AI-Based mechanism for the Predictive Resource Allocation of V2X related Network Services

Asterios Mpatziakas, Anastasios Sinanis, Iosif Hamlatzis, Anastasios Drosou, Dimitrios Tzovaras
Information Technologies Institute
Centre for Research and Technology (CERTH)
Thessaloniki, Greece
{ampatziakas, sinani, hamlatzis, drosou, Dimitrios.Tzovaras} @iti.gr

***Abstract*—This paper presents a mechanism that predicts the future position of a vehicle moving in both urban and/or highway environments. Based on this knowledge, it decides on the optimal position of the VNF and preemptively requests the allocation of network resources, before a cross-domain network service migration takes place. The objective of this mechanism is to ensure the uninterrupted, continuous connections of the vehicles, resulting in minimal or no service interruption time while ensuring an optimal utilization of Edge Cloud and Mobile Edge computational resources.**

***Index Terms*—Service Orchestration, Operations Support Systems, 5G networks, MEC, Artificial Intelligence, Deep Learning, Neuroevolution**

I. Introduction

Fifth generation mobile networks (5G) promise to enable innovative use cases for industries and vertical markets via numerous groundbreaking approaches that overcome limitations of legacy mobile networks. For the realization of the emerging 5G architecture, multiple existing and novel technologies are utilized. Two such technologies, essential for the realization of low-latency related use cases, are the virtualization of the network functions (NFV) composing the network services and the use of Multi-access edge computing (MEC).

The NFV concept, involves the replacement of the proprietary hardware used in the past for networking, with software that performing the same function allowing more efficient and flexible network deployment and operation [1]. This concept is further enhanced when combined with modern virtualization technologies such as containers, leading to some making a distinction between Virtual Network Functions(VNF) when referring to NFs hosted in Virtual Machines and Containerized or Cloud-Native Network Functions (CNF) when referring to NFs hosted in containers such as Docker [2]. Distributed MEC technologies combine telecommunications with IT [3]. The combination of these two technologies can be used to retain a minimal latency between the user and the infrastructure by moving the services the closest possible to the user at any given time. The ambition behind the proposed mechanism is to reach an optimal trade-off between network performance and deployment complexity and cost by leveraging Deep Reinforcement Learning that will be combined with an AI model trained from users' past mobility preferences to predict the correct placement of the services. To that end, any services with critical requirements used by the vehicles will be migrated and placed proactively, according to their requirements and the potential direction of the vehicle. In this way, it will be possible to: i) minimise the latency between the user and the service, ii) guarantee the service level agreements (SLAs).

Supported by the European Unions Horizon 2020 Research and Innovation Program through the 5G- Routes project under Grant Agreement No. 951867.

A. Key Contributions

This paper presents a novel approach to optimize the pre-emptive placement of a 5G network service based on minimizing Service Delays and preserving needed Quality-of-Service for services with Latency Requirements, while taking into account MEC server computational resources. To achieve this a mechanism composed of two parts both based on SotA AI algorithms is proposed: the first part predicts the location of moving vehicles to determine the need for service reallocation while the second part determines the optimal service positioning. To the best of the authors' knowledge, neither of the AI algorithms composing the proposed method have been yet applied to the specific problem; based on their application in other fields, detailed in II , results beyond SotA are expected.

The remainder of the paper is structured as follows: Section II presents a brief review of the relevant literature and Section III presents the methodology used to formulate the proposed approach while section IV contains the results of experiments that were performed to validate the proposed method. Finally, section V contains the conclusions of this paper along with future research directions and aims.

II. Related Work

The ultra-low latency and reliability requirements imposed by automotive services can be met through the extensive use of edge resources by deploying components of the end-to-end services as close as possible to the network edges. Relevant literature reports that using CNFs can

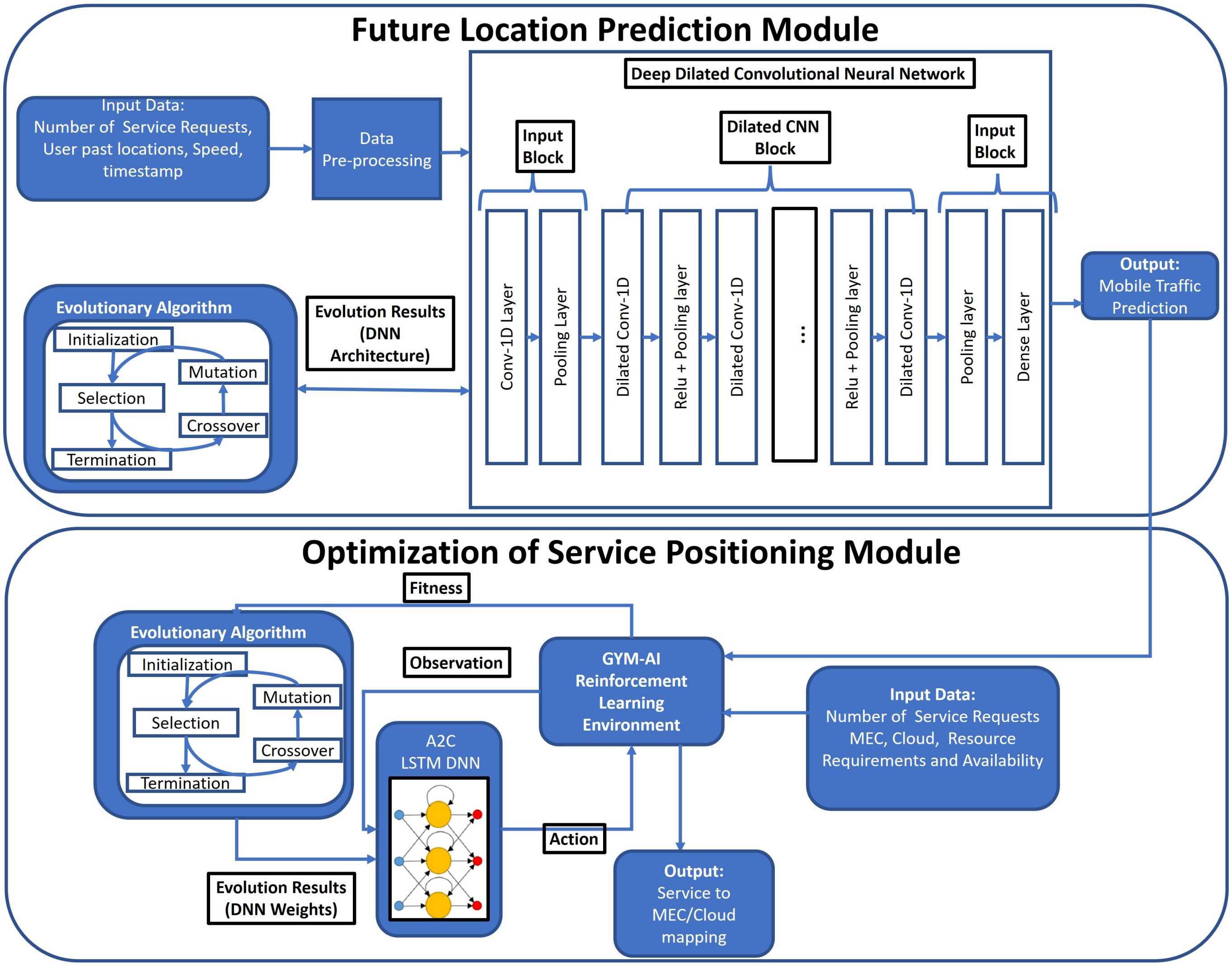

Fig. 1. Aggregated mean for the number of Quality of Service violations for N=500 experiments.

achieve Service Creation in the range of milliseconds [1], [2] while experimental results show a CNF service supported by MEC can be re-allocated with downtime less than 2.5 seconds [4]. However, edge resources are finite. Therefore, a large body of research leverages Artificial Intelligence (AI) algorithms to predict and pre-emptively ensure that NFs affecting V2X services have adequate resources allocated to perform their tasks.

A. Prediction of user mobility

Prediction of user equipment mobility is the first step of the proposed approach. It is a task already researched in the context for 3G/4G networks to account for the power consumption in cells or balance network radio resources or the number of successful handovers. With the advent of VNF this problem needs to be reformulated to account for MEC/edge cloud computational resources and to take into account the challenges presented by network slicing. In the case of legacy networks, the problem was solved by various approaches falling in the following three categories:

- Stochastic Methods such as Markov Models or Dynamic Bayesian Networks [5], [6];
- Heuristic Methods, such as the one presented in [7];
- Machine learning or Deep Learning based methods such as Poisson Regression Trees [8].

These approaches have also been applied in 5G networks. A review of the available literature shows that machine learning and artificial intelligence empowered approaches are more suitable [9], [10], achieving higher prediction accuracy scores compared to the other two methods. Therefore, AI is employed as a suitable tool for the development of the mechanism described in this paper. We propose the use of an AI model based on Dilated Causal Convolutional Neural Networks (DCNN) [11] trained via Deep Neuroevolution, to predict the incoming traffic and subsequently use this result as input to the resource allocation process. Deep Neuroevolutional techniques combine two sub-fields of AI, Deep Neural Networks (DNN) and Evolutionary Algorithms [12]. Applying

Evolutionary algorithms to determine the optimal architecture of DNNs has been shown to produce promising results, often surpassing the typically used gradient-based methods [13], [14]. Other SotA DNN architectures for time-series prediction such as the Sequence-to-Sequence DNN (Seq2Seq) solution presented in [15] and a Long Short-Term Memory (LSTM) CNN solution will also be implemented for comparison. Table I presents the Deep Neural Networks and DNN architectures most commonly applied to time-series along with a brief presentation for each one. DCNNs were originally created for applications in the field of Natural Language Processing. However, they have been shown to outperform both classic approaches and other SotA DNN when applied to timeseries data, for example in [16]–[18] since they effectively model sparse data, both long-term and short sequence relationships, and different time resolutions [16].

B. Optimization of service positioning

The second part of the proposed approach involves the development of AI empowered models that are triggered when service migration is required, to ascertain proper positioning of VNFs in the Edge Cloud and MEC network components. Based on the available literature, different approaches to solve this problem can be clustered to the following three categories [25]:

- based on prediction, i.e. adapting the network proactively to meet estimated changes,
- dynamically adapting the network as traffic fluctuates or
- on-the-fly, i.e. after a certain event occurs such as a number of requests are denied.

VNF migration is a hot research topic that has gained a lot of attention over the past years. In [26], Yi et al. propose an on-the-fly mechanism based on heuristics. In [25], Sarrigiannis et al. suggest a dynamic/online example of a process that handles VNF mitigation based on an iterative method. In [27], a Deep Neural Network architecture tuned by an Evolutionary Algorithm, is used to proactively assess the best server for migration, while in [28] the authors propose a context-aware stateful VNF migration using a Mixture Density Neural Network. In [29] a Deep Belief Neural Network is connected to a mechanism empowered by an Evolutionary Algorithm to handle the task. In [30] a DRL method is applied to ascertain change traffic point and avoid resource shortage via migration whereas in [31], a DRL neural network is used to migrate chains of VNF. The results presented in the literature, indicate that AI based methods produce better resource management schemes compared to other non-AI methods (such as heuristic methods).

We propose an AI model combining Evolutionary Algorithms with an Actor-Critic Deep Neural Network, employing Reinforcement Learning for the VNF migration part is being tested. Actor-Critic DNNs, as the name suggests, are composed of two parts. The actor part uses states, i.e. representations of the task environment, as input. It tries to output the optimal action by learning a series of optimal policies. The critic part evaluates the action decided by the actor part by computing a predefined function [32]. LSTM DNN will be used in internal parts of the network. This work expands the SotA Deep Reinforcement learning method presented in [31] by training the DRL model using Neuroevolution techniques. Based on the literature [13], [14], such approaches are expected to produce results that surpass the SotA.

III. Proposed Model

To model the trajectory of the vehicles monitored by the proposed mechanism we propose a mapping between their position and a predefined partition of the area of interest. Let $L = \{(lon_1, lat_1), ..., (lon_n, lat_n)\}$, be the set containing the position using longitude-latitude pairs for the n vehicles monitored at a certain time-point and $C = c_1, c_2, ...c_m,\ c \in \mathbf{N}$ be the set of cells resulting by partitioning the area of interest to a grid of equally sized square cells. Then, a function $f : L \to C$ provides the required mapping.

A. Future Location prediction Module

At any given time point t for a single vehicle, we define vector $a^t = \{c_a^{t-1}, v_a^t, t\}$, where c_a^{t-1} is the cell the vehicle was in the past location, and v is the speed of the vehicle. Having defined these variables, the task of predicting the future user location for time t can be defined as a function f that takes as input a sequence $\{a^t, a^{t-1}, ...a^{t-l}\}$ and outputs $c_a^{t+1} \in C$ i.e. the cell where the vehicle will be located in the next time point :

$$c_a^{t+1} = f(\{a^t, a^{t-1}, ...a^{t-l}\}), \forall c_a^t \in C, \tag{1}$$

which is learned by the Dilated Causal Convolutional Neural Network.

B. Optimization of Service Positioning Module

Moving on to the task of optimizing service positioning: Let $M = \{m_1, m_2, ..., m_n\},\ n \in N$ be the available MECs and C be the Regional Core Cloud Server. Let us assume that for each MEC a vector $R_{m_j} = \{CPU_{m_j}, MEM_{m_j}, HD_{m_j}, LAT_{m_j}\}$ and for the Core Cloud Server $R_C = \{CPU_C, MEM_C, HD_C, LAT_C\}$, are known, representing the number of CPU cores,the RAM memory, storage and Latency between the MEC and the switch they are connected respectively. Additionally, let LAT_{link} be the latency between different switches.

For a given time point t, let $S = \{s_1, s_2, ..., s_k\},\ k \in N$ be the number of services tracked by the mechanism. Each service requires a specific amount of computational resources $R_{s_k} = \{CPU_{s_k}, MEM_{s_k}, HD_{s_k}\}$ from the server it is deployed to optimally perform and is also linked to a Quality of Service level $Q_{s_k} = \{None, Low, High\}$.

The problem of optimal service placement can be considered a version of the multi-objective binary bin packing

TABLE I

A comparison of the Neural Network architectures commonly used to handle time-series data.

Deep Neural Network Type	Time Series Application
Recurrent Neural Networks (RNN)	RNNs can learn the relationship between present and past inputs, detecting patterns in sequential information. They have been shown to have problems capturing long term dependencies [18] due to vanishing gradient. LSTM and Gated Recurrent Units have been designed to overcome this problem.
Sequence to Sequence Networks (Seq2Seq)	This architecture is comprised by simpler DNN, usually RNN that encode and then decode sequential data. During the encoding phase the entire prior data is taken into account, effectively capturing long-term dependencies [18]. Additionally, these models generalize better and produce lower error results compared to RNN [15]. Seq2Seq DNN can be further enhanced by incorporating mechanisms such as 'Attention' or 'Beam Search' [19].
Convolutional Neural Networks (CNN)	CNN segment the input data using so called filters, which allows them to learn specific patterns. Simple CNN have been shown to model multi-dimension patterns, such as those in two-dimensional trajectories of vehicle movement more effectively in comparison to RNN [20]. They can also capture the high spatial and temporal correlation of movement data [21] providing comparable results to RNN. Additionally, CCN models scale better compared to RNN [22].
RNN and CNN Stacks	This architecture combines CNN to capture spatial and temporal correlations and RNN to capture temporal dependencies. This combination has been shown to outperform simple RNN or CNN [23].
Dilated Causal Convolutional Neural Networks (Proposed approach)	In this variant of the CNN, also called Wavenet, the filters are applied by skipping certain elements in the input, allowing the receptive field of the network to grow exponentially [22]. This property allows them to model even sparse data along with both long-term and short sequence relationships present. The literature suggests that approaches using DCNN outperform all other approaches presented in this table [16], [17], [22], [24].

problem, where k items i.e. services must be assigned to $n+1$ bins i.e. the MEC servers and the Core Cloud server. It can be formulated in the following manner: Let $y_j = 1$ if MEC/Cloud server j is used, $x_{kj} = 1$ of service k is placed on server j , $q_{kj} = 1$ if server j satisfies the SLA agreement regarding service k and l_{kj} the latency of service k is placed on server j.

The the aim is to produce an optimal solution satisfying the following two objectives a) maximize the number of services with satisfied QoS:

$$kpi_1 = argmax \sum_{k \in S}^{n+1} q_{kj}, \forall j \in \{1, ..., n+1\}, k \in S, \quad (2)$$

and b) minimize the total service delay

$$kpi_2 = argmin \sum_{k \in S}^{n+1} l_{kj} * x_{kj}, \forall j \in \{1, ..., n+1\}, k \in S, \quad (3)$$

, while taking into account the available computational resources i.e.

$$\sum_{k \in K}^{n+1} R_{s_k} * x_{kj} \leq R_{m_j}, \forall j \in \{1, ..., n+1\}. \quad (4)$$

Finally, the two objectives are combined to a single function by normalizing their values in a range of [0,1], which is used as the function to be optimized:

$$kpi_3 = \frac{kpi_1}{k} - \frac{kpi_2}{k * max(l_{kj})}, \forall j \in \{1, ..., n+1\}, k \in S, \quad (5)$$

The first task is learned by a Dilated Causal Convolutional network trained using a method based on Evolutionary Algorithms called Population Based training [33] while in the second task a Deep Reinforcement learning Algorithm is applied. The DRL algorithm is trained using Neuroevolution and more specifically the Particle swarm Optimization algorithm [34] is applied. A high level architecture of the proposed mechanism is shown in figure 1 .

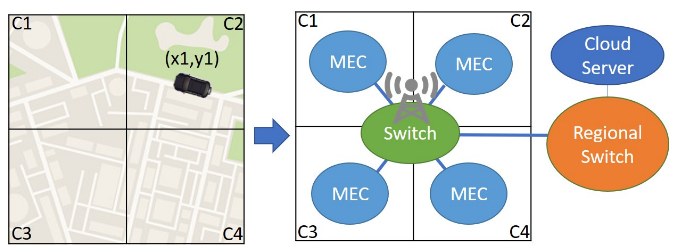

Fig. 2. High level overview of network model.

IV. Performance Evaluation

A. Experimental Setup

The AI algorithms composing the proposed mechanism were evaluated using a combination of two different data sets. The position prediction part is tested using an open dataset of a Large-Scale Urban Vehicular Mobility Data set, presented in [48]. This dataset covers the mobility pattern in a large German City for a day and comprises the traffic of more than 700,000 individual car trips. Apart from the model based on the proposed AI algorithm, three more AI models were trained based on the results of the literature review presented in II. The models where trained using 85% of the dataset, while 15% was held for evaluation. All models were extensively fine-tuned, using Population Based training.

To train and test the VNF Placement AI part of the mechanism, a synthetic dataset was created based on the assumptions presented in [20]. A network architecture and services as those described in [20] is assumed : Each cell has a MEC server with known CPU, RAM and HD resources and is connected to a switch collocated with a Base station. All the switches are connected to a regional switch, through which a cloud server that can also accommodate service requests can be reached. It is assumed that the latency for all the links in network is known. This network is used by a number of moving vehicles which use specific services. We assume that we know the computational needs for each such service, along with its' requirements for Latency and any predefined QoS agreement. For each time-point the first part of the mechanism predicts the next cells the vehicles will be located and based on that prediction, it tries to decide the optimal VNF placement based on minimizing Resource Utilization and Delay while satisfying QoS/Latency Requirements.

Table II, presents the various simulation parameters and algorithm hyper-parameters used in the experiments presented in section IV-B

TABLE II
Simulation parameters and algorithm hyper-parameters used for experiments

Variable	Values
MEC resources	CPU: 4 cores , Memory: 8GB , Storage: 32 GB
Cloud resources	CPU: 50 cores , Memory: 10 TB , Storage: 1000 TB
Mixture of QoS service level in used services (%)	Low 40%, High 40% , Critical 20%
Service resource requirements	All values where randomly sampled from the following ranges: CPU: [2,3], Memory: [1,2], Storage: [1,2]
Latency requirements	Low: 20 ms, High: 10 ms, Critical: 1ms
Connection Latency	Cloud: 22ms, Vehicle to same Cell MEC: 1ms, Switch to switch: 6 ms
DCNN hyper-parameters	Learning Rate: 1.54429e-07, Weight Decay : 0.280425, Optimizer : RMSprop, Activation: LeakyRelu, Weight initilization : k normal, Dilation Layers: 3, Layer size : 512
PSO hyper-parameters	public coefficient : 1.5, private coefficient: 1.5, inertia: 0.2
A2C LSTM hyper-parameters	Hidden Layers: 128, Hidden Layer Size: 64

B. Experimental Results

For the task of the prediction of the vehicle location, apart from the proposed algorithm (Dilated Causal Convolutional NN), three more algorithms (Seq2Seq, CNN, LSTM) where chosen for evaluation based on literature SotA. The Proposed model outperforms other models in terms of Accuracy for at least 2%. As shown in table IV-A, similar results occur for the other metrics commonly used to evaluate classification tasks i.e. the proposed algorithm produces results with at least 1% better Precision, 2% better Recall and 0.4% F1-Score.

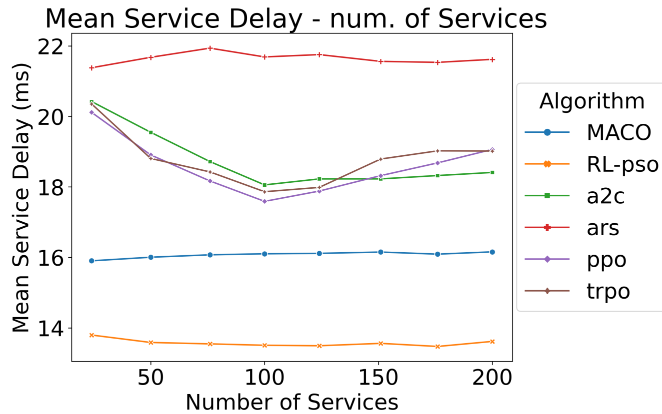

Fig. 3. Aggregated mean for service delay for N=500 experiments.

Fig. 4. Aggregated mean for the number of Quality of Service violations for N=500 experiments.

For the task of service placement, apart from the proposed algorithm (RL-PSO), four RL algorithms Asyncronous Actor-to-Critic (A2C), Proximaly Policy Optimization (PPO), Trust Region Policy Optimization (TRPO), Augmented Random Search (ARS) and a Evolutionary Algorithm named Multi-objective Ant Colony Optimizer (MACO) where chosen for evaluation based on literature SotA. For this task, multiple experiments were performed to account for the random factor introduced from the data creation process described in section IV-A. In these experiments, the number of services was scaled up from 25 to 200 instances. Each experiment was repeated N=500 times and the aggregated results for the mean Service Delay and number of QoS service violation are shown in table IV. Table IV presents the aggregated mean and standard deviation of the two objectives of interest for all the experiments. The proposed method outperforms all others, both in terms of minimizing the occurrence of QoS

TABLE III
RESULTS FOR THE TASK OF THE PREDICTION OF THE FUTURE LOCATION OF MOVING VEHICLES.

Model	Accuracy (%)	Precision(%)	Recall (%)	F1-Score(%)
Dilated Causal Convolutional NN (Proposed)	97.22	97.44	97.22	97.26
Seq2Seq	95.1	96.48	95.09	96.86
Simple CNN	94.98	96.22	94.98	96.46
CNN-LSTM stack	93.85	95.87	93.85	94.65

Service Violations and the mean Service Delay. Figures 3 and 4 show the aggregated mean and present these results in a visual manner.

V. Conclusions and Further Challenges

In this paper, a smart mechanism for the optimal Resource Allocation of V2X related Services is presented. It operates on two levels: Initially, it provides a prediction for the future location of moving vehicles that utilize the tracked services. Then, based on this prediction, it provides a mapping between services and their placements in either the MEC or Cloud servers available, taking into account both Quality of Service Agreements and Latency Requirements. The mechanism proposed, is based on two SotA Deep Learning Algorithms: the first one is called Dilated Causal Convolutional Neural Networks and is tuned using a method called Population Based Training while the second mechanism is based on Neuroevolution which combines Evolutionary Algorithms with Deep Neural Networks.

Experimental results showcase that the proposed approach outperforms similar SoA approaches, commonly used in the literature for the same task. As follow-up work, the proposed mechanism will be packaged in a docker container form and will be integrated with an Open Source MANO System [35]. Additionally, we plan to modify and extent the objectives currently used in the Reinforcement Learning algorithm to expand the aspects of the services taken into account for the optimization. Finally, following the integration with the MANO stack, we plan to integrate,test and demonstrate the mechanism in an actual 5G network implementation and apply it to automotive and railway related use-cases. The experiments performed in real life results will stress the proposed mechanism and validate its' effectiveness.

Acknowledgment

This work is supported by the European Unions Horizon 2020 Research and Innovation Program through the 5G-Routes project under Grant Agreement No. 951867. The opinions expressed in this paper are those of the authors and do not necessarily reflect the views of the European Commission.

References

[1] O. Arouk and N. Nikaein, "Kube5g: A cloud-native 5g service platform," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.

[2] A. Pino, P. Khodashenas, X. Hesselbach, E. Coronado, and S. Siddiqui, "Validation and benchmarking of cnfs in osm for pure cloud native applications in 5g and beyond," in *2021 International Conference on Computer Communications and Networks (ICCCN)*, 2021, pp. 1–9.

[3] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[4] C. Campolo, A. Iera, A. Molinaro, and G. Ruggeri, "Mec support for 5g-v2x use cases through docker containers," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1–6.

[5] N. A. Amirrudin, S. H. S. Ariffin, N. N. N. A. Malik, and N. E. Ghazali, "User's mobility history-based mobility prediction in lte femtocells network," in *2013 IEEE International RF and Microwave Conference (RFM)*, 2013, pp. 105–110.

[6] A. S. Gomes, B. Sousa, D. Palma, V. Fonseca, Z. Zhao, E. Monteiro, T. Braun, P. Simoes, and L. Cordeiro, "Edge caching with mobility prediction in virtualized lte mobile networks," *Future Generation Computer Systems*, vol. 70, pp. 148–162, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X16302072

[7] H.-L. Wang, S.-J. Kao, C.-Y. Hsiao, and F.-M. Chang, "A moving direction prediction-assisted handover scheme in LTE networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2014, no. 1, Nov. 2014. [Online]. Available: https://doi.org/10.1186/1687-1499-2014-190

[8] C. Ide, F. Hadiji, L. Habel, A. Molina, T. Zaksek, M. Schreckenberg, K. Kersting, and C. Wietfeld, "LTE connectivity and vehicular traffic prediction based on machine learning approaches," in *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*. IEEE, Sep. 2015. [Online]. Available: https://doi.org/10.1109/vtcfall.2015.7391019

[9] H. Zhang and L. Dai, "Mobility prediction: A survey on state-of-the-art schemes and future applications," *IEEE Access*, vol. 7, pp. 802–822, 2019. [Online]. Available: https://doi.org/10.1109/access.2018.2885821

[10] H. Gebrie, H. Farooq, and A. Imran, "What machine learning predictor performs best for mobility prediction in cellular networks?" in *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, May 2019. [Online]. Available: https://doi.org/10.1109/iccw.2019.8756972

[11] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," 2016. [Online]. Available: https://arxiv.org/abs/1609.03499

[12] A. Mpatziakas, S. Papadopoulos, A. Drosou, and D. Tzovaras, "Multi-objective optimisation for slice-aware resource orchestration in 5g networks," in *2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, 2020, pp. 79–86.

[13] F. P. Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, "Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning," 2017. [Online]. Available: https://arxiv.org/abs/1712.06567

[14] K. O. Stanley, J. Clune, J. Lehman, and R. Miikkulainen, "Designing neural networks through neuroevolution," *Nature Machine Intelligence*, vol. 1, no. 1, pp. 24–35, Jan. 2019. [Online]. Available: https://doi.org/10.1038/s42256-018-0006-z

TABLE IV

Aggregated results for N=500 experiment repetitions for the KPIs of Service and the number of QoS violations (mean and standard deviation). The number of services in each experiment was scaled from 25 to 200 with an increment of 25.

KPI / Algorithm	Service Requests							
	25		50		75		100	
	Delay	Violations	Delay	Violations	Delay	Violations	Delay	Violations
MACO	15.90 ± 0.77	5.58 ± 0.68	16.01 ± 0.51	11.28 ± 0.84	16.07 ± 0.38	17.83 ± 1.06	16.10 ± 0.36	23.55 ± 1.29
a2c	20.42 ± 0.71	5.79 ± 0.43	19.55 ± 0.59	11.23 ± 0.86	18.72 ± 0.49	17.61 ± 1.27	18.06 ± 0.50	22.30 ± 1.47
ars	21.38 ± 0.36	5.91 ± 0.29	21.68 ± 0.18	12.0 ± 0.0	21.94 ± 0.09	18.99 ± 0.1	21.69 ± 0.14	25.00 ± 0.20
ppo	20.11 ± 0.79	5.72 ± 0.53	18.91 ± 0.63	10.97 ± 0.99	18.17 ± 0.55	17.31 ± 1.3	17.59 ± 0.46	22.37 ± 1.56
pso	13.80 ± 1.05	4.72 ± 1.02	13.59 ± 0.75	9.36 ± 1.41	13.59 ± 0.66	14.71 ± 1.76	13.51 ± 0.54	19.64 ± 2.08
trpo	20.35 ± 0.72	5.77 ± 0.45	18.81 ± 0.67	10.91 ± 1.04	18.43 ± 0.55	17.18 ± 1.10	17.86 ± 0.47	22.45 ± 1.42
	Service Requests							
KPI / Algorithm	125		150		175		200	
	Delay	Violations	Delay	Violations	Delay	Violations	Delay	Violations
MACO	16.12 ± 0.31	29.17 ± 1.40	16.15 ± 0.26	36.22 ± 1.17	16.09 ± 0.29	41.51 ± 1.57	16.16 ± 0.06	47.0 ± 4.24
a2c	18.23 ± 0.42	27.4 ± 1.80	18.23 ± 0.34	34.22 ± 1.54	18.32 ± 0.36	40.69 ± 1.70	18.41 ± 0.57	47.0 ± 1.41
ars	21.75 ± 0.11	30.97 ± 0.17	21.56 ± 0.18	37.3 ± 0.81	21.53 ± 0.17	43.15 ± 0.84	21.62 ± 0.23	49.5 ± 0.71
ppo	17.88 ± 0.43	26.23 ± 1.96	18.319 ± 0.41	34.19 ± 1.94	18.68 ± 0.30	42.19 ± 1.23	19.06 ± 0.55	48.0 ± 1.41
pso	13.50 ± 0.52	24.09 ± 2.58	13.56 ± 0.49	30.15 ± 2.18	13.48 ± 0.38	34.39 ± 2.26	13.62 ± 0.43	40.5 ± 2.12
trpo	17.99 ± 0.41	27.21 ± 1.76	18.79 ± 0.35	34.98 ± 1.69	19.02 ± 0.38	42.76 ± 0.98	19.02 ± 0.26	50.0 ± 0.0

[15] C. Wang, L. Ma, R. Li, T. S. Durrani, and H. Zhang, "Exploring trajectory prediction through machine learning methods," *IEEE Access*, vol. 7, pp. 101 441–101 452, 2019.

[16] S. M. H. Rizvi, T. Syed, and J. Qureshi, "Real-time forecasting of petrol retail using dilated causal CNNs," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 2, pp. 989–1000, Feb. 2021. [Online]. Available: https://doi.org/10.1007/s12652-021-02941-3

[17] L. Börjesson and M. Singull, "Forecasting financial time series through causal and dilated convolutional neural networks," *Entropy*, vol. 22, no. 10, 2020. [Online]. Available: https://www.mdpi.com/1099-4300/22/10/1094

[18] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, and M. Weyrich, "A survey on long short-term memory networks for time series prediction," *Procedia CIRP*, vol. 99, pp. 650–655, 2021, 14th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 15-17 July 2020.

[19] F. Li, Z. Gui, Z. Zhang, D. Peng, S. Tian, K. Yuan, Y. Sun, H. Wu, J. Gong, and Y. Lei, "A hierarchical temporal attention-based lstm encoder-decoder model for individual mobility prediction," *Neurocomputing*, vol. 403, pp. 153–166, 2020.

[20] A. Dalgkitsis, P.-V. Mekikis, A. Antonopoulos, and C. Verikoukis, "Data driven service orchestration for vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4100–4109, 2021.

[21] N. Nikhil and B. Tran Morris, "Convolutional neural network for trajectory prediction," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, September 2018.

[22] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018.

[23] P. Shu, Y. Sun, Y. Zhao, and G. Xu, "Spatial-temporal taxi demand prediction using lstm-cnn," in *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, 2020, pp. 1226–1230.

[24] A. Borovykh, S. M. Bohté, and C. W. Oosterlee, "Dilated convolutional neural networks for time series forecasting," *Journal of Computational Finance*, 2018.

[25] I. Sarrigiannis, K. Ramantas, E. Kartsakli, P.-V. Mekikis, A. Antonopoulos, and C. Verikoukis, "Online vnf lifecycle management in an mec-enabled 5g iot architecture," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4183–4194, 2020.

[26] B. Yi, X. Wang, M. Huang, and K. Li, "Design and implementation of network-aware vnf migration mechanism," *IEEE Access*, vol. 8, pp. 44 346–44 358, 2020.

[27] D. M. Manias, H. Hawilo, and A. Shami, "A machine learning-based migration strategy for virtual network function instances," *ArXiv*, vol. abs/2006.08456, 2020.

[28] A. L. Ibrahimpasic, B. Han, and H. D. Schotten, "Ai-empowered vnf migration as a cost-loss-effective solution for network resilience," *2021 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, pp. 1–6, 2021.

[29] L. Tang, X. He, P. Zhao, G. Zhao, Y. Zhou, and Q. Chen, "Virtual network function migration based on dynamic resource requirements prediction," *IEEE Access*, vol. 7, pp. 112 348–112 362, 2019.

[30] K. Qu, W. Zhuang, X. Shen, X. Li, and J. Rao, "Dynamic resource scaling for vnf over nonstationary traffic: A learning approach," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 2, pp. 648–662, 2021.

[31] R. Chen, H. Lu, Y. Lu, and J. Liu, "Msdf: A deep reinforcement learning framework for service function chain migration," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, 2020, pp. 1–6.

[32] Z. Xiong, Y. Zhang, D. Niyato, R. Deng, P. Wang, and L.-C. Wang, "Deep reinforcement learning for mobile 5g and beyond: Fundamentals, applications, and challenges," *IEEE Vehicular Technology Magazine*, vol. 14, no. 2, pp. 44–52, Jun. 2019. [Online]. Available: https://doi.org/10.1109/mvt.2019.2903655

[33] A. Li, O. Spyra, S. Perel, V. Dalibard, M. Jaderberg, C. Gu, D. Budden, T. Harley, and P. Gupta, "A generalized framework for population based training," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 1791–1799. [Online]. Available: https://doi.org/10.1145/3292500.3330649

[34] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm Intelligence*, vol. 1, no. 1, pp. 33–57, Aug. 2007. [Online]. Available: https://doi.org/10.1007/s11721-007-0002-0

[35] A. Marsico, G. Garcia, F. J. Ramon, and A. Reid, *OSM in Action.* ETSI, Jul 2021. [Online]. Available: https://osm.etsi.org/images/OSM_EUAG_White_Paper_OSM_in_Action.pdf

.