 0.061 for Model4, where Model3 achieved 0.0002 which is smaller compared to Model4. However, in observation 47, Model4 recorded the best MAE value, and performed significantly better than the next performing model (model 3) with 34 % improvement. This indicates that the performance of a single model depends on the resource consumption distribution. Indeed, each model can learn a specific behaviour depending on the workload features that it can capture. Each model provides more accurate prediction compared to others for a specific workload, but cannot be applicable to other workloads. Therefore, we propose in this chapter a model

selector that automatically selects the most suitable model to be used depending on the analyzed workload and based on VNF consumption dependencies analysis.

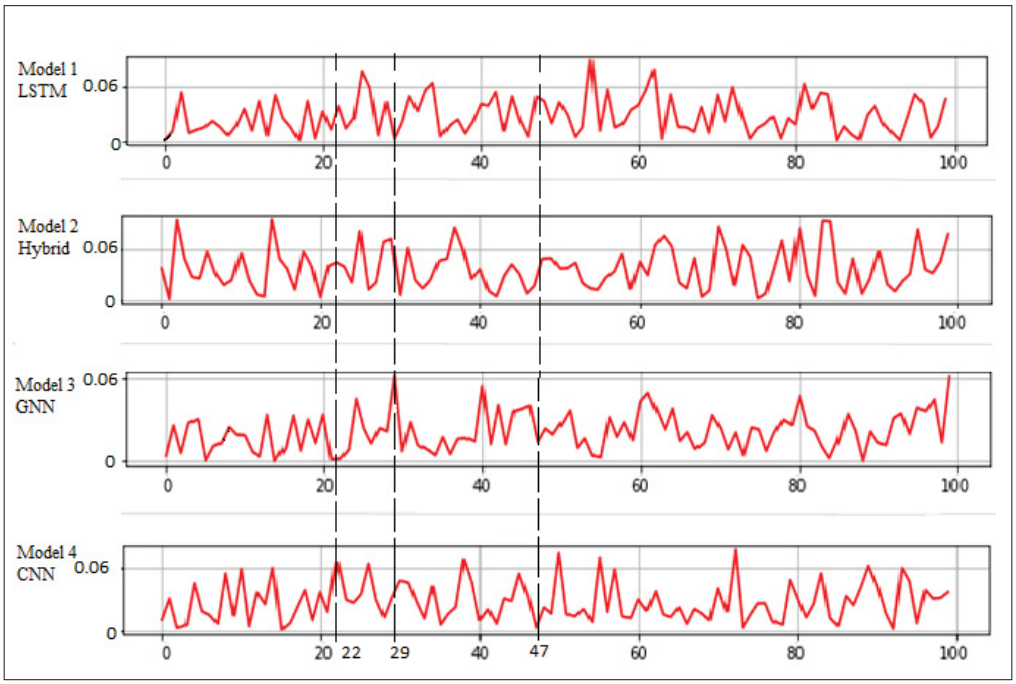

Figure 3.2 MAE errors of four different deep learning models predictors computed on 33 minutes CPU consumption with a sampling rate of 20 seconds of the same VNF

3.2 Proposed approach

This section provides a detailed explanation of the proposed resource consumption prediction system. The proposed system allows predicting accurately and efficiently the future resource consumption of a set of VNFs. In this application, we predict three resource types (memory, CPU, bandwidth) of five VNFs. We will present in the first part the overall working flow of the mechanism. Then, we will present the proposed MT-MLS that analysis the VNFs consumption correlation to enhance the prediction accuracy.

3.2.1 System overview

The proposed prediction system relies on VNF consumption correlation analysis to select, among a set of resource forecasting models, the best model to be used in the next predictions.

Our system contains four principal components (figure 3.3). In the first component, the monitored SFC historical consumption passes through a preprocessing step. In this step, invalid recorded data are first removed. That data corresponds to observations containing 'NAN' or negative value in one of the recorded resources, and the erroneous observation containing at least one outlier value detected using the modified z-scoreChoudhury *et al.* (2017). Then the data is normalized using min-max normalization in the range [-1,1]. Then the data is split into fixed size sliding windows w. Each window contains three types of consumed resources of each VNF of the whole SFC and each type contains h historical consumption of all VNFs of the SFC. The consumptions at $h + i$ are retained to represent the future consumption of the SFC, and are used as the base learner's output of the input window w. The preprocessing step results in a structured dataset, containing a set of historical windows of three resources, where the consumption of all resources are normalized between -1 and 1. Then the historical consumption are divided into 70% training set and 30% test set used to train and test the base learners.

The second component comprises a set of N resource forecasting models named base learners. The preprocessed historical consumed resources are fed to the base learner to be trained and evaluated. Each base learner is trained offline on a training set of historical consumption to forecast the anticipated resource consumption of the VNFs. Then, a test set of historical consumption is used to rate the base learner's forecasting on each test window w, generating a set of predictor evaluations on the test set.

In the third component of the system, the VNFs consumption and correlation are studied and analyzed to construct a metadata used to train the MT-MLS. The metadata is constructed mainly from the past base learners predictions. For each window w, the performances of base learners on each VNF are compared using the MAE measure Sihn & Park (2008), and the learner recording the lowest value is retained as the best predictor of the processed VNF for the window

w. Thus, we construct for each input sample w_i, a metadata output vector v_i to train the meta learner.

In the fourth component of the system, the online prediction process is executed by the MT-MLS. When a prediction R is required, the MT-MLS is the component responsible for choosing the predictor that maximizes the prediction accuracy to be used in a given future period depending on the analysis of the workflow preceding R and which maximizes the prediction accuracy. The VNF future consumption of the request R are then computed by the chosen predictor.

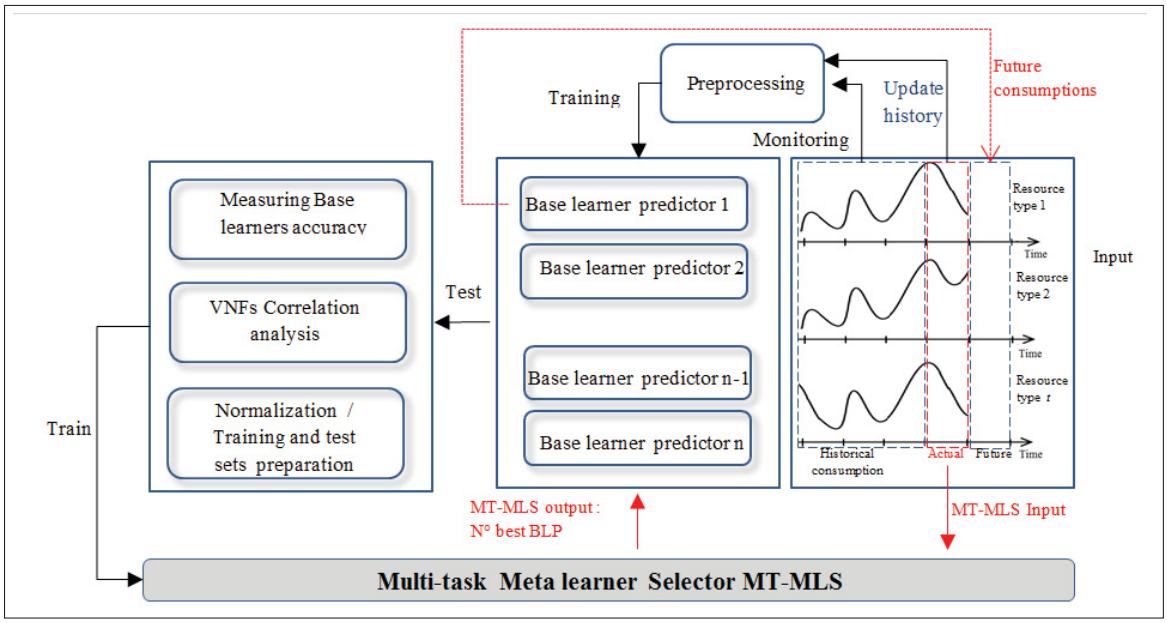

Figure 3.3 Overall scheme of the proposed architecture: the historical consumption is continuously monitored. After a pre-processing step, the processed data is fed to each base learner to be trained. Then the model measures the accuracy of each predictor and analyzes the correlation between the VNFs consumption and use them in the MT-MLS training. In the online process (red color) the request is presented directly to the MT-MLS that selects the best base learner to trigger in order to compute the future resource consumption

3.2.2 multitask Meta learner selector MT-MLS

The MT-MLS aims at selecting a set of best methods to be used in the future resource consumption computations. Based on the base learners evaluation and the correlation analysis, the MT-MLS is trained on the constructed metadata to learn the future ranking features of base predictors on

different workload scenarios. Thus, the meta learner selector is capable of evaluating the next prediction accuracy and selecting the most suitable learner for the future predictions of each VNF.

To achieve that, we used meta learning strategy Chen *et al.* (2021). This strategy consists of learning from information generated by learners. In other words, using the meta learning, we transform the learned information by the base learners into knowledge. In our work, the knowledge represents the relation between the different base learner's predictions in terms of accuracy in different workload scenarios. That knowledge is produced through the meta learner selector based on metadata.

In order to efficiently compute the future needed resources, we propose to simultaneously process multiple VNFs in the same computation. Thus, we formulated the selection problem as a multitask classification problem Aceto *et al.* (2021), in which, each VNF behaviour will be considered as an independent task to be learned, gathered together in the same end-to-end architecture. Figure 3.4 shows the architecture of the proposed multitask meta learner selector. The proposed architecture consists of an LSTM layer followed by a shared self attention layer Baffour *et al.* (2021) based similarity module and then a multi-branch classification (figure 3.4).

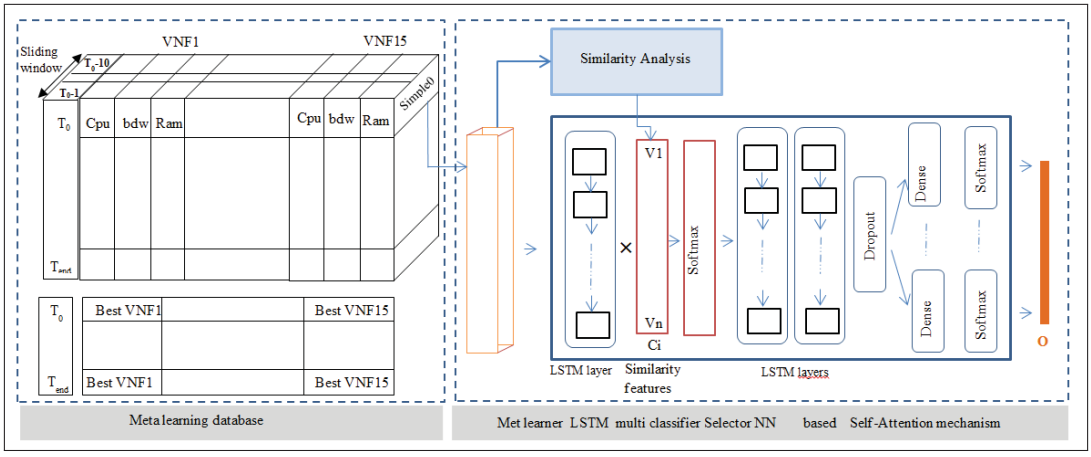

Figure 3.4 Intern architecture of the MT-MLS

3.2.2.1 Shared Self attention mechanism based VNFs similarity

In an SFC, the flow can pass from a VNF to another. In several cases, the flow passes from a VNF to its neighbor, which means that the increase in flow in a VNF may lead to the increase in its neighboring VNFs as well. This, generates a sort of dependency between the VNF in terms of resource consumption. This dependency can be expressed as a similarity of one or more types of resource consumption profiles between different VNFs. In this work, we intend to capture those similarities and use them in the learning process to allow the network focusing on the different consumption profiles. To achieve that, we propose an attention mechanism that assigns various weights to different VNFs consumption types. Therefore, we define a module that computes the similarity between two multi-type resource usage in the same sliding window w of two different VNFs. This is done by computing the correlation between two consumption curves using Kendall rank correlation coefficient given in equation 3.4.

In order to capture long-range dependencies in the multi-type historical consumption, the data of the sliding windows w are first fed to the LSTM layer, then to the attention mechanism that learns weight coefficients representing the importance of each VNF resource type. Let's R be the number of series resulting from the R LSTM cells, the attention consists of Computing c_i vector of characteristics which represents a sequence by a weighted sum of the hidden representation of length R where $R = resource - types \times Number - of - VNF$ as follows:

$$C_i = \sum_{j=1}^{R} \alpha_{ij} v_i \tag{3.1}$$

Where: α_{ij} represents the weight coefficient of each v_i, semantically it represents the normalized importance of each consumption in the forecasting process and is computed by equation 3.2.

$$\alpha_{ij} = \frac{exp\left(v_{ij}\right)}{\sum_{k=1}^{R} exp\left(v_{ik}\right)} \tag{3.2}$$

Where v_{ij} represents the dissimilarity score between two resource consumption, and is computed using 3.3.

$$v_{ij} = 1 - |\tau\left(v_i, v_j\right)| \tag{3.3}$$

Where $\tau(vi, vj)$ is the Kendall rank correlation between the historical records of the two VNFs resource type utilization for a given sliding window w.

Let $Rsc^j = \{Rsc^j_0, Rsc^j_1, ...Rsc^j_n\}$ be the consumption evolution of the resource j in time $t = 0, ..., n$ in the window w of size n. And $Rsc^i = \{Rsc^i_0, Rsc^i_1, ...Rsc^i_n\}$ be the consumption evolution of the resource i in the same window w. Then, the Kendall rank correlation between Rsc^i and Rsc^j is given by equation 3.4.

$$\tau = \frac{2}{n\,(n-1)} \sum sgn\left(Rsc^j_t - Rsc^j_{t-1}\right) sgn\left(Rsc^i_t - Rsc^i_{t-1}\right) \tag{3.4}$$

3.2.2.2 Multi-branch classification and model training

In this section, we explain how we trained the classifier for each VNF in one end-to-end learning architecture and in one single network. Our aim is to classify each sliding window into the best predictor class corresponding. We proposed a multitask model that is jointly trained by sharing the attention and the LSTM layers for each branch of the multitask network. Each VNF classifier has a separate branch but shares the same features extracted by the LSTM layers. Those features are extracted based on the VNF consumption behaviour, and also on their consumption dependencies on the other VNFs through the attention mechanism (section 3.2.2.1). This allows each branch of the multitask to be trained on a specific classification based on the own VNF consumption and dependencies features. In each classification task, the R-dimensional vector is mapped using a fully connected layer to y $\in n$, where n is the number of base learner predictors. Finally, we end up with a multi classifier network, in which the feature vectors extracted by the

LSTM network are fed to individual branches that predict the best algorithm for each VNF to be used for the future requests. The model optimization is achieved by minimizing the total sum of the VNFs classification loss.

$$L = \sum_{i=1}^{V} \alpha_i L_{vnfi} \tag{3.5}$$

Where: V is the number of VNFs. L_{vnfi} is the classification loss of the VNF number i. α_i is the weighting coefficient of each VNF in the SFC.

Each VNFs best algorithm classification was performed by minimizing the cross-entropy loss function given in equation 3.6.

$$L_{vnf} = -\frac{1}{w} \sum_{i=1}^{w} \sum_{n=1}^{N} 1_{y_i \in N_n} log\left(p_{model}\left[y_i \in N_n\right]\right) \tag{3.6}$$

Where L_{vnf} is the classification loss of each VNF, w is the window size, N is the number of classes corresponding to the number of base learners. $log p_{model}[y_i]$ is the likelihood log of the input y_i belonging to class n if it is mapped using model p.

3.2.3 Base learner techniques

In this section, we present a brief description of the four used base learners predictors based deep learning techniques. Note that the proposed MT-MLS architecture is not specific and is not limited to these models.

The first model is a GNN (graphical neural network) Scarselli *et al.* (2009) based on topological dependencies. Thanks to the graphical representation of the SFC, the model uses the neighboring relationships between VNFs to predict the future consumptions. The GNN model consists of three main components: a modelling module, in which the SFC is represented by a graph, where the vertices represent the VNFs and the edges represent the neighboring relationships between

different VNFs in the SFC. A transition function is used to compute the states of each node in the graph, and the output function computes the output of each node corresponding to the future utilization of the VNF. Both transition and output functions were modeled by two three-layered Feedforward NN Sanger (1989) trained on the historical consumptions of the SFC.

The second model is a CNN model Albawi *et al.* (2017) O'Shea & Nash (2015). It consists of five convolutional layers followed by a dropout layer and a max-pooling layer, then three other convolutional layers followed by three fully connected layers. Each convolutional layer is constituted of parallel stack channels, where each channel consists of a layer of neurons sharing the same weights, filter size, and biases. The neurons constituting the layers pass the sum of their weighted inputs through a ReLu activation function.

The third model is a recurrent NN comprised of multiple LSTM layers. More specifically, the model is composed of three LSTM layers, the two first are followed by a dropout layer, and the third by a fully connected layer. The prediction is then done using a Tanh activation function that predicts the future resource consumption of a set of VNFs.

The fourth model is a hybrid predictor that combines LSTM and MLP networks. The model principle consists of passing the original data directly after a preprocessing step to a three layered MLP network Gardner & Dorling (1998). This plays the role of a preliminary pretraining that produces discriminant features from the historical resource consumption of the VNFs. The resulting features are then fed to an LSTMNN Huang *et al.* (2015) that will learn and predict the future resource consumption of the VNFs.

3.3 Results

In this study, we intend to demonstrate the effect of a selection mechanism on the resource usage prediction. Thus, in the first part of our experiments, we evaluated the performances of each base learner alone as a prediction method. Then, in the second part, we evaluated the performance of the resource prediction using the adaptive selection mechanism as well as the cost entailed by the selection process.

3.3.1 Data set description and evaluation metrics

The main goal of resource prediction is to forecast the future needed resources of a network entity, this involves the use of past historical consumption of the network components. While most of the literature studies use simulated data Sciancalepore *et al.* (2017) Mijumbi *et al.* (2014b) Mijumbi *et al.* (2017), we propose in this work to experiment using two datasets gathered from tested deployed in real-world settings sets, in which the deployment consists of multiple nodes, with each deploying a specific service and consuming variable rates of resources. In this work, the CPU, the memory, and the bandwidth rates consumed by each node were monitored and recorded every 20 seconds during 16 days. Each node represents a virtualized network function and is processed as a single SFC node. In the first data set, the testbed was deployed using the Clearwater IMS solution and Openstack cloud infrastructure (figure 3.5). The SFC is composed mainly of five principal VNFs. The Bono node provides the anchor point for the client's connection to the Clearwater system. The Sprout node handles client authentication and the ISC interface to application servers. Cassandra and Astaire nodes are used to store the mastered and the cached data respectively of subscribers. Ellis is a sample provisioning portal that provides basic services such as password management. Homer is used to store the user's service settings documents. Homestead provides a web services interface. Homestead Prov also exposes a web services interface, allowing provisioning of subscriber data in Cassandra. Ralf provides an HTTP API for Bono and Sprout (table 3.1).

In the second dataset that we called the web data [1], the SFC comprises five nodes: the database VNF that stores words, three instances of Java REST API that read and serves words from the VNF database, and the Go web application which calls the API and builds words into sentences. When a request arrives, the traffic passes to the web VNF and then passes to one of the three Java API VNFs then the requested words are extracted from the database VNFs and printed on the website to construct a sentence. For the rest of the experiments, we will note the VNFs by their numbers from VNF1 to VNF15 (table 3.1).

[1] https://github.com/dockersamples/k8s-wordsmith-demo/blob/master/README.md

Table 3.1 IMS and WEB VNFs description

VNF	Description	
VNF1	Bono	Provides the anchor point for the client's connection to the Clearwater system, including support for various NAT traversal mechanisms.
VNF2	Sprout	Acts as a horizontally scalable, combined SIP registrar and authoritative routing proxy, and handle client authentication and the ISC interface to application servers. The Sprout nodes also contain the in-built MMTEL application server. Sprout is where the bulk of the I-CSCF and S-CSCF function resides.
VNF3	Cassandra	Stores the mastered data of subscribers
VNF4	Astaire	Stores the cached data of subscribers
VNF5	Ellis	A sample provisioning portal providing self sign-up, password management, line management and control of MMTEL service settings.
VNF6	Homer	A standard XDMS used to store MMTEL service settings documents for each user of the system.
VNF7	Homestead	Provides a web services interface to Sprout for retrieving authentication credentials and user profile information.
VNF8	Homestead-prov	Exposes a web services provisioning interface to allow provisioning of subscriber data in Cassandra
VNF9	Ralf	Provides an HTTP API that both Bono and Sprout can use to report billable events that should be passed to the CDF (Charging Data Function) over the Rf billing interface.
VNF10	Chronos	A distributed, redundant, reliable timer service developed by Clearwater. It is used by Sprout and Ralf nodes to enable timers to be run (e.g. for SIP Registration expiry) without pinning operations to a specific node (one node can set the timer and another act on it when it pops).
VNF11	Database	A Postgres database which stores words.
VNF12, VNF13, VNF14	Words	Java REST API which serves words read from the database.
VNF15	Web	A Go web application which calls the API and builds words into sentences.

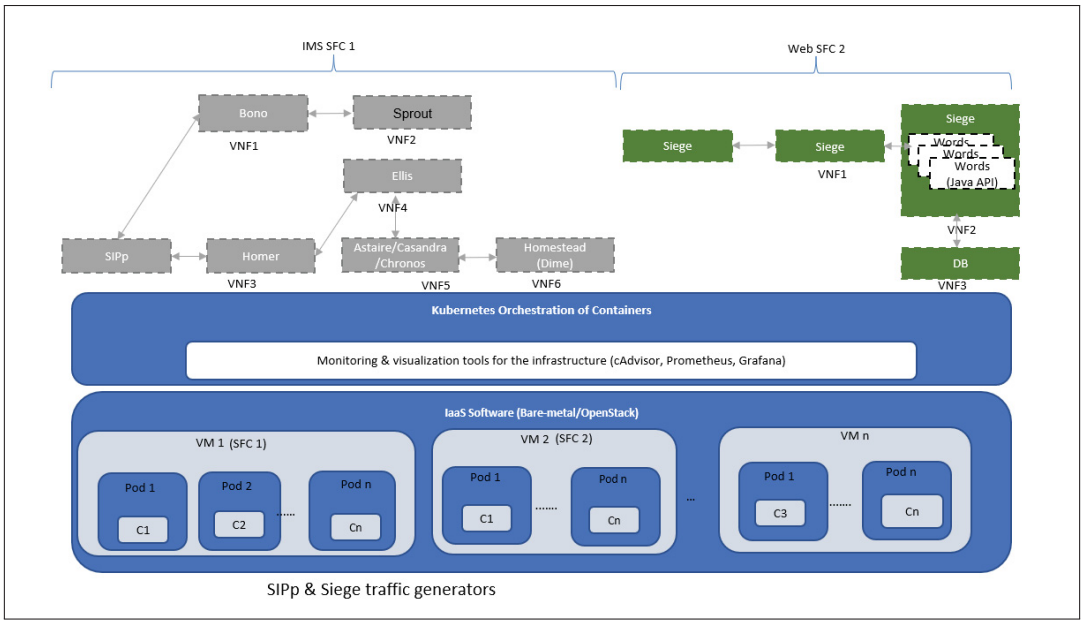

Figure 3.5 SFC deployment in cloud setting

In this experiment, we used five measurements of model prediction performance evaluation, including the Mean absolute error (MAE) known to be used in regression problems, it measures the errors between the predicted and the real observations. The Root mean square error (RMSE) Chai & Draxler (2014) and the mean squared error (MSE) that measure how close the estimated values to the real values. We have also computed the explained variance score which measures the dispersion of a given paired observation O'Grady (1982) and the coefficient of determination that measures the amount of variation between the real and the predicted values Ozer (1985), i.e. quantify the degree of relationship between the predicted and the real variables. The errors computations of the mean absolute error (MAE), the mean squared error (MSE) and the root-mean-square error (RMSE), the explained variance regression score function, and the coefficient of determination are respectively given by equation 3.7 - 3.10.

$$MAE = (\frac{1}{n}) \sum_{i=1}^{n} |(y_i - x_i)| \tag{3.7}$$

$$MSE = (\frac{1}{n}) \sum_{i=1}^{n} (y_i - x_i)^2 \tag{3.8}$$

$$RMSE = \sqrt{(\frac{1}{n}) \sum_{i=1}^{n} (y_i - x_i)^2} \tag{3.9}$$

$$R^2 = \frac{\sum_i (\hat{y}_i - \overline{y})^2}{\sum_i (y_i - \overline{y})^2} \tag{3.10}$$

To accurately verify the Meta learner efficiency, we used three measurements for classification problems: the recall, the precision, and the F-measure given respectively by equation from 3.11 to 3.13.

$$Precision = \frac{TP}{TP + FP} \tag{3.11}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{3.12}$$

$$F1 = \frac{2 \times Precision \times Sensitivity}{Precision + Sensitivity} \tag{3.13}$$

Where : TP and FN are respectively the true positives and false negatives.

3.3.2 Base learners setting and evaluation

In order to have a coherent comparison, the structure of each prediction model must be selected, the hyperparameters of the models should be optimized and the configuration of the base learners should also be tuned. In order to evaluate the generalization of the models with the WEB data, we trained the five networks on the same subset of only the IMS data set to tune the parameters

of each model. Each time, the trained system was tested to determine its prediction accuracy, in each case the system was run to determine an output, and then compared to the actual resource consumption using the mean absolute error. The hyperparameters were tuned by varying each one of the parameters on 10% of the original IMS data set. The best reported scores were adopted in the training process. We present in Table 3.2 the common hyperparameters adopted for each base learner.

Table 3.2 Common Hyperparameters setting

Parameter	**LSTM**	**GNN**	**MLP-LSTM**
Learning rate	0.05	0.01	0.02
optimizer	Adam	Adam	Adam
Loos	MAE	MAE	MAE
epochs	200	100	140

We also tuned the specific hyperparameters of each model in order to maximize the performances of the base learners' predictor. The GNN configuration consists on tuning mainly the transition and the output functions parameters represented by two feed forward NN. The resulting NNs architectures contain three layers in each network, with one hidden layer containing a fixed number of neurons to the average of the neurons in the output and the input layers.

Also, to establish the CNN architecture, we conducted experiments focusing on the following hyperparameters: number of convolutional layer with and without dropout layers, and the max-pooling vs average pooling layer. We performed several tests with different numbers of convolutional layers. In each time, we tested the architecture using a dropout layer and without a drop out layer. Then, we repeated the same experiments using a max-pooling layer and then an average pooling layer. Table 3.3 shows the resulting CNN architecture.

For the LSTM NN, we focused on the main specific hyperparameters, which are the number of LSTM layers and the number of LSTM unit in each layer. The number of unit were fixed equal in all LSTM layers and is equal to the features in the sliding windows. For the number of LSTM layers, we tested different possible configurations using two, three and four LSTM layers

Table 3.3 CNN architecture and parameters

Layer	Type	Parameters	Activation
layer 1,2,3,4	convolution	Filters size : 32, 128,164, 188, kernel 3	Tanh
Layer 5	Dropout	0.5	Tanh
Layer 6	MaxPooling1D	pool size = 2	Tanh
Layer 7,8	Convolution	filters=128, kernel_size=3	Tanh
Layer 9	Dense	–	Tanh

followed by a dropout layer and selected the optimal architecture based on the reported MAE and the training time. The selected architecture is presented in table 3.4.

Table 3.4 LSTM architecture and parameters

Layer	Type	Parameters	Activtion
1	LSTM layer	675 LSTM units	Tanh
2	Dropout	0.5	Tanh
3	LSTM layer	337 units	Tanh
4	Dense	–	–

For the hybrid model, we followed the same process done for the LSTM network, then we fixed the number of layers of the MLP NN to 3 and number of input neurons depending on the input sliding window features. In the same manner as in the LSTM network, we varied the layers structure and selected those reporting the best MAE scores. The resulted architecture is summarized in table 3.5.

Table 3.5 Hybrid LSTM-MLP hyper-parameters

Hyper-parameter	number of layers
Number of LSTM layer	3
Number of dropout layers	1
Number of LSTM units	15
Number of fully connected layers	1
MLP layer number	3

After establishing networks structure, we trained all base models on the training set, and reported the MAE average of the five metrics on the test set for each VNF in table 3.6.

We can clearly see that each model performs differently on each processed VNF. For instance, the MLP-LSTM model recorded the worst MAE value for VNF3 while it achieves the best error for VNF7. Similarly, the CNN model recorded the lowest MAE value for VNF3, VNF5 and VNF13. Meanwhile, it performs less in the other VNFs such as for VNF7. Extensive experiments were done in an unpublished previous work, intending to identify the scenarios in which each model performs the best. We dissected the problem into different workload cases and tested the models on each case. We conclude that defining a model for a specific workflow and a workload scenario may be a challenging task since the workload is continuously dynamic, and the best algorithm can achieve good results for specific consumption data, but not for other subsequent data. To clarify that, we present in figure 3.6 the recorded MAE value of the same VNF (VNF9) on 100 consecutive consumption sequences separated by 200 seconds interval time. The figure shows that even for the same VNF and in a close time, the performance of a given model may decrease or increase considerably, which demonstrates the complexity of identifying an appropriate algorithm for a given workload and workflow. Therefore, there is a need to define an ML architecture able to learn complex functions and allow predicting the most suitable prediction algorithm.

3.3.3 Adaptive Selection evaluation

In the first part of these experiments, we assess the meta learner classifier accuracy through the recall, the precision, and the F-measure (section 3.3.1), our aim is to evaluate the ability of the classifier to attribute the appropriate prediction model to a given workload. Then, in the second part, we investigate the performances of the resource consumption prediction using the adaptive selector and compare it to the performances of the base learners and based on the MAE, MSE, RMSE, explained variance and the coefficient of determination. Also, to explain the impact of the proposed attention mechanism, we compared the model performances with and without attention. The model was trained in a first time ignoring the attention layer and tested on the test

Table 3.6 Reported errors by each model for different scenarios using IMS and Web datasets

VNF	LSTM MAE	MLP-LSTM MAE	CNN MAE	GNN MAE
VNF1	0.151	0.092	0.088	0.120
VNF2	0.212	0.128	0.161	0.146
VNF3	0.086	0.129	0.031	0.041
VNF4	0.142	0.125	0.093	0.083
VNF5	0.174	0.092	0.047	0.056
VNF6	0.167	0.074	0.125	0.027
VNF7	0.085	0.072	0.110	0.162
VNF8	0.113	0.185	0.065	0.077
VNF9	0.052	0.079	0.044	0.074
VNF10	0.168	0.087	0.025	0.053
VNF11	0.065	0.061	0.0637	0.087
VNF12	0.192	0.092	0.014	0.02
VNF13	0.165	0.141	0.043	0.0723
VNF14	0.067	0.153	0.071	0.0631
VNF15	0.054	0.134	0.097	0.083

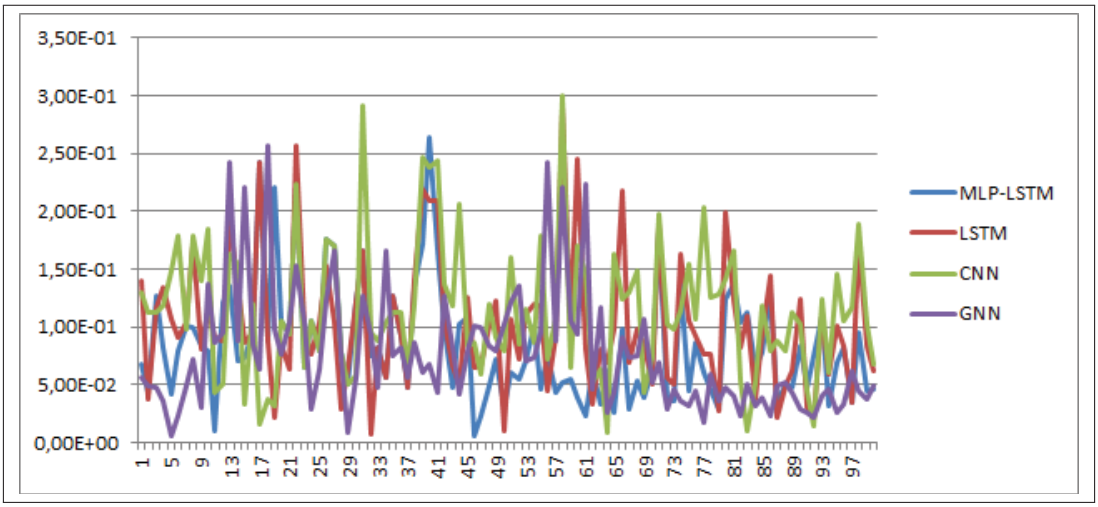

Figure 3.6 Comparison of MAE recorded by the four models on consecutive sliding windows

set, and then we trained the model a second time with the attention layer based VNF similarity and tested on the same test set as in the first experiment. The performance was assessed in both experiments, as well as the total cost entailed by the test process on the whole test set, and presented in Table 3.7. We can see throughout the table that the classification accuracy of the meta learner is pretty good and has been enhanced by 4% using the attention mechanism with an F-measure value of 0.81 instead of 0.77, this demonstrates the effectiveness of the attention layer.

Table 3.7 Meta learner selection evaluation: average scores on the five classes and total resource consumption on the whole test set by the LSTM-selector with and without attention mechanism

Measure	**Meta learner**	**Meta learner based attention**
Recall	0.77	0.81
Precision	0.78	0.82
F-measure	0.77	0.81

Table 3.8 Meta learner selection evaluation: reported classification metrics on each class resulted by the LSTM-selector with attention mechanism

Measure	**CNN class**	**LSTM class**	**GNN class**	**Hybrid class**
Sensitvity	0.85	0.79	0.84	0.80
Precision	0.81	0.85	0.79	0.82
Specificity	0.93	0.94	0.93	0.94
F-measure	0.83	0.81	0.81	0.81

To further investigate the classification accuracy, we present in figure 3.7 the confusion matrix of the multitask classification based attention mechanism on each class. The confusion matrix shows different classification errors.

Resource usage prediction evaluation

In this section, we intend to study the impact of the Meta learner algorithm selection on the resource consumption forecasting. Thus, we will be comparing the performances of the base

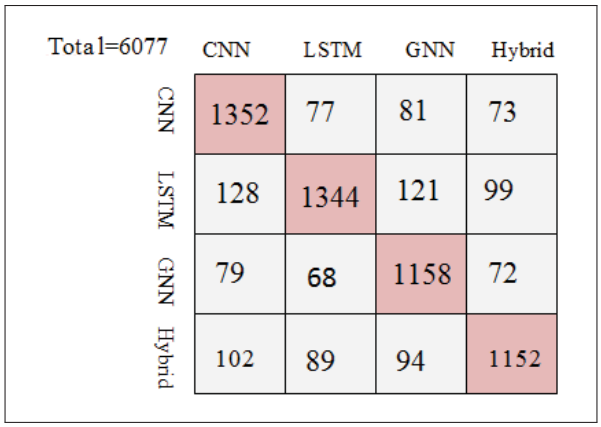

Figure 3.7 Confusion matrix on the four classes resulted from the selector based self attention

learners' resource forecasting with the adaptive selector resource forecasting. To achieve that, we computed for each base learner the average errors recorded on the test set of the selector. Then, we computed the average errors recorded by the chosen algorithm using the adaptive selector on the same test set. The obtained results are presented in table 3.9, which demonstrates that the results obtained by the dynamic selection are better than using a given algorithm at once, such as for MAE error, in which the CNN achieved 0.081 as the best base predictor for this test set while the dynamic forecasting achieved a lower MAE value equal to 0.021.

Table 3.9 Comparison between the five resource consumption forecasting models: Average errors reported on resource consumption prediction using the fourth models on the selector test set, and average errors computed on prediction of each observation by the selected algorithm (heterogeneous prediction algorithms)

Measure	**LSTM**	**MLP-LSTM**	**CNN**	**GNN**	**Algorithm selection**
MAE	0.114	0.084	0.072	0.081	0.021
MSE	0.096	0.092	0.087	0.082	0.032
RMSE	0.309	0.303	0.294	0.286	0.178
Explained variance	0,718	0,722	0,769	0,751	0.932
Coefficient of determination	0,727	0,718	0,774	0,756	0.926

3.3.3.1 Discussion

The proposed adaptive selector combines an LSTM NN to a time series analysis strategy that allows including the similarity dependencies between VNFs in the learning process. In order to get an accurate estimation of the attention layer efficiency, we presented in the first part of the experiments, the evaluation results of the adaptive classifier using the attention layer based VNF similarity and also without using it. The results show that the attention layer gives better classification results with an $F - measure = 0.81$ equivalent to 4% enhancement.

Thanks to the VNF similarity analysis used in the attention mechanism. The similarity information guides the network to focus more on VNFs having dissimilar consumption profile by given low importance (low weight) to VNF having an important correlation coefficient. Applying equation 3.2 in the attention mechanism is equivalent to assigning high weight to exceptional VNFs and smaller weight to similar VNFs. This fact explains the performance's enhancement.

In the second experiment, we tested and evaluated the prediction accuracy and efficiency of different methods, we observed that the performances of the base learners depend highly on the flow, which causes fluctuations in the MAE errors returned by each method (figure 3.6). Therefore, we assessed the prediction using the selected methods by the adaptive model. At a given time t, the adaptive model selects one base learner to be launched depending on the workload profile. Thus, the error is equal to the error of a base learner in a single observation. However, evaluating the resource prediction of the adaptive selector on a set of observations involves evaluating a mix of base learners. Therefore, the evaluation results of a set of observations is equal to the average errors of the set of selected base learners for each observation. This results in an average error less than the average error of the same base learner since the selector selects each time the model with the smallest error value.

The result demonstrated the utility and the efficiency of a selection algorithm that enhances considerably the prediction process in terms of accuracy where it achieves an average MAE=0.021 which corresponds to 30% improvement compared to the best base learner of this test set which

may also vary from a set to another. This enhancement is explained by the efficient selection throughout which the resource predictor is triggered each time needed, which minimizes the total error.

3.4 Conclusion

We proposed in this chapter, a resource prediction optimizer model MT-MLS. Depending on the current workload, the MT-MLS selects the most suitable prediction model among a set of already trained models. The selected model is used to predict the next resource consumption. The MTL-MLS is designed as a multitask classifier based on a meta learning strategy. The model is based on a VNF correlation analysis that allows capturing resource consumption dependencies between the multi-type resource of different VNFs. These dependencies are fed into the multitask meta learner via an attention mechanism, which allows the MT-MLS to learn the best predictor to be used in future times. Thus, the MT-MLS assigns simultaneously and independently to each VNF of the SFC the label of the most accurate model to be used in the future forecasting. This allows decreasing considerably the prediction error and hence increasing the prediction efficiency. The model was tested using two data sets gathered from two real world deployed SFCs. The performance analysis reveals that the MT-MLS enhances considerably the efficiency of the resource usage prediction.

CONCLUSION

In NFV environment, shareable resources are made available to users in a flexible manner (automatic scaling). However, the resources and the traffic load of each VNF vary continuously. Such variation requires proactive resource management to guarantee the QoS for the hosted VNFs despite the dynamic nature of the cloud environment. To guarantee this QoS, cloud providers generally opt for an over-provisioning solution, which is a viable solution in terms of operational costs. Proactive management of these resources would be therefore crucial to ensure the requested QoS, optimize the resources utilization, and minimize the costs. In this perspective, the main objective of this thesis was to define and validate an adaptive approach to predict resource consumption while meeting the SLA requirements of VNFs and optimizing the use of these resources. Based on a selection mechanism, our approach proved to be sufficiently generic to be applied to various data sets. It mainly includes multiple DL prediction models and a selection component to select the best method depending on the workload.

In the first contribution of this work, we developed four DL-based models, our goal was to analyze the behavior of DL models with the multi-type resource forecasting problem so that we would be able to define the most suitable DL technique for the problem. Thus, the resulting DL models were the results of a long process of experiment of trial and model enhancement based on the errors, the extensive experiments study performed on different workload scenarios revealed the uncertainty of the efficiency of each model that changes depending on the workload.

In the second contribution, we proposed an efficient GNN Model that uses SFC topological features to capture VNFs consumption dependencies. The GNN model demonstrates its performance in high-load traffic. The proposed GNN model has been compared with MLP, LSTM, hybrid LSTM, and CNN models to evaluate its accuracy and efficiency. Real-world datasets have been used to evaluate the proposed model using five performance metrics. The

performance analysis reveals that our graph-features-based GNN model outperforms the other models for SFCs with high traffic load variation.

In the third contribution of this work, we developed a VNF similarity module analysis based on VNFs consumption correlation to capture affinities between the multi-type resource consumption of the VNFs. This module has been used in the MT-MLS.

In the fourth contribution of this work, we proposed an automatic selection approach that enables us to find the most suitable DL model based on VNFs similarity analysis.is and errors of the base prediction models. The MT-MLS introduces a novel concept by analyzing similarities of multidimensional resource consumption between virtual network functions (VNFs) of a service function chain (SFC). Various SFC resource consumption datasets have been used to evaluate the proposed MT-MLS using three performance metrics for the evaluation of classification and five others for the evaluation of prediction. The performance analysis reveals that the MT-MLS enhance considerably the accuracy of prediction and thus the efficiency of the prediction mechanism. However, due to a lack of data, the generalisation of our model were not sufficiently assessed. In fact, we couldn't asses the performances of our model under various SFC structure and various workload. Indeed, we have chosen to work with data that is limited in quantity and diversity but collected from SFCs deployed in known environments (topologies and configuarations of the SFCs, the configurations of the cloud testing environement).

Our main contributions are summarized in the following:

- An efficient GNN Model that uses SFC topological features and demonstrates its performance in high load traffic.
- A deep investigation of five DL models for resource prediction reveals the weaknesses of DL-based solutions.
- A VNFs consumption similarity analysis module that guides a resource usage predictor to efficient forecasting.

- An efficient resource prediction mechanism adaptive to resource demand in virtualized systems. The proposed mechanism may be applied with many types and amount of resource prediction methods.

In our future work, we will enhance the VNF similarity analysis module by considering other VNF dependencies such as topological dependencies. We also intend to conduct real-world deployment studies to validate the effectiveness and practical utility of MT-MLS in operational NFV environments.

Our work has been published in two international journals In the first paper, we proposed a deep learning model to predict the resource consumption (e.g., CPU, memory) in network function virtualization infrastructures (NFVI).

- Bellili, Asma et Kara, Nadjia. 2023. «An efficient adaptive meta learning model based VNFs similarity for resource prediction optimization in virtualized networks». Journal of Network and Systems Management, vol. 31, nº 2, March 2023.

In the second paper, we proposed a multitask selector based on meta-learning strategy MT-MLS.

- Bellili, Asma et Kara, Nadjia. 2023. «A graphical deep learning technique-based VNF dependencies for multi-resource requirements prediction in virtualized environments». Journal of computing, October 2023.

APPENDIX I

EVALUATION METRICS

1. Confusion Matrix

We computed confusion matrix to asses the performances of our classifier and to help us understand how well our model is performing, by showing the number of correct and incorrect predictions across different classes. A confusion matrix is an N x N table used to evaluate the performance of a classifier, where N represents the number of target classes. It juxtaposes the actual target values with the predictions made by the model, providing a comprehensive overview of the model's performance and the types of errors it encounters. To compute the confusion matrix, we must define the following terms:

- True Positives (TP): The count of instances that were correctly predicted as belonging to a specific class.
- True Negatives (TN): The count of instances that were correctly predicted as not belonging to a particular class.
- False Positives (FP): The count of instances that were incorrectly predicted as belonging to a specific class when they do not.
- False Negatives (FN): The count of instances that were incorrectly predicted as not belonging to a class when they actually do.

Thus, each element of the matrix correspond to the number of element predicted as the row class where was expected to be predicted as the column class. Therefore, the diagonal entries indicate the number of correct predictions for each class. The Off-Diagonal entries represent classification errors and help in identifying which classes are frequently confused.

BIBLIOGRAPHY

A Vouk, M. (2008). Cloud computing–issues, research and implementations. *Journal of computing and information technology*, 16(4), 235–246.

Abohamama, A., El-Ghamry, A. & Hamouda, E. (2022). Real-Time Task Scheduling Algorithm for IoT-Based Applications in the Cloud–Fog Environment. *Journal of Network and Systems Management*, 30(4), 1–35.

Aceto, G., Ciuonzo, D., Montieri, A. & Pescapé, A. (2021). DISTILLER: Encrypted traffic classification via multimodal multitask deep learning. *Journal of Network and Computer Applications*, 183, 102985.

Al Wadia, M. & Ismail, M. T. (2011). Selecting wavelet transforms model in forecasting financial time series data based on ARIMA model. *Applied Mathematical Sciences*, 5(7), 315–326.

Albawi, S., Mohammed, T. A. & Al-Zawi, S. (2017). Understanding of a convolutional neural network. *2017 International Conference on Engineering and Technology (ICET)*, pp. 1–6.

Alidoost Alanagh, Y., Firouzi, M., Rasouli Kenari, A. & Shamsi, M. (2023). Introducing an adaptive model for auto-scaling cloud computing based on workload classification. *Concurrency and Computation: Practice and Experience*, 35(22), e7720.

Amiri, M. & Mohammad-Khanli, L. (2017). Survey on prediction models of applications for resources provisioning in cloud. *Journal of Network and Computer Applications*, 82, 93–113.

Andrikopoulos, V., Binz, T., Leymann, F. & Strauch, S. (2013). How to adapt applications for the cloud environment. *Computing*, 95(6), 493–535.

Antwi, E., Gyamfi, E. N., Kyei, K., Gill, R. & Adam, A. M. (2021). Determinants of commodity futures prices: decomposition approach. *Mathematical Problems in Engineering*, 2021, 1–24.

Anuradha, V. & Sumathi, D. (2014). A survey on resource allocation strategies in cloud computing. *International Conference on Information Communication and Embedded Systems (ICICES2014)*, pp. 1–7.

Auer, P. & Warmuth, M. K. (1998). Tracking the best disjunction. *Machine Learning*, 32(2), 127–150.

Baffour, A. A., Qin, Z., Wang, Y., Qin, Z. & Choo, K.-K. R. (2021). Spatial self-attention network with self-attention distillation for fine-grained image recognition. *Journal of Visual Communication and Image Representation*, 103368.

Bansal, S. & Kumar, M. (2023). Deep Learning-based Workload Prediction in Cloud Computing to Enhance the Performance. *2023 Third International Conference on Secure Cyber Computing and Communication (ICSCCC)*, pp. 635–640.

Barakabitze, A. A., Ahmad, A., Mijumbi, R. & Hines, A. (2020). 5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges. *Computer Networks*, 167, 106984.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. & Vance, P. H. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations research*, 46(3), 316–329.

Bellili, A. & Kara, N. (2023). An efficient adaptive meta learning model based VNFs affinity for resource prediction optimization in virtualized networks. *Journal of Network and Systems Management*, 31(2), 40.

Bellili, A. & Kara, N. (2024). A graphical deep learning technique-based VNF dependencies for multi resource requirements prediction in virtualized environments. *Computing*, 106(2), 449–473.

Bi, J., Li, S., Yuan, H. & Zhou, M. (2021). Integrated deep learning method for workload and resource prediction in cloud systems. *Neurocomputing*, 424, 35–48.

Bi, J., Yuan, H., Li, S., Zhang, K., Zhang, J. & Zhou, M. (2024). ARIMA-Based and Multi-application Workload Prediction With Wavelet Decomposition and Savitzkyâ€“Golay Filter in Clouds. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 54(4), 2495-2506. doi: 10.1109/TSMC.2023.3343925.

Biemann, C. (2016). Vectors or Graphs? On Differences of Representations for Distributional Semantic Models. *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex - V)*, pp. 1–7. Retrieved from: https://aclanthology.org/W16-5301.

Blenk, A., Kalmbach, P., Van Der Smagt, P. & Kellerer, W. (2016). Boost online virtual network embedding: Using neural networks for admission control. *2016 12th International Conference on Network and Service Management (CNSM)*, pp. 10–18.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010* (pp. 177–186). Springer.

Boyacioglu, M. A., Kara, Y. & Baykan, Ö. K. (2009). Predicting bank financial failures using neural networks, support vector machines and multivariate statistical methods: A comparative analysis in the sample of savings deposit insurance fund (SDIF) transferred banks in Turkey. *Expert Systems with Applications*, 36(2), 3355–3366.

Bradley, S. P. & Arnoldo, C. (1977). Hax, and Thomas L. Magnanti. Applied Mathematical Programming. *Addison-Wesley*, 445, 08855–1331.

Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A. & Buyya, R. (2011). CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience*, 41(1), 23–50.

Calheiros, R. N., Masoumi, E., Ranjan, R. & Buyya, R. (2014). Workload prediction using ARIMA model and its impact on cloud applicationsâ€™ QoS. *IEEE transactions on cloud computing*, 3(4), 449–458.

Cao, L.-J. & Tay, F. E. H. (2003). Support vector machine with adaptive parameters in financial time series forecasting. *IEEE Transactions on neural networks*, 14(6), 1506–1518.

Cao, L., Sharma, P., Fahmy, S. & Saxena, V. (2015). Nfv-vital: A framework for characterizing the performance of virtual network functions. *2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN)*, pp. 93–99.

Carella, G., Corici, M., Crosta, P., Comi, P., Bohnert, T. M., Corici, A. A., Vingarzan, D. & Magedanz, T. (2014). Cloudified IP multimedia subsystem (IMS) for network function virtualization (NFV)-based architectures. *2014 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–6.

Chai, T. & Draxler, R. R. (2014). Root mean square error (RMSE) or mean absolute error (MAE). *Geoscientific Model Development Discussions*, 7(1), 1525–1534.

Chen, J. & Wang, Y. (2020). An Adaptive Short-Term Prediction Algorithm for Resource Demands in Cloud Computing. *IEEE Access*, 8, 53915-53930. doi: 10.1109/ACCESS.2020.2981011.

Chen, J. & Wang, Y. (2020). An Adaptive Short-Term Prediction Algorithm for Resource Demands in Cloud Computing. *IEEE Access*, 8, 53915–53930.

Chen, Y., Liu, Z., Xu, H., Darrell, T. & Wang, X. (2021). Meta-baseline: Exploring simple meta-learning for few-shot learning. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9062–9071.

Choudhury, S. K., Sa, P. K., Choo, K.-K. R. & Bakshi, S. (2017). Segmenting foreground objects in a multi-modal background using modified Z-score. *Journal of Ambient Intelligence and Humanized Computing*, 1–15.

Clayman, S. The lattice monitoring framework open-source software.

Clayman, S., Maini, E., Galis, A., Manzalini, A. & Mazzocca, N. (2014). The dynamic placement of virtual network functions. *2014 IEEE network operations and management symposium (NOMS)*, pp. 1–9.

da Costa, L. A. L., Kunst, R. & de Freitas, E. P. (2022). Intelligent resource sharing to enable quality of service for network clients: the trade-off between accuracy and complexity. *Computing*, 1–13.

da Rosa Righi, R., Correa, E., Gomes, M. M. & da Costa, C. A. (2020). Enhancing performance of IoT applications with load prediction and cloud elasticity. *Future Generation Computer Systems*, 109, 689–701.

Dogani, J., Khunjush, F., Mahmoudi, M. R. & Seydali, M. (2023). Multivariate workload and resource prediction in cloud computing using CNN and GRU by attention mechanism. *The Journal of Supercomputing*, 79(3), 3437–3470.

Dubba, S., Gupta, S. & Killi, B. R. (2024). Predictive resource allocation and VNF deployment using ensemble learning. *Multimedia Tools and Applications*, 1–26.

Eramo, V., Lavacca, F. G., Catena, T. & Salazar, P. J. P. (2021). Application of a Long Short Term Memory neural predictor with asymmetric loss function for the resource allocation in NFV network architectures. *Computer Networks*, 193, 108104.

Farahnakian, F., Pahikkala, T., Liljeberg, P., Plosila, J., Hieu, N. T. & Tenhunen, H. (2016). Energy-aware VM consolidation in cloud data centers using utilization prediction model. *IEEE Transactions on Cloud Computing*, 7(2), 524–536.

Fei, X., Liu, F., Xu, H. & Jin, H. (2018). Adaptive vnf scaling and flow routing with proactive demand prediction. *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pp. 486–494.

Feng, J., Chen, X., Gao, R., Zeng, M. & Li, Y. (2018). Deeptp: An end-to-end neural network for mobile cellular traffic prediction. *IEEE Network*, 32(6), 108–115.

Fenton, N., Marsh, W., Neil, M., Cates, P., Forey, S. & Tailor, M. (2004). Making resource decisions for software projects. *Proceedings of the 26th international conference on software engineering*, pp. 397–406.

Freund, Y. & Schapire, R. E. (1999). Large margin classification using the perceptron algorithm. *Machine learning*, 37(3), 277–296.

Gambi, A. (2012). *Kriging-based self-adaptive controllers for the cloud*. (Ph.D. thesis, Università della Svizzera italiana).

Gardner, M. W. & Dorling, S. (1998). Artificial neural networks (the multilayer perceptron)â€"a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15), 2627–2636.

Gepp, A., Kumar, K. & Bhattacharya, S. (2010). Business failure prediction using decision trees. *Journal of forecasting*, 29(6), 536–555.

Gong, Z., Gu, X. & Wilkes, J. (2010). Press: Predictive elastic resource scaling for cloud systems. *2010 International Conference on Network and Service Management*, pp. 9–16.

Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep learning*. MIT press.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R. & Schmidhuber, J. (2016). LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10), 2222–2232.

Gulhane, V. A., Rode, S. V. & Pande, C. B. (2023). Correlation analysis of soil nutrients and prediction model through ISO cluster unsupervised classification with multispectral data. *Multimedia Tools and Applications*, 82(2), 2165–2184.

Gupta, L., Samaka, M., Jain, R., Erbad, A., Bhamare, D. & Metz, C. (2017). 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)*, pp. 1–9.

Gupta, S., Dileep, A. D. & Gonsalves, T. A. (2020). Online Sparse BLSTM Models for Resource Usage Prediction in Cloud Datacentres. *IEEE Transactions on Network and Service Management*, 17(4), 2335-2349. doi: 10.1109/TNSM.2020.3013922.

Gupta, S. & Dinesh, D. A. (2017). Resource usage prediction of cloud workloads using deep bidirectional long short term memory networks. *2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, pp. 1–6.

Gupta, V., Dharmaraja, S. & Arunachalam, V. (2015). Stochastic modeling for delay analysis of a VoIP network. *Annals of Operations Research*, 233(1), 171–180.

Hagenbuchner, M., Sperduti, A. & Tsoi, A. C. (2003). A self-organizing map for adaptive processing of structured data. *IEEE transactions on Neural Networks*, 14(3), 491–505.

Hartigan, J. A. & Wong, M. A. (1979). A k-means clustering algorithm. *JSTOR: Applied Statistics*, 28(1), 100–108.

Haykin, S. & Network, N. (2004). A comprehensive foundation. *Neural networks*, 2(2004), 41.

Herbst, N., Amin, A., Andrzejak, A., Grunske, L., Kounev, S., Mengshoel, O. J. & Sundararajan, P. (2017). Online workload forecasting. *Self-Aware Computing Systems*, 529–553.

Herrera, J. G. & Botero, J. F. (2016). Resource allocation in NFV: A comprehensive survey. *IEEE Transactions on Network and Service Management*, 13(3), 518–532.

Hinton, G. E., Osindero, S. & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7), 1527–1554.

Hoong, P. K., Tan, I. K. & Keong, C. Y. (2012). Bittorrent network traffic forecasting with ARMA. *arXiv preprint arXiv:1208.1896*.

Hsieh, S.-Y., Liu, C.-S., Buyya, R. & Zomaya, A. Y. (2020). Utilization-prediction-aware virtual machine consolidation approach for energy-efficient cloud data centers. *Journal of Parallel and Distributed Computing*, 139, 99–109.

Hu, R., Jiang, J., Liu, G. & Wang, L. (2013). CPU load prediction using support vector regression and Kalman smoother for cloud. *2013 IEEE 33rd international conference on distributed computing systems workshops*, pp. 88–92.

Huang, Z., Xu, W. & Yu, K. (2015). Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Imdoukh, M., Ahmad, I. & Alfailakawi, M. G. (2020). Machine learning-based auto-scaling for containerized applications. *Neural Computing and Applications*, 32(13), 9745–9760.

Iqbal, M. F. & John, L. K. (2012). Power and performance analysis of network traffic prediction techniques. *2012 IEEE International Symposium on Performance Analysis of Systems & Software*, pp. 112–113.

Jalalitabar, M., Guler, E., Zheng, D., Luo, G., Tian, L. & Cao, X. (2018). Embedding dependence-aware service function chains. *Journal of Optical Communications and Networking*, 10(8), C64–C74.

Jalodia, N., Henna, S. & Davy, A. (2019). Deep Reinforcement Learning for Topology-Aware VNF Resource Prediction in NFV Environments. *2019 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pp. 1–5.

Jarray, A. & Karmouch, A. (2013). VCG auction-based approach for efficient Virtual Network embedding. *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*, pp. 609–615.

Jeddi, S. & Sharifian, S. (2019). A water cycle optimized wavelet neural network algorithm for demand prediction in cloud computing. *Cluster Computing*, 22(4), 1397–1412.

Jeong, B., Baek, S., Park, S., Jeon, J. & Jeong, Y.-S. (2023). Stable and efficient resource management using deep neural network on cloud computing. *Neurocomputing*, 521, 99–112.

Jmila, H., Khedher, M. I. & El Yacoubi, M. A. (2017). Estimating VNF Resource Requirements Using Machine Learning Techniques. *International Conference on Neural Information Processing*, pp. 883–892.

Kalchbrenner, N., Danihelka, I. & Graves, A. (2015). Grid long short-term memory. *arXiv preprint arXiv:1507.01526*.

Kaur, K., Mangat, V. & Kumar, K. (2020). A comprehensive survey of service function chain provisioning approaches in SDN and NFV architecture. *Computer Science Review*, 38, 100298.

Khamsi, M. A. & Kirk, W. A. (2011). *An introduction to metric spaces and fixed point theory*. John Wiley & Sons.

Khandelwal, I., Adhikari, R. & Verma, G. (2015). Time series forecasting using hybrid ARIMA and ANN models based on DWT decomposition. *Procedia Computer Science*, 48, 173–179.

Kim, H., Park, S., Lange, S., Lee, D., Heo, D., Choi, H., Yoo, J.-H. & Hong, J. W.-K. (2020). Graph Neural Network-based Virtual Network Function Management. *APNOMS*, pp. 13–18.

Kim, I. K., Wang, W., Qi, Y. & Humphrey, M. (2018). Cloudinsight: Utilizing a council of experts to predict future cloud application workloads. *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, pp. 41–48.

Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kotsiantis, S. B., Zaharakis, I. D. & Pintelas, P. E. (2006). Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review*, 26(3), 159–190.

Krishnamoorthy, U., Karthika, V., Mathumitha, M., Panchal, H., Jatti, V. K. S. & Kumar, A. (2024). Learned prediction of cholesterol and glucose using ARIMA and LSTM models–A comparison. *Results in Control and Optimization*, 14, 100362.

Li, B., Lu, W., Liu, S. & Zhu, Z. (2018). Deep-learning-assisted network orchestration for on-demand and cost-effective vNF service chaining in inter-DC elastic optical networks. *IEEE/OSA Journal of Optical Communications and Networking*, 10(10), D29–D41.

Li, C., Zhang, Q., Huang, C. & Luo, Y. (2023). Optimal Service Selection and Placement Based on Popularity and Server Load in Multi-access Edge Computing. *Journal of Network and Systems Management*, 31(1), 1–18.

Li, H.-W., Wu, Y.-S., Chen, Y.-Y., Wang, C.-M. & Huang, Y.-N. (2017). Application execution time prediction for effective cpu provisioning in virtualization environment. *IEEE Transactions on Parallel and Distributed Systems*, 28(11), 3074–3088.

Li, J., Yao, J., Xiao, D., Yang, D. & Wu, W. (2024). EvoGWP: Predicting Long-Term Changes in Cloud Workloads Using Deep Graph-Evolution Learning. *IEEE Transactions on Parallel and Distributed Systems*, 35(3), 499-516. doi: 10.1109/TPDS.2024.3357715.

Lim, M. K. & Sohn, S. Y. (2007). Cluster-based dynamic scoring model. *Expert Systems with Applications*, 32(2), 427–431.

Lindemann, B., Müller, T., Vietz, H., Jazdi, N. & Weyrich, M. (2021). A survey on long short-term memory networks for time series prediction. *Procedia CIRP*, 99, 650–655.

Liu, H., Ding, S., Wang, S., Zhao, G. & Wang, C. (2022). Multi-objective Optimization Service Function Chain Placement Algorithm Based on Reinforcement Learning. *Journal of Network and Systems Management*, 30(4), 1–25.

Lu, J. & Turner, J. (2006). Efficient mapping of virtual networks onto a shared substrate.

Lu, Y., Panneerselvam, J., Liu, L., Wu, Y. et al. (2016). RVLBPNN: A workload forecasting model for smart cloud computing. *Scientific Programming*, 2016.

Markham, I. S. & Rakes, T. R. (1998). The effect of sample size and variability of data on the comparative performance of artificial neural networks and regression. *Computers & operations research*, 25(4), 251–263.

Masdari, M. & Khoshnevis, A. (2019). A survey and classification of the workload forecasting methods in cloud computing. *Cluster Computing*, 1–26.

Masoud, M., Lee, S. & Belkasim, S. (2016). Dynamic allocation of service function chains under priority dependency constraint. *2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pp. 684–688.

McMahan, H. B. (2010). A unified view of regularized dual averaging and mirror descent with implicit updates. *arXiv preprint arXiv:1009.3240*.

Mijumbi, R., Gorricho, J.-L. & Serrat, J. (2014a). Contributions to efficient resource management in virtual networks. *IFIP International Conference on Autonomous Infrastructure, Management and Security*, pp. 47–51.

Mijumbi, R., Gorricho, J.-L., Serrat, J., Claeys, M., De Turck, F. & Latré, S. (2014b). Design and evaluation of learning algorithms for dynamic resource management in virtual networks. *2014 IEEE Network Operations and Management Symposium (NOMS)*, pp. 1–9.

Mijumbi, R., Hasija, S., Davy, S., Davy, A., Jennings, B. & Boutaba, R. (2016a). A connectionist approach to dynamic resource management for virtualised network functions. *2016 12th International Conference on Network and Service Management (CNSM)*, pp. 1–9.

Mijumbi, R., Serrat, J., Gorricho, J.-L., Latré, S., Charalambides, M. & Lopez, D. (2016b). Management and orchestration challenges in network functions virtualization. *IEEE Communications Magazine*, 54(1), 98–105.

Mijumbi, R., Hasija, S., Davy, S., Davy, A., Jennings, B. & Boutaba, R. (2017). Topology-aware prediction of virtual network function resource requirements. *IEEE Transactions on Network and Service Management*, 14(1), 106–120.

Moradi, M., Ahmadi, M. & Nikbazm, R. (2022). Comparison of machine learning techniques for VNF resource requirements prediction in NFV. *Journal of Network and Systems Management*, 30, 1–29.

Morid, M. A., Sheng, O. R. L. & Dunbar, J. (2023). Time series prediction using deep learning methods in healthcare. *ACM Transactions on Management Information Systems*, 14(1), 1–29.

Mostafavi, S., Hakami, V. & Sanaei, M. (2021). Quality of service provisioning in network function virtualization: a survey. *Computing*, 103(5), 917–991.

Nehra, P. & Nagaraju, A. (2022). Host utilization prediction using hybrid kernel based support vector regression in cloud data centers. *Journal of King Saud University-Computer and Information Sciences*, 34(8), 6481–6490.

Nguyen, H., Shen, Z., Gu, X., Subbiah, S. & Wilkes, J. (2013). {AGILE}: Elastic Distributed Resource Scaling for Infrastructure-as-a-Service. *10th International Conference on Autonomic Computing ({ICAC} 13)*, pp. 69–82.

O'Grady, K. E. (1982). Measures of explained variance: Cautions and limitations. *Psychological Bulletin*, 92(3), 766.

Organization name. (1999). *Norm title*. Institution and norm number. Location: Organization name.

O'Shea, K. & Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.

Ouhame, S., Hadi, Y. & Ullah, A. (2021). An efficient forecasting approach for resource utilization in cloud data center using CNN-LSTM model. *Neural Computing and Applications*, 1–13.

Ozer, D. J. (1985). Correlation and the coefficient of determination. *Psychological bulletin*, 97(2), 307.

Patel, P., Ranabahu, A. H. & Sheth, A. P. (2009). Service level agreement in cloud computing.

Qian, L., Luo, Z., Du, Y. & Guo, L. (2009). Cloud computing: An overview. *IEEE International Conference on Cloud Computing*, pp. 626–631.

Qiu, F., Zhang, B. & Guo, J. (2016). A deep learning approach for VM workload prediction in the cloud. *2016 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pp. 319–324.

Rahmanian, A. A., Ghobaei-Arani, M. & Tofighy, S. (2018). A learning automata-based ensemble resource usage prediction algorithm for cloud computing environment. *Future Generation Computer Systems*, 79, 54–71.

Ramchoun, H., Idrissi, M. A. J., Ghanou, Y. & Ettaouil, M. (2016). Multilayer Perceptron: Architecture Optimization and Training. *IJIMAI*, 4(1), 26–30.

Salimian, L., Safi Esfahani, F. & Nadimi-Shahraki, M.-H. (2016). An adaptive fuzzy threshold-based approach for energy and performance efficient consolidation of virtual machines. *Computing*, 98(6), 641–660.

Sangani, S., Patil, R. & Goudar, R. (2024). Efficient algorithm for error optimization and resource prediction to mitigate cost and energy consumption in a cloud environment. *International Journal of Information Technology*, 1–11.

Sanger, T. D. (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural networks*, 2(6), 459–473.

Saxena, D., Kumar, J., Singh, A. K. & Schmid, S. (2023). Performance analysis of machine learning centered workload prediction models for cloud. *IEEE Transactions on Parallel and Distributed Systems*, 34(4), 1313–1330.

Scarselli, F. & Tsoi, A. C. (1998). Universal approximation using feedforward neural networks: A survey of some existing methods, and some new results. *Neural networks*, 11(1), 15–37.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M. & Monfardini, G. (2008). The graph neural network model. *IEEE transactions on neural networks*, 20(1), 61–80.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M. & Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1), 61–80.

Sciancalepore, V., Samdanis, K., Costa-Perez, X., Bega, D., Gramaglia, M. & Banchs, A. (2017). Mobile traffic forecasting for maximizing 5G network slicing resource utilization. *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pp. 1–9.

Sekwatlakwatla, S. P. & Malele, V. (2024). An approach for optimizing resource allocation and usage in cloud computing systems by predicting traffic flow. *Latin-American Journal of Computing*, 11(1), 80–89.

Shalev-Shwartz, S. et al. (2012). Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2), 107–194.

Shen, X. S., Huang, C., Liu, D., Xue, L., Zhuang, W., Sun, R. & Ying, B. (2021). Data Management for Future Wireless Networks: Architecture, Privacy Preservation, and Regulation. *IEEE Network*, 35(1), 8–15.

Shi, R., Zhang, J., Chu, W., Bao, Q., Jin, X., Gong, C., Zhu, Q., Yu, C. & Rosenberg, S. (2015). MDP and machine learning-based cost-optimization of dynamic resource allocation for network function virtualization. *2015 IEEE International Conference on Services Computing*, pp. 65–73.

Shyam, G. K. & Manvi, S. S. (2016). Virtual resource prediction in cloud environment: a Bayesian approach. *Journal of Network and Computer Applications*, 65, 144–154.

Sihn, S. & Park, J. W. (2008). MAE: an integrated design tool for failure and life prediction of composites. *Journal of composite materials*, 42(18), 1967–1988.

Silverstein, C. & Shieber, S. M. (1996). Predicting individual book use for off-site storage using decision trees. *The Library Quarterly*, 66(3), 266–293.

Simaiya, S., Lilhore, U. K., Sharma, Y. K., Rao, K. B., Maheswara Rao, V., Baliyan, A., Bijalwan, A. & Alroobaea, R. (2024). A hybrid cloud load balancing and host utilization prediction method using deep learning and optimization techniques. *Scientific Reports*, 14(1), 1337.

Singh, G., Sengupta, P., Mehta, A. & Bedi, J. (2024). A feature extraction and time warping based neural expansion architecture for cloud resource usage forecasting. *Cluster Computing*, 1–20.

Song, B., Yu, Y., Zhou, Y., Wang, Z. & Du, S. (2018). Host load prediction with long short-term memory in cloud computing. *The Journal of Supercomputing*, 74(12), 6554–6568.

St-Onge, C., Kara, N. & Edstrom, C. (2023). Multivariate outlier filtering for A-NFVLearn: an advanced deep VNF resource usage forecasting technique. *The Journal of Supercomputing*, 79(14), 16206–16232.

Sun, Q., Lu, P., Lu, W. & Zhu, Z. (2016). Forecast-assisted NFV service chain deployment based on affiliation-aware vNF placement. *2016 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6.

Taha, M. B., Sanjalawe, Y., Al-Daraiseh, A., Fraihat, S. et al. (2024). Proactive Auto-Scaling for Service Function Chains in Cloud Computing based on Deep Learning. *IEEE Access*.

Thantharate, A. & Beard, C. (2023). ADAPTIVE6G: Adaptive Resource Management for Network Slicing Architectures in Current 5G and Future 6G Systems. *Journal of Network and Systems Management*, 31(1), 1–24.

Torabi, R., Moradi, P. & Khantaimoori, A. R. (2012). Predict student scores using bayesian networks. *Procedia-Social and Behavioral Sciences*, 46, 4476–4480.

Tseng, F., Wang, X., Chou, L., Chao, H. & Leung, V. C. M. (2018). Dynamic Resource Prediction and Allocation for Cloud Data Center Using the Multiobjective Genetic Algorithm. *IEEE Systems Journal*, 12(2), 1688-1699. doi: 10.1109/JSYST.2017.2722476.

Tseng, F.-H., Wang, X., Chou, L.-D., Chao, H.-C. & Leung, V. C. (2017). Dynamic resource prediction and allocation for cloud data center using the multiobjective genetic algorithm. *IEEE Systems Journal*, 12(2), 1688–1699.

Tsoi, A. C., Morini, G., Scarselli, F., Hagenbuchner, M. & Maggini, M. (2003). Adaptive ranking of web pages. *Proceedings of the 12th international conference on World Wide Web*, pp. 356–365.

Vashistha, A. & Verma, P. (2020). A literature review and taxonomy on workload prediction in cloud data center. *2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pp. 415–420.

Waclawski, A. C. [US Patent 6,574,587]. (2003, 3). System and method for extracting and forecasting computing resource data such as CPU consumption using autoregressive methodology. Google Patents.

Wang, D. & Li, M. (2017). Stochastic configuration networks: Fundamentals and algorithms. *IEEE transactions on cybernetics*, 47(10), 3466–3479.

Wang, J., Tang, J., Xu, Z., Wang, Y., Xue, G., Zhang, X. & Yang, D. (2017). Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach. *IEEE INFOCOM 2017-IEEE conference on computer communications*, pp. 1–9.

Wang, T., Fan, Q., Li, X., Zhang, X., Xiong, Q., Fu, S. & Gao, M. (2021). DRL-SFCP: Adaptive Service Function Chains Placement with Deep Reinforcement Learning. *ICC 2021-IEEE International Conference on Communications*, pp. 1–6.

Weingärtner, R., Bräscher, G. B. & Westphall, C. B. (2015). Cloud resource management: A survey on forecasting and profiling models. *Journal of Network and Computer Applications*, 47, 99–106.

Page title. (1999a). In *Wikipedia*. Retrieved on 2013-05-16 from: https://www.wikipedia.org.

Page title. (1999b). In *Wikipedia*. Retrieved on 2013-05-16 from: https://www.wikipedia.org.

Wiley, W. L. & Bugenhagen, M. K. [US Patent App. 11/809,881]. (2008, 28). System and method for managing a service level agreement. Google Patents.

Wu, H., Zhang, W., Zhang, J., Wei, J. & Huang, T. (2013). A benefit-aware on-demand provisioning approach for multi-tier applications in cloud computing. *Frontiers of Computer Science*, 7, 459–474.

Xiao, Y., Zhang, Q., Liu, F., Wang, J., Zhao, M., Zhang, Z. & Zhang, J. (2019). NFVdeep: Adaptive online service function chain deployment with deep reinforcement learning. *Proceedings of the International Symposium on Quality of Service*, pp. 1–10.

Xu, D., Bai, M., Long, T. & Gao, J. (2021). LSTM-assisted evolutionary self-expressive subspace clustering. *International Journal of Machine Learning and Cybernetics*, 1–17.

Xu, Y., Cai, X., Wang, E., Liu, W., Yang, Y. & Yang, F. (2023). Dynamic traffic correlations based spatio-temporal graph convolutional network for urban traffic prediction. *Information Sciences*, 621, 580–595.

Yao, Y. & Rosasco, L. On early stopping in gradient descent learning. *Constructive Approximation*, 26, 289–315.

Yazdanian, P. & Sharifian, S. (2021). E2LG: a multiscale ensemble of LSTM/GAN deep learning architecture for multistep-ahead cloud workload prediction. *The Journal of Supercomputing*, 77, 11052–11082.

Younge, A. J., Von Laszewski, G., Wang, L., Lopez-Alarcon, S. & Carithers, W. (2010). Efficient resource management for cloud computing environments. *International conference on green computing*, pp. 357–364.

Yuan, H., Bi, J., Tan, W. & Li, B. H. (2016). Temporal task scheduling with constrained service delay for profit maximization in hybrid clouds. *IEEE Transactions on Automation Science and Engineering*, 14(1), 337–348.

Yuan, H., Bi, J., Li, S., Zhang, J. & Zhou, M. (2024). An Improved LSTM-Based Prediction Approach for Resources and Workload in Large-scale Data Centers. *IEEE Internet of Things Journal*.

Zeng, H., Zhang, H., Guo, J., Ren, B., Wu, J. et al. (2024). A novel hybrid STL-transformer-ARIMA architecture for aviation failure events prediction. *Reliability Engineering & System Safety*, 110089.

Zhang, L., Zhang, H., Tang, Q., Dong, P., Zhao, Z., Wei, Y., Mei, J. & Xue, K. (2020). LNTP: An End-to-End Online Prediction Model for Network Traffic. *IEEE Network*.

Zhang, Q., Cheng, L. & Boutaba, R. (2010). Cloud computing: state-of-the-art and research challenges. *Journal of internet services and applications*, 1(1), 7–18.

Zhang, Q., Yang, L. T., Yan, Z., Chen, Z. & Li, P. (2018a). An efficient deep learning model to predict cloud workload for industry informatics. *IEEE transactions on industrial informatics*, 14(7), 3170–3178.

Zhang, W., Guo, W., Liu, X., Liu, Y., Zhou, J., Li, B., Lu, Q. & Yang, S. (2018b). LSTM-based analysis of industrial IoT equipment. *IEEE Access*, 6, 23551–23560.

Zharikov, E., Telenyk, S. & Bidyuk, P. (2020). Adaptive workload forecasting in cloud data centers. *Journal of Grid Computing*, 18(1), 149–168.

Zhu, Y., Zhang, W., Chen, Y. & Gao, H. (2019). A novel approach to workload prediction using attention-based LSTM encoder-decoder network in cloud environment. *EURASIP Journal on Wireless Communications and Networking*, 2019(1), 1–18.