Fog and Edge Service Migration Approaches based on Machine Learning Techniques : A Short Survey

Nour El Houda Boubaker[1], Karim Zarour[1], Nawal Guermouche[2] and Djamel Benmerzoug[1]

[1]Constantine2 - Abdelhamid Mehri University, LIRE Laboratory, Constantine, Algeria
[2]LAAS-CNRS, University of Toulouse, INSA

Abstract

Service migration in Fog and Edge computing is a promising approach to avoid service interruption and improve quality of service (QoS) for users. However, finding optimal migration decisions in a highly dynamic environment is one of the challenging issues in the literature. This paper provides a short review of migration approaches using Machine Learning techniques. These approaches are studied and classified based on various aspects such as migration type and the optimized QoS metrics identified by the proposed taxonomies. Furthermore, different research questions are discussed and the main challenges in this field are explored.

Keywords

Service Migration, Fog, Edge, Cloud, QoS, Machine Learning

1. Introduction

Recently, Edge Computing, including its extension Mobile Edge Computing (MEC), and Fog Computing have emerged as promising paradigms to reduce the communication latency significantly by providing proximal offloading of Internet of Things (IoT) applications. The concept of Fog Computing has great similarity to Edge Computing. Both of the paradigms construct themselves on the edges of the network near data sources [1]. OpenFog Consortium makes the distinction that Fog Computing is hierarchical and it provides computing, networking, storage, control, and acceleration anywhere from Cloud to things while Edge Computing tends to be limited to computing at the edge [2].

However, the mobility of end-users and the limited coverage of MEC and Fog nodes can result in considerable network performance degradation and lower QoS support [3]. The service migration mechanism has a great potential to solve these issues by determining when, where, and how to migrate services from a node to another node[4].

Although many works [5, 6, 7] have been proposed to handle service migration in Edge-Fog-Cloud, only very few surveys focus on covering these studies. For instance in [8], service migration approaches in MEC are summarized based on many aspects such as strategies for service migration. The authors in [9] reviewed only migration approaches brought by the mobility

Tunisian Algerian Conference on Applied Computing (TACC 2022), December 13 - 14, Constantine, Algeria
nour.boubaker@univ-constantine2.dz (N. E. H. Boubaker); zarour.karim@univ-constantine2.dz (K. Zarour); nawal.guermouche@laas.fr (N. Guermouche); djamel.benmerzoug@univ-constantine2.dz (D. Benmerzoug)

of users. However, none of the existing studies focus on investigating the predictive migration approaches using the different techniques of Machine Learning (ML) including, Supervised Learning (SL), Deep Learning (DL), Reinforcement Learning (RL), and Deep Reinforcement Learning (DRL). To the best of our knowledge, this paper is the first that covers these aspects.

The remainder of this paper is organized as follows. Section 2 introduces the research methodology. Section 3 presents a classification of the reviewed approaches. Section 4 provides answers to the defined research questions. Finally, Section 5 concludes the survey.

2. Research Methodology

In this section, we present the followed steps for searching and filtering papers.

2.1. Research Questions

Since covering ML migration approaches is our main concern, we formulated the research questions as follows:

- Q1) What is the branch of Machine Learning mostly used to deal with migration problems?
- Q2) Are these migration approaches applied to entire workflow /application or at the service/task level ?
- Q3) What are the QoS metrics mostly considered by migration approaches?
- Q4) What is the migration environment mostly adopted in the literature?
- Q5) What are the domains of application related to the migration problem?
- Q6) What are the main challenges in this field?

2.2. Papers Selection

Firstly, the searching step was done using the following search string, which was used to query various scientific databases, such as IEEE, ResearchGate, Springer, and Elsevier:

Migration AND (Service OR Task Or Application) AND (Fog OR Edge OR Cloud) AND (Machine Learning OR Supervised Learning OR Deep Learning OR Reinforcement Learning OR Deep Reinforcement Learning).

Then, we conducted a filter step by applying the following exclusion and inclusion criteria:

- Including only papers from 2019 to 2022 because the different ML techniques have been widely used to solve migration problems since 2019.
- Excluding studies that do not focus on migration problems.
- Excluding studies that combine the techniques of ML with other fields (e.g. Combinatorial Optimization) as our main objective is to exclusively review ML-based approaches.

As a result, we obtained 20 papers. Fig. 1 captures plainly the repartition per year of the studied works.

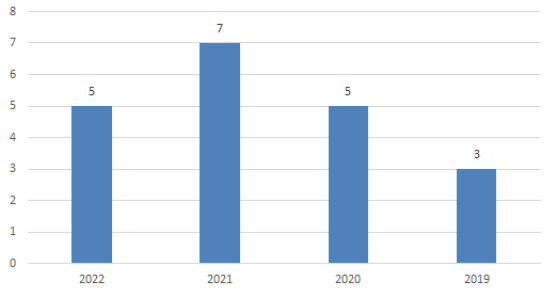

Figure 1: The number of reviewed papers per year

3. Migration Approaches Comparison

In order to answer the research questions 2.1, we have to compare the ML migration approaches. To do so, we have to identify the different criteria and aspects of classification.

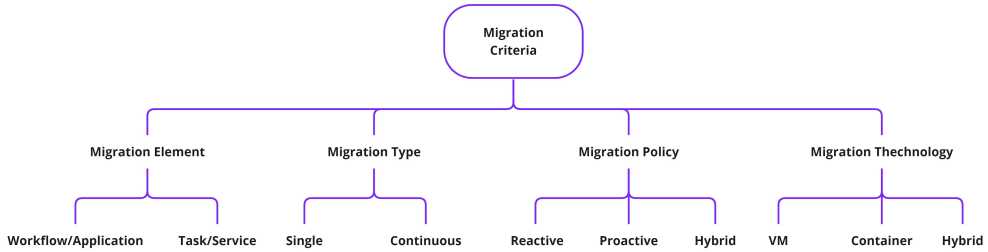

Figure 2: Migration Criteria

Figure 2 depicts the different migration criteria. We consider four main aspects, which are:

1. **Migration Element**: This aspect identifies the nature of the element concerned by the migration decision. We could identify two types, which could be either the entire application/workflow or a partial element of the application/workflow that may be a service or task.

2. **Migration Type**: This refers to the nature of migration. A single migration is generally considered in works when a task/service is offloaded from the end-device to the migration environment (Edge, Fog, or Cloud) for its execution. On the other hand, continuous migration occurs when the considered element is migrated multiple times due to many reasons, such as the continuous mobility of end-user or Edge-Fog nodes and the dynamic change in request pattern.

3. **Migration Policy**: This indicates the timing for performing the migration decision. In reactive policy, the migration is subject to change only after the system enters an undesirable state in terms of QoS degradation. On the other hand, the proactive policy anticipates the forthcoming disruptions in advance using generally predictive techniques and performs the migration decision before the system enters the undesirable state. Some

approaches combine the two policies, such as[10], who aim to efficiently balance reactive and proactive service migration decision making.

4. **Migration Technology**: This determines the mechanism used for migrating the element from one node to another in Edge/Fog/Cloud environments. In the literature, there are two dominant virtualization technologies: Virtual Machines (VMs) and containers. VM has been explored to move a service from one resource to another to support user mobility [11]. On the other hand, container as a lightweight virtualization technique has a lower management cost than VMs and performs much better on service migration process [12].

Table 1 depicts the classification of the approaches based on migration element, migration type, migration policy, the technology used for migration, and domains of application.

Table 1
Classification of migration approaches

Criteria		Works
Migration Element	Service/Task	[13],[14],[15],[16],[12],[17],[18],[19],[20],[21],[22],[23],[24],[25],[26],[27],[10]
	Workflow/ Application	[28] ,[29],[30]
Migration Type	Single	[13],[14],[15],[28]
	Continuous	[16],[12],[17],[18],[19],[20],[29],[21],[22],[23],[24],[25],[26],[27],[10],[30]
Migration Policy	Proactive	[16],[19],[23],[25]
	Hybrid (Proactive-Reactive)	[10]
Migration Technology	Virtual Machine (VM)	[12],[19],[21],[25]
	Container	[28] ,[22],[23],[30],[25]
	Hybrid	[17],[20]
domains of application	Vehicular	[16],[18],[20],[27],[24]
	Mobile	[17],[12],[19] ,[21], [25], [26]
	IoT	[28] [29],[22], [23],[30],[15]
	Industrial IoT (IIoT)	[13]
	Smart Healthcare	[14],[10]

Table 2 captures some features related to continuous migration. Continuous user mobility is usually considered when a user moves from a zone covered by a base station to another, and at each move step, the task/service has to be migrated to follow the end device. The user path could be defined or predicted in advance using ML techniques. On the other hand, very few studies have worked on continuous Fog mobility caused by the continuous movement of Fog nodes. For instance, in [30], the application has to be migrated to another Fog node whenever the hosted node in the current Fog domain becomes unavailable due to its mobility. Finally, in the dynamic change in request (demand) pattern, the task/service must be migrated according to the dominant request locations.

Table 2
Continuous Migration Features

Continuous Migration Features	Works
Continuous User Mobility	Predicted User Path [19] ,[25] Known User Path [16],[12],[18],[20],[29],[21],[22],[24],[26],[27],[10]
Continuous Fog Mobility	[30]
Changing in demands	[17],[18],[23]

To handle a predictive and a proactive migration, ML techniques have been mainly used. We could classify ML based approaches according to four parameters captured in Figure 3, which are:

1. **Migration Environment**: It specifies the target environment in which the migration element has to be placed. Single layer destination (SLD) refers to considering the resources of one specific layer for the migration approach, which might be Edge, Fog, or Cloud. On the other side, in multiple layers destination (MLD), the migration decision takes into account various layers, and according to resources/network states and task requirements, the most appropriate layer would be selected for migrating the service/task.

2. **Migration Nodes**: It identifies the nature of the considered resources in the migration environment. Mobile nodes are resources that may change their location in time, such as mobile devices, laptops, and vehicles. In contrast, static nodes can not change their location , as instance servers. Finally, hybrid nodes refer to considering an environment composites of static and mobile nodes.

3. **ML Technique**: This aspect identifies the different ML categories used in the literature for solving the migration problem. We distinguish four categories, which are:
   (a) **Supervised Learning**: In this category, the algorithms are given a labeled training dataset to build the system model representing the learned relation between the input and output. After training, when a new input is fed into the system, the trained model can be used to get the expected output[31]. SL approaches usually use features related to the history of tasks performed by several nodes, such as the computational capacities consumed and the processing time taken. Thereafter, the suitable node for offloading is estimated as in[15].
   (b) **Deep Learning**: It is a sub-field of machine learning that uses artificial neural networks (ANNs) containing two or more hidden layers to approximate some function that can be used to map input data to new representations or make predictions[32].
   (c) **Reinforcement Learning/Deep Reinforcement Learning**: They are self-learning techniques in which no prior knowledge of the environment is necessary. An agent learns the optimal behavior known as policy by interacting with the environment. At each decision step, the agent observes the state of the environment, takes an action, and receives a scalar reward value from the environment. Using this reward value, the agent adjusts its policy in order to maximize the long-term reward[33].
   (d) **Hybrid**: It regroups the approaches that combine the techniques of the previous categories to solve the migration problem. Generally, in this category, the techniques of SL/DL are used to predict the state of resources/network or the path of a user in advance. Next, the techniques of RL/DRL are applied for selecting the optimal resource for migration.

4. **QoS Metrics Type**: It determines the class of QoS metrics that are intended for optimization by migration approaches. We could identify three classes, which are:

(a) **Computation Node-Centric (CNC)**: This class aims to optimize metrics related to Edge/Fog/Cloud resources, as for instance:

- **Execution Time/Cost/ Energy**: They occur when a task is performed by the computation node.
- **Resource Utilization**: It is defined as the ratio between the resources that any service will consume and the available resources at the edge node[18].
- **Load Balancing**: This metric determines whether the computing load is distributed fairly among the computation node in the system[24].

(b) **Network-Centric (NC)**: It seeks to optimize network metrics. We cite:

- **Migration Time/Cost/Energy**: They occur when a task is migrated in network from node to another.
- **Communication Time/Cost/ Energy**: They occur when a task data is transmitted between end-user and their connected Edge/Fog nodes .
- **Network Throughput**: It indicates the amount of data moved successfully from one node to another in a given time period.

(c) **Joint-Centric (JC)**: It regroups the metrics that take into consideration optimizing the performance of nodes and network at the same time, among them:

- **Delay(Latency)/Cost/Energy**: In migration approaches, they refer to the summation of migration with execution metrics. Some approaches include the communication metric [13].

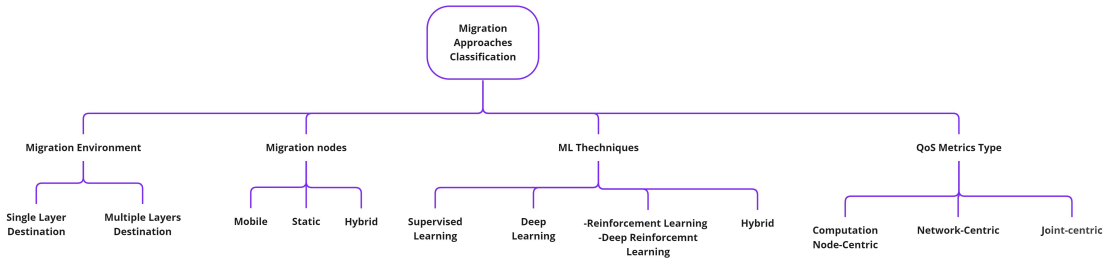

Figure 3: Migration Approaches Classification Taxonomy

Table 3 provides a comparison in terms of migration environment, migration nodes, ML category, used technique, algorithm name, and the optimized QoS metrics. We used the following acronyms for ML techniques:

RNN (Recurrent Neural Network), LSTM (Long Short Term Memory), Multivariate Linear Regression (MLR), Polynomial Multivariate Regression (PMR), Random Forest Regression (RFR), Support Vector Regression (SVR), Deep Q Network (DQN), Double Deep Q Network (DDQN), Deep Deterministic Policy Gradient (DDPG).

Table 3
Comparison of migration approaches

Works	Migration environement	Migration nodes	ML Category	Technique	Algorithm Name	QoS Metrics
[15]	Fog/Cloud (MLD)	Static	SL	Logistic Regression		-Latency (JC) -Energy Consumption (JC) -load balancing (CMC) -Operational Cost (CNC)
[28]	Cloud to Fog (SLD)	Static	SL	-MLR -PMR -RFR -SVR		Offloading Time (NC)
[14]	Edge/Cloud (MLD)	Static	RL	Q-Learning	CORL	-Latency (JC) -Energy Consumption (JC)
[13]	Edge/Cloud (MLD)	Static	DRL	Soft actor-critic	ISAC-CPTORA	-Delay (JC) -Energy Consumption (JC)
[16]	Fog (SLD)	Static	DL	RNN-LSTM		-Latency (JC) -Cost (JC)
[12]	MEC (SLD)	Static	RL	Q-learning	Mig-RL	Cost (JC)
[17]	MEC (SLD)	Static	DRL	DQN	SMDQN	-Delay (JC) -Migration Cost (NC)
[18]	MEC (SLD)	Static	DRL	actor-critic	DRLD-SP	-Delay (JC) -Resource Usage (CNC)
[19]	MEC (SLD)	Static	Hybird DL-DRL	-Seq2Seq -DQN	-Glimpse Mobility Prediction - M-DRL	Latency (JC)
[20]	MEC (SLD)	Static	RL	DDQN	DQL	-Delay (JC) -Migration Cost (NC)
[29]	Edge/Cloud (MLD)	Static	DRL	DDPG	DTASM	-Latency (JC) -Load forwarded to the cloud (NC)
[21]	MEC (SLD)	Static	RL	Q-learning	RLSMS	-Delay (JC) -Cost (JC)
[22]	MEC (SLD)	Static	DRL	DDPG	AWDDPG	-Migration Cost (NC) -Delay (JC)
[23]	Fog (SLD)	Static	DRL	DQN	IFSP	Maximizing the number of satisfied requests served in delay (JC)
[24]	MEC (SLD)	Static	DRL	DQN	Deep Q-learning	- Load balancing (CNC) -migration cost (NC)
[25]	MEC (SLD)	Static	Hybird DL-DRL	-RNN-based LSTM -Q-learning	RLSM	-Latency (JC) -Network Throughput (NC)
[26]	MEC (SLD)	Static	DRL	DQN		-Migration Cost (NC) -Transaction Cost (NC) -Migration Energy Consumption (NC)
[27]	MEC (SLD)	Static	RL	Q-learning	MS-Q	-Delay (JC) -Cost (JC)
[10]	Fog (SLD)	Static	DRL	DDPG	DDPG based schemes	-Latency (JC) -Energy Consumption (JC)
[30]	Fog (SLD)	Mobile	DRL	DQN	DRL based solution	Maximizing the number of satisfied user requests (JC)

4. Discussion

In this session, we answer the questions relieved in the subsection 2.1.

Q1) What is the branch of Machine Learning mostly used to deal with migration problems?

From Fig. 4, it is observed that the DRL category has the highest percentage with 50%, followed by RL with 25%. In the literature, the RL agent has proven its capabilities of learning and rapid decision-making in a highly dynamic environment. So, RL techniques are more suitable for handling unstable migration environments with dynamic features such as the mobility of users. On the other hand, traditional value-based algorithms (e.g. Q-learning) can yield optimal migration decisions due to their efficient balance of exploration and exploitation in the search space, but they suffer the most from scalability issues when the agent has to handle a large number of tasks and resources. In contrast, DRL techniques are capable of handling high-dimension data samples by using deep neural networks.

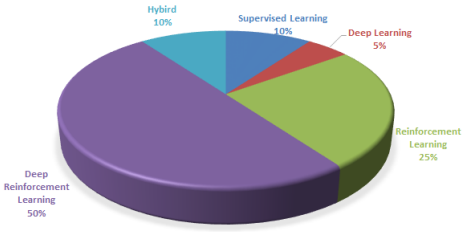

Figure 4: Classification based on the categories of techniques

Q2) Are these migration approaches applied to entire workflow/ application or at the service/task level ?

It is seen in Fig. 5 that service/task are the elements the most investigated by migration approaches, with a percentage of 85%. Considering a workflow with task dependencies is challenging and requires a lot of training time for the RL agent because at each state of environment(migration step), it has to find the set of resources that satisfy the computational requirements of tasks and ensure a high QoS.

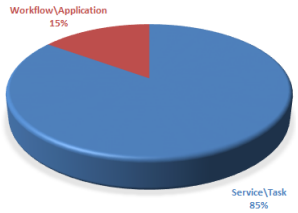

Figure 5: Classification based on migration element

Q3) What are the QoS metrics mostly considered by migration approaches?

Fig. 6 shows that performance(delay/latency, offloading time) and cost metrics are the most considered by 39% and 29%, respectively. It is comprehensible since the main goal of migration

strategies is to reduce user-centric metrics (latency, cost). Furthermore, we notice from Fig. 7 that joint-centric metrics are the most optimized. It is logical because migration approaches do not focus only on finding the optimal resources in terms of computation requirements. It mainly aim to reduce migration time/cost/energy incurred each time the services/applications migrate to another node.

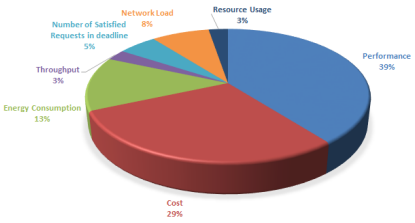

Figure 6: Classification based on QoS metrics

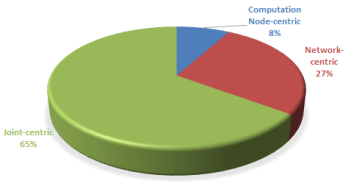

Figure 7: Classification based on QoS metrics type

Q4)What is the migration environment mostly adopted in the literature?

As depicted in Fig. 8, MEC is the environment mostly used for migration with a share of 11 papers, followed by Fog with 5 papers. The main reason is that services/applications have to be closer to end-users in order to satisfy QoS metrics in terms of high security constraints and low latency and cost.

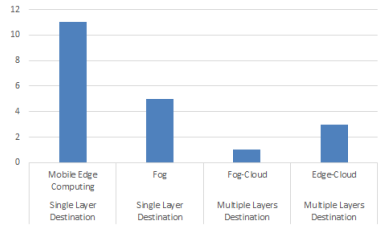

Figure 8: Classification based on migration environment

Q5) What are the domains of application related to the migration problem?

From Fig. 9, we notice that IoT, mobile, and vehicular applications are almost equally addressed. This is due to the fact that migration approaches deal mostly with applications that interact with mobile users in their daily lives such as smart parking systems.

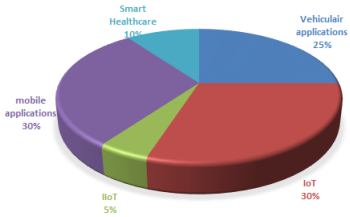

Figure 9: Classification based on migration domains of application

Q6) What are the main challenges in this field?

Service migration has been widely addressed in the literature. However, it still faces a lot of challenges that need further investigation. We identified the service migration challenges as follows:

- Although energy consumption minimization is treated in the literature, it is still one of the key challenges in MEC-Fog since achieving a trade-off between energy consumption and other QoS metrics is a challenging issue. Therefore, more research works need to be conducted in order to optimize this metric.

- The migration caused by the mobility of Fog/Edge nodes has not been well explored in the literature, unlike the one caused by user mobility. Indeed, this aspect is regarded as a major challenge due to the complexity and difficulty of understanding the mobility behavior of the heterogeneous Fog/Edge nodes. Thus, more efforts have to be applied in this context.

- The migration of dependent tasks is also one of the main obstacles in the migration field. Heuristics could be investigated to identify which tasks in a workflow should be migrated while handling tasks dependencies and avoiding network congestion.

- The majority of service migration approaches propose solutions in which services should be migrated among Edge-Fog nodes to follow user mobility. On the other hand, frequent migration may incur additional migration latency and energy consumption[34]. Therefore, it is important to conduct studies that also focus on identifying the only necessary migration. Heuristics could be applied in order to determine whether a migration decision at each user movement is necessary or will increase user-perceived latency and cause QoS degradation.

5. Conclusion

In this paper, we examined recent studies that were specifically focused on migration strategies of IoT applications in Edge-Fog-Cloud using Machine Learning techniques. Firstly, we classified the approaches according to multiple criteria such as migration type, policy, and domains of application. Next, we identified the used technique and the QoS optimized metrics for each work. Then, a statistical examination is conducted in order to answer the proposed research questions. Finally, we discussed the challenges in service migration, which need further investigation.

To cover more literature, we plan to conduct a resource management survey that reviews service placement, scheduling, offloading, and migration approaches in Edge-Fog-Cloud.

Acknowledgments

This work was supported by the ANR LabEx CIMI (grant ANR-11-LABX-0040) within the French State Programme “Investissements d’Avenir”.

References

[1] C. Li, Y. Xue, J. Wang, W. Zhang, T. Li, Edge-oriented computing paradigms: A survey on architecture design and system management, ACM Computing Surveys (CSUR) 51 (2018) 1–34.

[2] C. Byers, R. Swanson, Openfog consortium openfog reference architecture for fog computing, OpenFog Consortium Archit. Working Group, Fremont, CA, USA, Tech. Rep. OPFRA001 20817 (2017) 27–28.

[3] L. Liu, C. Chen, T. Qiu, M. Zhang, S. Li, B. Zhou, A data dissemination scheme based on clustering and probabilistic broadcasting in vanets, Vehicular Communications 13 (2018) 78–88.

[4] C. Puliafito, E. Mingozzi, G. Anastasi, Fog computing for the internet of mobile things: issues and challenges, in: 2017 IEEE International Conference on Smart Computing (SMARTCOMP), IEEE, 2017, pp. 1–6.

[5] J. P. Martin, A. Kandasamy, K. Chandrasekaran, Mobility aware autonomic approach for the migration of application modules in fog computing environment, Journal of Ambient Intelligence and Humanized Computing 11 (2020) 5259–5278.

[6] S. K. Pande, S. K. Panda, S. Das, Dynamic service migration and resource management for vehicular clouds, Journal of Ambient Intelligence and Humanized Computing 12 (2021) 1227–1247.

[7] N. Kaur, A. Kumar, R. Kumar, Promo: Proactive mobility-support model for task scheduling in fog computing, International Journal of Computers and Applications (2022) 1–10.

[8] S. Wang, J. Xu, N. Zhang, Y. Liu, A survey on service migration in mobile edge computing, IEEE Access 6 (2018) 23511–23528.

[9] Z. Rejiba, X. Masip-Bruin, E. Marín-Tordera, A survey on mobility-induced service migration in the fog, edge, and related computing paradigms, ACM Computing Surveys (CSUR) 52 (2019) 1–33.

[10] D. Lan, A. Taherkordi, F. Eliassen, Z. Chen, L. Liu, Deep reinforcement learning for intelligent migration of fog services in smart cities, in: International Conference on Algorithms and Architectures for Parallel Processing, Springer, 2020, pp. 230–244.

[11] K. Ha, Y. Abe, Z. Chen, W. Hu, B. Amos, P. Pillai, M. Satyanarayanan, Adaptive vm handoff across cloudlets, Technical Report CMU-CS-15-113 (2015).

[12] Y. Wang, S. Cao, H. Ren, J. Li, K. Ye, C. Xu, X. Chen, Towards cost-effective service migration in mobile edge: A q-learning approach, Journal of Parallel and Distributed Computing 146 (2020) 175–188.

[13] F. Zhang, G. Han, L. Liu, M. Martinez-Garcia, Y. Peng, Deep reinforcement learning based cooperative partial task offloading and resource allocation for iiot applications, IEEE Transactions on Network Science and Engineering (2022).

[14] R. Yadav, W. Zhang, I. A. Elgendy, G. Dong, M. Shafiq, A. A. Laghari, S. Prakash, Smart healthcare: Rl-based task offloading scheme for edge-enable sensor networks, IEEE Sensors Journal 21 (2021) 24910–24918.

[15] M. M. Bukhari, T. M. Ghazal, S. Abbas, M. Khan, U. Farooq, H. Wahbah, M. Ahmad, K. M. Adnan, An intelligent proposed model for task offloading in fog-cloud collaboration using logistics regression, Computational Intelligence and Neuroscience 2022 (2022).

[16] S. Memon, M. Maheswaran, Using machine learning for handover optimization in vehicular fog computing, in: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, 2019, pp. 182–190.

[17] H. Wang, Y. Li, A. Zhou, Y. Guo, S. Wang, Service migration in mobile edge computing: A deep reinforcement learning approach, International Journal of Communication Systems (2020) e4413.

[18] A. Talpur, M. Gurusamy, Drld-sp: A deep reinforcement learning-based dynamic service placement in edge-enabled internet of vehicles, IEEE Internet of Things Journal (2021).

[19] C.-L. Wu, T.-C. Chiu, C.-Y. Wang, A.-C. Pang, Mobility-aware deep reinforcement learning with glimpse mobility prediction in edge computing, in: ICC 2020-2020 IEEE International Conference on Communications (ICC), IEEE, 2020, pp. 1–7.
[20] A. Abouaomar, Z. Mlika, A. Filali, S. Cherkaoui, A. Kobbane, A deep reinforcement learning approach for service migration in mec-enabled vehicular networks, in: 2021 IEEE 46th Conference on Local Computer Networks (LCN), IEEE, 2021, pp. 273–280.
[21] X. Huang, Q. Zhang, Reinforcement learning based service migration strategy to minimize service cost with delay constraint in edge computing, in: 2021 7th International Conference on Computer and Communications (ICCC), IEEE, 2021, pp. 1341–1348.
[22] Y. Cui, D. Zhang, J. Zhang, T. Zhang, L. Cao, L. Chen, Distributed task migration optimization in mec by deep reinforcement learning strategy, in: 2021 IEEE 46th Conference on Local Computer Networks (LCN), IEEE, 2021, pp. 411–414.
[23] H. Sami, A. Mourad, H. Otrok, J. Bentahar, Demand-driven deep reinforcement learning for scalable fog and service placement, IEEE Transactions on Services Computing (2021).
[24] S. Moon, J. Park, Y. Lim, Task migration based on reinforcement learning in vehicular edge computing, Wireless Communications and Mobile Computing 2021 (2021).
[25] Z. Zhao, N. Emami, H. Santos, L. Pacheco, M. Karimzadeh, T. Braun, A. Braud, B. Radier, P. Tamagnan, Reinforced-lstm trajectory prediction-driven dynamic service migration: A case study, IEEE Transactions on Network Science and Engineering (2022).
[26] S. W. Park, A. Boukerche, S. Guan, A novel deep reinforcement learning based service migration model for mobile edge computing, in: 2020 IEEE/ACM 24th International Symposium on Distributed Simulation and Real Time Applications (DS-RT), IEEE, 2020, pp. 1–8.
[27] H. Zhang, Y. Wang, J.-T. Zhou, J. Gao, J. Liu, A migration method for vehicle mobility services based on road segmentation markov model, Wireless Communications and Mobile Computing 2022 (2022).
[28] A. Abdul Majeed, P. Kilpatrick, I. Spence, B. Varghese, Performance estimation of container-based cloud-to-fog offloading, in: Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing Companion, 2019, pp. 151–156.
[29] Y. Chen, Y. Sun, C. Wang, T. Taleb, Dynamic task allocation and service migration in edge-cloud iot system based on deep reinforcement learning, IEEE Internet of Things Journal (2022).
[30] A. Mseddi, W. Jaafar, H. Elbiaze, W. Ajib, Intelligent resource allocation in dynamic fog computing environments, in: 2019 IEEE 8th International Conference on Cloud Networking (CloudNet), IEEE, 2019, pp. 1–7.
[31] J. Xie, F. R. Yu, T. Huang, R. Xie, J. Liu, C. Wang, Y. Liu, A survey of machine learning techniques applied to software defined networking (sdn): Research issues and challenges, IEEE Communications Surveys & Tutorials 21 (2018) 393–430.
[32] J. M. Johnson, T. M. Khoshgoftaar, Survey on deep learning with class imbalance, Journal of Big Data 6 (2019) 1–54.
[33] R. R. Afshar, Y. Zhang, J. Vanschoren, U. Kaymak, Automated reinforcement learning: An overview, arXiv preprint arXiv:2201.05000 (2022).
[34] G. Liu, J. Wang, Y. Tian, Z. Yang, Z. Wu, Mobility-aware dynamic service placement for edge computing, EAI Endorsed Transactions on Internet of Things 5 (2019).