Federated Machine Learning for Resource Allocation in Multi-domain Fog Ecosystems

Weilin Zhang, Salman Toor
Department of Information Technology
Uppsala University
Uppsala, Sweden
weilin.zhang.7829@student.uu.se, salman.toor@it.uu.se

Mays AL-Naday
School of Computer Science and Electronic Engineering
University of Essex
Colchester, UK
mfhaln@essex.ac.uk

***Abstract*—The proliferation of the Internet of Things (IoT) has incentivised extending cloud resources to the edge in what is deemed fog computing. The latter is manifesting as an ecosystem of connected clouds, geo-dispersed and of diverse capacities. In such ecosystem, workload allocation to fog services becomes a non-trivial challenge. Users' demand at the edge is diverse, which does not lend to straightforward resource planning. Conversely, running services at the edge may leverage proximity, but it comes at higher operational cost let alone increasing risk of resource straining. Consequently, there is a need for intelligent yet scalable allocation solutions that counter the adversity of demand, while efficiently distributing load between the edge and farther clouds. Machine learning is increasingly adopted in resource planning. This paper proposes a federated deep reinforcement learning system, based on deep Q-learning network (DQN), for workload distribution in a fog ecosystem. The proposed solution adapts a DQN to optimize local workload allocations, made by single gateways. Federated learning is incorporated to allow multiple gateways in a network to collaboratively build knowledge of users' demand. This is leveraged to establish consensus on the fraction of workload allocated to different fog nodes, using lower data supply and computation resources. System performance is evaluated using realistic demand from Google Cluster Workload Traces 2019. Evaluation results show over 50% reduction in failed allocations when spreading users over larger number of gateways, given fixed number of fog nodes. The results further illustrate the trade-offs between performance and cost under different conditions.**

***Index Terms*—Workload Allocation, Federated Learning, Deep Q-network, Fog networks, Federated Average Aggregation**

I. Introduction

The number of Internet of Things (IoT) devices currently exceeds 13 billion, and is expected to reach 34.7 billion by the end of 2028 [1]. This is expected to increase the demand for cloud services of data offload, storage and processing to unprecedented levels [2]. Orthogonally, time-criticality and constraints on data sharing due to such reasons as cost and privacy, increasingly favour proximity between end-users and cloud services. This has incentivised shifting towards an edge-to-cloud ecosystem, deemed as a form of fog computing [3], [4]. The latter is a set of clouds: geo-dispersed, connected, decentralised and of variant resource capacity and locality to end-users. While the extension promises significant advantages, it comes with non-trivial challenges. The geo-dispersion and diversity of operational cost, energy efficiency and constraints, between the edge and the cloud, introduces trade-offs between performance and cost [5], [6]. Evidence from existing research and realistic cloud system indicates that edge resources are sparse and operationally expensive [7], [8]. This requires selective allocation to the edge on need-basis, to reduce the risk of resource straining and Quality of Service (QoS) degradation, for applications in need of the edge. Selective allocation is further need to maintain sustainable operational cost.

Orthogonally, local demand at the edge is known to be highly variant [9], and differs across multiple geographic regions (i.e. *hot spots*). This hinders the ability to plan edge resources as local demand is significantly less predictable. The complexity is amplified when entwining locality with user intents for data-consuming services, requiring user-generated data. Aside from privacy constraints, data offload to the ecosystem is associated with a network and storage costs, correlating with data size. This presents a trade-off between favouring the edge, to reduce network cost and the cloud to reduce storage counterpart.

Machine learning has been increasingly adopted in resource allocation to address some of the above challenges in fog computing [10], [11]. However, traditional central learning requires centralized data collation, in turn demand high storage capacity and computation power to learn over large datasets [12]. This further has a higher likelihood of conflicting with user intents for privacy and reduction in data offload cost. Instead, federated learning poses attractive opportunities to address these challenges. It enables collaborative learning, over distributed data owned by non-trusting entities [13]. This can be leveraged to learn patterns of local demand at the edge, in a scalable and intent-compliant manner, and optimized workload allocation across fog nodes to minimize operational costs while complying with user-intents. Moreover, federated learning enables fog providers to withhold business-sensitive information regarding the state of their fog nodes from end-users, while still optimizing workload allocation.

This work proposes a novel federated deep reinforcement learning system, for intelligent resource allocation in multi-domain fog ecosystems. The proposed solution combines deep Q-learning networks (DQN) with federated learning, to create a federated DQN system. Here, localized DQN agents train at the user side, while trained models are aggregated at the fog side. In doing so, the solution mitigates the adversity of local demand through consensus among access gateways, connecting local groups of users. Complementary, business-private information of fog nodes is preserved, by limiting input to access agents to the cost of fog resources. The contributions of this paper are three-fold:

- Propose a federated learning system, based on DQN for optimized resource allocation in multi-domain fog ecosystem, given limited information sharing.
- Formulate the proposed DQN structure and the mechanism for model aggregation, and device a reward scheme that maximizes compliance with user intents, while minimizing operational cost.
- Evaluate the performance of the proposed system for joint data offloading and workload allocation.

The remainder of this paper is organized as follows: Section II reviews state-of-the-art work in resource allocation in fog systems. Section III outlines the model of the fog ecosystem and formulates the problem of request allocation. Section IV describes the proposed federated learning system, the structure of the proposed DQN and the model aggregation mechanism and introduces the respective algorithms. Section V evaluates the performance of the proposed system under different conditions. Finally, Section VI draws the conclusions of this work.

II. Related Work

In recent years, the issue of workload allocation in fog networks has attracted increasing attention. The work of [14] proposes an approximation method by separating the main problem into three sub-problems, for optimal workload allocation to balance delay and minimize power consumption in a Fog-Cloud computing framework. However, this study does not consider the heterogeneity between fog and cloud nodes, which can negatively impact the algorithm's efficiency.

The work of [15] introduces a distributed framework that utilizes dual decomposition and develops two distributed algorithms to achieve optimization of workload. Fog nodes collaborate through an offload forwarding strategy, enabling them to transfer their workload to other fog nodes. With two algorithms that are based on the subgradient method with dual decomposition and the distributed ADMM-VS. The framework enhances the performance of fog computing. Even though distribution and privacy protection are considered in the research [15], it still faces issues when dealing with fog node communication. It needs to find a balance between workload and node communication, and the information of fog nodes is shared under the offload forwarding strategy. The work in [16] employs a deep reinforcement learning algorithm to achieve resource allocation in a dynamic fog computing environment. A deep Q-network (DQN) is applied to optimize resources to maximize the number of requests that the whole network can satisfy, which performs well in the result of the research. However, it only focuses on a single DQN system, lacking considerations of cooperation among multiple DQNs.

Based on the issues of workload allocation in fog networks above, the implementation of federated learning is being considered. In [17], an integrated edge-AI framework of deep reinforcement learning and federated learning is proposed to optimize edge computing, caching, and communication in mobile edge systems. The research in [18] proposes a distributed federated learning algorithm for the resource-constrained fog computing environment to reduce communication latency and energy consumption. Even though the solution provision a certain number of agents for local training, the performance of local models is not considered as a decision factor to trigger model aggregation. This causes significant increase in resource utilisation and churn in model performance. In [19], a federated learning framework named FedFog is proposed to balance the trade-off between network communication efficiency and model accuracy and achieve network-aware optimization of wireless fog-cloud systems. However, there is still a possibility of experiencing high latency as the aggregation occurs in the cloud layer, which is located at a far distance. To address the issues from the above discussion, we proposed a distributed federated learning system based on deep reinforcement learning, for workload allocation in multi-domain fog ecosystems where information sharing is limited across domains.

TABLE I
Summary of Notation

Notation	Definition
$g, \mathcal{G}$	Number and set of gateways
$f, \mathcal{F}$	Number and set of fog nodes
T_c, T_m, T_e	Cloud tier, Medium tier, Edge tier
C^f	Capacity of CPU of each fog node f
M^f	Capacity of memory of each fog node f
B^f	Bandwidth on the path between f and g
$D_g^{f,T}, D_g^f, D_g^T$	Distance between g and f per tier T
d_g^f	Hop count between gateway g and fog node f
E^f	Energy cost of each fog node f
S_g	Number of requests for gateway g
c_s	Required CPU of the request s
m_s	Required memory of the request s
b_s	Required bandwidth of the request s
l_s	Latency priority of the request s
$\alpha_{g,f}^s$	the binary decision variable of s
$\theta_f^{s,c}, \theta_f^{s,m}$	CPU and memory cost of serving s at f
$\theta_{f,g}^s$	network cost of sending s response back from f to g

III. The fog ecosystem and problem of resource allocation

This section outlines the model of the fog ecosystem and formulates the problem of resource allocation. The summary of notation for some key parameters is shown in Table I.

A. The Ecosystem Model

We consider a fog ecosystem the inline with the OpenFog reference architecture [4], and modeled similar to [20] with suitable adaptations. It is composed of three resource tiers, shown in Figure 1: the edge tier, medium tier and cloud tier. Each tier T, has $\mathcal{F}^T \subseteq \mathcal{F}$ set of fog nodes, and those in the same tier are assumed to have a comparable resource configuration. This means they have similar CPU and memory capacity, as well as energy cost. They are too connected by paths of similar bandwidth capacity and range of metric distance D^T to $\mathcal{G}$ set of access gateways. Each gateway is connected from one side to $\mathcal{F}$ fog nodes, and from the other side to a group of end-devices that request fog services, with different latency requirements. In this paper, we focus on three metrics of each $f \in \mathcal{F}$, i.e., the CPU capacity, C^f compute units; the memory capacity M^f in MB; and the energy cost E^f in \$ per compute unit. The latter is defined as 2×10^6MI, following [7]. Furthermore, each path between fog node f and gateway g has a bandwidth capacity B_g^f in Gb/s, and a metric distance, $D_g^f \in [D_{LB}^T, D_{UB}^T]$ in km. Where D_{LB}^T and D_{UB}^T are the lower and upper bounds on D^T, respectively. As fog nodes get closer to access gateways (i.e. approaching the network edge), the CPU, memory and bandwidth capacities as well as distance decrease, while energy costs increase. Each gateway $g \in \mathcal{G}$ receives a number of end-device requests, denoted by S_g. Each request can be described as $s_g = \langle c_s, m_s, b_s, l_s \rangle$, where c_s, m_s, b_s and l_s denote: the required CPU and memory of the request, the required bandwidth for sending the response back to the gateway, and the latency priority of the request. The latency priority determines the appropriate fog tier for allocation, whereby: high priority requests should be allocated to the edge tier, medium priority ones should be allocated to the medium tier and low priority counterparts should be allocated to the cloud tier.

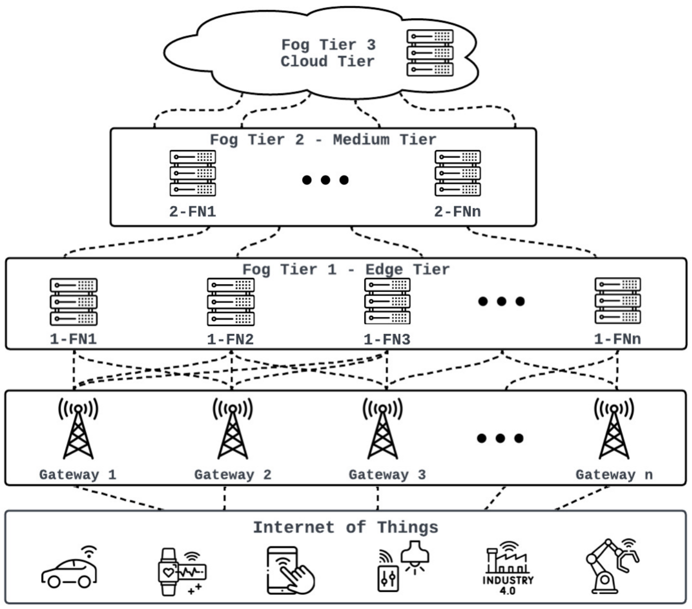

Fig. 1. The three-tier architecture of the fog ecosystem

B. The Problem of Resource Allocation

Given the above the problem of resource allocation in such a fog ecosystem can be described as that of maximizing total allocations with the minimum overall cost, incurred by both parties: access gateways and fog nodes. Subject to capacity constraints of the fog nodes and latency constraints of requests. Mathematically, the problem can be formulated as:

$$\min \sum_{g\in\mathcal{G}} \sum_{s\in S_g} \sum_{f\in\mathcal{F}} \alpha_{g,f}^s \big(c_s * \theta_f^{s,c} + m_s * \theta_f^{s,m} + b_s * \theta_{f,g}^s\big) \quad (1)$$

subject to:

$$\sum_{g\in\mathcal{G}} \sum_{s\in S_g} \alpha_{g,f}^s c_s \quad \leq \quad C^f,\ \forall f \in \mathcal{F} \quad (2)$$

$$\sum_{g\in\mathcal{G}} \sum_{s\in S_g} \alpha_{g,f}^s m_s \quad \leq \quad M^f,\ \forall f \in \mathcal{F} \quad (3)$$

$$\sum_{f\in\mathcal{F}} \alpha_{g,f}^s = 1 \quad , \quad l_s = T \quad (4)$$

whereby $\alpha_{g,f}^s$ is a binary decision variable, defined as:

$$\alpha_{g,f}^s = \begin{cases} 1, & \text{if } s_g \text{ is allocated to } f \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

$\theta_f^{s,c}$, $\theta_f^{s,m}$ denote the CPU and memory costs of serving request s at fog node f, while $\theta_{f,g}$ is the network bandwidth cost for sending the response data back to the gateway. Each cost may be a combination of the energy cost incurred by using the resources along with value of the resource in the fog node. The constraints of (2) and (3) the total allocation of CPU and memory resources does not exceed the respective CPU and memory capacity of any fog node. The constraint of (4) ensures that each request is allocated to a fog node in the appropriate tier, restricted by the mapping between the latency priority of the request, l_s and the tier T. Notably, when DQN does not obey these constraints, a violation and request failure is recorded. Note that we omit the bandwidth capacity constraint, assuming that the network has sufficient bandwidth to accommodate all traffic demand. This is represented by an equality between the priority and the tier. Such a problem is well-known to be NP-hard and difficult to solve centrally, as that requires all stakeholders in the ecosystem to share state information with the solver. However, such information are typically business-sensitive and not subject to external sharing. Hence, to solve the problem of (1), we propose a federated learning system described next.

IV. The Federated Deep Q-Network System

This section presents the FDQN system in two aspects, i.e., the network architecture and model structure.

A. Federated DQN Structure

Given the model and problem formulations above, we propose a federated DQN system (FDQN) for intelligent workload allocation to fog nodes, shown in Figure 2. DQN is a reinforcement learning algorithm that combines deep learning techniques with the Q-learning algorithm [21]. Local agents learn by using deep neural networks to estimate the Q-values for each (state, action) pair, aiming to maximize the total reward. A FDQN system comprises local learning agents and a centralised aggregator.

Since a gateway serves as a point of connection for end-devices, we consider gateways to host the local agents. Each agent trains a DQN for one round over a subset of local requests, to determine an optimal local offloading strategy. A local score is then calculated based on accumulated reward from requests allocations. Meanwhile, the aggregator is hosted at the fog side. For example, co-located with an orchestrator of fog nodes. At the end of each round of training, agents send their local models and their respective scores to the aggregator, along with the volume of demand allocated to each fog node. The aggregator calculates a global score from local counterparts. If the global score has improved, from the last calculation, the aggregator generates a global model from local alternatives and send it back to learning agents. Irrespective of the score, the aggregator further calculates new capacity allocation per access gateway per fog node, along with costs of CPU and memory resources per fog node based on available capacities and allocated demand in the round. These updates are communicated to learning agents at the end of each round.

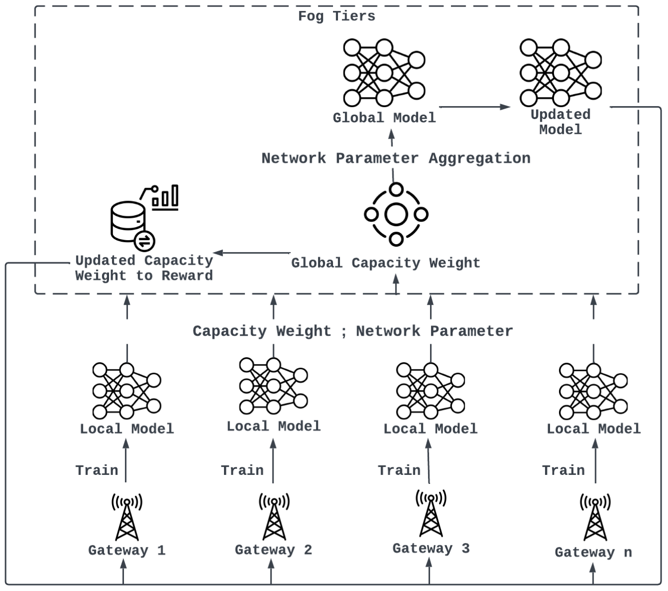

Fig. 2. Process of Federated Deep Q-Network Model

B. Local DQN Model

Each agent trains a DQN to allocate a local request to one of the fog nodes and reserve CPU and memory resources. Here, we formulate the problem of fog node selection and resource allocation as a Markov decision process (MDP) with: a state space, action space and a reward scheme. The set of local requests represent the state space, while the set of fog nodes represent the action space. The reward scheme incorporates the memory-CPU capacity along with the energy price and PUE of each fog node, as well as the length and bandwidth on the path between an access gateway and a fog node.

- *State Space*: the state space for each gateway $g \in \mathcal{G}$ is the set of requests, S_g, described earlier in Section III-A. Note that $t = \{1, 2, \ldots, |S_g|\}$ is the requests' index of S_g, with s_t referring to the t-th request.
- *Action Space*: the action space for gateway g is denoted by A_g. Each action represents a fog node and can be described as $a_g^f = \langle C_g^f, M_g^f, E_g^f, B_g^f, D_g^f, d_g^f, W_{g,i-1}^f \rangle$, where C_g^f and M_g^f are the current CPU and memory availability of f to the g. E_g^f is the energy cost of f at g. B_g^f, D_g^f and d_g^f are: the bandwidth on the path between f and g, the nominal distance of g from the respective tier of the fog node and the hop count on the path. $W_{g,i-1}^f$ represents the weight of fog node f after the last round $i-1$ of local training and cost updates. Notably, $C_{g,t}^f$, $M_{g,t}^f$ are updated before each request allocation (i.e. state-action mapping of t), calculated as:

$$C_{g,t}^f = C_{g,t-1}^f - c_{g,t} \tag{6}$$

$$M_{g,t}^f = M_{g,t-1}^f - m_{g,t} \tag{7}$$

- *Reward Scheme*: the objective of the reward scheme is to maximize the number of successfully allocated requests, while minimizing the allocation cost per request. Accordingly, the reward scheme is an inverse representation of the cost functions of (1), $\theta_f^{s,c}, \theta_f^{s,m}, \theta_{f,g}^s$, defined as a joint function of: the current resource availability, the energy cost, the path conditions and the latency requirement. The function is formulated as:

$$R(s_t^g, a_g^f) = R_{g,t}^{cap,f} + W^f + R_{g,t}^{E,f} + R_{g,t}^{B,D,f} + R_{g,t}^{l,f} \tag{8}$$

where $R_{g,t}^{cap,f}$ is derived from the current capacity of fog node f when processing state t, defined as:

$$R_{g,t}^{cap,f} = \begin{cases} \lambda_{cap} \log_5(cap_{g,t}^f + 1), & C_{g,t}^f \geq 0, M_{g,t}^f \geq 0 \\ -1 - e^{-C_{g,t}^f}, & C_{g,t}^f < 0, M_{g,t}^f \geq 0 \\ -1 - e^{-M_{g,t}^f}, & C_{g,t}^f \geq 0, M_{g,t}^f < 0 \\ -1 - e^{-cap_{g,t}^f}, & C_{g,t}^f < 0, M_{g,t}^f < 0 \end{cases} \tag{9}$$

$cap_{g,t}^f = C_{g,t}^f + M_{g,t}^f$ and λ_{cap} is a tuning parameter that controls the weight of $R_{g,t}^{cap,f}$ in the reward function. Recall that W^f is an attraction term that represents the weight of f for all gateways based on its total capacity. The term is calculated by the aggregator, and shared with all gateways following each round of training. $R_{g,t}^{E,f}$ corresponds to the energy reward of f, denoted as:

$$R_{g,t}^{E,f} = \frac{\lambda_e}{E_{g,t}^f} \tag{10}$$

where λ_e is a tuning parameter that controls the weight of $R_{g,t}^{E,f}$ in (8). Notice that a higher energy cost incurs lower energy reward. $R_{g,t}^{B,D,f}$ corresponds to the reward component of the network path. It is derived from the bandwidth capacity of the path, the nominal metric distance between any gateway and fog nodes of tier T and the hop count of the path, denoted as:

$$R_{g,t}^{B,D,f} = \frac{B_g^f}{d_g^f D_g^f} \tag{11}$$

Note that the term $d_g^f D_g^f$ represents the effective distance, i.e. *proximity*, between gateway g and fog node f. Smaller values indicates higher proximity between the gateway and fog node. $R_{g,t}^{l,f}$ is the reward-penalty term, corresponding to the satisfaction of the latency requirement. Recall that the latency priority maps to an appropriate tier of allocation. A request allocated to a fog node in the correct priority-tier map incurs a reward (i.e. *match*), otherwise incorrect allocation incurs a penalty. The term is defined as:

$$R^{l,f}_{g,t} = \begin{cases} +\lambda_l, & \text{if } l_{g,t} \text{ matches } T \\ -\lambda_{\hat{l}}, & \text{otherwise} \end{cases} \tag{12}$$

where λ_l and $\lambda_{\hat{l}}$ are configurable parameters. Notice that the magnitude of the reward may not necessarily equal that of the penalty. This is to allow for added flexibility to independently control the two sides of the term.

C. An Agent: Training Local DQN

Each gateway g deploys an agent, which trains their DQN as shown in Algorithm 1. Note that ϵ, σ and β are the hyper parameters of DQN. ϵ is the exploration parameter while σ is the decrement rate of ϵ, and β is the minibatch size. For any training round, e, an agent runs an internal loop of size $|S_g|$, corresponding to the total state size at the gateway (i.e. total number of requests). For each state t, the gateway chooses an action $a^f_{g,t}$ (i.e., a fog node) and obtains the reward $R_t(.)$ and a next state s_{t+1} (Lines 2-7). After that, a transition containing the current state, selected action, new state and the reward is stored in the experience replay memory (Line 9). Before the iteration ends, the gateway randomly samples a minibatch of transitions from the replay memory buffer and updates the Q network (Lines 9-10). At the end of the round, the gateway calculates the total reward of this round, regarded as a score reflecting the performance of DQN in this round. The score and the model parameters are then shared with the aggregator to calculate the global score and generate a global model of the round.

D. The Aggregator: Generating A Global Model

The algorithm for calculating a global score and model is outlined in Algorithm 2, based on earlier work in [22] with suitable modifications. Similar to [22], the global score $\mathcal{R}_e$ is calculated as an average of the local scores $\{\mathcal{R}_{e,g} | g \in \mathcal{G}\}$, following the end of each round e (Lines 3-4). Note that the averaging strategy is a mere example here. Alternative strategies can be adopted in a straightforward manner to suit different federation approaches. If $\mathcal{R}_e$ exceeds the best score so far $\mathcal{R}_{best}$, local models are aggregated to generate a global counterpart, using a federation function of choice. This work adopts Federated Averaging (FedAvg) as an example function, while noting that alternatives can substitute FedAvg in a straightforward manner. The global model parameters are sent to the gateways for the next round of local training, $e+1$, (Lines 10-12). Otherwise, if $\mathcal{R}_e < \mathcal{R}_{best}$, the gateways are notified to continue training using their pre-existing local models. In parallel, the aggregator assesses the volume of request allocations to each fog node against the capacity of the node to update the value of $\{W^f | \forall f \in \mathcal{F}\}$, tuning the attraction of f. In cases where a fog node f decides to scale up or down the capacity allocation per gateway, the aggregator may further calculate a new C^f_g, representing a fraction of the actual total capacity of f and share it with gateway g.

Algorithm 1 Local DQN Model Training

```
Input: last aggregate model, hyper parameters ($\epsilon$, $\sigma$, $\beta$), state
    space $\{s^g_t | t \in 1, 2, \ldots, |S_g|\}$, action space $\{A_f | f \in \mathcal{F}\}$,
    initial local score $\{\mathcal{R}_{0,g} | g \in \mathcal{G}\}$, experience replay mem-
    ory $\mathcal{M}$.
Output: local model parameters $(w_{e,g}, b_{e,g})$, local score
    $\mathcal{R}_{e+1,g}$.
 1: Initialization: $\mathcal{R}_{e,g}$
 2: for t = 1 to $|S_g|$ do
 3:     if $(Random() > \epsilon)$ then
 4:         Select a random action $a^f_{g,t}$
 5:     else
 6:         Select $a^f_{g,t} \leftarrow argmax_a Q(s_{t+1}, a^f_{g,t})$
 7:     end if
 8:     Take action $a^f_{g,t}$, observe reward $R(s^g_t, a^f_{g,t})$ and the next
        state $s_{t+1}$
 9:     Store transitions $\{s_t, a^f_{g,t}, s_{t+1}, R(s^g_t, a^f_{g,t})\}$ into $\mathcal{M}$
10:     if $t \geq \beta$ then
11:         Randomly sample a $\beta$-size minibatch of transitions
            from $\mathcal{M}$
12:     end if
13:     Update evaluation $Q$ network and target $\hat{Q}$ network
14: end for
15: Calculate the score of the DQN agent $\mathcal{R}_{e,g} \leftarrow$
    $\sum_{t=1}^{|S_g|} R(s^g_t, a^f_{g,t})$
16: Send local model parameters $(w_{e,g}, b_{e,g})$, the score $\mathcal{R}_{e,g}$
    and the volume of request allocations to the aggregator.
```

Algorithm 2 Global Model: Federated Averaging

```
Input: the number of gateways (agents) $|\mathcal{G}|$, best global score
    $\mathcal{R}_{best}$
Output: the aggregated model parameter
 1: Initialisation : $\mathcal{R}_{best}$, number of episodes $H$
 2: for episode $e = 0$ to $H$ do
 3:     for each agent of $g \in \mathcal{G}$ in parallel do
 4:         Obtain the locally trained DQN model of $g$, the local
            score $\mathcal{R}_{e,g}$ and volume of request allocations
 5:     end for
 6:     Calculate the global score $\mathcal{R}_e \leftarrow (\sum_{g \in \mathcal{G}} \mathcal{R}_{e,g})/|\mathcal{G}|$
 7:     if $e > 0$ and $\mathcal{R}_e > \mathcal{R}_{best}$ then
 8:         $\mathcal{R}_{best} \leftarrow \mathcal{R}_e$
 9:         Calculate $(w_{e+1}, b_{e+1})$ parameters of the global
            model using FedAvg: $w_{e+1} \leftarrow (\sum_{g \in \mathcal{G}} w_{e,g})/|\mathcal{G}|$,
            $b_{e+1} \leftarrow (\sum_{g \in \mathcal{G}} b_{e,g})/|\mathcal{G}|$
10:         Calculate a new $\{W^f \mid \forall f \in \mathcal{F}\}$ based on local
            request allocations per $g$
11:         Send aggregated model parameter $\{w_{e+1}, b_{e+1}\}$ and
            $\{W^f\}$ to local agents.
12:     end if
13: end for
```

V. Evaluation

Here, we evaluate the performance of the proposed solution over realistic cloud requests from Google Cluster Workload Traces 2019 [23]. First, we obtain a total number of requests for cloud services, by randomly sampling the data from the instances table of the traces dataset. The requests are then evenly distributed among one or multiple gateways to simulate a geo-distribution of demand. We consider requests from various users, and the number of users in a particular region remains constant. In this case, the number of requests per gateway decrease as the number of gateways increases. Each request specifies: the latency priority along with the CPU and memory requirements. Based on the explanation of the priority range in [24], we assume priority classes to represent the latency requirement of requests, indicating the preferred fog tier for allocation.

To emulate a realistic fog ecosystem, we extract details of Google machines from the traces and use them to represent tier-based fog nodes. To do so, machines are grouped together to provided aggregate memory and CPU capacities following the distribution described in [7]. This distribution model is further used to define the energy price and PUE per fog node, along with the average bandwidth, range of metric distance and hop count on the path between a fog node and any access gateway. The energy cost per request per fog node is calculated by a combination of the PUE and the price per computing unit in each fog node along with the task size. The range of PUE values per fog tier is assumed to follow the distribution of the average PUE of data centers [25].A summary of the evaluation parameters and settings is provided in Table II. All results are shown for the validation dataset, using trained models. Note, all models have been trained until convergence is reached where the reward value is within acceptable bounds of variation. Such convergence required an average of 200 training rounds to be achieved.

A. Request Allocation Failure Rate

Here, we evaluate performance given a priority-tier-based allocation strategy. Requests are successfully allocated if fog resources are provisioned and the request priority matches the tier of the fog node, serving the request. We focus on the rate of allocation failure, as opposite to the rate of successful allocation. Because, the former allows a microscopic zoom on performance under different conditions. Figure 3 shows the ratio of request allocation failures given increasing spread of demand over a larger access network. This is illustrated by increasing number of gateways. The results indicate that allocation failures primarily occur within requests of medium-to-high priority, need to be allocated to fog nodes in medium and edge tiers, respectively. This is caused by the higher constraint on capacities of fog nodes in those tiers compared to the cloud counterpart, as shown in Table II, leading to resource strain in those nodes that results in failed allocations.

The failure rate decreases from $\approx 60\%$ to $\approx 2\%$ as the number of gateways increases from 1 to 20. This is caused by the difference between the volume of demand per gateway and capacity per fog node closest to the gateway. Since fog nodes in the same tier have similar capacities, selection of a fog node in a tier depends on the path settings. When the volume of high-priority demand is concentrated in one gateway, it poses higher strain on the edge node closest to the gateway and requests will fail to allocate under the dominance of latency requirement in the reward function even with a higher attraction of the medium and/or cloud tiers than edge nodes farther from the gateway. This results in a higher failure rate of allocation in the requests with medium and high priority latency demand in one gateway. As the number of gateways increases and demand spreads, the number of edge nodes closest to gateways grows too. The combined growth and demand spread alleviates the resource burden on specific, 'nearest', fog nodes in each tier. As a result, the failure rate decreases significantly.

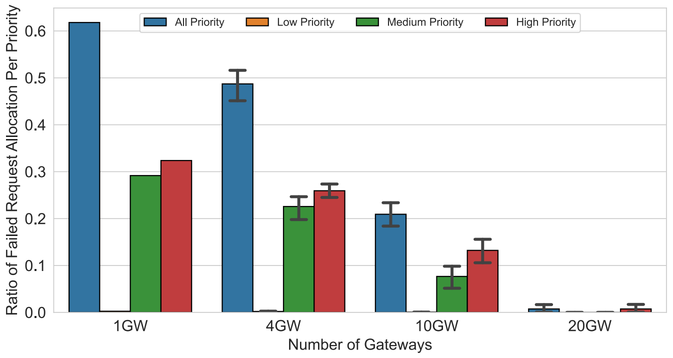

Fig. 3. Failed request allocation per priority having fixed set of fog nodes $\{2, 15, 20\}$ for increasing number of gateways.

B. Resource Utilization

Figure 4(a) shows resource utilization per tier when increasing the number of gateways. Notice, that there is no guarantee allocations always have a priority-tier match. Hence, figures 4(b)-4(c) reveal incorrect allocations per priority class and per tier, respectively. The former shows the volume of incorrect allocations per priority, while the latter shows to which tier they have been directed. The cloud tier maintains a consistent ratio of request allocation, hovering around 35%, regardless of the number of gateways. The allocation ratio to the medium and edge tiers increases as the number of gateways increases. The issue is attributed to the limited capacity and communication distance of these tiers. Increasing the number of gateways leads to higher spread of the workload across more fog nodes. With the proximity advantage of fog nodes in the medium and edge tiers, a trade-off in choosing fog nodes in different tiers arises in DQN. Due to this, the allocation request ratio has increased from $\approx 40\%$ to $\approx 100\%$.

Figure 4(b) shows that incorrect allocation primarily happens with low-priority requests, $\approx 0.02\% - 0.25\%$ are allocated incorrectly. Figure 4(c) shows incorrect allocations mainly observed in the medium tier, meaning that non-medium-priority requests are allocated incorrectly to the medium tier with a range of $\approx 0.02\%–0.25\%$. Meanwhile, a few non-high-priority requests are allocated to the edge tier. The reason for this is that the reward values of fog nodes in the cloud and medium tiers are close in some situations, particularly if there is enough

TABLE II
EVALUATION PARAMETERS AND SETTINGS

Parameter	Setting
Number of gateways	[1, 4, 10, 20]
Total number of requests for all the gateways	40000
Number of requests per gateway (gw)	$\{1gw : 40000, 4gw : 10000, 10gw : 4000, 20gw : 2000\}$
Latency priority (p) of requests per tier (T)	$\{T_{cloud} : p \leq 115, T_{mid} : 116 \leq p \leq 119, T_{edge} : p \geq 120\}$
Number of Fog nodes per tier (T)	$\{T_{cloud} : 2, T_{mid} : 15, T_{edge} : 20\}$
Capacity ratio per node per tier	$\{T_{cloud} : T_{mid} : T_{edge} = 60 : 3.75 : 1\}$
Average energy cost per node per tier	$\{T_{cloud} : [0.04 - 0.05], T_{mid} : [0.12 - 0.15], T_{edge} : [0.97 - 1.22]\}$
Bandwidth capacity between fog nodes and gateways [Gbps]	$\{T_{cloud} : [100 - 150], T_{mid} : [10 - 15], T_{edge} : [1 - 1.5]\}$
Metric distance between fog nodes and gateways [km]	$\{T_{cloud} : [100 - 120], T_{mid} : [10 - 20], T_{edge} : [3 - 5]\}$
Hop count between fog nodes and gateways	$\{T_{cloud} : 100, T_{mid} : [12, 13], T_{edge} : [1 - 10]\}$

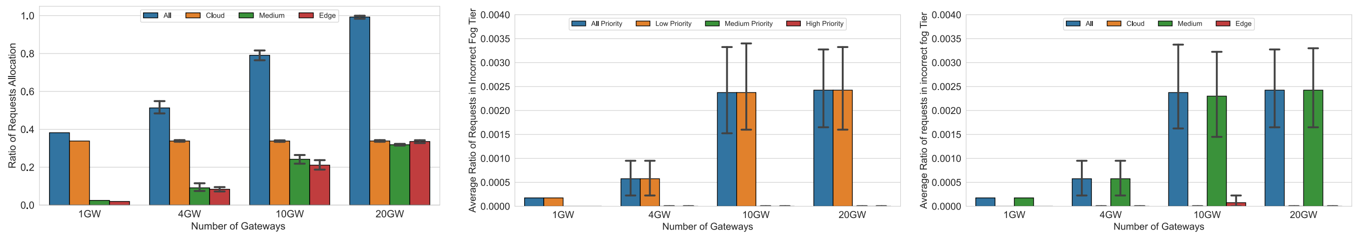
(a) Gateways Request Allocation - Overall (b) Gateways Incorrect Allocation per Priority (c) Gateways Incorrect Allocation per Tier

Fig. 4. Resource utilisation having fixed set of fog nodes $\{2, 15, 20\}$ for increasing number of gateways.

TABLE III
SETTING OF ENERGY COST

Tier	PUE Range	Price (\$) [7]
Cloud Tier	$[1.25 - 1.55]$	0.032339
Medium Tier	$[1.55 - 1.8]$	0.080848
Edge Tier	$[2 - 2.5]$	0.485090

available capacity in the medium tier. A small part of cloud-priority requests are allocated to the medium tier because of the higher reward. Notably, overall the rate of incorrect allocation is low due to the severe penalty in DQN.

C. Energy Cost

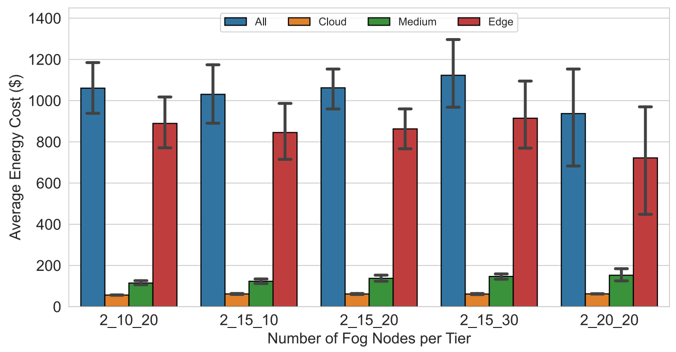
Fig. 5. Average energy cost per tier having fixed number of 10 gateways (i.e. 10GW) for different sets of fog nodes

Figure 5 presents the results of the energy cost, incurred when varying the number of fog nodes per tier. The number of gateways is fixed at 10, with 4000 requests per gateway. We analyze the energy cost given variant number of nodes in the edge and medium tiers. First, we compare $\{10, 20, 30\}$ edge-tier nodes, having fixed 2 cloud-tier nodes and 15 medium-tier nodes. Second, we compare $\{10, 15, 20\}$ medium-tier nodes for fixed 2 cloud-tier counterparts and 20 edge-tier nodes.

The results show that the total energy cost of the fog ecosystem strongly correlated with the energy cost of the edge tier. The energy cost of the cloud tier is the lowest, while the edge tier has the highest energy cost, as adapted in Table III from [7]. The energy cost increases when scaling the edge tier, resulting in an overall cost increase. On the other hand, scaling up the medium tier only results in small increase of the energy cost, with a lower overall cost than that when scaling the edge. Combined with the resource allocation in Figure 4(a), the number of fog nodes in each tier becomes crucial for lowering the overall energy cost, given similar request rate from a fixed number of gateways.

D. Sensitivity

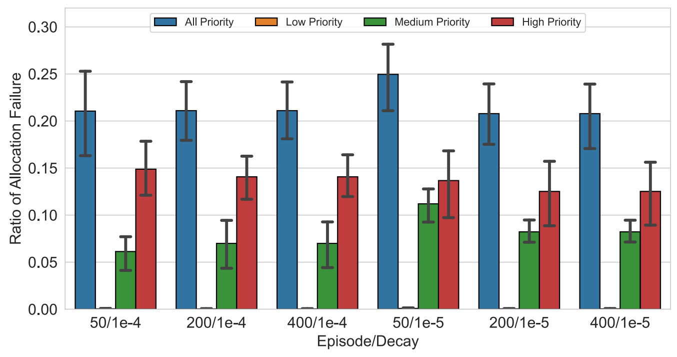
Fig. 6. Request allocation sensitivity with the episode parameter $\{50, 200, 400\}$ and decay rate $\{10^{-4}, 10^{-5}\}$ (fixed number of gateways: 10GW, fixed set of fog nodes: $\{2, 15, 20\}$)

Figure 6 shows the sensitivity of the FQDN system to different episode numbers (i.e. learning rounds) and epsilon decay rate. The number of gateways is fixed at 10 with 4000 requests per gateway, while the number of fog nodes is fixed at: 2 nodes in the cloud tier, 15 nodes in the medium tier and 20 nodes in the edge tier. We use the rate of allocation failure to describe the sensitivity, due to its microscopic zoom on performance under different conditions. Increasing the number of episodes from 50 to 400 incurs fewer failed allocations with less variation of the standard error. On the other hand, a faster decay rate 10^{-4} causes a DQN to have insufficient exploration, of the diversity of states and actions and immaturity of the trained model. This can lead to a sub-optimal allocation policy of DQN, particularly when the number training episodes is low. In contrast, a slower decay rate of 10^{-5} allows for greater exploration of states and actions space, leading to higher maturity of the trained model and better allocation policy . This is reflected by having $\approx 16\%$ reduction in failed allocations from 0.25 to ≈ 0.21 across all priorities, when increasing the number of episodes from 50 to 400.

Comparing the different episodes between the decay rate 10^{-4} and 10^{-5}, the performance with a slower decay rate 10^{-5} is worse than that with a faster decay rate 10^{-4} under 50 episodes but the performance of 10^{-5} is better as the episodes increase. This is caused by the exploration under different decay rates. With the decay rate 10^{-5}, the probability of choosing exploration is still larger than exploitation, which can cause a higher failure rate in 50 episodes because of the higher randomness. This increases the chance to obtain the optimal policy of DQN with a more sufficient exploration. Even though the performance of 10^{-4} is better than that of 10^{-5} in 50 episodes, the failure rate under 10^{-4} is slightly higher than under 10^{-5} in 200 and 400 episodes. This is because of the less sufficient exploration in 10^{-4}, which means less experience for training.

For the request with medium priority, the reason for the increasing failure rate from 50 to 200 episodes under 10^{-4} is that some medium fog nodes gain the advantage too early during the training under the less sufficient exploration (i.e. less experience for training), resulting in a shortage of capacity in these fog nodes and causing an increase in failed allocation of requests with medium priority.

E. Reward Scheme

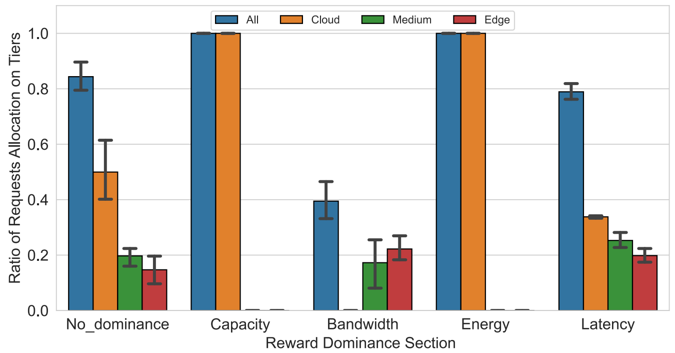

Fig. 7. Reward discussion of request allocation per tier per dominance with fixed gateways: 10GW, fixed set of fog nodes: $\{2, 15, 20\}$

Figure 7 shows the resource allocation based on the dominance of different terms in the reward scheme. Recall that each term corresponds to a cost metric of interest to fog nodes or access gateways or both. The number of gateways is fixed at 10 with 4000 requests per gateway, while the number of fog nodes is fixed at: 2 in the cloud tier, 15 in the medium tier and 20 in the edge tier. When there is no dominant metric in the reward function, most of the requests are allocated to the cloud tier due to the advantage of high capacity and low energy cost. This trend becomes clearer when the capacity or energy metrics are dominant in the reward function. Almost all the requests are allocated to the cloud tier. In contrast, less than 40% of the requests are allocated in total when the bandwidth metric is dominant. Here, requests are allocated to the medium and edge tiers because the reward is determined by the bandwidth capacity, hop count and metric distance of each gateway-fog node path. Although the bandwidth capacity on paths towards the cloud tier is greater than the other two tiers, the metric distance is farther too. In this case, requests tend to be allocated to the tier closer to the gateways, i.e., the edge and medium tiers. This causes high resource strain in those tiers, resulting in performance degradation. When latency is the dominant metric, requests with different latency priorities are allocated to different tiers. Comparing with the no-dominance alternative, the total request allocation when latency is dominant is worse. Since requests can only be sent to the matching tier based on latency priority, resource strain is more likely to happen in the edge and/or medium tiers. In other words, if a tier is strained, there is no leniency to allocate requests in a different tier when latency is the dominant metric.

VI. Conclusions

This work proposed a novel federated deep reinforcement learning system, for efficient workload allocation in multi-domain fog computing ecosystems. Our system incorporated a set of local agents, training DQN models to intelligently allocate requests of a single gateway to different fog nodes. The adversity of demand at the edge is mitigated by introducing consensus among gateways through federated learning, achieved by aggregating local models. This has further allowed for faster model convergence and higher workload balance. Moreover, by limiting the information sharing from fog nodes to the cost of resources and allocated capacity per gateway, the solution allows fog nodes to maintain their autonomy and preserve their private information. We evaluated the performance of the solution with respect to allocation failure, resource utilization and energy cost. The results showed that the allocation failure decreases with the increase of gateways. Whilst the low rate of incorrect allocation mainly happens between the cloud tier and the medium tier. Moreover, the sensitivity of the system is evaluated when varying the number of learning rounds, the exploration decay rate and the dominance of different terms in the reward scheme. Evaluation results have shown CPU or energy dominance in the reward scheme results in dominant allocation to the cloud due to the cheaper costs. Our future work intends to evaluate larger volume of requests, addressing other trade-offs between the distribution of gateways and fog nodes.

References

[1] "Ericsson Mobility Report November 2022," *Ericsson*, p. 11, 2022.

[2] "Cisco Annual Internet Report - Cisco Annual Internet Report (2018–2023) White Paper," *Cisco*, March 2020.

[3] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, ser. MCC '12, Helsinki, Finland: Association for Computing Machinery, 2012, pp. 13–16.

[4] O. C. A. W. Group, "Openfog reference architecture for fog computing," Fremont, CA, USA, Tech. Rep. OPFRA001.020817, February 2017.

[5] J. Vergara, J. Botero, and L. Fletscher, "A comprehensive survey on resource allocation strategies in fog/cloud environments," *Sensors*, vol. 23, no. 9, p. 4413, 2023.

[6] J. B. Jr, B. Costa, L. R. Carvalho, M. J. Rosa, and A. Araujo, "Computational resource allocation in fog computing: A comprehensive survey," *ACM Computing Surveys*, 2022.

[7] S. Shaik and S. Baskiyar, "Distributed service placement in hierarchical fog environments," *Sustainable Computing: Informatics and Systems*, vol. 34, p. 100 744, 2022.

[8] A. M, B. P, and C. L, "Trends in data centre energy consumption under the european code of conduct for data centre energy efficiency," *ENERGIES*, vol. 10, no. 10, p. 1470, 2017.

[9] K. Cao, Y. Liu, G. Meng, and Q. Sun, "An overview on edge computing research," *IEEE Access*, vol. 8, pp. 85 714–85 728, 2020.

[10] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Communications Surveys Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019.

[11] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655–1674, 2019.

[12] K. H. Abdulkareem, M. A. Mohammed, S. S. Gunasekaran, M. N. Al-Mhiqani, A. A. Mutlag, S. A. Mostafa, N. S. Ali, and D. A. Ibrahim, "A review of fog computing and machine learning: Concepts, applications, challenges, and open issues," *IEEE Access*, vol. 7, pp. 153 123–153 140, 2019.

[13] A. M. Abdelmoniem, C.-Y. Ho, P. Papageorgiou, and M. Canini, "Empirical analysis of federated learning in heterogeneous environments," in *Proceedings of the 2nd European Workshop on Machine Learning and Systems*, ser. EuroMLSys '22, Rennes, France: Association for Computing Machinery, 2022, pp. 1–9.

[14] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1171–1181, 2016.

[15] Y. Xiao and M. Krunz, "Distributed optimization for energy-efficient fog computing in the tactile internet," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 11, pp. 2390–2400, 2018.

[16] A. Mseddi, W. Jaafar, H. Elbiaze, and W. Ajib, "Intelligent resource allocation in dynamic fog computing environments," in *2019 IEEE 8th International Conference on Cloud Networking (CloudNet)*, 2019, pp. 1–7.

[17] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, 2019.

[18] R. Saha, S. Misra, and P. K. Deb, "Fogfl: Fog-assisted federated learning for resource-constrained iot devices," *IEEE Internet of Things Journal*, vol. 8, no. 10, pp. 8456–8463, 2021.

[19] V.-D. Nguyen, S. Chatzinotas, B. Ottersten, and T. Q. Duong, "Fedfog: Network-aware optimization of federated learning over wireless fog-cloud systems," *IEEE Transactions on Wireless Communications*, vol. 21, no. 10, pp. 8581–8599, 2022.

[20] M. AL-Naday, N. Thomos, J. Hu, B. Volckaert, F. de Turck, and M. J. Reed, "Service-based, multi-provider, fog ecosystem with joint optimization of request mapping and response routing," *IEEE Transactions on Services Computing*, pp. 1–15, 2022.

[21] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[22] M. Al-Naday, M. Reed, V. Dobre, S. Toor, B. Volckaert, and F. De Turck, "Service-based federated deep reinforcement learning for anomaly detection in fog ecosystems," in *2023 26th Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, 2023, pp. 121–128.

[23] J. Wilkes, *Yet more Google compute cluster trace data*, Google research blog, Posted at https://ai.googleblog.com/2020/04/yet-more-google-compute-cluster-trace.html., Mountain View, CA, USA, April 2020.

[24] J. Wilkes *et al.*, "Google cluster-usage traces v3," *Google Inc., Mountain View, CA, USA, Technical Report*, 2020.

[25] Statista. "Data center average annual pue worldwide." (2021), [Online]. Available: https://www.statista.com/statistics/1229367/data-center-average-annual-pue-worldwide/.