Ensemble Deep Learning Aided VNF Deployment for IoT Services

Mahzabeen Emu
Department of Computer Science
Lakehead University
Thunder Bay, Canada
memu@lakeheadu.ca

Salimur Choudhury
Department of Computer Science
Lakehead University
Thunder Bay, Canada
salimur.choudhury@lakeheadu.ca

***Abstract*—In the sixth generation (6G) networks, due to the massive Internet of Things (IoT) connectivity and substantial growth of communication traffic, an effective Virtual Network Function (VNF) orchestration scheme is anticipated to function dynamically and intelligently. Moving beyond the traditional paradigm of the VNF orchestration and employing VNFs on the network edge located cloudlets based on the inspiration from multi-access edge computing can intensify the overall performance of delay-sensitive applications. In this paper, we intend to investigate how to simultaneously leverage the ensembling of multiple deep learning models for proper calibration to provide real-time VNF placement solutions. We also address the challenges associated with state-of-the-art approaches to deal with dynamic network traffic and topology patterns. Our envisioned methods, based on Convolutional Neural Networks and Artificial Neural Networks named as E-ConvNets and E-ANN respectively, suggest two proactive VNF deployment strategies. These VNF placement strategies demonstrate (simulation results) encouraging performance (optimality gap nearly 7%) in terms of minimizing relocation and communication costs, and high scalability intelligence factor (around 0.93). Moreover, the presented results are further indications of integrating edge computing and deep learning-based strategies into similar research enigmas for future telecommunication networks.**

***Index Terms*—Virtual Network Function, Deep Learning.**

I. Introduction

Based on the expectations to fulfill the demands of ultra-high processing speed and low communication delay sensitive applications, 6G cellular networks are envisioned to support an extensive variety of vertical use cases [1]. Some of the applications can be the massive connectivity of Internet of Things (IoT), collaborative computing, remote surgery and machinery, augmented reality (AR), virtual reality (VR), and autonomous driving [2]. Nevertheless, the current network service orchestration schemes become incompetent to handle numerous service specifications and various device types due to not implying sustainability for real-time applications and poor administration capability [2]. Thus, Network Function Virtualization (NFV) [3] pledges to facilitate network service provisioning at considerably decreased capital costs and operational expenditure. The intention is to replace the necessity of proprietary hardware devices with the software enabled implementation of Virtual Network Functions (VNFs) on conventional virtualized platforms such as virtual machines (VMs) running in cloudlets (small scale data centers at the edge of Internet) [3]. As suggested by the concept of multi-access edge computing (MEC) or fog computing [4], VNFs (e.g., firewall, load balancer, WAN accelerator, and intrusion detection system) placed at cloudlets in closer proximity to the users diminishes the burden of unnecessary data traversal and bandwidth consumption through the centralized cloud. The vision towards future telecommunication networks anticipates that the third parties will designate the content-aware and user-specific services along with their corresponding specifications, for example, the highest tolerable latency or least throughput limits, to the network administrator, expedited by NFV and software-defined networking (SDN) [5]. Mostly, state-of-the-art NFV resource orchestrators consider the static condition of networks, while ignoring the temporal differences in network traffic and topology due to mobility of users or congestion [6] [3]. Moreover, the lack of considering the re-computation of VNF placements in these methods makes them ill-equipped to be employed in practical settings, and often the consequences are violations of the Quality of Service (QoS) and Service Level Agreement (SLA) [7]. Ongoing researches evolve around different optimization formulations using Integer Linear Programming (ILP) and Mixed Integer Linear Programming (MILP) to outline the VNF orchestration scheme, which is NP-hard by nature [8] [9], and fail to offer fast VNF placements decisions at different times. To approach intelligent VNF orchestration, meta-heuristic based swarm intelligence algorithm, specifically, Ant Colony Optimization (ACO), has been proposed in [9]. However, the family of swarm intelligence algorithms, including ACO, requires extensive parameter tuning of exploitation-exploration ratios, making them heavy weighted to accommodate for real-time orchestration [9]. In distinction to the existing works, we have aimed to propose lightweight and dynamic deep learning [10] aided strategies for the VNF orchestration and deployment that facilitates both users and services provides exclusively by collaborative minimization of communication, relocation delay and costs in real-time. We have considered two popularly known approaches, Convolutional Neural Networks [10] and Artificial Neural Networks [10] blended with the twist of ensemble training and prediction fashion [11].

978-3-903176-31-7

The pre-trained models can be placed on the cloudlets (local, device hosted or infrastructure based clouds) so the VNF orchestration process may be swift and prompt enough for delay-sensitive IoT applications [2].

The remainder of this paper has been structured as follows. We describe the literature review of the subject of our interest in Section II. Section III includes the explanation of the model that we have considered. The ILP model of the optimization framework for VNF deployment problem is exhibited in Section IV. Section V covers the discussion of the proposed deep learning aided strategies named, E-ConvNets and E-ANN. Next, simulation results using different performance metrics for all the placement strategies are represented in Section VI. Finally, Section VII concludes the paper.

II. Literature Review

A broad plethora of research studies related to VNF deployment in the hybrid cloud has been emerging lately [12] [13]. To enhance the Quality-of-Experience (QoE) of the users and for massive computation time critical applications, a large volume of communication resources are required. Thus, a lot of focus has been drawn towards resource constraints management based VNF deployment strategies [14] [15]. The authors of [15] have proposed distributed VNF deployment by caching resources, yet this mechanism is unable to manage dynamic network situations. Moreover, some researches focus solely on different VNF migration schemes [7]. Ben et al. have proposed a capacitated VNF migration scheme with the help of Virtual Network Embedding (VNE) [7]. However, the main focus has been deviated away from communication delay concerns that may affect the overall user experience. A stable matching algorithm to reduce the execution time by introducing mixed-integer programming and obtaining the near-optimal results in terms of latency has been approached [6] [16]. This algorithm can prevent the failure of the model in extreme cases. Even so, this model has been designed according to static network arrangements, which would require to be initiated every time instance eventually not being feasible for online applications. As a step closer towards intelligent VNF orchestration, swarm-based intelligence inspired by the natural behavior of ant colony has been approached [9], which considers both user mobility based VNF relocation and communication costs. However, the difficulty is that these types of algorithms need a lot of attention towards parameter tuning that require longer execution times to deal with large scale and real-time scenarios [17]. To the best of our knowledge, none of the existing works suggested the use of ensemble deep learning assisted strategies for VNF orchestration in order to function within reasonable running time limits, promote scalability, and support both the providers and users interest mutually, while approaching towards 6G cellular networks.

III. System Model

The system model includes two distinct domains. The cloudlet domain consists of a set of small scale data centers (DCs), D at the edge of the Internet, having secure and robust wired connections among them. On exploiting cloudlet confederation, the connected cloudlet DCs can offer and receive services from one another.

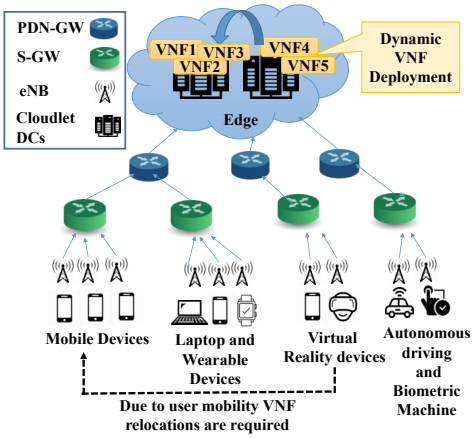


Fig. 1: A high level system architecture for VNF orchestration.

Besides, the domain of Radio Access Network (RAN) incorporates a set of access points, for example, base stations termed as evolved NodeB (eNB). A number of users can be connected to each eNB through radio signal. A base station controller usually manages a collection of eNBs. A single eNB is allowed to be connected to only one cloudlet DC via the Serving Gateway (S-GW) and Packet Data Network Gateway (PDN-GW) of the particular DC. However, numerous numbers of eNBs can be connected to a cloudlet DC. A set of eNBs, E can receive service from its associated DC. A data center can provide direct services by running the corresponding VNF of a client or user under the eNB connected to that cloudlet DC. Moreover, it can offer passive services via neighbouring DCs, which demands additional service cost. Due to user mobility, V_j can be considered as the set of VNFs of eNB, $e_j \in E$ that are required to be relocated. The requests for VNF migration usually occur because of hand off between an eNB, $e_j \in E$ and other eNBs that are connected to different cloudlet DCs unlike e_j. Table I exhibits the major notations along with their description used to implement the optimization framework.

IV. Optimization Framework for VNF Deployment

The ILP formulated optimization framework considered in this paper has been proposed in [9]. In order to deploy VNF, $v_i \in V_j$ of eNB, $e_j \in E$ to cloudlet DC, $d_k \in D$, the relocation time $R^i_{k,j}$ can be calculated using the following equation:

$$R^i_{k,j} = \{(1 - \epsilon^i_k) \times b^i_{k,j}\} \times Q^i_k \quad (1)$$

TABLE I: Description of parameters for the system model

Notation	Description
D=$\{d_1, d_2, ..., d_D\}$	The set of cloudlet DCs in the network
E=$\{e_1, e_2, ..., e_E\}$	The set of all eNBs connected to the cloudlet DC where the system is running
V=$\{v_1, v_2, ..., v_V\}$	The set of all VNFs
V_j	The set of VNFs of eNB, $e_j \in E$ that are required to be relocated, where $V_j \subseteq V$
ϑ_{worst}	Maximum communication delay toleration limit of the network
$t_{k,l}$	The communication delay between a cloudlet DC, $d_k \in D$ and the cloudlet DC, $d_l \in D$, provided that $k \neq l$
$t_{j,k}$	The communication delay between eNB, $e_j \in E$ and the cloudlet DC, $d_k \in D$
S_i	Size of VNF, $v_i \in V$
ϕ_k	Cost to place any VNF to some cloudlet DC, $d_k \in D$
ψ_k	Cost to take service from cloudlet DC, $d_k \in D$
C_k	Capacity of the cloudlet DC, $d_k \in D$ for holding VNFs
σ_i	Execution time of VNF, $v_i \in V$
η	Priority factor of VNF migration or relocation
τ_i	Transfer time of VNF, $v_i \in V$
N_k	Number of VNFs that are already executing in cloudlet DC, $d_k \in D$
$\Upsilon^i_{k,j}$	Summation of communication, relocation, and execution time if VNF, $v_i \in V_j$ of eNB, $e_j \in E$ is placed at cloudlet DC, $d_k \in D$

where, ϵ^i_k holds 1, if the VNF instance have been earlier running on cloudlet, DC $d_k \in D$, otherwise 0. Therefore, in case $(1 - \epsilon^i_k)$ is 1, the corresponding VNF instance can be considered for relocating to a cloudlet, DC $d_k \in D$. Likewise, the decision variable $b^i_{k,j}$ holds 1 if VNF, $v_i \in V_j$ of eNB, $e_j \in E$ is placed at cloudlet DC, $d_k \in D$, otherwise 0. The relocation cost to migrate VNF, $v_i \in V_j$ of eNB, $e_j \in E$ to some cloudlet DC, $d_k \in D$ can be represented by Q^i_k and calculated as follows:

$$Q^i_k = (1 - n^i_k) \times \tau_i \tag{2}$$

Again, n^i_k holds 1, if the expected VNF, $v_i \in V_j$ of eNB, $e_j \in E$ is running on the cloudlet DC, $d_k \in D$, otherwise 0. Therefore, upon the value of $(1 - n^i_k)$ being 1, we need to relocate or transfer the VNF from the previous DC. In such case, the relocation cost is equal to transfer time τ_i, which can be calculated using the following equation:

$$\tau_i = \frac{S_i}{r} \tag{3}$$

where, r is the achievable data rate to relocate any VNF and S_i represents the size of the VNF, v_i. The communication delay to get service for a VNF, $v_i \in V_j$ is calculated from the following equation:

$$T^i_{k,j} = b^i_{k,j} \times (t_{j,k} + t_{k,l}) \tag{4}$$

Here, the summation of communication delay between eNB, $e_j \in E$ and the cloudlet DC where the solution is executing $(t_{j,k})$ along with the communication delay between cloudlet DC, $d_k \in D$ and the cloudlet DC, $d_l \in D$, which holds the running VNF $(t_{k,l})$ contribute to the total communication delay. Finally, the objective function formulation using ILP can be presented as the following:

$$min \sum_{e_j \in E} \sum_{v_i \in V_j} \sum_{d_k \in D} \{\eta \times R^i_{k,j} \times \phi_k + (1-\eta) \times T^i_{k,j} \times \psi_k\} \tag{5}$$

A trade-off is introduced by estimating the priority factor denoted as η in the objective function. The objective is to minimize the overall network relocation and communication costs of the network. The objective function presented in the Eq. (5) is subject to the following constraints:

$$\mathcal{C}1 : \sum_{d_k \in D} b^i_{k,j} = 1, \ \forall_{e_j \in E}, \ \forall_{v_i \in V_j} \tag{6}$$

$$\mathcal{C}2 : \sum_{v_i \in V_j} \sum_{d_k \in D} b^i_{k,j} = |V_j|, \ \forall_{e_j \in E} \tag{7}$$

$$\mathcal{C}3 : \sum_{d_k \in D} (R^i_{k,j} + T^i_{k,j} + \sigma_i) \leq \vartheta_{worst}, \ \forall_{e_j \in E}, \ \forall_{v_i \in V_j} \tag{8}$$

$$\mathcal{C}4 : \sum_{e_j \in E} \sum_{v_i \in V_j} b^i_{k,j} \leq C_k, \ \forall_{d_k \in D} \tag{9}$$

$$\mathcal{C}5 : b^i_{k,j} \in \{0, 1\}, \ \forall_{e_j \in E}, \ \forall_{v_i \in V_j}, \ \forall_{d_k \in D} \tag{10}$$

The constraint $\mathcal{C}1$ is basically an atomicity constraint, ensuring the single assignment of each VNF, $v_i \in V_j$ of eNB, $e_j \in E$ to exactly one cloudlet DC, $d_k \in D$. Another constraint $\mathcal{C}2$ specifies that all VNFs, $v_i \in V_j$ of eNB, $e_j \in E$ must be allocated to some cloudlet DC, $d_k \in D$. Next, The QoS constraint $\mathcal{C}3$ guarantees the summation of relocation delay, communication delay, and execution time of VNFs to remain below a certain pre-defined threshold ϑ_{worst}, which can be varied according to application nature. The capacity constraint $\mathcal{C}4$ assures not to overload cloudlet DCs. Hence, the number of VNFs executing in a cloudlet DC is not allowed to exceed the capacity of that cloudlet DC. Finally, the constraint $\mathcal{C}5$ is a binary constraint representing the value of decision variable $b^i_{k,j}$ to be 1, in case VNF, $v_i \in V_j$ of eNB, $e_j \in E$ is placed at cloudlet DC, $d_k \in D$, otherwise remains 0.

V. Proposed Deep Learning Aided VNF Deployment

The future of cellular networks and NFV infrastructure manager expect to exploit AI for offering intelligent orchestration and management systems [1]. The concept is to locate the pre-trained models in cloudlet DCs so that the deployment decisions induced by the testing phase can offer real-time solutions with ultra-low execution time required for prediction.

A. Labeled Dataset Generation

We leverage the ILP optimization framework solver described in section IV to optimize different VNF deployment scenarios, and then record the respective solutions to generate labeled data for training purpose. We merge the features along with the decision variable $b^i_{k,j}$ found by ILP solver to produce a labeled dataset for training purpose in lines 5-7 of algorithm 1. We denote the labeled dataset as $\mathcal{L}$.

Algorithm 1: Ensemble training phase of deep learning aided VNF Deployment at each cloudlet DC $d_k \in D$

Input: D, E, V_j, ϑ_{worst}, $R^i_{k,j}$, $T^i_{k,j}$, ϕ_k, ψ_k, C_k, σ_i, η, τ_i, and N_k
Result: Set of trained models $\mathcal{M}^*$

1 $\mathcal{M}^* \leftarrow \varnothing$
2 **foreach** *model* $m_t \in \mathcal{M}$ **do**
3 $\mathcal{L} \leftarrow \varnothing$
4 **for** $epoch <$ *total simulation epochs* **do**
5 $\mathcal{S} \leftarrow$ Generate a random system using input parameters
6 $b^i_{k,j} \leftarrow$ Assign the decision variable by solving system $\mathcal{S}$ through the ILP optimizer framework
7 $\mathcal{L} \leftarrow \mathcal{S} \cup b^i_{k,j}$
8 Train the model m_t using labeled dataset $\mathcal{L}$
9 $\mathcal{M}^* \leftarrow \mathcal{M}^* \cup m_t$

B. Ensemble Convolutional Neural Networks (E-ConvNets)

To generate a well-calibrated model due to the uncertain nature of the network parameters, we have incorporated the ensembling technique into the model utilizing E-ConvNets [11]. The E-ConvNets method consists of a set of alternative different convolutional network models $\mathcal{M}$. Each model $m_t \in \mathcal{M}$ is trained by different randomly generated datasets as explained in lines 2-7 of algorithm 1. Finally, at the end of this algorithm as suggested in line 8, we receive a set of trained ensemble models $\mathcal{M}^*$ that are further used for deploying VNF $v_i \in V_j$ of eNB $e_j \in E$ to some cloudlet DC $d_k \in D$ using algorithm 2. The testing or prediction phase of ensemble techniques for E-ConvNets have been exhibited in algorithm 2. We have generated random unlabeled data $\mathcal{U}$ for performance evaluation in line 1. In lines 5-11, we verify the constraints and apply trained models $m_t \in \mathcal{M}^*$ exploiting E-ConvNets on unlabeled dataset $\mathcal{U}$ to generate the confidence scores for placing all VNF, $v_i \in V_j$ of every eNB, $e_j \in E$ to each cloudlet DC, $d_k \in D$. We update this confidence score obtained by each model in variable $\hat{\mathcal{Y}^i_{k,j}}$ that can range between 0.0 and 1.0. We accumulate the cumulative prediction confidence score of all trained models for deploying VNFs, $v_i \in V_j$ of eNB, $e_j \in E$ to cloudlet DC, $d_k \in D$, and store it in the variable $\mathcal{X}^i_{k,j}$ through line 12. In lines 14-18, we select the cloudlet DC, $d_k \in D$ that holds the highest cumulative confidence score generated by all trained models for every VNF, $v_i \in V_j$ of eNB, $e_j \in E$ pair. Finally, a set of solutions $\mathcal{F}$ is constructed iteratively containing respective eNB, $e_j \in E$, VNF, $v_i \in V_j$, and selected cloudlet DC, $d_k \in D$ in line 19. The sequential styled E-ConvNets feature a set of typical model designs [10]. Each model includes several convolutional layers that are followed by a pooling layer. Batch normalization is performed to enhance the performance, speed, and stability of models, thus require less computational complexity. For the convolutional layers, we select rectified linear (ReLU) activation function, while output layers use softmax function by following the cross-entropy loss function.

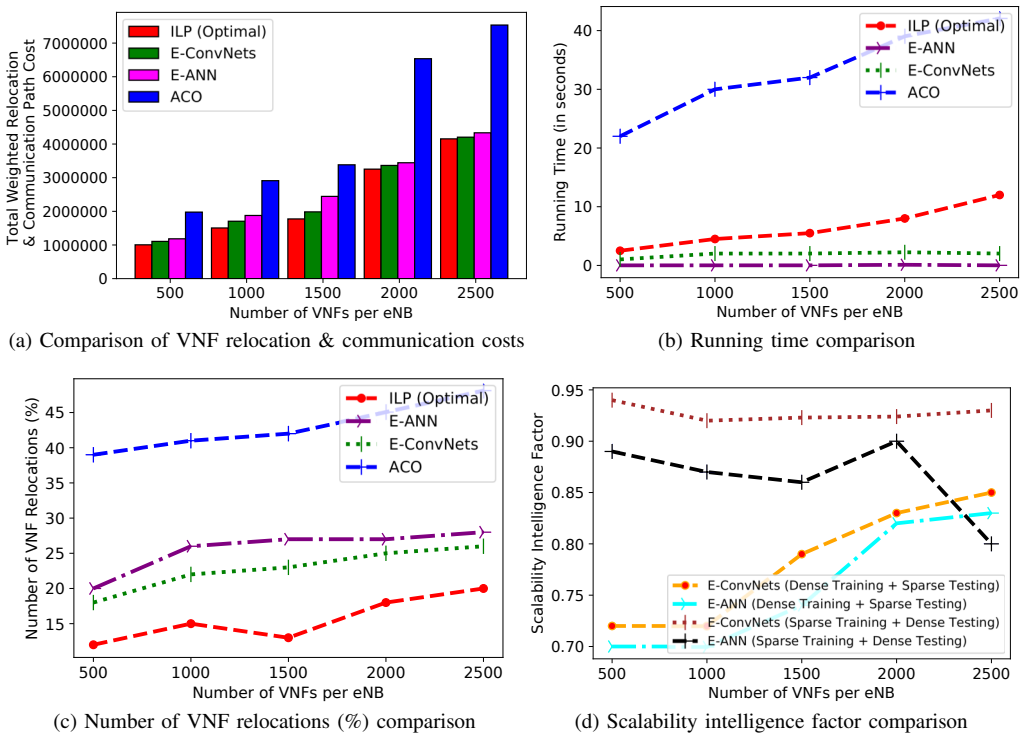
Algorithm 2: Ensemble testing phase of deep learning aided VNF Deployment at each DC $d_k \in D$

Input: D, E, V_j, ϑ_{worst}, $R^i_{k,j}$, $T^i_{k,j}$, ϕ_k, ψ_k, C_k, σ_i, η, τ_i, and N_k
Result: A set of solutions $\mathcal{F}$

1 $\mathcal{U} \leftarrow$ Generate a random system using input parameters
2 $\mathcal{F} \leftarrow \varnothing$
3 $\mathcal{X}^i_{k,j} \leftarrow 0$
4 **foreach** *trained model* $m_t \in \mathcal{M}*$ **do**
5 **foreach** *eNB* $e_j \in E$ **do**
6 **foreach** *VNF* $v_i \in V_j$ **do**
7 **foreach** *cloudlet DC* $d_k \in D$ **do**
8 **if** ($N_k > C_k$ ***and*** $\Upsilon^i_{k,j} > \vartheta_{worst}$) **then**
9 $\hat{\mathcal{Y}^i_{k,j}} \leftarrow 0$
10 **else**
11 $\hat{\mathcal{Y}^i_{k,j}} \leftarrow$ Set the confidence score between 0 and 1 by applying trained model m_t on $\mathcal{U}$
12 $\mathcal{X}^i_{k,j} \leftarrow \mathcal{X}^i_{k,j} + \hat{\mathcal{Y}^i_{k,j}}$
13 **foreach** *eNB* $e_j \in E$ **do**
14 **foreach** *VNF* $v_i \in V_j$ **do**
15 $temp_k \leftarrow \varnothing$
16 **foreach** *DC* $d_k \in D$ **do**
17 $temp_k \leftarrow temp_k \cup \mathcal{X}^i_{k,j}$
18 $max \leftarrow \underset{k}{\operatorname{argmax}}\ temp_k$
19 $\mathcal{F} \leftarrow \mathcal{F} \cup (e_j, v_i, d_{max})$

C. Ensemble Artificial Neural Networks (E-ANN)

The Artificial Neural Networks (ANN) [10] is a paradigm for processing information and usually configured according to the requirement of applications through the learning phase. Our proposed E-ANN consists of multiple models $m_t \in \mathcal{M}$. In order to train the E-ANN models, we apply the algorithm 1 on some randomly generated labeled data $\mathcal{L}$, as explained earlier in the subsection V-A. The output layer of the E-ANN has equal number of nodes specifying cloudlet DCs for every model of the ensemble learning. The output nodes indicate the probability or confidence score of placing a VNF to each cloudlet DC. Finally, we place a VNF on the cloudlet DC of the corresponding output node having the highest cumulative confidence score summed up from all the trained models. We employ the same algorithm 2 for the prediction phase of VNF deployment, while applying the E-ANN architecture. Instead of employing a single architecture of ANN, we utilize a set of trained models that altogether contribute to the final outputs. We assemble these models to accumulate the confidence score of each output node to interpret the ultimate set of output nodes for proper calibration [11]. The final layer of the E-ANN architecture utilizes softmax function under cross-entropy loss regime, while the nodes from hidden layer employ hyperbolic tangent function [10].



(a) Comparison of VNF relocation & communication costs

(b) Running time comparison

(c) Number of VNF relocations (%) comparison

(d) Scalability intelligence factor comparison

Fig. 2: Comparison of the performance impacts of different VNF deployment strategies for varying number of VNFs under each eNB for 12 data centers in total.

VI. Performance Evaluation

To solve the ILP formulation of the problem, we have used the Gurobi optimization solver. Python's TensorFlow libraries have been utilized to support the experiments concerning our proposed deep learning-based approaches (E-ConvNets and E-ANN).

A. Simulation Environment

We have studied a network consisting of 12 cloudlet DCs. The number of eNBs under each cloudlet DC can differ in the range of 5 - 25, while the number of VNFs under each eNB can be between 500 - 2500. The communication delay between different pairs of cloudlet DCs can vary between 10 - 200 milliseconds. However, to get service from the cloudlet DC directly connected to the respective eNB, a trivial amount of time ranging between 2 - 5 milliseconds has been considered. We assume the data rate of transferring VNFs between distinct pairs of cloudlet DCs to be 1 - 50 Mbps, and the size of VNFs are allowed from 100 to 300 KB. The value of priority factor η has been selected as 0.7. All the mentioned network parameters have been adapted from the existing literature [9]. For the experimental purpose, we have used three hidden layers each containing 156 nodes, and the number of output nodes being equal to the number of the cloudlet DCs in the system in case of E-ANN. Additionally, the number of convolution layers have been considered as 5. The dropout rate have been selected as 0.3. These hyperparameters have been selected through simulation study and tuning. We have used the same hyperparameters to compile and train each model in order to retain simplicity.

B. Performance Metrics

The performance metrics analyzed for the performance evaluation of different VNF deployment strategies have been described in the following:

Total Weighted VNF Relocation and Communication Costs: This metric is the interpretation of objective function value mentioned in the equation Eq. (5).

Running Time: By reporting the execution times of algorithms, we can recognize how quickly a VNF placement method can extend its orchestration services to the users.

Number of VNF Relocation: The total number of VNF relocations required to migrate the VNFs to the selected cloudlet DC is defined by this metric, which directly impacts the administration of the whole network.

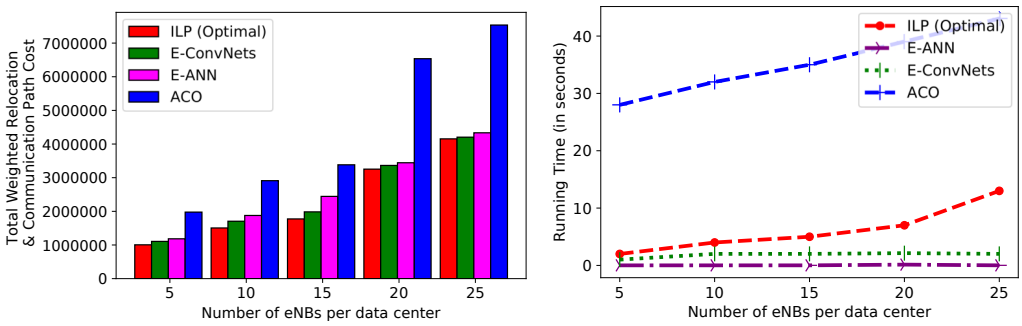
Scalability Intelligence Factor: To calculate this metric, we have trained the models on dense and sparse networks individually and tested on the other way around to illustrate how much the performance of these models deviate from the optimization framework. Then, we have normalized the resultant metric values between 0 and 1. To generate a sparse network system, we have varied the communication delay 10 times more than the dense networks.

C. Results and Discussion

We have categorized the analysis of the results into following two kinds:

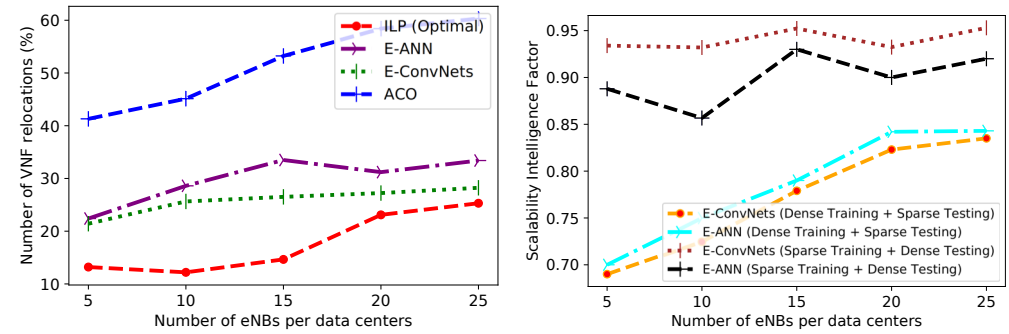
1) Varying number of VNFs under each eNB

Fig. 2 represent the effects on the mentioned performance metrics associated with the experiment of varying the number



(a) Comparison of VNF relocation & communication costs

(b) Running time comparison



(c) Number of VNF relocations (%) comparison

(d) Scalability intelligence factor comparison

Fig. 3: Comparison of the performance impacts of different VNF deployment strategies for varying number of eNBs under each data center for 12 data centers in total.

of VNFs under each eNB. From Fig. 2a, it can be easily seen that with the growing number of VNFs, the performance of ACO based deployment strategy declines in terms of relating to the ILP formulation unlike the smaller instances of VNFs. However, the E-ConvNets and E-ANN continue to maintain their performances through the increasing number of VNFs. E-ConvNets exhibit the most promising objective function value resembling the ILP formulation for all the cases. The running time of E-ANN has been shown to be the lowest in 2b, and the E-ConvNets manifest very similar execution time as well. Contrarily, the traditional ILP and ACO-based approaches require higher running time to take VNF deployment decisions comparatively that is not suitable for latency sensitive real-time IoT applications. The number of VNF relocations affect the migration overhead of networks that have been presented in the Fig. 2c. E-ConvNets and E-ANN incur migrations ranging between around 15% - 30%. However, ACO based placement strategy induces relocations above 40%, while the optimal percentage of relocations shown by ILP remains approximately 10% - 20% for different numbers of VNFs. For future networks, orchestration systems demand scalability. Hence, to support the experiments of E-ConvNets and E-ANN, we train the models in different settings of sparse and dense networks and test their performance on vice-versa. Fig. 2d illustrates that both of the deep learning models perform significantly well when they are trained using a sparse network.

2) Varying number of eNBs under each cloudlet DC

Fig. 3 illustrates the performance impacts due to the varying the numbers of eNBs from 5 to at most 25 under each cloudlet DC, while keeping the number of VNFs fixed at 1000. The results found in this experiment somewhat resemble the ones found in the earlier simulations represented in this paper.

VII. Conclusion

The conventional optimization techniques due to the various drawbacks, mostly lacking agility, fail to be qualified for real-time adaptions in dynamic network perspectives. Thus, we have stressed on designing a prompt technique for intelligent networks to proactively assign VNFs to the edge cloudlets DCs with best possible relocation and communication costs as the outcome, while considering the predictable rapid growth of IoT services in near future. For the sake of the model calibration process, we have considered utilizing multiple models instead of relying on a single one for the training and prediction phase. Hence, we have applied E-ConvNets and E-ANN in simulated network arrangements and compared the results with other existing conventional approaches. Experimental results suggest that E-ConvNets outperforms all other methods in terms of minimizing costs and relocation burden with significantly improved scalability intelligence factor. Although, E-ANN performs best according to running time, yet being very close to execution times of E-ConvNets.

REFERENCES

[1] W. Saad, M. Bennis, and M. Chen, "A vision of 6g wireless systems: Applications, trends, technologies, and open research problems," *IEEE Network*, vol. 34, no. 3, pp. 134–142, 2020.

[2] R. Cziva, C. Anagnostopoulos, and D. P. Pezaros, "Dynamic, latency-optimal vnf placement at the network edge," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, Honolulu, HI, USA, April 2018, pp. 693–701.

[3] X. Fei, F. Liu, H. Xu, and H. Jin, "Adaptive vnf scaling and flow routing with proactive demand prediction," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, Honolulu, HI, USA, April 2018, pp. 486–494.

[4] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, "Survey on multi-access edge computing for internet of things realization," *IEEE Communications Surveys Tutorials*, vol. 20, no. 4, pp. 2961–2991, Fourthquarter 2018.

[5] M. Gharbaoui, C. Contoli, G. Davoli, G. Cuffaro, B. Martini, F. Paganelli, W. Cerroni, P. Cappanera, and P. Castoldi, "Demonstration of latency-aware and self-adaptive service chaining in 5g/sdn/nfv infrastructures," in *2018 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Verona, Italy, Nov 2018, pp. 1–2.

[6] K. S. Ghai, S. Choudhury, and A. Yassine, "A stable matching based algorithm to minimize the end-to-end latency of edge nfv," *Procedia Computer Science*, vol. 151, pp. 377 – 384, 2019.

[7] H. Ben-Ammar, Y. Hadjadj-Aoul, and S. Ait-Chellouche, "Efficiently allocating distributed caching resources in future smart networks," in *2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*, Jan 2019, pp. 1–4.

[8] M. R. Garey. and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1990.

[9] P. Roy, A. Tahsin, S. Sarker, T. Adhikary, M. A. Razzaque, and M. Hassan, "User mobility and quality-of-experience aware placement of virtual network functions in 5g," *Computer Communications*, vol. 150, pp. 367–377, 12 2019.

[10] Q. Mao, F. Hu, and Q. Hao, "Deep learning for intelligent wireless networks: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 20, no. 4, pp. 2595–2621, Fourthquarter 2018.

[11] H. Ge, F. Jiang, and Z. Zhang, "A hybrid localization algorithm of rss and toa based on an ensembled neural network," in *2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, Chongqing, China, May 2019, pp. 1280–1284.

[12] C. Zhang, H. P. Joshi, G. F. Riley, and S. A. Wright, "Towards a virtual network function research agenda: A systematic literature review of vnf design considerations," *Journal of Network and Computer Applications*, vol. 146, p. 102417, 2019.

[13] S. Ayoubi, S. Sebbah, and C. Assi, "A cut-and-solve based approach for the vnf assignment problem," *IEEE Transactions on Cloud Computing*, pp. 1–1, 06 2017.

[14] J. Fu and G. Li, "An efficient vnf deployment scheme for cloud networks," pp. 497–502, June 2019.

[15] X. Chen, Z. Zhu, J. Guo, S. Kang, R. Proietti, A. Castro, and S. J. B. Yoo, "Leveraging mixed-strategy gaming to realize incentive-driven vnf service chain provisioning in broker-based elastic optical inter-datacenter networks," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 10, no. 2, pp. 232–240, Feb 2018.

[16] K. S. Ghai, S. Choudhury, and A. Yassine, "Efficient algorithms to minimize the end-to-end latency of edge network function virtualization," *Journal of Ambient Intelligence and Humanized Computing*, 01 2020.

[17] M. Abu-Lebdeh, D. Naboulsi, R. Glitho, and C. W. Tchouati, "On the placement of vnf managers in large-scale and distributed nfv systems," *IEEE Transactions on Network and Service Management*, vol. 14, no. 4, pp. 875–889, Dec 2017.