Future Generation Computer Systems 125 (2021) 891–907

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Energy-conscious optimization of Edge Computing through Deep Reinforcement Learning and two-phase immersion cooling

Sergio Pérez [a,*], Patricia Arroba [a,b], José M. Moya [a,b]

[a] Laboratorio de Sistemas Integrados (LSI), Universidad Politécnica de Madrid, ETSI Telecomunicación, Avenida Complutense 30, Madrid 28040, Spain
[b] Center for Computational Simulation, Universidad Politécnica de Madrid, Campus de Montegancedo UPM, Boadilla del Monte, 28660 Madrid, Spain

ARTICLE INFO

Article history:
Received 4 December 2020
Received in revised form 15 July 2021
Accepted 20 July 2021
Available online 26 July 2021

Keywords:
Energy-aware optimization
Deep Reinforcement Learning
Edge Computing
Two-phase immersion cooling
Advanced driver assistance systems

ABSTRACT

Until now, the reigning computing paradigm has been Cloud Computing, whose facilities concentrate in large and remote areas. Novel data-intensive services with critical latency and bandwidth constraints, such as autonomous driving and remote health, will suffer under an increasingly saturated network. On the contrary, Edge Computing brings computing facilities closer to end-users to offload workloads in Edge Data Centers (EDCs). Nevertheless, Edge Computing raises other concerns like EDC size, energy consumption, price, and user-centered design. This research addresses these challenges by optimizing Edge Computing scenarios in two ways, two-phase immersion cooling systems and smart resource allocation via Deep Reinforcement Learning. To this end, several Edge Computing scenarios have been modeled, simulated, and optimized with energy-aware strategies using real traces of user demand and hardware behavior. These scenarios include air-cooled and two-phase immersion-cooled EDCs devised using hardware prototypes and a resource allocation manager based on an Advantage Actor–Critic (A2C) agent. Our immersion-cooled EDC's IT energy model achieved an NRMSD of 3.15% and an R^2 of 97.97%. These EDCs yielded an average energy saving of 22.8% compared to air-cooled. Our DRL-based allocation manager further reduced energy consumption by up to 23.8% in comparison to the baseline.

1. Introduction

Technological breakthroughs such as the Internet of Things (IoT) or Artificial Intelligence (AI) facilitate the establishment of Smart Cities. This novel conception of modern cities has the potential to empower people in truly diverse ways [1]. According to McKinsey [2], IoT-connected devices will reach 43 billion by 2023, almost three times more than in 2018. Rethinking their deployment and operation becomes a necessity to adapt and scale the infrastructure required to satisfy the ever-growing demand for new applications.

Today, Cloud Computing (CC) stands as the reigning computing paradigm, relying heavily on massive infrastructures that manage myriads of data in a centralized fashion. These so-called Cloud Data Centers (CDCs) will ultimately suffer under an increasingly saturated network, negatively affecting the delay perceived by the end-users. While these circumstances might not be a problem for some applications, those with critical latency and bandwidth constraints, such as autonomous driving [3] or smart healthcare [4], will undoubtedly experience performance issues.

Edge Computing (EC) emerges as an alternative computing paradigm that rather complements CC. Unlike CC, EC relies on bringing computing resources closer to data sources, in other words, closer to the end-users [5]. Edge Data Centers (EDCs) are placed at the edge of the network, where data sources concentrate, offloading CDCs' tasks. These smaller EDCs have enough computing resources to process small-to-medium workloads. EDCs work as a gateway, connecting cloud facilities to edge devices in the event of more demanding tasks, helping to foster a more sustainable, scalable, and flexible model. Gartner predicts that by 2022, 75% of enterprise-generated data will be created and processed outside centralized CDCs, a massive leap from just 10% in 2018 [6].

This new paradigm presents many advantages [7]: (i) network congestion is reduced as EC encourages decentralization; (ii) latency plummets as EDCs are closer to the end-users than CDCs; (iii) security is improved as EDCs can further process information utilizing safer methods; and (iv) reliability is improved as decentralization in EC-based architectures avoids potential central outages.

1.1. Research challenges and motivation

Although EC solves several problems found in CC, it is not without its own challenges. Since EDCs are close to end-users,

* Corresponding author.
E-mail addresses: s.pmorillo@alumnos.upm.es (S. Pérez), p.arroba@upm.es (P. Arroba), jm.moya@upm.es (J.M. Moya).

https://doi.org/10.1016/j.future.2021.07.031

many of them will be placed in high-density urban areas. Thus, the space employed in these infrastructures must be minimized and the energy efficiency should be maximized to avoid potential power-grid issues. Furthermore, unlike in CC, where resources are concentrated in a few CDCs, EC requires many smaller EDCs. Hence, their unit cost should be decreased to scale the solution. Besides, their dimensioning should consider both end-users and applications in the area of their deployment to offer better and cheaper services. Note that while most CDCs are deployed in cold climates to improve their energy efficiency, EDCs are deployed near data sources, so they cannot benefit from suitable weather conditions in most cases.

Our research addresses these challenges in two ways. On the one hand, State-of-Art (SOA) cooling systems reduce both the required area and the operational and unit costs of EDCs since their space and energy efficiency are greatly improved. On the other hand, the intelligent and dynamic management of EDCs' computing resources also reduces energy costs and provides better services by adapting to the volume and location of the applications and end-users.

1.1.1. Improving edge computing cooling systems

Regardless of the computing paradigm, data centers have two primary energy contributors, the Information Technology (IT) systems and the cooling infrastructures [8]. On average, the latter accounts for 40%, oscillating between 24% and 61%. Historically, most data centers have built air-cooled systems to refrigerate their rooms. Although widely adopted, these systems are not the most efficient solution [9]. SOA cooling systems are based on immersion-cooled architectures, in particular two-phase immersion cooling [10]. In this cooling method, the IT equipment is submerged in a close bath full of a dielectric liquid. It aims to passively (i.e., with no energy consumption) flow the engineered fluid by boiling and condensing it. During operation, the electronic equipment evaporates the coolant. Then, the coolant vapor rises to the top, where a heat exchanger condenses it back to the liquid phase.

Its potential is unmatched [11], being able to reduce cooling energy consumption by 95%. While air solutions have a power density between 4 and 40 kW per rack, two-phase immersion solutions can get up to 250 kW per rack. It yields Power Usage Effectiveness (PUE) values around 1.02-1.03, close to the global minimum, and only achieved before with big IT companies' free cooling solutions. This strategy is also space-efficient as it has ten times less physical footprint than air solutions (100 kW/m^2 compared to 10 kW/m^2). Not only that, but thanks to the high temperatures required to cool equipment down, this method is practically independent of climatic conditions.

1.1.2. Improving edge computing resource management

On the other hand, resource allocation and dynamic management in data centers have always drawn Academia's attention [12,13]. Conventional methods are usually based on control theory [14]. However, data-driven solutions are gaining momentum, thanks to the rise of AI. As reported by Google [15], the use of these data-driven solutions in their data centers has yielded cooling and total energy savings of 40% and 15%, respectively. One of the most promising AI technologies is Deep Reinforcement Learning (DRL). It has been employed in essential fields such as games, robotics, health, or economics, achieving superhuman performance [16]. In resource management, DRL can outperform theory-driven solutions in large infrastructures by capturing better their ever-changing and extremely volatile nature [17].

These two strategies help to scale EC facilities further. Minimizing the needed space while maximizing the energy efficiency allows the deployment of more computing resources, thereby making the network less CC-dependent. Furthermore, complex and powerful systems such as large Graphics Processing Unit (GPU)-based architectures become viable in this scenario. Thus, it enables the adoption of advanced applications in EC, as those based on SOA Deep Learning (DL) (e.g., real-time Computer Vision (CV), Nature Language Processing (NLP), or IoT-related [18]).

Specific applications that would benefit vastly from EC are Advanced Driver Assistance Systems (ADAS). They aim to enhance road and vehicle safety by assisting drivers, who cause around 94% of accidents in the USA [19]. Many of these services utilize high-resolution cameras, and according to NVIDIA [20], vehicles with ten of these cameras produce, on average, two gigapixels per second. Moreover, those vehicles using DL applications may reach 250 TOPS (trillions of operations per second). This massive volume of data, together with the critical latency constraints of these applications [3], makes EC architectures a perfect fit. In fact, it is actively being researched in Academia, often referred to as Vehicular Edge Computing (VEC) [21].

1.2. Our proposed approach

To the best of our knowledge, there are not projects in the Literature that combines both SOA cooling systems (two-phase immersion cooling) and intelligent resource management (AI data-driven optimization) in the data center scope. Moreover, most works on EC center around the deployment of small nodes, often called Mobile Edge Computing (MEC) servers [22], instead of EDCs with enough computing resources for more demanding tasks like ADAS. Our research intends to contribute to this matter.

The research presented in this paper is focused on the design, implementation, and optimization of energy-efficient deployment and dynamic operation in realistic ADAS-based EC scenarios. It is achieved through the strategies mentioned above: two-phase immersion cooling and smart resource management via DRL. In the evaluated EC scenarios, the immersion-cooled EDCs obtained an average energy reduction of 22.8% compared to air-cooled ones. On the other hand, our devised DRL agent produces savings of up to 23.8% compared with the baseline.

This research's main contributions are summarized as follows:

- Energy-aware Edge Data Center modeling of two-phase immersion systems using real Edge Computing hardware prototypes with GPUs and an application based on ADAS (intelligent driving assistance) and Deep Learning.
- Energy-aware resource management optimization of realistic Edge Computing scenarios using Deep Reinforcement Learning, our two-phase immersion cooling models, and Mercury, a SOA 5G-Edge simulator, providing the most realistic simulation possible.

1.3. Organization of the paper

The remainder of this paper is organized as follows. Firstly, Section 2 gives further information on related work concerning our research. Section 3 presents the EC scenario to be optimized, featuring the two-phase immersion-cooled EDC model. Then, Section 4 covers the DRL-based resource allocation manager. Section 5 describes the experimental results. Finally, Section 6 presents the main conclusions of this work.

2. Related work

As mentioned in Section 1, the combination of two-phase immersion cooling and resource allocation optimization in EC scenarios has not yet been explored in current research. Some works deal with the former [23,24] and others with the latter [25,26]. Therefore, we introduce these two topics separately in the subsequent paragraphs. The summary of each part is found in Tables 1 and 2, respectively.

2.1. Cooling strategy in edge computing

As for cooling strategies in EC, most of them resemble the ones used in CC with a focus on space and energy efficiency. Historically, most data centers have built air-based cooling systems to refrigerate their rooms. The same goes for EC. Huawei [27] and Rittal [28], which already offer EDCs with air-based cooling methods. Hot aisle & cold aisle strategies [29] are the foundation of this type of solution. It consists in lining up the data center's racks to face cold air intakes and hot air exhausts alternatively, thereby forming cold and hot rows.

However, the most efficient air strategy is free cooling [30], which leverages favorable climatology of cold locations to cool down the data center's equipment. Cooling costs drop drastically since much of the cooling infrastructure is gone. Big IT companies like Facebook tend to locate their largest cloud centers in these areas, achieving unprecedented PUE factors [31], where PUE is the ratio between the total and IT energy (i.e., useful) energy costs in a data center. However, this is not viable for EDCs as location-dependent solutions limit the reach of EC by increasing the perceived delay. Moreover, studies suggest that free cooling solutions are prone to failures, hurting efficiency [32].

Air-based system's drawbacks outweigh the advantages [33], more so in EC, as free cooling is not feasible in all locations. Air flows lead to corrosion, vibration, and turbulence. Hence, potential failures. Air-cooling infrastructure is not space-efficient, critical in EC. Furthermore, the air heat transfer coefficient is the lowest among solutions in data centers.

On the other hand, water-based cooling systems solve these problems and have better transfer characteristics, thus offering higher energy savings. It might save around 50% of energy cooling costs compared to an air-based system [33]. However, it has a fatal flaw, leakages. Water leakages in data centers may be lethal, even more in EDCs, since they are deployed in densely populated areas.

One of the most important water-based systems is rear door cooling, which features both passive and active versions. It uses exchangers that remove heat from the source but rely on IT equipment's fans to push the airflow. They can handle up to 30 kW per rack in optimal situations, but it might be better to aim for lower levels to be cost-effective [34]. Direct water cooling (or cold plates) [35] offers the maximum power density per rack among water solutions, amounting to 80 kW in some commercial products. It also works at higher operating temperatures, thus saving energy costs. The problem lies in the need for specific hardware, which limits its repercussion.

Dielectric-liquid-based cooling is on the rise thanks to its outstanding potential that has put it into the spotlight. The solutions vary widely. Indirect cooling [36] resembles some water solutions as it evades direct contact with the equipment. Unlike those strategies, leakages are not such a thread anymore. However, indirect cooling does not take advantage of the heat removal potential of dielectric liquids since it is not in contact. Another disadvantage, as in water-based solutions, is that its design is specific to the equipment. Hence, it usually cannot be repurposed. Moreover, the design, which is centered around many pipes, makes the equipment challenging to manipulate.

The last dielectric-liquid-based subgroup and the most energy-efficient is Immersion Cooling. There are two main methods. First, one-phase immersion cooling [37] submerges the equipment in either an open or semi-open bath. The coolant remains in the liquid phase. This method leverages the heat transfer potential that the liquid has but needs an extensive recirculation system to cool it down, therefore worsening energy and space efficiency. Furthermore, the coolant needs to work at low temperatures to avoid evaporation. Hence, extra energy is needed for its correct functioning.

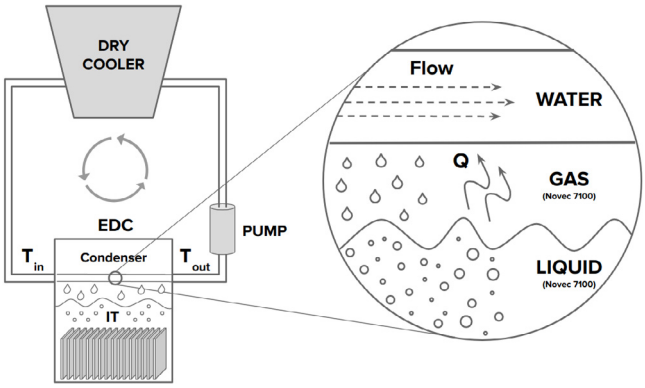

Fig. 1. Simplified two-phase immersion cooling tank.

This solution uses two types of coolants, mineral oils and engineered fluids [38]. The first ones are more widely used due to their market availability. These chemicals are petroleum-based distillates (mainly composed of alkanes and cycloalkanes) that are usually further processed with hydrotreatments. These compounds present many disadvantages. Even with the special treatment that adds resistance, oils are still flammable. Their viscosity hurts the heat transfer capabilities and an efficient recirculation through the cooling system. Mineral oils have high Global Warming Potential (GWP) (i.e., significant toxicity) and are not entirely biodegradable, just around 30%–40%. They are also not clean, thus hindering maintenance operations by data center operators. On the other hand, engineered fluids are specifically designed for tasks such as cooling. Even though they tend to be more expensive, they are usually non-flammable, cleaner, safer to operate, almost entirely biodegradable (+90%). Moreover, they also have low toxicity, a low GWP, and higher heat transfer coefficients.

Two-phase immersion cooling is the most promising method of all. As stated before [11], it can reduce cooling energy consumption by 95%. While air solutions have a power density between 4 and 40 kW per rack, two-phase immersion solutions can get to 250 kW per rack. It yields PUE values of 1.02-1.03, close to the global minimum, and only achieved before with free cooling solutions of large IT enterprises. The advantage over this air method is that it is geographically agnostic, vital for EC solutions. This strategy is also space-efficient as it has ten times less physical footprint than air solutions (100 kW/m^2 compared to 10 kW/m^2). This space reduction is also huge for the deployment of EDCs. Moreover, it does not need specific casing or equipment. Thus it can adapt better than other water or dielectric liquid solutions.

The coolant's recirculating method is the key to both outstanding energy and space efficiencies [39]. It aims to passively (i.e., with no energy consumption) flow the engineered fluid by boiling and condensing it, using high operating temperatures (more than 60 °C). During operation, the electronic equipment immersed in a closed bath evaporates the coolant. Then, the coolant's vapor rises to the top, where a heat exchanger condenses it back to the liquid phase. The heat exchanger placed inside the tank circulates water and transfers it to a dry cooler, where it is cooled down. This part is the only source of energy consumption in this method, and it is close to zero since the operating temperatures of both water and coolant are the highest among cooling solutions. Fig. 1 depicts a simplified version of a two-phase immersion cooling tank as an EDC to illustrate how it works better.

In the literature, most two-phase immersion cooling works focus on analyzing a prototype without any specific real-world

Table 1
Cooling systems overview.

Cooling System	Advantages	Disadvantages
Air-based Cooling	Most widespread systems that can harness climatology.	High energy and space costs, and location dependent.
Hot Aisle & Cold Aisle	Aisle distribution of separated hot and cold airflows.	Prone to undesired hotspots and turbulence.
Free Cooling	Outdoor heat exchange. Massive energy savings.	Needs favorable climatology and another cooling system for safety.
Water-based Cooling	Solves air problems like corrosion and vibration. Higher heat transfer coefficient.	Lower heat transfer coefficient than dielectric-liquid-based. Leakages are lethal.
Rear Door Cooling	Remove heat from the source using water.	Relies on server's fans. Not cost efficient.
Cold Plates	Ultra-low temperature water not needed.	High economic costs. Needs specific IT equipment.
Dielectric-liquid-based Cooling	Much more safe and energy efficient than water systems.	Still under development. Not widely used.
Indirect Cooling	No lethal leakages and better heat removal coefficient.	Does not leverage the liquid's heat removal potential (not direct contact).
One-phase Immersion Cooling	Heat transfer potential in open baths. Low pressures.	Needs low coolant temperatures using an external circuit. Still in development.
Two-phase Immersion Cooling	Passive method that saves up to 95% of cooling energy (most among all options).	Potential high pressures in closed baths. Still in development.

applications since this technology is still in development. Kanbur et al. [23] propose a system featuring a pump, a server tank, and a dry tower. Both thermal and economic analyses were carried out using workloads between 3.43 and 9.17 kW. The achieved PUE lies between 1.15-1.4. Wu et al. [40] build and analyze a similar system, but placed in a tropical environment instead. Like this research, they use the coolant Novec 7100. The authors again focus on agnostic workloads. On the other hand, our work shifts to applications in real-world settings and their optimization.

Outside of Academia, two-phase immersion cooling is also a work in progress. Not many companies have ventured to develop a market-ready product. BitFury and its wholly-owned subsidiary Allied Control, renamed in 2021 as Liquid Stack [41], developed in 2014 the award-winning DataTank, a container unit with a two-phase immersion-cooled rack of each for up to 252 kW. It has a cost of less than one dollar per watt and is designed for 19-inch rails. They claim to achieve a PUE of 1.02-1.03, and it has only been used for mining cryptocurrency since its release. This research wants to broaden the scope of a cooling system that presents many opportunities.

Though still a prototype, another solution is the Open Compute Project (OCP)-compliant two-phase immersion cooling tank by Wiwynn [42]. The tank has a thermal design power of 100 kW, although it was tested using just 60 OCP hardware nodes of 1 kW each. It achieves a PUE of 1.02. The tank was first showcased at the OCP 2019 Summit. Something that was not explored in either of these works is the engineered fluid's underlying heat transfer potential. The coolant must be explained first to understand why this property is crucial.

Both products and this research use Novec engineered fluids by 3M. In our case, the chosen coolant is 3M's Novec 7100. It is a hydrofluoroether with a boiling point of 61 °C, albeit a boiling point range between 50 °C and 99 °C. This dynamic feature is stressed in work presented by T. L. Bergman et al. [43]. It shows the relation between the heat transfer capacity and the surface temperature. There are four boiling regimes, and most solutions work in the first regime (free convection) due to unaware or conservative views. However, the coolant's true potential is way up high at the end of the second regime (nucleate). This fact could further enhance two-phase immersion cooling systems by making them more efficient in stable conditions.

Trying to work at maximum efficiency is challenging as surpassing that range would cause a sharp drop in performance. In addition, the last two regimes (transition and film) also have larger gas volumes making the coolant unstable. It is referred to as the boiling crisis. For this reason, predictive and proactive resource allocation is vital in EC scenarios that utilize two-phase immersion cooling. This issue leads us to the optimization of resource allocation. Combining both two-phase immersion cooling and smart resource management would boost the energy efficiency in EC architectures.

2.2. Resource allocation in edge computing

For many years, the optimization of resources has been explored in EC, CC, and data centers in general. Resource allocation strategies can improve the performance of many aspects (e.g., delay and energy) vastly. There exist many approaches to it [44,45]. We describe and compare several works that at least partially relate to what we want to achieve with this research.

M. Ghobaei-Arani et al. [26] propose a Learning Automata called ControCity that balances elasticity, delay, and utilization for a CC case study. Compared to other solutions, it improves elasticity and resource allocation by 5.4% and 8.4%. Another work from the same authors [46] addresses a Fog Computing (FC) case study closer to the EC case. They employ a moth-flame algorithm to optimize the quality of service and total execution time. Both papers make an effort to evaluate their solutions with a well-established simulator for CC and FC, respectively (CloudSim and iFogSim). They also use real-world traces to obtain the most realistic results possible. However, they do not consider energy costs that are relevant in contexts like these.

Regarding EC, some works apply DRL techniques to solve resource allocation problems in MEC-based applications like in our research. J. Xu et al. [47] propose a DRL algorithm to minimize delay and energy costs by offloading tasks to an energy harvesting MEC server. They mix online and offline learning to speed up training. X. Chen et al. [48] optimize a densely populated MEC scenario for offloading strategies based on task, energy queue state, and channel qualities between user equipment and access points. They devise a double Deep Q-Network (DQN) to solve it.

D. Zheng et al. [49] propose a model-free DQN-based resource manager for EC applications. It is not energy-aware as the only inputs are utilization and distance of edge servers. Z. Ning et al. [50] optimize a VEC application using a two-way algorithm, being one part a double DQN. They achieve a 90% execution time reduction for the non-DRL-based part and 15% for the DRL-based part compared with another DQN model.

Finally, R. Cárdenas et al. [51–53] present a heuristic algorithm in their work for the 5G-enabled and data-intensive simulator Mercury. For testing it, they envision an extensive and well-thought-out use case for high-performance EC with multiple EDCs and evaluate it in Mercury. Furthermore, they analyze the impact of their algorithm on energy efficiency thoroughly in their results. However, they lack a proactive approach to the energy problem, as their heuristics only take into account location-aware information (it assigns workloads to the closest EDCs). While this helps end-users reduce the delay they perceive, EC solutions already

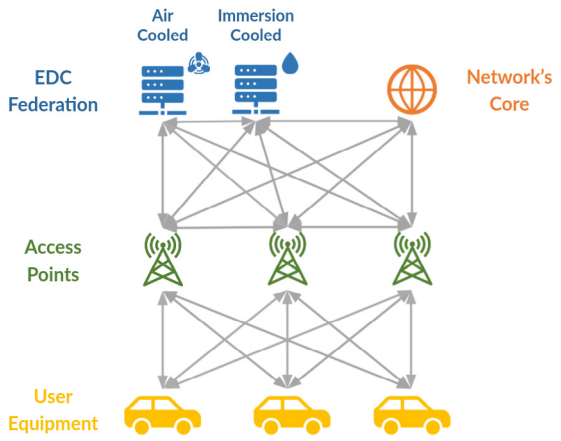

Fig. 2. The devised Edge Computing scenario.

achieve extremely low delays thanks to how they are devised. Thus, energy efficiency becomes as important as the perceived delay, if not more.

What makes our research different from the cited EC papers is the optimization objective and the EC scenario. On the one hand, their optimization focuses most times on the utilization of MEC servers. The only contemplated energy costs are derived from latency issues due to saturation. Instead, our research provides a holistic view of the energy consumed in EDCs, both in IT and cooling. On the other hand, their EC infrastructure concept is limited to independent MEC servers, often with limited CPU hardware. In our case, the EDCs' equipment includes multi-GPU architectures that can process bigger workloads.

For these two reasons, our research envisions a more realistic perspective of the energy problem in EC. These works usually employ agnostic applications to which they assign a utilization factor. On the contrary, this paper uses realistic IT and cooling energy models obtained with EC hardware data that runs a specific EC application. Hence, this work presents a comprehensive and energy-conscious approach to EC.

3. Edge computing scenario modeling

This Section covers the design and implementation of power consumption models required to simulate and optimize the energy efficiency in EC scenarios. To that extent, the devised EC scenarios are introduced first. This paper has designed a single Radio Access Network (RAN), as shown in Fig. 2. The User Equipment (UE) (i.e., the vehicles) runs an ADAS application and requests sessions to the EC network for processing the corresponding workloads. The network's core decides in which EDC are new incoming sessions processed. In the EDC, the sessions are stored and computed using one of its available Processing Units (PUs). The offloading model is akin to the Function as a Service (FaaS) concept [54]. For this purpose, resources are reserved temporarily, and only when UE requests them, the sessions are created. Communication-wise, the sessions are conveyed through the Access Points (APs). First, the UE interacts with the nearest AP. Then, the AP does so with the EDC assigned by the network's core. The assignment is performed by the Software-Defined Network (SDN) controller that uses a DRL-based agent that analyzes the energy status of the EDCs in order to make its decision.

Since we do not have access to an already deployed EC infrastructure, our focus remains on simulating and optimizing the most realistic EC scenarios possible. To this end, we leverage the potential of the EC simulator Mercury [51–53]. Mercury features many options to implement EC energy-related models while offering SOA 5G-enabled EC dynamics simulation. To obtain a realistic energy behavior of the scenario, Energy models of EDCs are developed and implemented in the simulator. This decision was made because their power consumption is by far the highest amongst the elements in the scenario, so it becomes crucial to model and optimize them.

To obtain these models, we have built different EDC prototypes and devised a real-world ADAS application. This application has been tested in these prototypes to analyze the energy profile. Consequently, the first step of the solution is to explain the ADAS application. To make its design realistic, it should be viable in an actual EC context. The ADAS application is based on one of our previous publications [55]. To this end, we devised a DL-based service that alerts drivers when they lose concentration on the road. A Convolutional Neural Network (CNN) estimates the driver's head position using video footage. If drivers spend an excessive amount of time looking to the sides, the algorithm warns them. Hence, it aims to improve road safety. The workloads generated by the vehicles and processed in the EDCs are for training the algorithm instead of making predictions. Training DL algorithms are known for being computationally expensive. There exist in both Academia and Industry many efforts to accomplish efficient DL training in EC-like layouts (see Federated Learning). Thanks to the offloading, the hardware in the vehicles would need less processing power, thereby lowering the economic costs. Moreover, the quality of the predictions would also improve as they can update more frequently with minimal delay. In fact, the information from different users in the network can be shared thanks to the EC layout.

3.1. Edge data center modeling

Now that the application has been defined, it needs to be tested on EC hardware (i.e., the mentioned prototypes) to analyze the energy behavior and obtain a model. This hardware would comprise the EDCs' PUs (i.e., the IT equipment) in an actual EC scenario (Fig. 2). Unequivocally, the key energy contributors in EDCs and data centers alike are the IT equipment and the cooling system. Therefore, our EDC power consumption model considers both sources (Eq. (1)). Others, such as room lighting, are neglected since their contribution is much lower and relatively constant. As introduced in Section 1, a key element of our research is two-phase immersion cooling. For this reason, we built one prototype based on this SOA cooling system and another one based on air cooling that serves as a baseline. The IT equipment and cooling system are modeled with data from EC-like hardware to achieve the most realistic simulations for each of them. Since the application is based on a DL algorithm, both prototypes employ GPUs as the IT equipment, which are well-suited for this type of workload.

$$P_{EDC} = P_{IT} + P_{cooling} \tag{1}$$

P_{EDC} = EDC power consumption [W]
P_{IT} = IT Equipment power consumption [W]
$P_{cooling}$ = cooling power consumption [W]

3.1.1. Air-cooled edge data center

The air-cooled prototype features a Sapphire Pulse Radeon RX 580 GPU as the IT equipment. The energy model was published in one of our previous publications [55], achieving a Normalized Root Mean Squared Error (NRMSD) of 99.01% and a Coefficient of determination (R^2) of 2.45%. The model was also already tested successfully in Mercury [51]. In short, the modeling process consisted of three well-distinguished parts: (i) the ADAS application running on the GPU for different configurations of the GPU's clock frequencies (main and memory) and number of parallel

Table 2
SOA resource allocation solutions.

Reference	Case Study	Technique	Metric	Tool	Advantages	Limitations
M. Ghobaei-Arani et al. [26]	Cloud Computing	Learning automata	Elasticity, delay, and utilization	CloudSim	Multi-objective optimization. Tested with real data traces.	No energy optimization. Cloud limitations (compared to Edge).
M. Ghobaei-Arani et al. [46]	Fog Computing	Moth-flame algorithm	QoS and total execution time	iFogSim	Execution and transfer time evaluation. Real-world case study. Tested with a Fog-specific simulation tool.	No energy optimization. No load balancing.
J. Xu et al. [47]	Mobile Edge Computing	DRL (DQN)	QoS and energy	Simulation (NA)	Comprehensive environment model. Energy harvesting. Auto-scaling. Fast convergence.	Cloud data traces. Generic energy consumption model. Lack of heterogeneous infrastructure.
X. Chen et al. [48]	Mobile Edge Computing	DRL (DDQN)	Task execution delay, drops, queuing delay, and failure penalty	Simulation (NA)	UE-based optimization for tasks. Comprehensive task model. Price implications.	Generic energy consumption models. Not tested with real-world applications. No real data traces.
D. Zheng et al. [49]	Edge Computing	DRL (DQN)	Operational costs	Simulation (NA)	Simple yet effective solution. High scalability.	No energy model. Ad-Hoc simulation tool.
Z. Ning et al. [50]	Vehicular Edge Computing	DRL (DDQN)	QoE	Simulation (Python)	Joint task scheduling and resource allocation optimization. Mobility-based layout.	No specific application. No real data traces. Ad-Hoc simulation tool.
R. Cárdenas et al. [51–53]	High-Performance Edge Computing	Heuristics	Delay	Mercury	Real-world use case. High performance IT (GPUs) energy models. Tested with a specific 5G Edge simulator.	No proactive energy approach. No cooling models, especially two-phase immersion.

sessions (several ADAS workloads running in the GPU at the same time); (ii) the monitoring system in the background that collects the GPU's status (e.g., power consumption, temperature, and utilization); (iii) and the power consumption model that fits the collected GPU data using the GPU's clock frequencies and the number of parallel sessions as inputs. The model is based on a FeedForward Neural Network (FNN) and summarized in Eq. (2).

$$P_{IT} = P_{GPU} = f(w_{ADAS}, freq_{main}, freq_{mem}) \tag{2}$$

P_{GPU} = GPU power consumption [W]
w_{ADAS} = ADAS workload (Sessions running concurrently in the GPU)
$freq_{main}$ = GPU's main clock frequency [Hz]
$freq_{mem}$ = GPU's memory clock frequency [Hz]

On the other hand, the cooling energy model of this EDC is based on a publication from J. Moore et al. [56]. It envisions the most conventional air cooling strategy, hot aisle & cold aisle. The model has only two parameters: the IT power consumption P_{IT} that depends on the user demand, and the equipment inlet temperature T_{inlet} that is fixed during the simulations. Eq. (3) shows the power consumption model adapted to our air-cooled system.

$$P_{cooling} = \frac{P_{IT}}{0.0068 \cdot T_{inlet}^2 + 0.0008 \cdot T_{inlet} + 0.458} \tag{3}$$

3.1.2. Two-phase immersion-cooled edge data center

The two-phase immersion-cooled EDC prototype employs a Sapphire Pulse Radeon RX 570 for its IT equipment. The characteristics are practically similar to the previous model. While the procedure to obtain the IT equipment's power consumption model is the same as in the air-based case, the prototype is entirely different to fit into a two-phase immersion cooling system. Fig. 3 shows the prototype that was utilized to obtain the power consumption model.

Fig. 3. The modeled two-phase immersion-cooled prototype.

This prototype features a heat exchanger mechanism, unlike in our previous publication [57]. The system circulates cold water that passes through two liquid cooling blocks. On their surface, the gas originated from the coolant's evaporation during runtime is condensed. The main GPU chip has a graphite and copper sheet

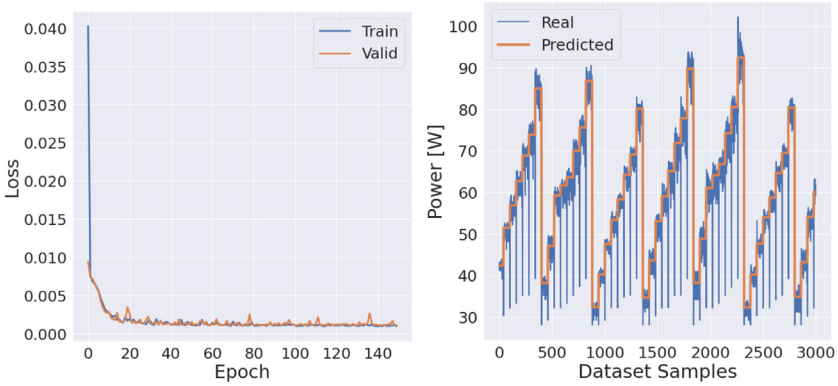

Fig. 4. Energy model's loss curves and predictions of the immersion-cooled EDC's IT equipment.

Fig. 5. Front (left) and rear (right) views of our full-size EDC currently in development.

Table 3
Hyperparameters of the immersion-cooled EDC's IT equipment power consumption model.

Hyperparameter	Value
Hidden layers	x1 FeedForward
Neurons	512
Train-Val-Test split	70%, 20%, 10%
Scaling method	Min–Max
Scaling range	[0,1]
Batch size	128
Epochs	150
Loss function	MSD
Activation function	ReLU
Optimizer	Nadam
Learning rate	0.001

mechanically attached to distribute the heat better. There were no gas leakages around the sealing, and the coolant could reach 61 °C (i.e., the boiling point).

Table 3 lists the final model's hyperparameters. The results show an NRMSD of 3.15% and an R^2 of 97.97%. On the other hand, Fig. 4 depicts the training and validation curves (left) and the power consumption predictions using some samples from the test set (right). It presents stable training without overfitting issues. As the model's inputs comprise three discrete inputs, the model's output is also discrete.

Two-phase immersion cooling systems consume very little energy. The only significant element that needs consideration is the pump that recirculates the heat exchanger's water. Therefore, the cooling system energy model is based solely on this element. This research models the pump Wilo IPL 50/115-0,75/2 installed in our full-size EDC that we are developing for this project [58] (Fig. 5).

The pump's power consumption model employs the information found in its datasheet. In this documentation, the pump's shaft power P_s and efficiency η_m are expressed as a function of the flow rate f. The pump's power consumption P_m is derived from these two variables, as in Eq. (4). Fig. 6 shows this power consumption in terms of the flow rate. Pumps usually achieve their highest efficiency around the middle of the flow range, coinciding with the impeller at its full size. Pumps are meant to work at maximum efficiency, not just to save energy but also to avoid potential damage because of saturation.

$$P_{cooling}\ [W] = P_m = \frac{P_s(f)}{\eta_m(f)} \tag{4}$$

The last modeling step is to compute the flow rate as a function of the heat dissipation in the IT equipment and the desired temperature difference between the heat exchanger's input and output in the two-phase immersion EDC. Fig. 1 in Section 2 depicts this architecture. The dissipated power Q is the heat generated on the GPUs' chips surface (i.e., the IT power consumption). Since the IT power consumption depends on the IT demand, the only control variable is the temperature difference $\Delta T = T_{out} - T_{in}$. Eqs. (5) to (7) show how to compute the flow

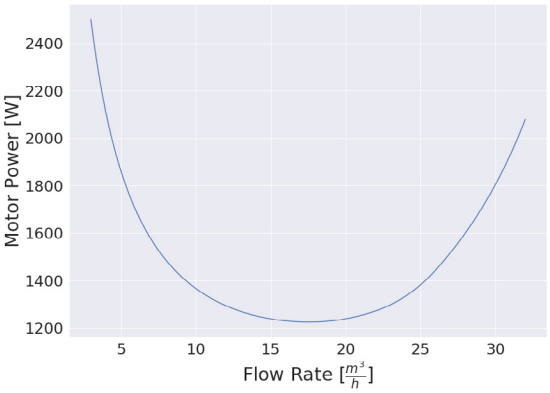

Fig. 6. Pump's power consumption P_m as a function of the flow rate f.

rate f as a function of the temperature difference ΔT through a simplified scenario using a circular pipe.

$$Q = f_m \cdot C_p \cdot \Delta T \tag{5}$$

$$f = \frac{f_m}{277.78 \cdot \rho} \tag{6}$$

$$f = \frac{Q}{277.78 \cdot \rho \cdot C_p \cdot \Delta T} \tag{7}$$

$Q = P_{IT}$, power dissipation in the IT equipment [W or J/s]
f_m = mass flow rate [g/s]
C_p = specific heat capacity [4.18 J/g° K]
$\Delta T = T_{out} - T_{in}$, temperature difference [° K]
f = flow rate [m^3/h]
ρ = density [1 g/cm^3]
1 m^3/h = 277.78 cm^3/s

4. Deep reinforcement learning-based resource allocation manager

All elements in the scenario but the network's core have been defined already. As mentioned in Section 3, the core's role is to map the AP-EDC connections dynamically. So, allocating computing resources in the EDCs to process the vehicles' workloads is done according to this mapping. The goal of this mapping is to obtain the highest energy efficiency possible. As stated in Section 1, our key contributions to the energy problem in EC are two-fold: two-phase immersion cooling (discussed in Section 3) and smart resource allocation via DRL. In this Section, the resource manager of the network's core is devised as a DRL agent using an Advantage Actor–Critic (A2C) model.

4.1. Problem description

During the simulation, several vehicles are driving around a city using the DL-based ADAS application. These vehicles request computing resources to the EC network for processing the training of the DL algorithm. Each of these requests generates a session in the EC network. Then, the network's core allocates these sessions through the APs to the available GPUs in the EDCs to compute them. Once a session has finished, the results are sent back to the corresponding vehicle, if needed. To decide which EDC processes each session, the core updates the AP-EDC connections dynamically based on the energy status of the EDCs. Since the resources (i.e., the GPUs) in the EDCs are limited and the power consumption behaves non-linearly, the network's core is in charge of efficiently allocating these sessions to optimize the energy efficiency in the network. To learn more about the scenario and simulation dynamics, there exist other works that address them thoroughly [51–53].

4.2. Deep reinforcement learning formulation

As explained in the previous Section, during the EC simulation, the resource manager via the SDN controller updates periodically and one-by-one the connections between APs and EDCs. To decide how to assign them, it receives the energy status of the EDCs. This information features metrics, such as power consumption or utilization. Therefore, the DRL agent's goal is to manage the AP-EDC routing using the information from the EDCs to optimize energy efficiency.

The foundation of DRL is Reinforcement Learning (RL). In RL problems, an agent interacts with an environment in discrete timesteps. In each timestep, it takes an action based on the current state of the environment, moves to a new state, and receives a reward. The goal is to maximize the cumulative reward. Formally speaking, it can be described as a Markov Decision Process (MDP). As in any other Markov Process, it obeys the Markov property (i.e., a process depends only on the present state). It is formed by a 5-tuple, $\langle S, A, P, R, \gamma \rangle$, where:

- S is the set of states of the environment. In our problem, the states are information about the EDCs' status. For each EDC status EDC^i_{st}, the information is comprised of: the IT power consumption P^i_{IT}, the cooling power consumption $P^i_{cooling}$, the total power consumption P^i_{EDC}, and the utilization factor u^i. The state space for N EDCs is formulated as:

$$S = \{s = (EDC^0_{st}, \ldots, EDC^i_{st}, \ldots, EDC^N_{st})\} \tag{8}$$

$$EDC^i_{st} = \{P^i_{IT}, P^i_{cooling}, P^i_{EDC}, u^i\} \tag{9}$$

- A is the set of actions the agent may take. The actions correspond to the selection of an EDC (EDC^i) to pair a certain AP. Thus, the action space is comprised of all the EDCs in the scenario. For N EDCs, it is defined as follows:

$$A = \{a = (EDC^0, \ldots, EDC^i, \ldots, EDC^N)\} \tag{10}$$

- $P : S \times A \times S \rightarrow \mathcal{P}(S)$ is the transition probability matrix. Each value is a probability of going from one state s to another s' after taking action a. It is a property of the environment, the EC scenario. Since our DRL algorithm is model-free, it does not learn the state-transition probabilities and only samples from the environment.
- $R : S \times A \times S \rightarrow \mathbb{R}$ is the reward function. Our idea is to minimize the total power consumption to maximize energy savings in the scenario. Consequently, the reward function for timestep t and N EDCs that minimizes the consumed energy is denoted as in Eq. (11). Note that for a state–action pair, the reward is based on the state that follows.

$$r_t = R(s_t, a_t, s_{t+1}) = -\sum_{i=0}^{N} P^i_{IT,t+1} + P^i_{cooling,t+1}$$
$$= -\sum_{i=0}^{N} P^i_{EDC,t+1} \tag{11}$$

- $\gamma \in (0, 1)$ is the discount factor for balancing the importance of immediate and future rewards, and converging the cumulative reward, return R, to finite values. In our case, the discounted return is applied for calculating the return of a trajectory T (i.e., a sequence of state–action pairs). The return for a timestep t is as follow:

$$R_t = R(s_0, a_0, \ldots, s_t, a_t) = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'} \tag{12}$$

To reduce energy consumption, we now need an agent that follows a policy based on this MDP. The algorithm and its architecture are thoroughly explained in Section 4.3. Nevertheless, let us first clarify how the agent can attain its energy-conscious goal with an example that combines the scenario dynamics explained in Section 3 with this DRL formulation.

In the scenario, we have an EC network with EDCs and APs distributed across a populated region. Each EDC features a fixed number of GPUs that can store up to a predefined number of sessions. On the other hand, drivers around this region use the ADAS application described in Section 3. Each vehicle runs a DL-based model whose inferences assist the driver during their rides. The users send petitions to the EC network periodically for training the application's algorithm. The 5G protocols that use these petitions and sessions are described in-depth in previous works [51–53]. Once the request is accepted, the network's core creates and assigns a session to an EDC. During the session, the user sends the images collected during their drive to the EDC's GPU, where the training happens. Once the training is done, if the new model shows a better performance than the onboard one, it is sent back to the vehicle. In parallel to this, the network's core awaits new incoming petitions.

It is here, in the core, where the agent's goal is to interact in the simulation by selecting an EDC to send the sessions (action a_t) each timestep t. This action a_t enables the connection to the selected EDC via the APs. The policy's decision is based on the energy information of the EDCs (state s_t). After the action is taken, the agent and simulator move to the next timestep $t+1$, when interacting again with a new energy situation in the EDCs (state s_{t+1}). It also receives a reward r_t for the action it previously took. Note that the frequency of new timesteps t is high enough so that the core is aware of all new incoming sessions.

This procedure is repeated until the end of the simulation. We collect all interactions during the entire process (i.e., states, actions, and rewards) to update the agent's policy before rerunning it again (Section 4.5 explains how the update works). It is expected that with enough simulations and a reward based on minimizing the energy consumption, the agent will learn a good policy for distributing the sessions in the EDC in an energy-conscious manner.

4.3. Advantage actor–critic agent

Agents must act accordingly to a policy to try to maximize the return. A policy is a rule that the agent obeys to take an action based on a state. As stated before, the DRL model employed is an A2C from the actor–critic family. In actor–critic algorithms, the actor selects the action to take, while the critic assesses how good the selection was. These methods are also classified as policy gradients since the actor uses gradient ascent to update the parameters of a parametrized function, the policy $\pi_\theta(s,a) = \mathbb{P}[a \mid s, \theta]$, that maps states to action. In A2C models, the function's parameters are updated for maximizing the return as in Eq. (13).

$$\Delta\theta = \alpha \nabla_\theta log \pi_\theta(s,a) A_{w,v}(s,a) \tag{13}$$

Where (i) $\Delta\theta$ is the parameters update; (ii) α is a step-size parameter, (iii) $\nabla_\theta log \pi_\theta(s,a)$ is the score function that shows the direction in which the gradient moves (i.e., the actor); and (iv) $A_{w,v}(s,a) = Q_w(s,a) - V_v(s)$ is the advantage function that indicates how good it is to take a specific action in a state, the action-value function $Q_w(s,a)$, compared to the overall state value, the value function $V_v(s)$ (i.e., the critic). These two new functions are also parametrized with parameters set v, w. In practice, the action-value function $Q_w(s,a)$ can be computed using the return (our case) or Temporal Difference (TD) Learning. The remaining parameters $V_v(s)$ are updated by minimizing the mean squared error of $A_v(s,a)$.

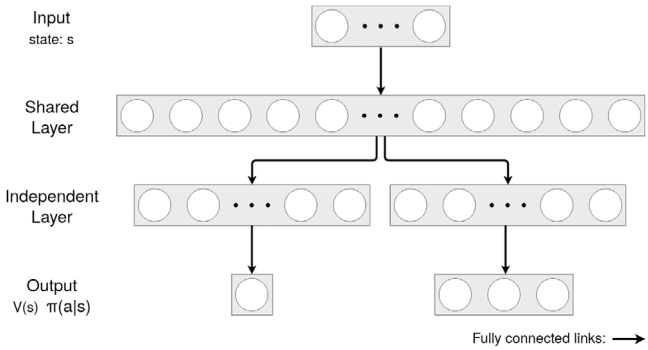

Fig. 7. Two-headed FeedForward Neural Network.

Table 4
Two-headed FeedForward Neural Network's hyperparameters.

Hyperparameter	Value
Input size	12 (state space)
Output size	3+1 (action space + $V(s)$)
Hidden layers	x1 shared, x1 independent
Neurons	128, 64
Activation function	ReLU, Softmax

DRL agents utilize Deep Neural Networks (DNNs) as the parametrized function. DL models work better with non-linear high-dimensional spaces than regular Machine Learning (ML) models [16]. In A2C models, the DNN receives the state and outputs the probability distribution of actions from which a single action is sampled. This procedure enhances exploration. The devised DNN is a two-headed FNN with parameters θ (Fig. 7). This architecture outputs the probability distribution of actions $\pi(s \mid a)$ and the critic's value function $V(s)$. In practice, both $\pi(s \mid a)$ and $V(s)$ can be obtained from the same DNN. The two outputs also share the first layers in which the gradient is propagated jointly. Although this method might seem unusual, it was successful in renowned RL projects such as AlphaGo [59]. Table 4 shows the two-headed FNN's hyperparameters.

The loss function $\mathscr{L}$ is comprised of both the policy loss $\mathscr{L}_{policy}$ and value (or advantage) loss $\mathscr{L}_{value}$. The policy loss is the A2C update term (Eq. (13)), and the value loss is the mean squared error of the advantage function. In practice, an additional entropy term $\mathscr{L}_{entropy}$, which enhances exploration, is also added. The entropy loss is computed for the probability distribution of the actions, $\pi(A_t \mid s_t, \theta_\pi)$. This term penalizes unbalanced distributions to avoid always selecting the same actions. A constant β is applied to balance this term that should not dominate over the other losses. Eq. (14) shows the final loss term, and Eq. (15) describes it for timestep t. Note that the policy and entropy losses should be maximized while the value loss should be minimized. Thus, the signs of the first two loss terms are adjusted since the selected DL framework minimizes losses. This loss is computed for each timestep and then accumulated during a trajectory. Finally, with a learning rate α equal to 0.001, the optimizer Adam updates the parameters using the accumulated mean loss for the entire trajectory (Eq. (16)).

$$\mathscr{L} = \mathscr{L}_{policy} + \mathscr{L}_{entropy} + \mathscr{L}_{value} \tag{14}$$

$$\begin{aligned} \min \mathscr{L}_t = & - log(\pi(a_t \mid s_t, \theta_\pi)) \cdot (R_t - V(s_t)) \\ & - \beta \cdot \pi(A_t \mid s_t, \theta_\pi) \cdot log(\pi(A_t \mid s_t, \theta_\pi)) \\ & + 0.5 \cdot (R_t - V(s_t))^2 \end{aligned} \tag{15}$$

$$\Delta\theta_\pi = \alpha \nabla_{\theta_\pi} \bar{\mathscr{L}} \tag{16}$$

4.4. Problem complexity

Even though DL-based solutions such as DRL have shown promising results in a wide range of applications, they do not guarantee global convergence and tend to be time-consuming and computationally expensive. Therefore, they usually need careful tuning of both problem and algorithm parameters (often called hyperparameters) to yield satisfactory results. In our case, we focus on a few key elements to reduce the complexity of our setting to facilitate fitting the DRL model.

First and foremost, we consider the MDP formulation. Exploring a large state space can potentially slow down the training process due to the curse of dimensionality. So, although the simulator Mercury offers other information that can be modeled within the states, such as delays, we prefer to utilize just energy-related features. Something similar happens with the action space. It can add unnecessary overhead to the problem. We have devised it as efficiently and straightforward as possible by mapping the actions to the EDCs. Reward schemes vary widely, from sparse values at the end of episodes to continuous values based on several variables. We found out that the quickest and most stable way for the algorithm to converge to good results is to simplify the reward function as much as possible. Consequently, we use the most meaningful metric to our problem: the total energy consumption of the scenario as the reward.

Then, it is also important to take into account the DRL algorithm selection based on our requirements. The chosen algorithm (i.e., A2C) belongs to the actor–critic family. These models combine both policy-based and value-based methods. In DRL Literature, they often antagonize with Q-learning as both are the foundation of the most advanced algorithms in the field, such as Proximal Policy Optimization (PPO) or Soft Actor–Critic (SAC). There is a trade-off between the two approaches. Among many aspects, the former has better convergence properties, while the latter is more sample-efficient [60]. This work utilizes actor–critic algorithms for several reasons. First, the simulator Mercury produces cheap data samples, so being sample-efficient is less of a concern. Second, the improved stability helps to find robust solutions easily, which is a common pitfall in DRL models. Third, the implementation is more straightforward than Q-learning (e.g., it does not require a replay buffer), helping avoid potential code-related issues.

Last but not least, the DNN-based policy runs into the same issues as other DL solutions. Specifically, bigger networks can result in better performance at the expense of slower training and inference times. The selected architecture aims to reduce both the simulator's speed performance and the model convergence as much as possible. Hence, it is relatively small without sacrificing performance. In our experiments, larger neural networks did not yield noticeable improvements.

On the other hand, analyzing the time complexity is rather unfeasible due to the mentioned convergence properties of DL-based models. However, our problem could at least be categorized as an online bin packing problem [61], which is NP-hard. In our environment, the GPUs hosted by the EDCs would be the bins or containers. Furthermore, the sessions from the vehicles would be the items. These items have to be distributed to fit into the containers to minimize, in our case, the energy consumption. Recent works [62,63] suggest that DRL solutions, such as the one proposed in this work, improve the performance of existing methods based on heuristics in online bin packing problems.

4.5. Training procedure

After formulating the problem and the A2C agent, the next part is to devise the training procedure to fit the two-headed FNN's parameters and achieve the best results (i.e., the highest returns). Algorithm 1 summarized this process. The first element to consider is feature scaling. It has a notable impact on speed convergence in DL models. Since each simulation results in a different data distribution, our feature scaling strategy utilizes fixed scaling parameters to perform standardization. So, before training the A2C model, a first simulation is run to obtain the scaling parameters, the mean μ and variance σ, used during the entire training process for standardizing the DNN input (i.e., the state).

Table 5
Model's training hyperparameters.

Hyperparameter	Value
Feature scaling	Fixed standardization
Parallel simulations	3
Discount factor γ	0.99
Return & Reward	Standardization
Loss $\mathscr{L}$	Equation Eq. (14) & Eq. (15)
Entropy term β	0.001
Optimizer	Adam
Learning rate α	0.001
Batch size	Sum of trajectory lengths
Total episodes	Meeting convergence criterion
Convergence criterion	Max reward or train time

After obtaining the scaling parameters, the DNN's weights and biases are initialized randomly. Then, several parallel simulation instances are launched. This method reduces the variance found in the model. The number of parallel simulations was set to three since higher values did not show significant improvements. Finally, the DNN model is broadcasted to these simulations. The trajectory (the sequence of states, actions, and rewards in each timestep) is stored for later use in each simulation.

Once all simulations are over, the return of each timestep of each trajectory is computed (Eq. (12)). The discount factor γ is set to 0.99 to give similar importance to immediate and future rewards. This way of computing the return is usually enough, but its standardization for each trajectory can speed up the convergence. It might also mess up the training procedure, but our experiments showed that it was beneficial. The reward is also standardized. Without this procedure, the return is biased towards the end of the trajectory due to the rewards being inherently negative (Fig. 8).

After obtaining the returns, the losses are computed and the parameters updated as explained in Eqs. (15) and (16). The process from the simulation launching to the parameter updating constitutes an episode. Episodes are repeated until the convergence criterion is met. This criterion is either achieving the maximum reward over several consecutive episodes or finishing predefined training times to avoid excessively long procedures. Table 5 shows a summary of the final hyperparameters. Our devised model is model-free, on-policy, online, and an actor–critic within the RL taxonomy. Several elements such as rewards, returns, loss terms, and probability distributions were monitored to evaluate the whole process properly during the training. Fig. 9 illustrates some of these monitored elements.

5. Results

So far, this paper has presented models for the elements that comprise the devised EC scenario. From these models, three different scenarios are proposed for their energy optimization. Firstly, an air-based scenario that only includes air-cooled EDCs. This configuration corresponds to the current state of deployments of EC infrastructure. Then, a heterogeneous scenario in which the two types of EDC, air-cooled and two-phase

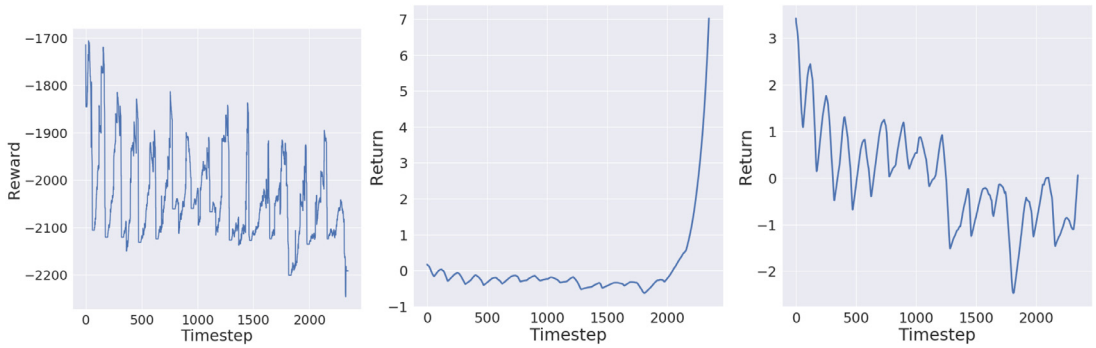

Fig. 8. Return without (center) and with (right) prior reward (left) standardization.

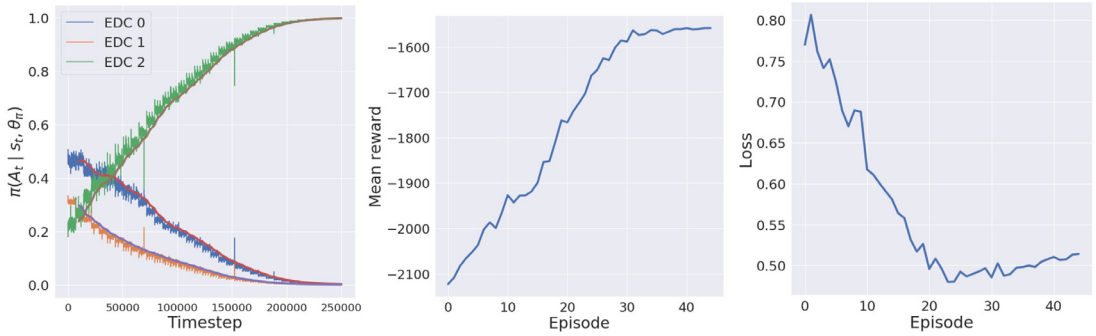

Fig. 9. Some monitored elements from a sample training procedure: probability distribution of actions over consecutive simulation timesteps (left), mean reward (center) and loss (right) over episodes.

Algorithm 1 A2C model training procedure

Preliminary simulation to obtain feature scaling parameters μ, σ
Initialize DNN's parameters θ_π randomly
while maximum reward or training time not reached **do**
  Launch N parallel simulations for a preset time
  Broadcast DNN to parallel simulations
  while parallel simulations running **do**
    for parallel simulation, timestep t **do**
      Collect states s_t, actions a_t, and rewards r_t
    end for
  end while
  Initialize loss: $\mathscr{L} \leftarrow 0$
  for simulation's trajectory T **do** ▷ $T = \{t_0, t_1 \ldots, t_T\}$
    Standardize rewards r_T
    for timestep t **do**
      Compute return: $R_t = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'}$
    end for
    Standardize returns R_T
    for timestep, t **do**
      Compute loss: $\mathscr{L}_t = \mathscr{L}_{p,t} + \mathscr{L}_{v,t} + \mathscr{L}_{H,t}$
      $\mathscr{L}_{p,t} = -log(\pi(a_t \mid s_t, \theta_\pi)) \cdot (R_t - V(s_t))$ ▷ $a_t \in A_t$
      $\mathscr{L}_{v,t} = +0.5 \cdot (R_t - V(s_t))^2$
      $\mathscr{L}_{H,t} = -\beta \cdot \pi(A_t \mid s_t, \theta_\pi) \cdot log(\pi(A_t \mid s_t, \theta_\pi))$
      $\mathscr{L} \leftarrow \mathscr{L} + \mathscr{L}_t$
    end for
  end for
  Update DNN's parameters: $\theta_\pi \leftarrow \theta_\pi + \alpha \nabla_{\theta_\pi} \bar{\mathscr{L}}$
end while

immersion-cooled, share the stage, showing the latter's gradual adoption. Finally, a two-phase immersion-based scenario in the same fashion as the first scenario but using two-phase immersion-cooled EDCs. This last scenario illustrates the final and complete adoption of immersion-cooled solutions in the future. Moreover, we have carried out several hypothesis tests to evaluate the robustness of our solution and results. Before presenting them, we introduce the simulation configuration and evaluation criteria for reproducibility and previous works that serve as the baseline for comparing our model.

5.1. Simulation configuration

The configuration of the EC network's elements, presented in Sections 3 and 4 , is based on previous publications [51], which also aimed for realistic simulation of EC scenarios. In short, the location of the scenario is in the San Franciso Bay Area since UE (i.e., the vehicles) mobility traces are from taxis circulating in this area. The number of vehicles is fixed at 50 to avoid increasing training times excessively while still offering appealing layouts. Aside from UE, there exist three EDCs and ten APs in the scenario. Their location is the same as in the cited works [51]. The number of PUs (i.e., the GPUs) per EDC varies amongst 5, 10, and 15 to analyze how the size affects energy efficiency.

Each PU within each EDC processes up to 4 (immersion-cooled) or 5 (air-cooled) parallel sessions, which corresponds with the number of parallel sessions used for training their models. The models are different due to some issues related to the hardware and DL platform. Bear in mind that more parallel sessions do not translate into better performance since the processing speed is inversely proportional to this number [55]. This parameter is configured via the PU utilization factor (25% for immersion-based and 20% for air-based). In scenarios with both EDCs, the utilization factor is set to 25% to compare them better. The other PU parameters, the main and memory clock frequencies, are set to their maximum value in each GPU to offer their best performance. Once the resource allocation manager

Table 6
Configuration of simulation parameters.

Parameter	Value	Parameter	Value
Scenario Location	San Francisco Bay Area	User Equipment	50 vehicles
Edge Data Centers	3	Access Points	10
EDCs & APs location	See work [51]	Processing Units	5,10,15 GPUs
Dispatch strategy	*Maximum*	Hot Standby	2 GPUs
GPU main clock	$freq_{max}$	GPU memory clock	$freq_{max}$
Utilization (immersion)	25%	Utilization (air)	20%
Utilization (both)	25%	Other parameters	Default [64]
Mercury version	*Lite*	Simulation time	180 s

has sent a session to an EDC, it is allocated using the dispatch strategy *Maximum*, which puts new sessions in the GPU with the highest utilization that can still host a session. This strategy yielded the best results energy-wise in previous works [51,55]. In addition to this strategy, there are always two GPUs in hot standby (i.e., switched on but not hosting sessions) to improve the users' perceived delay.

This work employs the version *lite* of Mercury, which contributed to a massive reduction in training time. The simulation time is set to 180 s, enabling the computation of 30 episodes in 45 min on average. Table 6 lists the configuration of the parameters above. The parameters of Mercury not mentioned in this Section are set to default values [64].

5.2. Baseline and evaluation criteria

As explained in Section 2, most works do not picture their algorithms as a holistic solution to the energy problem in high-performance EC scenarios. For instance, cooling consumption, a key energy factor in this technology, is often overlooked. Moreover, the focus is usually on theoretical formulations of the IT consumption, unlike our empirical GPU models, which are based on real-world workloads. For these reasons, we compare our DRL algorithm to the papers that have addressed the EC problem with a clear energy-oriented use case, including location awareness and high-performance IT consumption modeling [51–53].

Regarding the baseline, their authors propose a heuristic algorithm in which they measure the distance between the drivers and the EDCs. In short, it assigns the sessions conveyed through the APs to the closest EDC possible. While this approach can potentially reduce the perceived delay by the users, the delay is already extremely low in EC layouts. Moreover, they lack a proactive approach for the energy contribution in their heuristics. It is worth noting that they do analyze the energy impact thoroughly as a consequence of their location-aware model. In the EC scenarios, we assess how much energy can be saved using our DRL agent.

As for the evaluation criteria, the main evaluation metric is the total power consumption in the scenario P, which comprises the sum of the IT power consumption P_{IT} and cooling power consumption $P_{cooling}$ of the tree EDCs P_{EDCs} (Eq. (17)). The mean $\bar{P}$ and peak P_p values are computed to evaluate the results. Moreover, the PUE is also used. Its purpose is the assessment of the cooling system's efficiency. It is computed as the total power consumption P divided by the IT power consumption P_{IT} (Eq. (18)). Better values are closer to one. Note that the *PUE* does not reflect the overall energy optimization in the scenario but the cooling system's efficiency. So, for instance, if the IT power consumption is optimally reduced while maintaining the same cooling consumption, the PUE would worsen. The metric helps to assess whether our cooling consumption models behave as the SOA suggests. The mean value $\bar{PUE}$ is employed for this metric.

$$P\ [W] = P_{EDCs} = P_{IT} + P_{cooling} \tag{17}$$

$$PUE = \frac{P}{P_{IT}} = \frac{P_{IT} + P_{cooling}}{P_{IT}} \tag{18}$$

Table 7
Air-based scenario results.

Strategy	*GPUs*	$\bar{P}\ [kW]$	[%]	$P_p\ [kW]$	[%]	$\bar{PUE}$	[%]
Baseline	5	1.86	−13.12	1.94	−13.34	1.31	-
Energy		1.61		1.68		1.31	
Baseline	10	2.23	−15.47	2.40	−20.41	1.31	-
Energy		1.88		1.91		1.31	
Baseline	15	2.22	−6.19	2.40	−10.77	1.31	-
Energy		2.09		2.14		1.31	

5.3. Simulation results

All experiments were executed using an ASUS TUF Gaming FX505GD laptop with an Intel-Core i7-8750H CPU 2,2 GHz processor and a 16 GB 2667 MHz DDR4 memory. The DRL agent (PyTorch) and the GPU models (TensorFlow) were implemented in the CPU. Most training times ranged from 30 minutes to 2 hours.

5.3.1. Air-based scenario

The air-based scenario features three air-cooled EDCs. Their energy model, which was devised in Section 3, has a unique parameter that needs to be configured. The air cooling system model includes one control variable, the equipment inlet temperature T_{inlet} (Eq. (3)), which usually varies between 16 °C and 27 °C [65]. The final value was set to 20 °C, yielding a PUE of 1.31. The model is reasonably optimistic since the average PUE in today's data centers is around 1.58, with frontrunners such as Google and Facebook achieving values lesser than 1.1 [66]. As stated in the previous Section, the parameter GPUs per EDC, or just *GPUs*, is tested for three different values to assess how the EDC dimensioning contributes to the scenario's total energy consumption. Table 7 lists the results. It shows the mean $\bar{P}$ and peak P_p power consumption, and the mean PUE $\bar{PUE}$ for the baseline and our energy-aware model. Each pair's relative improvement is also provided to compare both strategies seamlessly.

All configurations yielded noticeable energy savings, varying from 6.19% to 15.47% in mean power consumption. The best results are found in the medium-sized EDCs closely followed by the small-sized ones. This slight difference in the outcome arises from their size since the smaller EDCs have less room for improvement. The scenario with the most equipped EDCs presents the most moderate results. This fact is related to the GPUs in hot standby, which usually remains completely unused while consuming energy as larger EDCs are less prone to saturation. Thus, this scenario hardly leverages the GPUs in hot standby of underutilized EDCs.

On the other hand, the PUE does not fluctuate since the air cooling model is linearly proportional to the IT demand (Eq. (3)). Fig. 10(a) illustrates the power consumption in the best scenario, ten GPUs per EDC, over the simulation's duration. During almost the entire simulation, our energy-aware model consumed less energy than the baseline. Closer values are found during the simulation's start.

Table 8
Two-phase immersion-based scenario results.

Strategy	*GPUs*	$\bar{P}$ [*kW*]	[%]	P_p [*kW*]	[%]	$\bar{PUE}$	[%]
Baseline	5	1.33	−4.87	1.39	−0.25	1.02	+0.24
Energy		1.26		1.39		1.03	
Baseline	10	1.64	−6.60	1.79	−7.54	1.05	+1.61
Energy		1.53		1.65		1.06	
Baseline	15	1.72	−0.56	1.88	−2.13	1.10	−1.36
Energy		1.72		1.84		1.09	

Table 9
Heterogeneous scenario results.

Strategy	*GPUs*	$\bar{P}$ [*kW*]	[%]	P_p [*kW*]	[%]	$\bar{PUE}$	[%]
Baseline	5	1.80	−7.67	1.86	−0.69	1.24	−1.84
Energy		1.67		1.85		1.22	
Baseline	10	2.31	−23.75	2.51	−26.30	1.27	−10.26
Energy		1.77		1.85		1.14	
Baseline	15	2.34	−22.61	2.53	−21.89	1.29	−14.93
Energy		1.81		1.98		1.10	

5.3.2. Two-phase immersion-based scenario

As with the air-based scenario, there exist some unique parameters. The cooling model includes one control variable, the temperature difference ΔT (Eq. (7)). As stated in Section 3, the model is based on the pump installed in our full-size EDC. This EDC is designed for workloads up to 50 kW. We are simulating scenarios with a maximum of 15 GPUs, each with a peak power consumption of around 100 W. So, while the real-world system would need a ΔT of a few degrees, our simulated EDC needs less than 1 °C to work in the optimal range (Fig. 6). Besides this parameter, a power reduction factor γ is also applied to the model for adjusting its output to our small-scale scenario. The final cooling power consumption employed in the simulations $P^{sim}_{cooling}$ is computed as follows:

$$P^{sim}_{cooling}\,[W] = \gamma \cdot P_{cooling} = \frac{GPUs \cdot 100\,W}{50,000\,W} \cdot P_{cooling} \tag{19}$$

As in the previous scenario, our energy-aware model is tested against the baseline using three different configurations of GPUs per EDC. Table 8 lists the results for the two-phase immersion-based scenarios. The improvement obtained relative to the baseline is somewhat less than in the air-cooled case, showing energy savings between 0.56% and 8.16%. It was expected for several reasons. On the one hand, the immersion-cooled GPU model behaves more linearly than the air-cooled one. On the other hand, this GPU employs a higher utilization factor. Thus, the EDCs have less capacity and possible combinations to allocate the sessions and improve the situation. These two facts diminish potential optimization savings.

Size-wise, the results follow the same trend as the air-based case. The medium-sized EDCs yield the best results, with the small-sized ones close behind. The largest EDCs show almost no improvement. The reasons are the same as in the air-based scenario (large EDCs are likely never to reach their limit, so they usually have these GPUs available), but with the addition of the immersion-cooled GPU model (less potential energy improvements). Finally, the cooling model's configuration proved successful as the PUE fluctuates between 1.02 to 1.09, close enough to what the Literature suggests for these systems and much more efficient than air-cooled ones nowadays.

Figs. 10(c) and 10(d) depict the model's performance for the scenario with the highest savings using the mean power consumption and PUE, respectively. Looking at the power consumption, it is clear that the difference is much narrower than in the air-based scenario during most of the simulation. Our resource allocation manager obtains better results during most of the simulation, with rather stable and constant power consumption, unlike the baseline. In contrast, the baseline strategy scores better PUE values than ours by a small margin. This outcome strengthens the idea described in Section 5.2 that a better PUE does not mean better energy efficiency, but rather an IT energy optimization of the scenario whose contribution is much more significant than the cooling, hurting the PUE final value.

Besides strategy comparisons, the most notable finding is that immersion-based scenarios consume significantly less energy than air-based ones, thanks to both IT and cooling systems being less power-greedy. In the evaluated simulations, the mean energy reduction found with two-phase immersion-cooled EDCs compared to air-cooled ones is 22.6%, with a maximum of 28.5%. Fig. 10(b) depicts the air-based and immersion-based scenario comparison with the highest savings corresponding to the 5 GPUs per EDC configuration and the baseline.

5.3.3. Heterogeneous scenario

Finally, this paper presents a heterogeneous scenario that leverages both EDC types, mimicking the gradual adoption of two-phase immersion-cooled EDC in EC deployments. The layout features two air-cooled EDCs and one immersion-cooled EDC. If our models behave as expected, the DRL agent will prefer to allocate sessions in the immersion-cooled EDC as it clearly consumes less energy.

Table 9 illustrates the results. These scenarios show the highest energy savings with up to 23.75% reduction compared to the baseline. However, this time, the five-GPUs EDCs are the ones with rather modest results. The reason behind this is straightforward. As expected, the energy-aware model tries to allocate as many sessions as possible in the immersion-cooled EDC. This behavior hurts the EDCs with limited resources since there is only one immersion-cooled. On the other hand, the largest EDCs are almost as good as the medium-sized EDCs, lagging behind as their use of hot standby GPUs is less efficient, like in other scenarios. Fig. 10(e) shows the results for the best scenario. The difference in power consumption between both strategies is second to none, reducing more than 500 W on average. Since this scenario includes ten GPUs per EDC, the utilization (Fig. 10(f)) depicts how the immersion-cooled EDC fills up completely, and one of the other two air-cooled EDCs takes the few remaining sessions during the interval with the highest demand.

From the analysis of the heterogeneous scenarios, some conclusions are inferred. The most energy-efficient EDC was the immersion-cooled one, which yielded the most substantial energy savings. Perhaps, the most significant finding is that heterogeneous scenarios, which are more diverse with higher complexity, bring out the best of algorithms such as DRL models. The results have been notably better in contrast to more simple scenarios. They pave the way to formulate even more complex layouts where DRL-based solutions might make the difference and help obtain considerable energy savings in real EC setups. Aside from the three types of scenarios, the dimensioning of EDCs through their resources, the GPUs, proved vital to optimize future deployments. As seen in these results, the medium-sized configurations (ten GPUs per EDC) harness much better the additional hot standby GPUs than the small-sized and large-sized ones (five and fifteen GPUs each).

5.3.4. Statistical hypothesis testing

In addition to the previously discussed results, we have carried out statistical tests to prove the benefits of our solution compared to the baseline in a more rigorous manner. To this end, we first obtained the mean power consumption $\bar{P}$ (our optimization metric) of 10 simulations per evaluated scenario

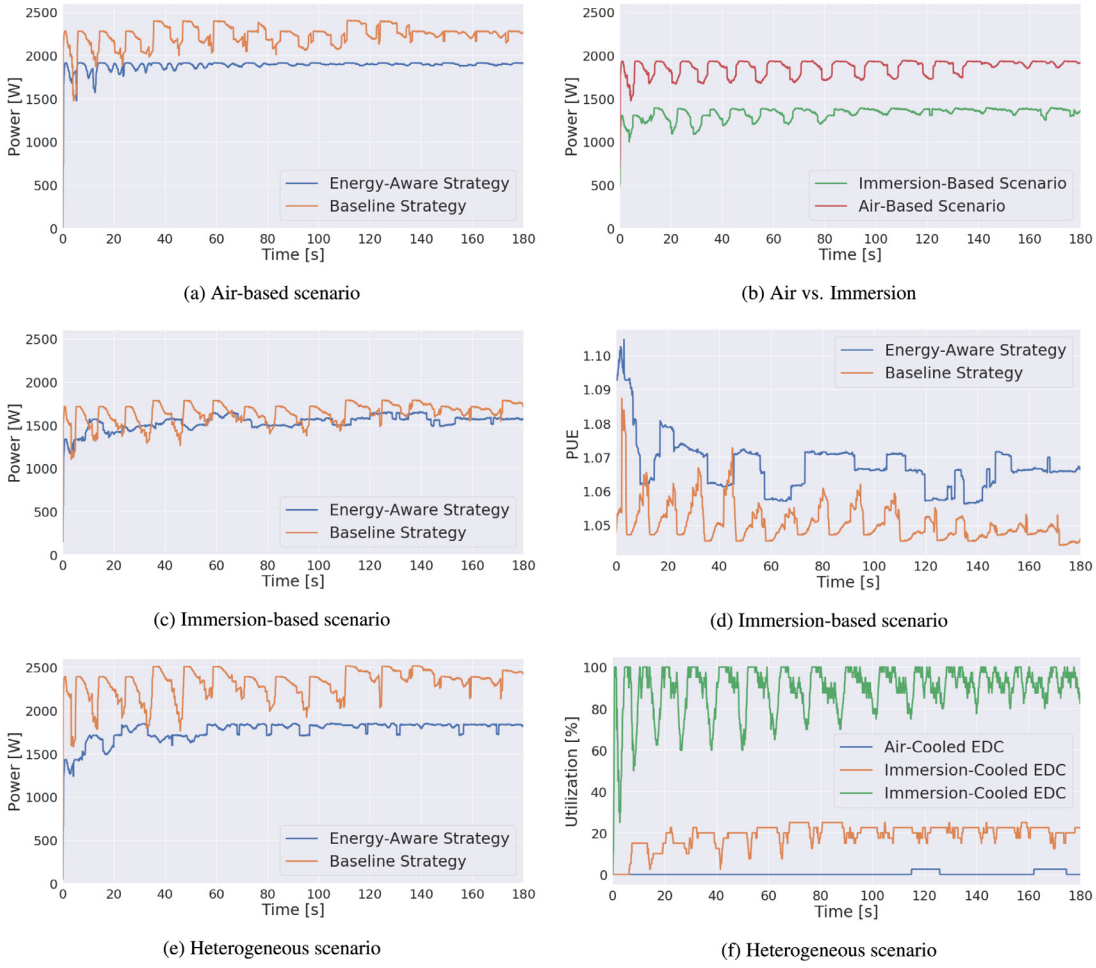

Fig. 10. Results of the three evaluated scenarios: (a) strategy comparison for the best air-based scenario by power consumption; (b) EDC type comparison for the best configuration by power consumption; (c) strategy comparison for the best immersion-based scenario by power consumption; (d) strategy comparison for the best immersion-based scenario by PUE; (e) strategy comparison for the best heterogeneous scenario by power consumption; and (f) strategy comparison for the best heterogeneous scenario by utilization.

using our energy model, 90 simulations overall. Fig. 11 depicts the outcome. We can infer that the variation in energy consumption among different simulations in the same scenario is definitely low. Thus, suggesting that our DRL agent provides consistent performance. The two relevant observations are: (i) the noticeable energy savings using immersion-based scenarios compared to air-based ones, (ii) and the potential of optimizing heterogeneous scenarios since increasing the number of GPUs does not impact the power consumption that much.

Finally, we make use of statistical hypothesis testing and the 90 simulations to compare our model against the baseline. After analyzing several options, we found that the paired sample t-test [67] was the most suitable for our case. T-tests are well-known and robust parametric tests that are used today in many fields. In fact, the evidence shows that in DRL, they are the best option even when the assumptions are partially met [68]. These assumptions are (i) independent observations, (ii) approximately normally distributed data, (iii) no outliers, and (iv) continuous data. In our case, we meet the requirements, but with a small sample size, it is better to utilize a robust test such as this one to guarantee its validity. Another critical factor is the number of variables. Since there are only two (our model's and the baseline's results), the paired sample t-test becomes a perfect fit for evaluating our research. Accordingly, two hypotheses are considered and defined as follows:

- The null hypothesis (H_0) suggests that both the model and the baseline are similar.
- The alternative hypothesis (H_1) tells us that the two solutions are different.

To further prove the viability of the A2C agent, we use different criteria to evaluate it. We calculate the median, mean, maximum, and minimum values of the ten simulations for each of the nine scenarios using the mean power consumption $\bar{P}$, our optimization metric. Table 10 lists the outcome of the test. It shows the sample size (N), the sample mean and standard deviation (SD) of the pair-wise difference, the degrees of freedom (d.f.), and the obtained t-statistic and p-value, using the four different criteria. To determine the statistical significance, the cut-off value is set to 0.05, a typical significance level utilized in Academia. From worst (maximum $\bar{P}$) to best (minimum $\bar{P}$), all four criteria yield p-values far lower than 0.05. Therefore, we can reject the null hypothesis H_0 and conclude that the mean power consumption of our DRL model compared to the baseline is significantly lower, as suggested by the evidence.

6. Conclusions and future directions

This paper presents novel research for the energy optimization of realistic EC scenarios utilizing two-phase immersion cooling systems and data-driven resource allocation via DRL. As proof

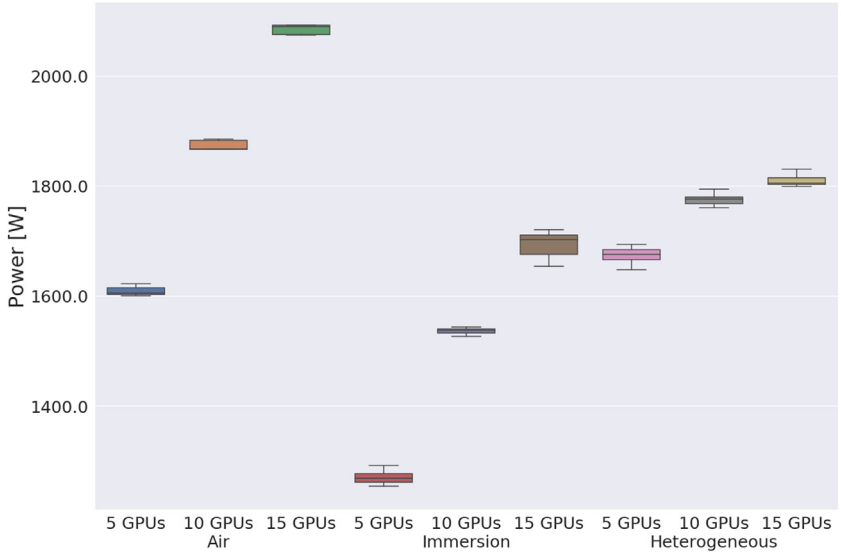

Fig. 11. Mean power consumption of several simulations ($N = 9$) per evaluated scenario.

Table 10
Statistical comparison between our DRL agent and the baseline.

Metric	Criteria	N	Mean	SD	d.f.	t-statistic	p-value	$p > 0.05$
$\bar{P}\,[W]$	Mean	9	237.01	184.54	8	3.6325	0.0067	✗
	Median	9	236.48	187.06	8	3.5755	0.0072	✗
	Maximum	9	220.41	184.30	8	3.3828	0.0096	✗
	Minimum	9	252.74	179.70	8	3.9780	0.0041	✗

of concept, multiple EC scenarios have been modeled, simulated, and optimized, harnessing these two methods. The scenarios feature actual traces from taxis around San Francisco and employ a DL-based ADAS application well suited for EC workloads. In the scenario, these workloads are offloaded to EDCs. We also used this application to develop energy models of two types of EDC prototypes that include GPUs for the IT and either air or two-phase immersion cooling systems. The two-phase immersion-cooled prototype's IT energy model achieved an NRMSD of 3.15% and an R^2 of 97.97%. Its cooling energy model is theoretically devised using data from a real heat exchanger mechanism.

On the other hand, a resource allocation manager is responsible for assigning the workloads to the EDCs. It is a DRL-based A2C agent with the goal of minimizing the EDCs' energy consumption. A two-headed FNN is employed for its architecture, which was fine-tuned considering both speed and performance. Three main scenarios were evaluated: air-based, immersion-based, and heterogeneous, which features both air-cooled and immersion-cooled EDCs. Firstly, the air-cooled scenario yielded considerable energy savings compared with the baseline, 12% on average. Then, immersion-cooled EDCs obtained an average energy reduction of 22.8% in contrast to air-cooled ones, and immersion-based scenarios obtained an additional 4% below the baseline. Finally, the heterogeneous scenario presents maximum savings of 23.8% in comparison to the baseline.

Numerous future lines can be drawn from here, such as (i) multi-objective optimization that involves delays, energy consumption, and prices; (ii) more complex and realistic scenarios with smart-grid, renewable energies, and size limitations; (iii) multiple applications based on different fields like health or gaming; (iv) thermal-aware modeling to leverage the maximum potential of two-phase immersion cooling; and (v) new DRL agents, DL architectures, IT equipment, cooling systems or baselines to add further variety to the experiments. Perhaps, the most important conclusion of this research is that the combination of DRL optimization and two-phase immersion cooling enables many possibilities to optimize highly complex EC scenarios going forward. This fact would potentially accelerate the deployment of EC solutions and, thus, a more scalable, sustainable, and greener future.

CRediT authorship contribution statement

Sergio Pérez: Conceptualization, Methodology, Software, Validation, Supervision, Formal analysis, Data curation, Investigation, Writing – original draft, Visualization. **Patricia Arroba:** Conceptualization, Methodology, Validation, Writing – review & editing, Supervision. **José M. Moya:** Supervision, Funding acquisition, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This project has been partially supported by the Centre for the Development of Industrial Technology (CDTI) and State R&D Program Oriented to the Challenges of the Society (Retos Colaboración 2017) under contracts IDI-20171194 and RTC-2017-6090-3 from the Spanish Ministry of Science and Innovation. This paper has also been supported by the Spanish Ministry of Economic Affairs and Digital Transformation (MINECO) under grant PID2019-110866RB-I00. This work has also been supported by the Directorate-General for Rural Development, Innovation and Agri-Food Training (DGDRIFA) of the Spanish Ministry of Agriculture, under Operational Group MESRASA.

References

[1] R. Sánchez-Corcuera, A. Nuñez-Marcos, J. Sesma-Solance, A. Bilbao-Jayo, R. Mulero, U. Zulaika, G. Azkune, A. Almeida, Smart cities survey: Technologies, application domains and challenges for the cities of the future, Int. J. Distrib. Sens. Netw. 15 (6) (2019).
[2] F. Dahlqvist, M. Patel, A. Rajko, J. Shulman, Growing opportunities in the Internet of Things, Tech. rep., McKinsey, 2019.

[3] D.A. Chekired, M.A. Togou, L. Khoukhi, A. Ksentini, 5G-slicing-enabled scalable SDN core network: Toward an ultra-low latency of autonomous driving service, IEEE J. Sel. Areas Commun. 37 (8) (2019) 1769–1782.
[4] A. Ahad, M. Tahir, K.A. Yau, 5G-based smart healthcare network: Architecture, taxonomy, challenges and future research directions, IEEE Access 7 (2019) 100747–100762.
[5] R. van der Meulen, What Edge Computing Means for Infrastructure and Operations Leaders, Tech. rep., Gartner, 2018.
[6] W. Jonshon, K. Sparks, B. Daly, R. Gyurek, K. Balachandran, J. Barnhill, L. Merrill, B. Markwalter, A. Drobot, J. Foerster, D. Hatfield, 5G Edge Computing Whitepaper, Tech. rep., FCC, 2018.
[7] C. Chang, S.N. Srirama, R. Buyya, Internet of things (IoT) and new computing paradigms, Fog Edge Comput.: Princ. Paradigms 6 (2019) 1–23.
[8] L. Newcombe, M. Acton, R. Tozer, J. Booth, P. Bertoldi, S. Flucker, A. Rouyer, 2018 Best Practice Guidelines for the EU Code of Conduct on Data Centre Energy Efficiency, Tech. rep., European Commission, 2018.
[9] R. Bunger, W. Torell, V. Avelar, Capital Cost Analysis of Immersive Liquid-Cooled vs. Air-Cooled Large Data Centers, Tech. rep., Schneider Electric, 2019.
[10] T. Day, P. Lin, R. Bunger, Liquid Cooling Technologies for Data Centers and Edge Applications, Tech. rep., Schneider Electric, 2019.
[11] Electronics Materials Solutions Division, Two-Phase Immersion Cooling: A revolution in data center efficiency, Tech. rep., 3M, 2019.
[12] F. Hussain, S.A. Hassan, R. Hussain, E. Hossain, Machine learning for resource management in cellular and IoT networks: Potentials, current solutions, and open challenges, IEEE Commun. Surv. Tutor. 22 (2) (2020) 1251–1275.
[13] R.T. Rodoshi, T. Kim, W. Choi, Resource management in cloud radio access network: Conventional and new approaches, Sensors 20 (9) (2020).
[14] A. Leva, PID-based controls in computing systems: A brief survey and some research directions, IFAC-PapersOnLine 51 (4) (2018) 805–810, 3rd IFAC Conference on Advances in Proportional-Integral-Derivative Control.
[15] C. Gamble, J. Gao, Safety-First AI for Autonomous Data Centre Cooling and Industrial Control, Tech. rep., DeepMind, 2018.
[16] N.C. Luong, D.T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, D.I. Kim, Applications of deep reinforcement learning in communications and networking: A survey, IEEE Commun. Surv. Tutor. 21 (4) (2019) 3133–3174.
[17] N. Lazic, C. Boutilier, T. Lu, E. Wong, B. Roy, M. Ryu, G. Imwalle, Data center cooling using model-predictive control, in: Advances in Neural Information Processing Systems, 2018, pp. 3814–3823.
[18] J. Chen, X. Ran, Deep learning with edge computing: A review, Proc. IEEE 107 (8) (2019) 1655–1674.
[19] T.S. Combs, L.S. Sandt, M.P. Clamann, N.C. McDonald, Automated vehicles and pedestrian safety: Exploring the promise and limits of pedestrian detection, Am. J. Prev. Med. 56 (1) (2019) 1–7.
[20] NVIDIA, Self-driving safety report, Tech. rep., NVIDIA Corp, 2018.
[21] L. Liu, C. Chen, Q. Pei, S. Maharjan, Y. Zhang, Vehicular edge computing and networking: A survey, Mob. Netw. Appl. (2020) 1–24.
[22] F. Hussain, S.A. Hassan, R. Hussain, E. Hossain, Machine learning for resource management in cellular and IoT networks: Potentials, current solutions, and open challenges, IEEE Commun. Surv. Tutor. 22 (2) (2020) 1251–1275.
[23] B.B. Kanbur, C. Wu, S. Fan, W. Tong, F. Duan, Two-phase liquid-immersion data center cooling system: Experimental performance and thermoeconomic analysis, Int. J. Refrig. 118 (2020) 290–301.
[24] P. Malagón, P. Arroba, S. Briongos, A.M. Santana, J.M. Moya, Modeling tree-structured I2C communication to study the behavior of a dielectric coolant in a two-phase immersion cooling system, in: Proceedings of the 2020 Summer Simulation Conference, SummerSim '20, Society for Computer Simulation International, San Diego, CA, USA, 2020.
[25] Y. Li, Y. Wen, D. Tao, K. Guan, Transforming cooling optimization for green data center via deep reinforcement learning, IEEE Trans. Cybern. 50 (5) (2020) 2002–2013.
[26] M. Ghobaei-Arani, A. Souri, T. Baker, A. Hussien, Controcity: An autonomous approach for controlling elasticity using buffer management in cloud computing environment, IEEE Access 7 (2019) 106912–106924.
[27] Huawei, Fusionmodule1000a all-in-one data center, 2021, https://e.huawei.com/en/products/network-energy/dc-facilities/ids1000-a, [Online; accessed 26-Jul-2021].
[28] Rittal, Modular data centers in containers, 2021, https://www.rittal.com/it-solutions/en/solution/data-center-container/, [Online; accessed 26-Jul-2021].
[29] H. Lu, Z. Zhang, L. Yang, A review on airflow distribution and management in data center, Energy Build. 179 (2018) 264–277.
[30] H.M. Daraghmeh, C.-C. Wang, A review of current status of free cooling in datacenters, Appl. Therm. Eng. 114 (2017) 1224–1239.
[31] Facebook Sustainability, 2020 Sustainability Report, 2020, https://sustainability.fb.com/wp-content/uploads/2021/07/2020_FB_Sustainability-Report.pdf, [Online; accessed 26-Jul-2021].
[32] K. Heinemeier, Free cooling: At what cost? ACEEE Summer Study Energy Efficiency Build. (2014).
[33] C. Nadjahi, H. Louahlia, S. Lemasson, A review of thermal management and innovative cooling strategies for data center, Sustainable Comput.: Inform. Syst. 19 (2018) 14–28.
[34] K. Karki, S. Novotny, A. Radmehr, S. Patankar, Use of passive, rear-door heat exchangers to cool low to moderate heat loads, ASHRAE Trans. 117 (2011) 26–33.
[35] T.J. Chainer, M.D. Schultz, P.R. Parida, M.A. Gaynes, Improving data center energy efficiency with advanced thermal management, IEEE Trans. Compon. Packag. Manuf. Technol. 7 (8) (2017) 1228–1239.
[36] J. Li, G. Zhou, T. Tian, X. Li, A new cooling strategy for edge computing servers using compact looped heat pipe, Appl. Therm. Eng. 187 (2021) 116599.
[37] T. Day, P. Lin, R. Bunger, Liquid Cooling Technologies for Data Centers and Edge Applications, Tech. rep., Schneider Electric, 2019.
[38] D.W. Sundin, Mineral Oil, White Oil and Synthetic Dielectric Coolants, Tech. rep., Engineered Fluids, LLC, 2017.
[39] Electronics Materials Solutions Division, The next generation of data centers is here, Tech. rep., 3M, 2019.
[40] C. Wu, W. Tong, B.B. Kanbur, F. Duan, Full-scale two-phase liquid immersion cooing data center system in tropical environment, in: 2019 18th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm), 2019, pp. 703–708.
[41] Liquid Stack, Datatank 48U, 2021, https://liquidstack.com/solutions/datatank-48u, [Online; accessed 26-Jul-2021].
[42] Cliff Robinson, Wiwynn two-phase immersion cooling system for OCP nodes, 2019.
[43] T.L. Bergman, F.P. Incropera, D.P. DeWitt, A.S. Lavine, Fundamentals of Heat and Mass Transfer, John Wiley & Sons, 2011, pp. 656–660.
[44] Z. Qu, Y. Wang, L. Sun, D. Peng, Z. Li, Study QoS optimization and energy saving techniques in cloud, fog, edge, and IoT, Complexity 2020 (2020) 1–16.
[45] M. Ghobaei-Arani, A. Souri, A. Rahmanian, Resource management approaches in fog computing: A comprehensive review, J. Grid Comput. 18 (2020).
[46] M. Ghobaei-Arani, A. Souri, F. Safara, M. Norouzi, An efficient task scheduling approach using moth-flame optimization algorithm for cyber-physical system applications in fog computing, Trans. Emerg. Telecommun. Technol. 31 (2) (2020).
[47] J. Xu, L. Chen, S. Ren, Online learning for offloading and autoscaling in energy harvesting mobile edge computing, IEEE Trans. Cogn. Commun. Netw. 3 (3) (2017) 361–373.
[48] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, M. Bennis, Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning, IEEE Internet Things J. 6 (3) (2019) 4005–4018.
[49] D. Zeng, L. Gu, S. Pan, J. Cai, S. Guo, Resource management at the network edge: A deep reinforcement learning approach, IEEE Netw. 33 (3) (2019) 26–33.
[50] Z. Ning, P. Dong, X. Wang, J.J.P.C. Rodrigues, F. Xia, Deep reinforcement learning for vehicular edge computing: An intelligent offloading system, ACM Trans. Intell. Syst. Technol. 10 (6) (2019).
[51] R. Cárdenas, P. Arroba, R. Blanco, P. Malagón, J.L. Risco-Martín, J. Moya, Mercury: A modeling, simulation, and optimization framework for data stream-oriented IoT applications, Simul. Model. Pract. Theory 101 (2019) 102037.
[52] R. Cárdenas, P. Arroba, J.M. Moya, J.L. Risco-Martín, Edge federation simulator for data stream analytics, in: Proceedings of the 2019 Summer Simulation Conference, SummerSim '19, Society for Computer Simulation International, 2019, pp. 1–12.
[53] R. Cárdenas, P. Arroba, J.L. Risco-Martín, Bringing AI to the edge: A formal M&S specification to deploy effective IoT architectures, J. Simul. (2021) 1–18.
[54] G.C. Fox, V. Ishakian, V. Muthusamy, A. Slominski, Status of serverless computing and function-as-a-service(faas) in industry and research, CoRR abs/1708.08028 (2017).
[55] S. Pérez, J. Pérez, P. Arroba, R. Blanco, J. Ayala, J. Moya, Predictive GPU-based ADAS management in energy-conscious smart cities, in: 2019 IEEE International Smart Cities Conference (ISC2), 2019, pp. 349–354.
[56] J.D. Moore, J.S. Chase, P. Ranganathan, R.K. Sharma, Making scheduling "Cool": Temperature-aware workload placement in data centers, in: USENIX Annual Technical Conference, General Track, 2005, pp. 61–75.
[57] J. Pérez, S. Pérez, J.M. Moya, P. Arroba, Thermal prediction for immersion cooling data centers based on recurrent neural networks, in: Intelligent Data Engineering and Automated Learning, IDEAL, Springer International Publishing, 2018, pp. 491–498.

[58] HDC–NOVEC: High-density data center server cooling by immersion in bi-phase refrigerant fluid (IDI-20171194), funded by 3M and the Spanish ministry of science and innovation, 2020.
[59] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al., Mastering the game of go without human knowledge, Nature 550 (7676) (2017) 354–359.
[60] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 2018.
[61] S. Angelopoulos, C. Dürr, S. Kamali, M.P. Renault, A. Rosén, Online bin packing with advice of small size, Theory Comput. Syst. 62 (8) (2018) 2006–2034.
[62] H. Zhao, Q. She, C. Zhu, Y. Yang, K. Xu, Online 3D bin packing with constrained deep reinforcement learning, 2021, arXiv:2006.14978.
[63] H. Hu, X. Zhang, X. Yan, L. Wang, Y. Xu, Solving a new 3D bin packing problem with deep reinforcement learning method, 2017, arXiv:1708.05930.
[64] R. Cárdenas, Mercury M&S&O framework for fog computing, 2020, https://www.github.com/greenlsi/mercury_mso_framework, [Online; accessed 26-Jul-2021],
[65] W. Torell, K. Brown, V. Avelar, The Unexpected Impact of Raising Data Center Temperatures, Tech. rep., Schneider Electric, 2016.
[66] C. Koronen, M. Åhman, L. Nilsson, Data centres in future European energy systems—energy efficiency, integration and policy, Energy Efficiency 13 (2020) 129–144.
[67] A. Ross, V.L. Willson, Paired samples T-test, in: Basic and Advanced Statistical Tests, Brill Sense, 2017, pp. 17–19.
[68] C. Colas, O. Sigaud, P.-Y. Oudeyer, A Hitchhiker's guide to statistical comparisons of reinforcement learning algorithms, in: ICLR Worskhop on Reproducibility, Nouvelle-Orléans, United States, 2019.

Sergio Pérez is a Research Assistant at the Technical University of Madrid (UPM). He received his M.Sc. in Telecommunication Engineering from UPM in 2020. His research interests include energy-aware modeling and resource allocation optimization in edge data centers. His email address is s.pmorillo@alumnos.upm.es.

Patricia Arroba is an Assistant Professor at the Technical University of Madrid (UPM), Spain. She received her Ph.D. degree in Telecommunication Engineering from UPM in 2017. Her research interests include energy and thermal-aware modeling and optimization of data centers. Her email address is p.arroba@upm.es.

José M. Moya is an Associate Professor at the Technical University of Madrid (UPM). He received his Ph.D. degree in Telecommunication Engineering from UPM in 2003. His research interests include proactive and reactive thermal-aware optimization of data centers. His email address is jm.moya@upm.es.