Journal of Network and Computer Applications 216 (2023) 103648

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

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

Review

AI augmented Edge and Fog computing: Trends and challenges

Shreshth Tuli [a,*], Fatemeh Mirhakimi [b], Samodha Pallewatta [c], Syed Zawad [d], Giuliano Casale [a], Bahman Javadi [b], Feng Yan [d], Rajkumar Buyya [c], Nicholas R. Jennings [e]

[a] *Imperial College London, UK*
[b] *Western Sydney University, Australia*
[c] *University of Melbourne, Australia*
[d] *University of Nevada, Reno, USA*
[e] *Loughborough University, UK*

ARTICLE INFO

Keywords:
AI
Edge computing
Fog computing
Cloud computing
Deployment
Scheduling
Fault-tolerance

ABSTRACT

In recent years, the landscape of computing paradigms has witnessed a gradual yet remarkable shift from monolithic computing to distributed and decentralized paradigms such as Internet of Things (IoT), Edge, Fog, Cloud, and Serverless. The frontiers of these computing technologies have been boosted by shift from manually encoded algorithms to Artificial Intelligence (AI)-driven autonomous systems for optimum and reliable management of distributed computing resources. Prior work focuses on improving existing systems using AI across a wide range of domains, such as efficient resource provisioning, application deployment, task placement, and service management. This survey reviews the evolution of data-driven AI-augmented technologies and their impact on computing systems. We demystify new techniques and draw key insights in Edge, Fog and Cloud resource management-related uses of AI methods and also look at how AI can innovate traditional applications for enhanced Quality of Service (QoS) in the presence of a continuum of resources. We present the latest trends and impact areas such as optimizing AI models that are deployed *on* or *for* computing systems. We layout a roadmap for future research directions in areas such as resource management for QoS optimization and service reliability. Finally, we discuss blue-sky ideas and envision this work as an anchor point for future research on AI-driven computing systems.

1. Introduction

In the past decade, the evolution of our digital lives has accelerated across multiple facets, including efficient computation (Gill et al., 2019), communication (Shi et al., 2020) and transportation (Nguyen et al., 2021), making our lives simpler and more convenient. This evolution has been driven by several factors such as the rising concern for climate change and sustainable computing (Tuli et al., 2021b), the expected end of Moore's law for silicon-based compute systems (Theis and Wong, 2017) and the recent lifestyle-changing pandemics (Ndiaye et al., 2020) to name a few. With changing user demands and application scenarios, novel techniques are required to fuel further growth for high fidelity and scalable computation. There are two trends at the center of this growth: Artificial Intelligence (AI) and the Internet of Things (IoT). In the context of resource management, the field of AI aims to build intelligent entities that automate the process of dynamically making various design decisions for industrial computational deployment. The shift from relying on hand-encoded algorithms and human domain experts to AI or Machine Learning (ML) agents has enabled computation on scale and facilitated servicing the computational needs of the rising world population (Gill et al., 2019). The other field, IoT, promises ubiquitous connectivity across various computational and networking devices using the Internet. It presents a broad spectrum of computational resources, ranging from resource-limited devices at the Edge of the network (Mao et al., 2017) to the heavy physical or virtual machines in the Cloud (Tuli et al., 2019b), all connected to the users via gateway devices. Fog computing devices connect Edge and Cloud resources, giving rise to a paradigm that holistically considers the entire continuum of resources from Edge to the Cloud, often referred to as *Fog Continuum* (Bittencourt et al., 2018).

The convergence of AI and Fog Continuum, presents a massive opportunity for research and enterprise. Technological research organizations, such as Gartner, predict that in the coming years, AI-augmented computing will be at the forefront of technical advancements in Internet and Communication Technologies (ICT) (Costello, 2019). When deploying AI-based applications on Fog continuum or leveraging AI to manage running applications, novel resource management issues

* Corresponding author.
E-mail address: s.tuli20@imperial.ac.uk (S. Tuli).

https://doi.org/10.1016/j.jnca.2023.103648
Received 14 October 2022; Received in revised form 19 January 2023; Accepted 13 April 2023
Available online 3 May 2023

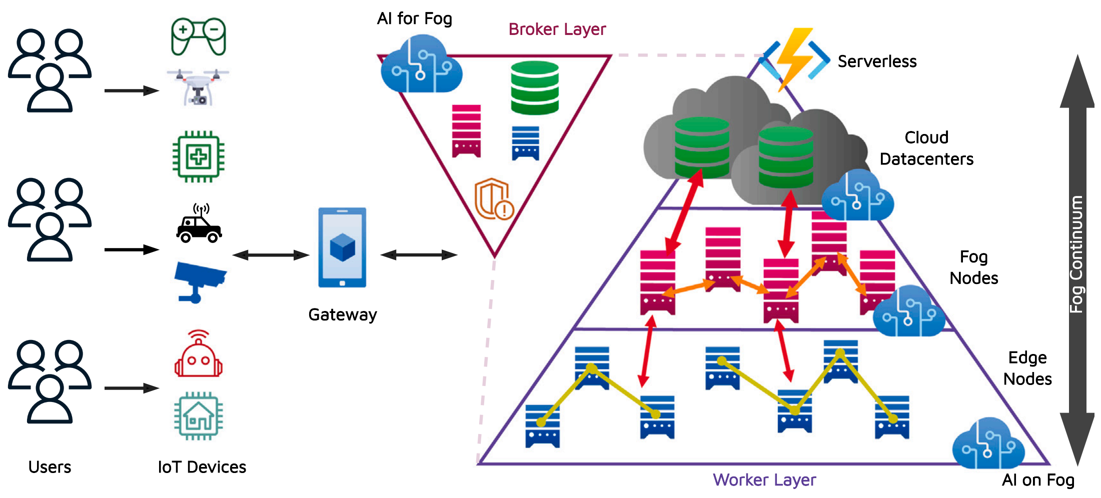


Fig. 1. AI *on* and *for* the Fog continuum.

arise corresponding to the maintenance of optimal Quality of Service (QoS). As part of this paper, we explore the latest trends in the domain of AI-augmented resource management and the challenges it presents to deliver upon the promise of improving QoS of existing and next-generation computational infrastructures.

1.1. Motivation of research in AI-based augmentation

A typical Fog environment consists of two computational layers: broker and worker (see Fig. 1). The worker layer consists of generic compute nodes that execute incoming applications by processing incoming data from the users and return the results *via* gateway devices (Tuli et al., 2019b) (see nodes in the purple triangle in Fig. 1). The broker layer consists of compute nodes that monitor and manage the back-end infrastructure, including the worker nodes (see nodes in the inverted red triangle in Fig. 1). This includes deciding where to deploy/place incoming applications as tasks or migrate running tasks to optimize system performance. This difference in broker and worker roles are tied closely with the classification of AI based approaches of AI *on* and *for* Fog that we describe later. Recent research in AI has shown some promise in the direction of improving QoS of Fog systems, thanks to their higher inference speeds and accuracy compared to classical techniques (Liang et al., 2020). AI research for Fog systems has spanned diverse categories including (1) classical AI that covers informed and uninformed search methods, (2) machine learning that encompasses unsupervised, supervised and semi-supervised methods, (3) reinforcement learning that includes tabular and deep reinforcement methods, and (4) deep learning that uses deep neural networks as function approximators to model complex relationships across data in Fog systems (Russell and Norvig, 2009; Goodfellow et al., 2016). A brief taxonomy from Russell and Norvig (2009) is presented in Fig. 2. We shall leverage this taxonomy in Section 4 to discuss and classify state-of-the-art AI research for Fog systems.

AI-based augmentation of Fog systems has traditionally been in two major directions. First, where AI models have replaced conventional applications, for instance Deep Neural Networks (DNNs) have replaced prior methods in domains such as traffic surveillance using computer vision, chat bots using natural language processing and smart homes using robotics (Shi et al., 2020; Park et al., 2018; Amini et al., 2020), giving fast, scalable and accurate results. This entails augmenting the workloads that are run *on* the Fog worker nodes, and hence we call this domain *AI on Fog*. AI on Fog has been a key driving factor many AI based practical deployments, such as self-driving cars, smart-cities and automated surveillance systems (Wang et al., 2020f). Second, where the AI models are used to determine optimal workload placement, service level schedules and fault remediation steps. This augments the resource management services at the broker layer *for* decision making, and hence we call this domain *AI for Fog*. This domain has been crucial for efficient resource management for modern distributed services such as Netflix and cloud platforms (Tuli, 2022; Varghese and Buyya, 2018). We elucidate the challenges presented by each paradigm below.

AI on Fog. This domain is primarily concerned with the applications running on the worker layer of a Fog system. As modern applications have turned heavily dependent on AI-based models, specifically those that utilize deep learning, we observe that DNNs are becoming the backbone of many industrial tasks and activities (Gill et al., 2019). As the computational capabilities of devices have improved, new deep learning models have been proposed to provide improved performance (Zhu et al., 2018; Li et al., 2019c). Moreover, many recent DNN models have been incorporated with mobile edge computing to give low latency services with improved accuracy compared to shallow networks. Specifically in time-critical complex tasks such as image segmentation, high frame-rate gaming and traffic surveillance that require latency in the order of 10–100 milliseconds (Khanna et al., 2020). The performance of such neural models reflects directly on the reliability of application domains like self-driving cars, healthcare and manufacturing (Gill et al., 2019; Kraemer et al., 2017). The integration of such AI models with various computational systems has led to the rise of EdgeAI services, *i.e*, applications that utilize AI to process data at the edge. To provide high accuracy, neural models are becoming increasingly demanding in terms of data and compute power, resulting in many challenging problems. To accommodate these increasing demands, such massive models are often hosted as web services deployed on the public Cloud (Zhang and Zhang, 2017). On the other hand, mobile edge devices in typical Fog deployments face severe limitations in terms of computational and memory resources as they rely on low power energy sources like batteries, solar, or other energy scavenging methods (Mao et al., 2016). This is not only because of the requirement of low cost, but also the need for mobility in such nodes (Khanna et al., 2020). In such systems, it is possible to handle the processing limitations of massive AI models by effective preemption and prolonged job execution. However, memory bottlenecks are much harder to solve as shown in prior work (Shao and Zhang, 2020b). In a practical distributed edge environment where storage spaces are typically mapped to a network-attached-media, a large virtual memory imposes high network bandwidth overheads that make performing large-scale distributed computations hard (Laskaridis et al., 2020). Thus, as part of this paper, we explore various methods developed to efficiently deploy and manage AI-based applications on Fog infrastructures by possibly decomposing DNNs and running distributed training and inference (Li et al., 2020b).

Table 1
A comparison of our work with existing surveys based on key parameters and domain coverage.

Ref.	Fog continuum				AI domains		Problem copes		
	IoT	Edge	Cloud	Serverless	AI on Fog	AI for Fog	Deployment	Scheduling	Maintenance
Yang et al. (2019)		✓			✓				✓
Wang et al. (2020c)		✓			✓		✓	✓	
Liu et al. (2021a)		✓			✓		✓		
Murshed et al. (2021)	✓	✓			✓		✓		
Varghese and Buyya (2018)			✓			✓	✓	✓	
Hasan and Goraya (2018)			✓			✓			✓
Zhong et al. (2021)		✓	✓			✓		✓	
Singh et al. (2021)		✓	✓			✓	✓	✓	
Nayeri et al. (2021)	✓	✓	✓			✓	✓	✓	
Mampage et al. (2021)				✓		✓	✓	✓	
Duc et al. (2019)		✓	✓			✓	✓		✓
Deng et al. (2020b)	✓	✓			✓	✓	✓	✓	
This review	✓	✓	✓	✓	✓	✓	✓	✓	✓

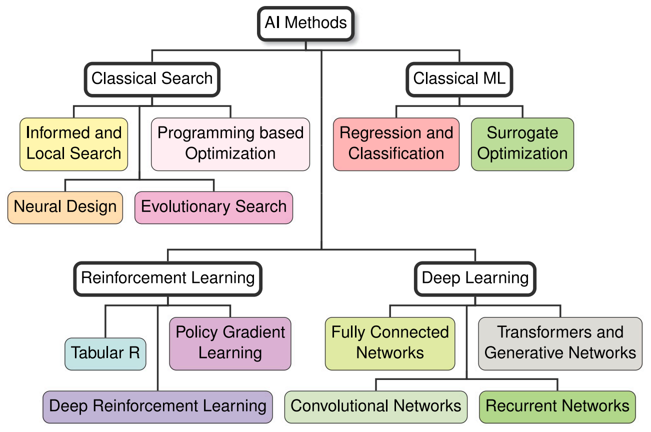


Fig. 2. A brief taxonomy of AI methods for Fog systems that extends the one proposed by Russell and Norvig (Russell and Norvig, 2009).

AI for Fog. This domain is primarily concerned with resource management level decision making at the broker layer of a Fog system. The problem of efficiently managing Fog resources is hard (Tuli et al., 2022e). One of the challenges we face in such a system is the heterogeneity of resources across the Edge and Cloud (Li et al., 2020a; Kaur et al., 2020; Hosseinalipour et al., 2020). Another challenge in industrial settings is to deliver low latencies for time-critical applications, for instance, healthcare, robotics and smart-cities. These challenges are exacerbated by modern-day applications, wherein the workloads are highly dynamic and host machines having volatile resource capabilities. Furthermore, as applications become more demanding and privacy-sensitive, Fog devices have become more prone to breakdowns, malicious attacks and intrusions (Zhang et al., 2019a). This entails taking recovery steps required to deal with the diverse effects of system faults, such as network packet drops, memory errors or disk failures requiring different remediation steps. So far, the industrial and research landscape of Fog resource management has been dominated by heuristics and classical optimization-based methods. Such approaches have low scheduling times and work well for general cases, but due to steady-state or stationarity assumptions, they provide poor performance in non-stationary heterogeneous environments with dynamic workloads (Tuli et al., 2022e). To address these challenges, various AI methods have been recently proposed that utilize adaptive schemes based on evolutionary methods and reinforcement learning. These methods adapt to changing scenarios and offer promising avenues for dynamic optimization (Fox et al., 2019). For accurate and scalable modeling of the Fog environment, such methods use deep learning-based local search or learning models with neural networks which approximate an objective function such as energy consumption or response time (Tuli et al., 2020b; Liu and Wang, 2020; Basu et al., 2019). However, the most accurate AI methods typically have high decision times or resource footprints, making it hard to deploy them in budget or resource-constrained settings. Thus, as part of this paper, we also explore various advancements in AI methods for efficient resource management in Fog systems (Li et al., 2020b).

1.2. Our contributions

Our primary focus lies at the intersection of the two fields of AI and Fog, particularly for resource management decision making to optimize system performance measured using metrics like systems QoS. We review a broad range of techniques developed for optimizing QoS by efficiently deploying AI applications in Fog systems (AI on Fog), utilizing AI methods for resource management decision making (AI for Fog), or both of these together. We partition the entire resource management domain into three scopes based on the decisions we need to optimize: deployment, scheduling and maintenance.

1. **Deployment** deals with intelligent resource provisioning and versioning of workloads on Fog infrastructures to optimize QoS (Tuli, 2022; Calheiros et al., 2014).
2. **Scheduling** deals with arranging and controlling deployed workloads on compute infrastructure for QoS efficient execution (Tuli et al., 2022e; Kadota et al., 2018; Matrouk and Alatoun, 2021).
3. **Maintenance** aims at securing, preventing and recovering from failures the deployed and scheduled workloads in a Fog environment (Du et al., 2020; Tuli et al., 2022c).

We present a comprehensive literature review of the state-of-the-art approaches in the above three scopes. We devised a search query based on the formulated research questions: (edge computing) OR (fog computing) OR (cloud computing) AND (AI) and (resource management) OR (scheduling) OR (provisioning) OR (fault-tolerance). We classify the reviewed methods by their essential characteristics and methodologies. We identify the future directions of AI based augmentation technologies for Fog platforms.

1.3. Related surveys

As summarized in Table 1, some previous surveys have already explored the use of AI for enhancing Fog environments; however, they do not consider the diverse use cases together and cover the complexity of the domains to a limited extent. The first four surveys cover the domain of *AI on Fog*. Yang et al. (2019) introduce and review AI-based methods for data integrity, specifically utilizing Blockchain and deep learning technologies. Wang et al. (2020c) present an exhaustive review of methods for efficient deployment and scheduling of DL-based applications on Edge infrastructures. They discuss several advancements in Edge hardware for accelerating AI training and inference.

However, these reviews do not include the specific advancements in DNN models that focus on deployments in heterogeneous Edge-cloud infrastructures. Liu et al. (2021a) present several techniques to deploy massive DNNs in Edge environments, particularly focusing on model compression and neural architecture search. Here, we also consider the recent developments in distributed split neural models. Murshed et al. (2021) consider distributed DNN training and inference for EdgeAI applications; here, we also cover the impact on the resource management back-ends in Edge systems.

Furthermore, there have been some recent studies that investigate the *AI for Fog* domain. Varghese and Buyya (2018) discuss various technological advancements in Cloud computing domain that leverage AI models for task placement and scheduling. Hasan and Goraya (2018) summarize the research in fault-tolerant Cloud computing using AI-based methods. These works ignore the effects of merging the Cloud paradigm with Edge nodes. Zhong et al. (2021) discuss various methods to schedule workloads in the form of containers in Edge and Cloud environments. Similar surveys by Singh et al. (2021) and Nayeri et al. (2021) describe the methods for provisioning nodes and scheduling tasks in a Fog environment. Duc et al. (2019) discuss similar methods for reliable resource provisioning in Edge-cloud environments. Mampage et al. (2021) describe resource management techniques for serverless computing environments. However, these works consider AI as black-box models and do not discuss the specific advancements in the underpinning AI techniques for QoS improvement in the context of deployment, scheduling or maintenance. Finally, Deng et al. (2020b) discuss the AI methods for and on Edge platforms, but only in the context of task allocation and AI model compression. They do not discuss the use of latest technologies such as coupled-simulation (Tuli et al., 2022e) in solving major challenges faced when utilizing AI models for efficient resource management. Further, they restrict their descriptions to edge-only environments and do not consider the complete fog continuum.

This work builds upon the previous surveys to present a holistic view of how AI models have augmented Fog systems, particularly focusing on the overlap among *AI on Fog* and *AI for Fog* methods. We emphasize the diversity and complexity of QoS aware resource management schemes in the Fog continuum by categorizing the landscape into deployment, scheduling and maintenance related strategies. Unlike previous surveys, we present a classification of AI and fog methods that highlights the intersection between data-driven models and resource management distributed systems encompassing AI design, system modeling and workload-injection frameworks. Using such a holistic approach, we consolidate trends to present root-cause issues that limit the performance of AI or fog systems and share possible future directions to tackle them.

1.4. Article structure

The rest of the paper is organized as follows: Section 2 reviews the computing paradigms of IoT, Edge, Cloud and serverless and how Fog harnesses them. We describe the various service architectures and elucidate the main control knobs and optimization parameters. We discuss state-of-the-art AI methods in Section 4. This section presents these methods in the scopes of deployment, scheduling and maintenance. We then perform a detailed trend analysis and methodological overlap in Section 5. Such trend analysis facilitates in determining root-causes for current limitations and possible solutions as future directions as detailed in Section 6. Finally, Section 7 concludes the survey.

2. Background

In this section, we present the various computing paradigms that form the Fog continuum, service architectures and parameters offered from the systems aspect for AI methods to exploit and optimize the overall QoS.

2.1. Related computing paradigms

We now describe the computing paradigms of Cloud, Edge and serverless. We mention their merits and limitations to motivate the need for a continuum of resources.

Cloud Computing. The Cloud computing paradigm consists of an inter-connected and virtualized pool of resources (computing, storage, network, etc.) that can be dynamically provisioned on-demand, as per user specifications and with minimal management effort (Buyya et al., 2009) (see top tier in Fig. 1). Cloud resources may be publicly accessible or privately deployed. Traditionally, workloads are run in Cloud nodes as distinct virtual machines (VMs), allowing Cloud providers to migrate running workloads from one Cloud node to another for load balancing and tuning various QoS parameters. A significant challenge in the Cloud paradigm is that Cloud datacenters are located multiple hops away from the IoT devices, which increases the data transmission time between the devices and the Cloud instances hosting the applications. To overcome these limitations of Cloud computing, a new paradigm called Edge computing was introduced to meet the service requirements of large-scale IoT applications.

Edge Computing. Recently, Edge computing (Satyanarayanan, 2017) has grown dramatically. The network edge, defined as the computational layer that resides closest to the end-user, is where most data sources are present. Edge computing follows the data gravity principle, *i.e.*, it moves the computational resources close to the data sources or the network edge (see bottom tier in Fig. 1). This leads to a multitude of benefits (Hu et al., 2017). First, it offers low response times, possibly in milliseconds, crucial for time-critical tasks such as flight control, healthcare, autonomous cars and gaming (Li et al., 2019c; Gill et al., 2019). Second, it allows us to build reliable systems where service resilience is provided at the node level, allowing other compute devices to act as backups and ameliorating performance degradation by reducing service downtimes using failover and fallback mechanisms (Bagchi et al., 2019). A major challenge at the Edge is that devices have limited computational capabilities and therefore suffering significantly under stress. There also exists a vast amount of devices in an IoT system, giving rise to bandwidth contentions (Belcastro et al., 2021).

Serverless Computing. Serverless computing emerged as a solution for the complexity of Cloud and Edge computing that hides server usage and runs user codes on-demand automatically with high scalability at the function level, such that the users are only billed for the code execution time (Castro et al., 2019). It is agnostic to the specific set of resources we utilize, Edge or Cloud. Platforms and architectures have been recently proposed in the literature to extend serverless capabilities to Edge computing (Javadi et al., 2020; Cicconetti et al., 2020). In serverless, the applications use precisely the amount of resources needed at any one point in time and charge accordingly, making the costs proportional to the exact resource usage (Hendrickson et al., 2016). Even though the tight integration in serverless makes it friendly for user, it also makes it hard for developers to optimize QoS when running serverless applications due to the lack of data management in serverless. Unlike containers and VMs that allow independent monitoring of each running service, serverless frameworks abstract out the active functions in the system, reducing the viability of tuning them for performance optimization.

2.2. Shift to fog continuum

There are typically several resource-constrained edge nodes in close proximity to the users and resource-abundant cloud nodes that are at multi-hop distance. This imposes the challenge of managing the resource-latency trade-off between edge and cloud layers, which fog continuum aims to address. None of the previously mentioned paradigms are ideal for building a generic computational platform for the end-users. The high latency of Cloud nodes, the unreliability of

Table 2
The importance level of performance parameters in different industries.

Industry	Resp. Time	Cost	Energy	Accuracy	Reliability
Agriculture	Medium	High	Medium	Medium	High
Healthcare	High	Medium	Medium	High	High
Construction	Low	High	Low	Medium	Low
Food	Medium	High	Medium	Medium	Medium
Transport	High	Low	Medium	High	High
Textile	Low	High	Medium	Medium	High
Gaming	High	Low	Low	Medium	Medium
Aviation	High	Low	High	High	High
Smart Cities	Medium	Medium	High	High	High

Edge devices, and the limited exposure of the resource management level controls offered by serverless frameworks motivate researchers to leverage all these paradigms in tandem, giving rise to the Fog continuum.

Fog Continuum. Fog is a parallel and distributed computing paradigm introduced by CISCO in 2012 as an interface between the Cloud and Edge computing systems to support latency-critical and resource-hungry application services by providing an interface between the computation and storage offered by Cloud and Edge (Bonomi et al., 2012). Fog introduces a hierarchical architecture with an intermediate layer between end-users and Cloud datacenters which utilizes computational, storage, and networking resources that reside within the path connecting users to the Cloud (Mahmud et al., 2018). These resources known as Fog nodes include gateways, switches, routers, nano datacenters, Cloudlets, etc. Unlike traditional fog or mist platforms, *fog continuum* is an umbrella term that includes edge only, cloud only and hybrid edge-fog-cloud resources. As Fog resources are distributed, heterogeneous, and resource-constrained compared to Cloud data centers, efficient resource provisioning and application placement algorithms are vital for harvesting the full potential of the Fog continuum.

2.3. Services

We now describe the various architectures utilized by the Fog continuum to service user requests. Each service architecture imposes disparate set of constraints on and the control surface expose to the underlying resource management techniques, possibly utilizing AI models.

Infrastructure-as-a-Service (IaaS). IaaS provides physical or virtual hardware resources (*i.e.*, compute, storage, network infrastructure, etc.) on a pay-for-what-you-use basis. This eliminates the need for the initial investment in hardware and provides users with an easy and convenient way to remotely access, monitor, and configure infrastructure as a service (Soualhia et al., 2019; Gill et al., 2019). IaaS gives AI-based resource managers control over provisioning, scaling of hardware resources, and deploying software on available hardware resources to maintain required levels of QoS for their deployed applications without having the responsibility of managing and controlling the underlying infrastructure.

Platform-as-a-Service (PaaS). PaaS provides consumers with a development and execution environment that consists of a set of tools to create and deploy their own applications (Varghese and Buyya, 2018; Buyya et al., 2009; Zhang et al., 2015). This service simplifies application deployment by providing only platform level controls and hiding infrastructure level controls from the user. However, PaaS allows underpinning AI-based resource management solutions to control the applications and configurations of the platform that hosts the applications. A specific type of PaaS, Machine Learning-as-a-Service (MLaaS), presents ML technologies such as Deep Learning require large-scale computation power to be viable. MLaaS abstracts out the deployment aspects and is used to describe Fog systems that provide out-of-the-box support for enabling ML technologies such as data pre-processing, model training and inference. Such systems aim to provide ease of use to users who are looking to develop and deploy their own machine learning applications efficiently.

Software-as-a-Service (SaaS). SaaS provides the highest level of abstraction by providing consumers with the capability to use applications running within Fog or Cloud resources that the service provider manages (Gill et al., 2019; Buyya et al., 2009; Varghese and Buyya, 2018). This only provides AI resource managers with limited capability to control certain application configurations. This is because the underlying architecture and application capabilities are controlled and managed by the service provider.

2.4. Optimization parameters

We now describe the various Quality of Service (QoS) parameters of a Fog system that we expect an AI-based resource manager to optimize for ideal system performance.

Response Time. This parameter indicates the service delivery time. Within distributed Fog environments, the response time of a service depends on multiple parameters such as data transmission time, propagation time, processing time, and service deployment time (Mahmud et al., 2020a). Thus, Fog resource provisioning and application scheduling consider the response time as a vital parameter for utilizing distributed and heterogeneous Fog resources, along with remote Cloud datacenters to prioritize applications/services with stringent latency requirements for placement within Fog environments. In Cloud and Edge environments, consumers and providers negotiate these QoS parameters to establish a Service Level Agreement (SLA) (Buyya et al., 2009). SLAs are critical in deadline-oriented tasks such as flight management systems, self-driving car networks and gaming. As IoT applications are heterogeneous in their characteristics (e.g., time-sensitive healthcare applications, data-intensive surveillance applications, etc.), QoS-aware scheduling mechanisms are necessary to utilize resource-constrained devices.

Cost. The cost of using Cloud and Edge environments depends on the type of service used by the consumer and the pricing model (*i.e.*, on-demand, reserved or spot pricing) employed by the service provider. Cloud allows potential cost savings in case of computation on large-scale. Due to the limited computation capacity of the Fog nodes, novel pricing models are introduced for Fog environments (Mahmud et al., 2020b). Thus, Fog application placement aims to reach a tradeoff between cost and response time, to minimize the cost of deployment while satisfying deadline requirements of the applications (Deng et al., 2020a).

Energy. IoT is highly scalable, with a large number of sensors generating a significant amount of data for processing. This results in higher energy consumption and carbon footprint in Cloud datacenters during data transmitting and processing (Oma et al., 2018). Fog continuum, with its distributed architecture, has the potential to achieve higher energy efficiency by relying on low-power edge nodes when possible (Gill et al., 2019; Mahmud et al., 2020a), but is limited by the energy and computation capacity of the Fog nodes (Mahmud et al., 2020a). This motivates resource provisioning and application placement algorithms to reach a tradeoff between time and energy in an IaaS or PaaS platform (Ghanavati et al., 2020). When utilizing resource-heavy AI models for resource provisioning in broker nodes, the brokers themselves can lead to high energy consumption. This makes it crucial to develop AI methods that are energy efficient also in terms of their inference.

Reliability. Reliability of Fog systems is quantitatively defined as a probability measure of how frequently a system delivers the services it has been designed for. Edge and Fog nodes are prone to different types of failures, including hardware failures, software failures, network failures and resource overflow (Bagchi et al., 2019). Dynamic issues such as battery constraints, connection fluctuations, resource availability, and mobility problems can contribute to the complexity of the reliability of such systems (Carvalho et al., 2021). These failures are likely to be more frequent in Edge and Fog servers due to their geographical dispersion, distributed deployment, and lack of maintenance and support from providers. Even a small failure probability per node is cascaded by the presence of a large number of interconnected nodes. Therefore, reliable Fog systems must be implemented that has a low failure rate, and when it does fail, it recovers quickly.

Accuracy. We use accuracy as a general term to highlight the performance of an AIaaS/MLaaS service in terms of the closeness of model output to the true or expected outputs. This can include classification performance, detection accuracy or prediction error. Several metrics exist in literature to measure the performance of an AI model, such as precision, recall, F1 score, confusion matrix and area under the receiver operating characteristic (AUROC). When deploying AI-based workloads on Fog systems, it is crucial that the choice of AI models is based on the accuracy specifications from the user. Some application use-cases, such as healthcare, require extremely accurate results. On the other hand, other scenarios, for instance autonomous systems, need near real-time inference. AI models have distinct performance and inference times.

2.5. Synergy with industrial IoT/industry 5.0 applications

Growth in adoption of various technologies including Industrial Internet of Things (IIoT), Industry 5.0 and aforementioned computing systems has been unprecedented in recent years and as a result several industries are utilizing these technologies to improve their productivity and services (Liu et al., 2017a). We now provide a brief overview for some important industrial applications under the umbrella terms of Industry 5.0 and how they relate to performance parameters including response time, cost, energy, accuracy and reliability when they adopt Cloud, Fog and Edge platforms. We consider energy as an indicator of the carbon footprint of the different services. The overview of this analysis is presented in Table 2 where the importance level of parameters are classify as high, medium, and low.

The *Agriculture* industry widely uses various sensors for monitoring humidity, temperature, soil moisture to better control and maintain the plants and trees in the large scale agricultural fields (Misra et al., 2020). Important performance parameters for Fog systems would be cost and reliability as they have direct impact on the final cost and the quality of the harvest. System response time, accuracy and energy are in the medium level of importance. *Healthcare* leads to the adoption of various sensors for patient monitoring and providing real-time feedback to the patient and caregivers (Kumari et al., 2018). Healthcare systems need low response time with high accuracy and reliability as they need to provide real-time response. The *Construction* industry aims to keep tracking of the projects, and site safety. The most important metric is cost which is the main decision point for the adoption of such systems in the construction industry (Abioye et al., 2021). *Food* industry has widely adopted IIoT, Cloud and AI in different stages including production, transport, storage and consumption which led to the proposing of “Internet of Food” (Boulos et al., 2015). The potential Fog system for this industry should be very cost effective to minimize the overall cost of the food. The *Transport* industry aim to make travel more efficient by utilizing a large number of IIoT sensors, especially by the advent of autonomous vehicles (Nikitas et al., 2020). Here, reliability, accuracy and response time are the most important metrics for self-driving cars. The *Textile* industry uses Fog continuum in smart textile for cost effective and reliable system to curtail supply chain costs. The *Gaming* industry is the growing entertainment industries, the quality of user experience is highly dependent on low latency and reliable response to the users. The *Aviation* industry is now leading to the new evolutionary era called Aviation 5.0 impacting manufacturing, through aircraft operation and air traffic management. Reliability is the key importance metrics for such a system. This development of *Smart cities* spans from intelligent traffic management to trash collection and air quality control. The main performance metrics for such a system are energy, accuracy and reliability. An overview of highly important

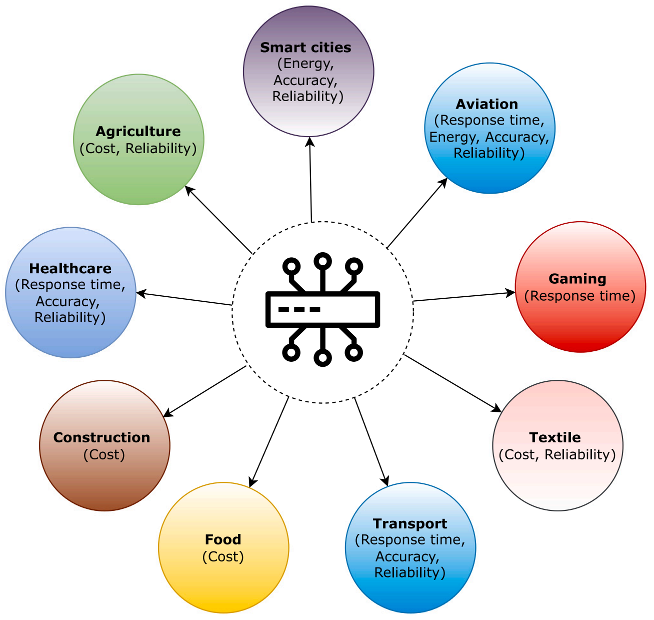

Smart cities (Energy, Accuracy, Reliability)
Agriculture (Cost, Reliability)
Aviation (Response time, Energy, Accuracy, Reliability)
Healthcare (Response time, Accuracy, Reliability)
Gaming (Response time)
Construction (Cost)
Textile (Cost, Reliability)
Food (Cost)
Transport (Response time, Accuracy, Reliability)

Fig. 3. Critical performance metrics of Fog continuum in Industry 5.0 applications.

performance metrics for Fog continuum systems adopted in industrial applications is illustrated in Fig. 3, which indicated reliability is the most common metrics in these applications. The rest of the discussion considers all mentioned metrics used to measure the performance of AI based resource management solutions. However, the specific choice of metrics is subject to the application use-case and deployment scenario as mentioned above.

3. AI integration in systems

Considering the background discussion in Section 2, we have established the control surface provided by Edge and Cloud paradigms for resource management. We also present the parameters optimized by AI models to generate management decisions in Fog systems. This needs extensive integration between the Fog systems and AI methods. To this end, a plethora of approaches have been developed, both at simulation and deployment levels, which provide an interface between the two technologies. We discuss these interfacing technologies in this section.

3.1. Simulators and frameworks for fog research

We first discuss the tools that allow modeling and testing of Fog systems.

3.1.1. Simulated platforms

A simulated platform enables researchers to test their methods at scale quickly. However, as simulators are approximations of the physical systems, they may provide noisy results or deviate from observations.

Popular Fog simulators, such as iFogSim, provide a modular, event-driven simulation platform, created on top of CloudSim, a widely used simulator for Cloud environment simulations (Gupta et al., 2017; Calheiros et al., 2011). iFogSim enables simulation of distributed and heterogeneous Fog nodes and scheduling of IoT based application workflows. Prior work (Shahidinejad and Ghobaei-Arani, 2021; Suryadevara, 2021; Etemadi et al., 2021; Tuli et al., 2020b) uses this simulator to analyze a wide range of scheduling algorithms such as evolutionary algorithms, machine learning, deep learning and reinforcement learning algorithms. Another CloudSim based simulator, IoTSim-Edge, allows users to test IoT infrastructure and framework by providing a testbed for deploying IoT Edge devices as a simulation in a single application (Jha et al., 2020). IoTSim-Edge also separates the broker and worker layers by explicitly defining an *Edge Broker* that acts as a simulated Fog device that manages Edge resources, and an *Edge Device* as simulated worker nodes. Similarly, PureEdgeSim (Mechalikh et al., 2019b) takes an edge focused control of Fog systems, particularly used for disease diagnosis (Javaid et al., 2021) and fuzzy tree based decision making (Mechalikh et al., 2019a). Others, like SimEdgeIntel, provides cross-platform and cross-language support, thus enabling easy integration of machine learning-based resource management policies (Wang et al., 2021a). It supports mobility modeling, network configuration and implementation of multiple handover mechanisms. A similar simulator, Deep FogSim, is designed to support large-scale evaluations of the delay-energy performance of Conditional Neural Networks (CDNNs) within Fog environments (Scarpiniti et al., 2021). It provides a software platform to model computing and network aspects of Fog environments and simulates the performance of the inference phase of CDNNs on top of Edge or Cloud nodes.

ECSim++ is a simulator (Nguyen and Huh, 2018) that extends the OMNetpp++ (Varga, 2010), which presents capabilities of power control and cache management, making it more realistic than other simulated devices. RelIoT This is a reliability simulator for IoT-based Fog systems (Ergun et al., 2020) and presents metrics such as power consumption, execution time, breakdown time and network characteristics such as throughput, delay, network and jitter. Unlike other simulators, it offers several combinations of reliability metrics to measure the fault resilience of a Fog system. Yet Another Fog Simulator (YAFS) (Lera et al., 2019) is a simulator that allows users to monitor network topologies, device resources and network resources. Unlike other simulators, it includes network path routing and user or device level movement as part of the control knobs it offers. A serverless simulator, SimFaaS (Mahmoudi and Khazaei, 2021) acts as a platform with serverless functionalities. It contains out-of-the-box support for simulating essential serverless properties such as *cold/warm starts* and auto-scaling. It supports the stateless/function based programming paradigm and has been demonstrated to effectively simulate real usage scenarios (Mahmoudi and Khazaei, 2021). However, it still lacks support for simulating heterogeneous systems, node failures and large-scale deployments.

Apart from the above simulators, there are also simulators such as EmuFog (Mayer et al., 2017), FogTorch (Brogi and Forti, 2017), BigHouse (Meisner et al., 2012) and Sim4DEL (Liu et al., 2021b). They are all simulators that focus on other aspects of Cloud systems such Fog topologies, storage and sensor infrastructures, accurate device simulations, streaming systems and federated deep Edge learning. There are also two Cloud-based Fog and Edge device simulators: Azure IoT (Stackowiak, 2019) and AWS IoT device simulator (AWS, 2021). These focus on simulating large-scale IoT systems with support for simulating thousands of devices, serverless functions within Cloud VMs and integrating live sensors and actuators. As such, these two can be used since they provide support using their large pool of back-end Cloud resources.

3.1.2. Physical platforms

For credible AI-augmented Fog research, testing developed solutions on emulators that duplicate industrial deployment scenarios on physical platforms is increasingly important.

OpenStack is an open-source platform developed by Rackspace Inc. and NASA, originally developed for Cloud environment, but later also extended to support Edge devices, thanks to its modularized APIs (Sefraoui et al., 2012). OpenStack has custom hypervisor drivers that can support a variety of virtualization technologies such as KVM, QEMU, UML, Xen, VMware, Docker and many more, making it very versatile option for Edge virtualization. Other platforms, such as KubeEdge (Wang et al., 2020d) and OpenEdge (OpenEdge, 2021), are based on Kubernetes virtualization technology (Kristiani et al., 2018). They provide functionalities for efficient communication between Edge and Cloud as well deployment of various AI-based applications (Wang et al., 2020d). They also contain APIs that control assignment of device resources to different workloads, which allows for efficient use of resources for the already resource-constrained Edge devices.

Another framework, FogBus, facilitates IoT-Fog-Cloud integration to run multiple applications using platform-independent interfaces provided by the platform (Tuli et al., 2019b) by following master-worker topology where master nodes known as Fog Brokers are responsible for delegating data processing tasks to the worker Fog nodes. Similarly, EiF *i.e.*, Elastic Intelligent Fog, An et al. (2019) is a framework that supports AI-based service migration, predictive network resource allocation and predictive QoS-aware orchestration along with support for distributed AI. A recent framework, COSCO, *i.e.*, Co-Simulation based Container Orchestration (COSCO) (Tuli et al., 2022e), is a framework that presents AI-based resource management modules to not only utilize the workload resource utilization characteristics, but also simulated characteristics at a future state of the system. The interleaved execution of AI models and coupled simulation (referred to as co-simulation in literature) enables long-term optimization (Tuli et al., 2022b) and quick adaptation in volatile system settings (Tuli et al., 2022a,c).

3.2. AI benchmarks for fog systems

For research related to *AI on Fog*, several workload like benchmarks have been utilized to test the efficacy of Fog systems, such as Raspberry Pi clusters, when dealing with AI-based applications. These are summarized in Table 3.

A popular Fog benchmark, DeFog (McChesney et al., 2019),[1] consists of six real-time heterogeneous workloads: Yolo, Pocketspinx, Aeneas, FogLamp, iPokeMon and RealFD. Yolo uses Convolution Neural Network (CNN) for object classification in images. Pocketsphinx is a Natural Language Processing (NLP) based speech-to-text synthesis engine that utilizes an AI-based search strategy. Aeneas is a text and audio synchronization tool that utilizes text to speech tools with AI-based search for minimizing speech deviation metrics. iPokeMon is an adaptation of the game Pokemon Go with simulated players and service requests for network testing in Fog. FogLamp is an application that uses aggregated sensor data and simulated data retrieval requests to test the storage bandwidth of Fog devices. RealFD uses computer vision for face detection in video streams. Other benchmarks, such as AIoTBench (Luo et al., 2018)[2] and EdgeAIBench (Hao et al., 2018),[3] are AI-based Edge computing based benchmark suites that consist of various real-world computer vision application instances. The former consists of CNN neural networks for image classification. These include three typical heavy-weight networks: ResNet18, ResNet34, ResNext32 × 4d, as well as four light-weight networks: SqueezeNet, GoogleNet, MobileNetV2, MnasNet. The latter includes applications such as ICU patient monitoring and heart failure prediction using attention-based LSTMs, surveillance camera video face-detection using CNN networks and road-sign detection for autonomous vehicles using CNNs.

Another AI benchmarking suite, AIBench (Gao et al., 2018),[4] includes a wide variety of AI applications including image classifications using CNNs, image generation using Generative Adversarial Networks (GANs), text-to-text translation using recurrent neural networks (RNNs), speech-to-text using Gated Recurrent Units (GRUs) and LSTMs, recommendation system using collaborative filtering and spatial image transformations using Transformer neural networks. EdgeBench (Das

[1] DeFog: https://github.com/qub-blesson/DeFog.
[2] AIoTBench: https://www.benchcouncil.org/aibench/aiotbench/index.html.
[3] EdgeAIBench: https://www.benchcouncil.org/aibench/edge-aibench/index.html.
[4] AIBench: http://www.benchcouncil.org/AIBench/index.

Table 3
Comparison of AI based benchmarks for Fog systems in terms of workload coverage.

Benchmark	AI Search	AI Filtering	Robotics	Vision/ CNN	NLP/ RNNs	Trans- formers	GANs
DeFog (McChesney et al., 2019)	✓			✓			
AIoTBench (Luo et al., 2018)				✓			
AIBench (Gao et al., 2018)		✓		✓	✓	✓	✓
EdgeAIBench (Hao et al., 2018)				✓	✓		
EdgeBench (Das et al., 2018)	✓			✓			
IoTBench (Celik et al., 2018)			✓	✓			

et al., 2018)[5] includes audio to text translation using AI search and object recognition using CNNs. IoTBench (Celik et al., 2018) consists of multiple AI models run simultaneously under the same input workloads. It includes applications for image classification using CNNs and robotics workloads related to Simultaneous Localization and Mapping (SLAM) of robot environments.

Apart from the above mentioned benchmarks, several execution traces are used by state-of-the-art AI augmentation techniques as datasets for simulation based testing. Bitbrain consists of traces of resource utilization metrics from 1750 VMs running on BitBrain distributed datacenter (Shen et al., 2015). Azure2017 and Azure2019 are collected from Microsoft Azure public Cloud platform and are representative workload traces across thirty consecutive days (Cortez et al., 2017). Google Cluster is another dataset of CPU and memory utilization traces of multiple nodes in a high-performance cluster in Google Cloud Platform (Reiss et al., 2011). Server Machine Dataset (SMD) (Su et al., 2019) is a five-week long dataset of resource utilizations of 28 machines from a compute cluster. Other request trace datasets include HDFS (HDFS, 2021), MHealth (MHealth, 2021), PlanetLab (PlanetLab, 2021), Wikimedia (Wikimedia, 2021), Bikeshare (Bikeshare, 2021), Shakespeare (Shakespeare, 2021), SETI (Javadi et al., 2009), Crawdad (Piorkowski et al., 2009), Traffic (Sivanathan et al., 2018), T-Drive (Yuan et al., 2010), Tutoring (Metevier et al., 2019), WfCommons (Coleman et al., 2021) and Yahoo Webscope (Yahoo Webscope, 2021).

3.3. AI modeling and engineering

Now that we have described the various simulation and emulation platforms for Fog systems, AI toolkits and benchmarking suites, we elucidate the challenges faced while training or running inference using an AI model. Model training is a highly resource-intensive task due to the large number of parameters in modern AI and Deep Learning models, and traditionally requires the use of high-performance clusters, Graphical Processing Units (GPUs) or Tensor Processing Units (TPUs). As such, given that Edge devices usually contain limited resources, the training overheads are significant and can take significantly longer time compared to Cloud nodes. Additionally, the hardware resources for Edge devices are used by other applications in parallel to the training process, giving rise to frequent resource contentions. Furthermore, unlike traditional clusters, Cloud engineers have little to no control over the availability of the Edge devices, making training a challenging task.

Currently, two mainstream methods of model development and training exist: Centralized and Federated Learning (FL) (Lim et al., 2020). In a centralized learning system, the AI model, typically a DNN, with the training dataset is kept in a single resource-intensive machine with the training framework updating the parameter updates iteratively until convergence to minimize a developer-defined loss function (Lim et al., 2020). In a federated learning setup, the model to be trained is sent to multiple Edge devices that contain a subset of the training data. The model is trained locally using the local hardware and the parameter updates are iteratively aggregated into a global copy of the model. Centralized learning requires a high-end system, but does not lead to high bandwidth use as in FL. Federated learning requires nodes to synchronize models iteratively that can lead to network contentions and increased wait times. However, as only the parameter updates are shared across the nodes and not the local data, it ensures data privacy.

4. State of the art methods

Given the optimization parameters in Fog systems, infrastructure level constraints offered by the simulator/frameworks, we now review the state-of-the-art methods for the three aspects resource management: provisioning, scheduling and maintenance to optimize the QoS metrics including response time, cost, energy, accuracy and resiliency.

4.1. Interface between AI and fog

Data sources and inputs for AI models. For any resource management system in the Cloud, for instance, an AI model, this paradigm provides multi-modal data sources to analyze the system. Traditionally, these include workload resource utilization traces in the form of a fraction of CPU, RAM, Disk and Network bandwidth utilization and host resource capacities in the form of instruction per second (IPS), available RAM and Disk space and parameters of the network interface (Tuli et al., 2022e; Mao et al., 2016; Jalali et al., 2019; Basu et al., 2019; Tuli et al., 2021b). Other parameters include gateway bandwidths, geographical location of users and Fog nodes (Lera et al., 2019), communication latencies and mobility characteristics (Ye et al., 2018).

Control Knobs and outputs of AI models. We categorize the state-of-the-art approaches for AI-augmented Fog continuum as per the decisions they aim to optimize.

1. *Deployment:* This deals with the initial decisions of how to efficiently execute resource-intensive AI applications on constrained Fog systems. In *AI on Fog*, this concerns with appropriate methods to deploy resource-intensive AI/ML models on constrained nodes. This entails deciding the appropriate strategy to compress AI models without compromising on performance. We discuss this aspect in Section 4.2.1. In *AI for Fog*, this concerns the efficient allocation of resources for the input workloads. This includes the provisioning of resources, *i.e.*, allocation of new and deallocation of existing Fog devices. We discuss this aspect in Section 4.2.2.
2. *Scheduling:* This deals with scheduling the deployed workloads on existing Fog infrastructure. This concerns with optimal placement of tasks on Fog nodes to optimize system QoS. As we consider a dynamic setting, our scheduling decisions also include task migration decisions, *viz*, the relocation of one or a group of tasks from one node to another, allowing the system to adapt to changes in the environments, workloads or user demands. In *AI on Fog* setups, if the incoming workloads have tasks that impose precedence constraints, such as different layers of a neural network, we categorize these as workflows and discuss relevant methods in Section 4.3.2. Schedulers for independent tasks are discussed in Section 4.3.1.
3. *Maintenance:* Even though there are several methods based on redundancy that have been proposed for Cloud computing systems (Sharma et al., 2016), these techniques cannot be directly applied to Fog systems due to resource limitations. This means that Fog servers may have less capability to use redundancy techniques

[5] EdgeBench: https://github.com/akaanirban/edgebench.

Table 4
Summary of state-of-the-art methods for AI augmented deployment. Color scheme as per Table 7.

Decision type	Category	Ref.	Method	Infra.	Benchmark	Framework/Simulator	Merits	Limitations
DNN deployment	AutoML	Xia et al. (2019), Li et al. (2019c) and Zhao et al. (2021)	Policy gradient learning	E	AIoTBench	F: Custom	Fast execution	Generalizability
	Model pruning	Jiang et al. (2019b) and Liu et al. (2017b)	Search	E+C	AIoTBench	F: Custom	Cost efficient	Accuracy
		Shao and Zhang (2020a), Yu et al. (2020) and Huang et al. (2020c)	Neural design	E+C	AIoTBench	–	Generalizable	Overhead
		Zhou et al. (2021)	Search	E+C	AIoTBench	F: Custom	Cost efficient	Overhead
	Gradient pruning	Sattler et al. (2019) and Luo et al. (2021b)	Search	E	AIBench	F: Custom	Low overhead	Accuracy
		Reisizadeh et al. (2020), Hamer et al. (2020) and Tran et al. (2019)	Neural design	E+C	Shakespeare	S:Custom	Low overhead	Accuracy
	Low precision	Coelho et al. (2021) and Jain et al. (2018)	Neural design	E+C	–	–	Generalizable	Scalability
		Lane et al. (2016) and Imani et al. (2019)	Search	E+C	AIBench*	F: Custom	Memory efficient	Generalizability
	Layer splitting & Early exits	Kang et al. (2017) and Zhang et al. (2021b)	Search	E+C	AIBench*	S: Bighouse	Energy efficient	Generalizability
		Callegaro et al. (2020)	Linear programming	E	–	–	Fast execution	Scalability
		Matsubara et al. (2019), Teerapittayanon et al. (2017) and Goli et al. (2020)	Neural design	E+C	AIBench*	–	Low overhead	Accuracy
		Yu et al. (2021)	Policy gradient learning	C	AIBench*	F: AWS IoT	Memory efficient	Generalizability
	Splitting	Kaplunovich and Yesha (2020)	Search	C	AIBench*	F: AWS IoT	Generalizable	Overhead
		Chen et al. (2018), Huang et al. (2020d) and Kim et al. (2017)	Neural design	E+C	AIoTBench*	F: Custom	High accuracy	Generalizability
		Tuli (2022)	DQN + Multi armed bandits	E+C	AIoTBench*	F: COSCO	High QoS	Overhead
Resource provisioning	Demand prediction	Hyndman and Athanasopoulos (2018)	Linear regression	E+C	–	S: Custom	Fast execution	Accuracy
		Zhu et al. (2016)	Support vector regression	C	Google Cluster	S: Custom	Memory efficient	Scalability
		Chen and Wang (2018), Luo et al. (2020) and Luo et al. (2021a)	Gaussian process regression	E+C	Azure2017/19	S: Custom	High accuracy	Scalability
		Taylor and Letham (2018)	Modular regression	E+C	–	–	Scalable	Accuracy
		Singh et al. (2019) and Calheiros et al. (2014)	ARIMA	C	Wikimedia	S: Custom	High accuracy	Scalability
		Xu et al. (2020a)	Decision regression tree	E	Bikeshare	–	Fast execution	Only univariate
		Bega et al. (2019) and Jeddi and Sharifian (2019)	CNN	E+C	Custom	–	High accuracy	Interpretability
		Ouhame et al. (2021) and Yazdanian and Sharifian (2021)	LSTM	C	Custom	S: Custom	High accuracy	Interpretability
	Decision optimization	Aliyu et al. (2020)	Ant colony optimization	E+C	Bitbrain	S: Cloudsim	High util. ratio	Scalability
		Luo et al. (2020) and Luo et al. (2021a)	Bayesian optimization	E+C	Azure2017/19	S: Custom	High util. ratio	Scalability
		Zhu et al. (2016) and Chen et al. (2020b)	Particle swarm optimization	C	Google cluster	S: Custom	Memory efficient	Scalability
		Asghari et al. (2021)	Genetic algorithm	C	–	S: Cloudsim	High util. ratio	Execution time
	Hybrid provisioning	Wilder et al. (2019)	Deep surrogate optimization	E+C	Yahoo Webscope	S: Custom	Cost efficient	Generalizability
		Stuckey et al. (2020)	DNN + Dynamic Prog.	E+C	–	–	High util. ratio	Scalability
		Levy et al. (2020)	FCN + Multi armed bandits	C	Custom	F: Kafka	Fast & Scalable	Exposure bias
		Chen et al. (2019)	FCN + Monte Carlo tree search	C	–	–	High QoS	Execution time
		Xu et al. (2020b), Bitsakos et al. (2018) and Sami et al. (2021)	Deep Q learning	E+C	Google Cluster	S: Custom	Scalable	Interpretability
		Xu et al. (2020c), Chen et al. (2020a) and Chen et al. (2021)	Policy gradient learning	E+C	–	S: iFogSim	Low energy	Scalability

like replication (Shivakumar, 2015). This leads to higher response times and SLA violations that can lead to significant financial losses (Nicoletti, 2013). Thus, it is critical to develop a mechanism for the maintenance of Fog environments. This deals with detecting faults/anomalies in real-time, discussed in Section 4.4.1. Also, we consider works that develop appropriate proactive or reactive recovery mechanisms to prevent service downtime. They are either related to load-balancing methods that aim at preventing faults or scaling the Fog infrastructure (see Section 4.4.2).

Most data-driven methods achieve local optimum. However, some approaches are developed to avoid getting stuck in such local optima (Loshchilov and Hutter, 2016), although they do not guarantee achieving global optima. We now describe how data-driven AI methods can be used to solve the deployment, scheduling and maintenance challenges in fog continuum.

We now move to the three aspects of AI augmented resource management introduced in Section 1.2. We consider works that leverage either a physical framework or a simulated environment.

4.2. AI augmented deployment

A summary of all AI augmented deployment methods is presented in Table 4. Here, a *benchmark* corresponds to the workloads used to train and test the presented methods. *Infra.* column represents whether the methods utilize Edge (E) or Cloud (C) or both (E+C). Asterisk with framework/simulator means that the respective papers utilize a modified version of the base platforms.

4.2.1. AI augmented DNN deployment

Running training or inference procedures for AI/ML models is computationally expensive. Given that Edge devices tend to have limited compute resources that are usually shared across multiple running applications, it is essential to develop resource-efficient training and inference mechanisms to ensure short training times and resource load. Several solutions have been proposed in the past to address this.

AutoML. For instance, several AutoML (automated machine learning) techniques run search in the space of neural network models, *i.e.*, NAS, to find out the optimal DNN model to execute a task in a given system (Xia et al., 2019; Zhou et al., 2019; Li et al., 2019c; Zhao et al., 2021). These methods can be run to find the optimal DNN architecture for a given set of constraints such as training or inference times, memory footprint, computational requirements, etc.

Model Pruning. Another direction is to take existing AI models and prune their parameters to reduce overall local training cost, for instance, PruneFL (Jiang et al., 2019b). Model pruning is a commonly used strategy in ML that intelligently cuts away parts of the model architecture without compromising the quality of model inference (Shao and Zhang, 2020a; Yu et al., 2020; Huang et al., 2020c). Compared to AutoML, by pruning out parts of the model, the computational resources required to train or run inference on the model are reduced, making it more amenable to Edge and Fog systems. Similar model pruning works are dependent on the DNN design. For instance, Generative Optimization Networks (GON) (Tuli et al., 2021c) are generative models inspired by GANs that use two neural networks in tandem: generator and discriminator. Unlike GANs, GONs do not use the generator and create new samples only using the discriminator network. Other examples include SlimGAN (Hou et al., 2021), Gradient Origin Networks (Bond-Taylor and Willcocks, 2021) and similar GAN slimming techniques (Wang et al., 2020a). Similar works exist that perform model pruning for other DNN types, for instance, CNNs (Liu et al., 2017b). Pruning reduces both the memory and compute requirements of models. Similarly methods like BBNet utilize multiple techniques together, such as model pruning and data compression (Zhou et al., 2021). BBNet decides the optimal pruning and compression parameters using local search-based techniques.

Gradient Pruning. In FL systems, all worker nodes need to send gradient updates of their models over the network, which generally translates to up to gigabytes of data depending on the size of the model. To avoid bandwidth contentions, several works discuss solutions to prune the gradient updates improving memory and network efficiency in federated setups (Yang et al., 2020a). Some approaches, such as Sparse Ternary Compression (STC) (Sattler et al., 2019) and Deep Gradient Compression (DGC) (Luo et al., 2021b), employ compression mechanisms to reduce the communication bandwidth required for distributed training or inference. Other methods, like FedPaq (Reisizadeh et al., 2020), perform periodic aggregation and quantization to reduce communication frequency. A similar approach, CMFL (Luping et al., 2019), intelligently decides which model updates give the maximum boost in performance and only sends the top-performing gradient updates. Another work, FedBoost (Hamer et al., 2020), uses *ensemble training* to boost model training efficiency and offload only a small part of the ensemble to the Edge devices with predefined intervals to reduce communication overheads. Another work, FEDL (Tran et al., 2019), theoretically demonstrates the relationship between convergence rate and energy consumption of an FL system and formulates the computation and communication models as a non-convex optimization program to optimize the distribution of federated networks and outperformed other learning methods.

Low-Precision. Energy efficiency is also one of the major concerns when designing efficient ML training algorithms due to FL parties generally being battery-powered devices. Deep learning is inherently very power-consuming due to the large amounts of computation that need to be performed. As such, there has been a relatively large amount of work in energy efficiency by discrete quantization and using low-precision hardware architectures (Coelho et al., 2021; Gong et al., 2019; Jain et al., 2018; Langroudi et al., 2019b,a). This not only reduces the computational overheads, but also gives significant gains in terms of memory and energy footprints (Gong et al., 2019; Tuli et al., 2021c). The level of precision cannot be changed dynamically at test time as changing precision requires re-training the models. Thus, these decisions either need to be performed at the setup time or multiple models need to be trained of different precision levels, at the cost of higher training time, to provide control to the resource manager to tradeoff between accuracy and memory footprint. Other optimization methods, such as DeepX (Lane et al., 2016), focus on the development of deep learning models on mobile devices by runtime control of the memory to reduce the layer-wise operations, such that only the most important operations use larger bytes. Further, it efficiently identifies unit blocks of a DNN architecture and allocates them to local or remote memory caches depending on the access frequency, improving memory footprint. A similar method, FloatPIM (Imani et al., 2019) provides an interface between software and hardware by using Processing in-memory (PIM) to reduce memory usage.

Layer Splitting and Early-Exits. For typical DNNs, it is possible to run inference without performing operations across all layers. Methods, such as Neurosurgeon (Kang et al., 2017) and DeepSlicing (Zhang et al., 2021b), decide the optimal layer partitions of a neural network using grid-search at run time to maximize system QoS. Others, like SplitComp (Callegaro et al., 2020) model the problem of deciding the optimal splitting strategy as a Markov Process and leverage Linear Programming to converge to the optimal splitting strategy. Further, to reach the best tradeoff between model accuracy and processing delay, many early-exit strategies have been proposed where the inference is performed only through a few layers instead of the entire DNN and do not use the complete DNN (Li et al., 2019c; Pacheco et al., 2021; Wang et al., 2019a). Most work in this category aims at segregating these network splits into different devices based on their computational performance (Matsubara et al., 2019; Teerapittayanon et al., 2017; Goli et al., 2020; Zhang et al., 2021b; Kang et al., 2017). Thus, fast and localized inference using shallow portions of DL models can allow quick inference, possibly at the cost of poorer resulting accuracy. This gives a tradeoff between result fidelity and response time. Several works have been proposed to leverage this tradeoff for multi-objective optimization, especially to reduce the frequency of SLA violations (Tuli et al., 2019a; Yang et al., 2020b). Other recent methods aim at exploiting the resource heterogeneity in the same network layer by splitting and placing DNNs based on user demands and host capabilities (Gunasekaran et al., 2020). Such methods can split DNNs and choose from different architectural choices to reach the maximum accuracy while agreeing to the latency constraints. Other works aim at accelerating the model run-times by appropriate scheduling of a variety of DNN models on Edge-clusters (Liang et al., 2020). Another method, Gillis, uses a hybrid model, wherein it employs either model-compression or layer-splitting based on the application SLA demands (Yu et al., 2021). The decision is taken using a reinforcement-learning model which continuously adapts in dynamic scenarios. It is a serverless-based model serving system that automatically partitions a large model across multiple serverless functions for faster inference and reduced memory footprint per function. The Gillis method employs two model partitioning algorithms that respectively achieve latency optimal serving and cost-optimal serving with SLA compliance. However, this method cannot jointly optimize both latency and SLA. Moreover, it does not consider the mobility of devices or users and hence is ineffective in efficiently managing large DNNs in mobile Edge computing environments.

Splitting. Two types of splitting strategies exist: data splitting and semantic model splitting. Data splitting splits the input data batch across multiple instances of the neural networks for parallel inference. Data splitting allows reducing the response time of inference over input data, at the cost of higher network overheads (Kaplunovich and Yesha, 2020). Semantic model splitting divides the network weights into a hierarchy of multiple groups that use a different set of features. Here, the neural network is split based on the data semantics, producing a tree structured model that has no connection among branches of the tree, allowing parallelization of input analysis (Kim et al., 2017). Due to limited information sharing among the neural network fragments, the semantic splitting scheme gives lower accuracy in general compared to unsplit networks. Semantic splitting requires a separate training procedure where publicly available pre-trained models cannot be used. This is because a pre-trained standard neural network can be split layer-wise without affecting output semantics. For semantic splitting, we would need to first split the neural network based on data semantics and re-train the model. However, semantic splitting provides parallel task processing and hence lower inference times, more suitable for mission-critical tasks like healthcare and surveillance. Examples of such methods include ThriftyEdge (Chen et al., 2018), CLIO (Huang et al., 2020d), SplitPlace (Tuli, 2022) and SplitNet (Kim et al., 2017; Ushakov et al., 2018).

TinyML. This is a paradigm where the objective is to run complex deep learning models within resource constrained embedded devices (Ray, 2021). Although many of the above approaches have high overlap with the methods considered in the scope of TinyML, we specifically discuss the advances in computational algorithms to augment resource management in fog environments. For instance, hyper-dimensional computing (HDC) is an approach that consumes much lower energy compared to conventional methods. Here the tensors of DNNs are mapped to higher dimensional tensors (Ge and Parhi, 2020). Another approach to improve the memory footprint and minimize the read/write latencies is swapping (Miao and Lin, 2021) where DNN models are efficiently swapped between the on-chip memory of the microcontroller and external flash memory. Another recent approach is *attention condenser* that is an auxiliary neural network that learns self-attention to condense the size of the input (Wong et al., 2020).

4.2.2. AI augmented resource provisioning

Systematic resource provisioning is central to cost and resource-efficient computation in Fog systems. Bootstrapping resources, such as Cloud VMs or Edge nodes is time-consuming for latency-critical tasks; a key challenge is to predict future workload demands to provision resources to optimize QoS. Resource management is a key aspect of resource provisioning, which instantiates and deallocates resources based on dynamic workload demands. Most prior work aims to automate resource provisioning to optimize various performance measures such as energy consumption, cost, and task response time (Tuli et al., 2021b; Levy et al., 2020). However, this problem is challenging due to the non-stationary utilization characteristics of most workloads (Ebadifard and Babamir, 2021), requiring methods to dynamically adapt their provisioning policies. Most dynamic resource provisioning methods decouple the provisioning problem into two stages: demand prediction and decision optimization (Luo et al., 2020). This is commonly referred to as the *predict+optimize* framework in literature. Thus, we divide prior approaches based on their decision type.

Demand Prediction. Methods that forecast demands at a future state of a Fog system need data corresponding to historical workload demands on the same system. Several methods have been proposed that leverage a forecasting model. For instance, a class of methods utilize regression models such as Linear Regression (LR) (Hyndman and Athanasopoulos, 2018), Support Vector Regression (SVR) (Zhu et al., 2016), Gaussian Process Regression (Chen and Wang, 2018; Luo et al., 2020, 2021a) or modular regression (Prophet) (Taylor and Letham, 2018). Others utilize auto-regressive models such as AutoARIMA (Singh et al., 2019; Calheiros et al., 2014) or other ML models like Regression Tree (RT) (Xu et al., 2020a), time series decomposition (TSDec) (Hyndman and Athanasopoulos, 2018) or unobserved component model (UCM) (Durbin and Koopman, 2012) based forecasting. Recent works utilize DNNs to perform forecasting, for instance, using LSTM neural networks (Ouhame et al., 2021), CNNs (Bega et al., 2019), convolutional wavelet neural networks (Jeddi and Sharifian, 2019) or LSTM based GANs (Yazdanian and Sharifian, 2021). DNN based demand prediction models are known to outperform classical AI or regression based approaches (Yazdanian and Sharifian, 2021; Jeddi and Sharifian, 2019).

Decision Optimization. Using a demand prediction model, several previous works optimize the provisioning decision to minimize execution costs or maximize the utilization ratio. Conventional methods often use evolutionary search strategies such as Ant Colony Optimization (ACO) (Aliyu et al., 2020), which has been shown to exhibit state-of-the-art QoS scores in recent work (Luo et al., 2021a). Others use Bayesian Optimization (BO) (Luo et al., 2020, 2021a), Particle Swarm Optimization (PSO) (Zhu et al., 2016; Chen et al., 2020b) or Genetic Algorithms (Asghari et al., 2021). Among the different approaches, ACO and PSO are appropriate for static scenarios, whereas GA and BO are more suitable for highly dynamic settings (Tuli et al., 2022e; Asghari et al., 2021).

Hybrid Provisioning. Other methods, such as Decision-NN, combine the prediction and optimization steps by modifying the loss function to train neural networks in conjunction with the optimization algorithm (Wilder et al., 2019). This method uses a neural network as a surrogate model to directly predict optimization objectives and uses the concept of neural network inversion, wherein the method evaluates gradients of the objective function with respect to inputs and runs optimization in the input space. However, continuous relaxation of the discrete optimization problem used in this work has been shown to adversely impact performance (Luo et al., 2020). A similar method, Semi-Direct, utilizes dynamic programming to find the optimal provisioning decision (Stuckey et al., 2020), but offers limited scalability with workload size. Similarly, Narya (Levy et al., 2020) is built for mitigating VM interruptions in Cloud machines, but can be straightforwardly extended to resource provisioning. It uses a neural network as a surrogate model with a multi-armed bandit model to decide provisioning actions. However, it faces the problem of exposure bias, *i.e.*, the neural model is biased to the trends in the training data and is unable to forecast in unseen cases.

Reactive Provisioning. Recently, RL based methods have been proposed for reactive provisioning. For instance, Intelligent Resource Allocation Framework (iRAF) (Chen et al., 2019) solves the complex resource allocation problem for the collaborative mobile Edge computing (CoMEC) network using Deep Reinforcement Learning (DRL) with a multi-task objective formulation. It makes resource allocation decisions based on network states and other task characteristics such as the computing capability of devices, network quality, resource utilization, and latency requirements. iRAF automatically takes into account the network environment and makes resource allocation decisions to maximize the performance over latency and power consumption. It uses self-play training where the agent becomes its own teacher and learns over time in a self-supervised learning manner. Specifically, it uses a fully connected network (FCN) with Monte Carlo Tree Search (MCTS) to optimize the provisioning decision. Some other works, such as DDRM (Chen et al., 2020a), focus on the integration of IoT and industrial manufacturing systems (IIoT). The authors argue that due to the limitation of computing capacity and battery, computation-intensive tasks need to be executed in the mobile Edge computing (MEC) server. Another similar work (Baek and Kaddoum, 2020) focuses on optimizing the Fog nodes by selecting the suitable nodes and proper resource management while guaranteeing the QoS requirements of the users. It designs a joint task offloading and resource allocation control for heterogeneous service tasks in multi-fog nodes systems. It applies a deep recurrent Q-network (DRQN) approach to approximate the optimal value functions and applies an adjusted exploration-exploitation method to make the optimization process more efficient. Similarly, ReCARL (Xu et al., 2020b) focuses on Cloud Radio Access Networks (CRANs). It proposes a resource allocation scheme in CRANs to improve the objective of power consumption and SLA violations of wireless users over a long time period. To do this, it uses DRL to solve a custom convex optimization problem and apply a Deep Neural Network (DNN) to approximate the action-value function. It uses two DRL agents: ReCARL-Basic (requiring limited training) and ReCARL-Hybrid (requiring deep learning training). It has been evaluated via extensive simulation to demonstrate that ReCARL achieves significant power savings in highly dynamic settings while meeting user SLA demands. Similarly, Deep Elastic Resource Provisioning (DERP) (Bitsakos et al., 2018) uses Deep-Q learning to optimize provisioning decisions with utilization ratio as a reward for the DRL agent. Unlike Q-learning based agents that utilize a neural network to predict the expected reward for each action (Sami et al., 2021), recent methods also use neural networks to approximate the optimal policy. Such approaches are called policy gradient methods and include (Xu et al., 2020c; Chen et al., 2021). The state-of-the-art policy gradient methods outperform traditional reinforcement learning (Q learning) and Monte Carlo based approaches (Xu et al., 2020c; Chen et al., 2021).

Table 5
Summary of state-of-the-art methods for AI augmented scheduling. Color scheme as per Table 7.

Decision type	Category	Ref.	Method	Infra.	Benchmark	Framework/Simulator	Merits	Limitations
Bag-of-Tasks scheduling	Maxweight scheduling	Krishnasamy et al. (2018)	Maxweight	E+C	–	S: Custom	Fast execution	Resp. Time
		Liu et al. (2020)	Maxweight	E+C	Tutoring	S: Custom	Fast execution	Resp. Time
	Surrogate modeling	Jiang et al. (2019a)	DNN + Linear programming	E	–	S: Custom	Energy efficient	Accuracy
		Han et al. (2018)	DNN + GA	E+C	–	–	Generalizable	Overhead
		Tuli et al. (2020a)	GMM + GA	E+C	Bitbrain	S: iFogSim	Fast execution	Overhead
		Tuli et al. (2022e)	DNN + Gradient Opt.	E+C	DeFog	F: COSCO	High QoS	Interpretability
		Tuli et al. (2021b)	GNN + Gradient Opt.	C	DeFog	F: COSCO	High QoS	Exposure bias
	Stochastic modeling	Jamshidi and Casale (2016) and Bui et al. (2017)	Gaussian process regression	C	Google Cluster	S: Custom	Fast execution	Overhead
		Panda et al. (2015) and Jawad et al. (2018)	Robust Search	C	Custom	–	Energy efficient	Overhead
		Alelaiwi (2019)	DBN + Search	E+C	–	S: Custom	Fast execution	Accuracy
		Tuli et al. (2022a)	NPN + Gradient Opt.	E+C	DeFog	F: COSCO	High QoS	Interpretability
	Reinforcement learning	Tang et al. (2018) and Li et al. (2019b)	Deep Q learning	E+C	Crawdad	S: Cloudsim	Generalizable	Scalability
		Wang et al. (2021b)	Minimax Q learning	C	Traffic	S: Clousim	Generalizable	Scalability
		Sheng et al. (2021)	Policy gradient learning	E	–	S: Custom	High QoS	Adaptability
		Tuli et al. (2020b) and Cheng et al. (2021)	Policy gradient learning	E	Bitbrain	S: iFogSim	High QoS	Adaptability
	Co-Simulation	Tuli et al. (2022e) and Tuli et al. (2022a)	FCN + Co-Simulation	E+C	DeFog	F: COSCO	Generalizable	Overhead
Workflow Scheduling	Meta-heuristic methods	Wang et al. (2020e)	Particle swarm optimization	C	WfCommons	F: AWS IoT	Generalizable	Low QoS
		Huang et al. (2020a)	Ant colony optimization	E+C	WfCommons	F: AWS IoT	Generalizable	Low QoS
		Ghanavati et al. (2020)	Ant mating optimization	E+C	WfCommons	S: Custom	Energy efficient	Overhead
	Surrogate optimization	Ismayilov and Topcuoglu (2020)	DNN + GA	C	WfCommons	–	Generalizable	Overhead
		Pham and Fahringer (2020)	DNN + GA	C	WfCommons	S: Cloudsim	Cost efficient	Overhead
		Tuli et al. (2022b)	DNN + Gradient Opt.	E+C	WfCommons	F: COSCO	High QoS	Interpretability
	Game theory	Wang et al. (2020b)	Attack-Defense Model	C	WfCommons	F: Openstack	Cost efficient	Overhead
	Reinforcement learning	Wang et al. (2019b)	Deep Q learning	C	WfCommons	F: AWS IoT	Generalizable	Scalability
		Kaur et al. (2020)	Deep Q learning	C	WfCommons	S: Clousim*	Generalizable	Scalability
		Ghosal et al. (2020)	Policy gradient learning	E+C	–	–	High QoS	Adaptability
		Hu et al. (2019)	Policy gradient learning	C	WfCommons	F: Custom	High QoS	Adaptability
	Other	Feng et al. (2019)	LSTM	E+C	Custom	–	Generalizable	Scalability
		Wang et al. (2020f)	Support vector regression	E	–	S: Custom	Fast execution	Scalability
		Alsurdeh et al. (2021)	Gradient descent search	E+C	Custom	S: CloudSim	High QoS	Overhead

4.3. *AI augmented scheduling*

A summary of recent AI-augmented scheduling methods is presented in Table 5.

4.3.1. *AI augmented scheduling of bag-of-tasks*

QoS-aware placement of IoT applications requires reaching a trade-off among multiple conflicting QoS parameters such as response time, cost and energy. In the *bag-of-task* workload model, each task can be independently scheduled.

MaxWeight-Scheduling. Over the years, many scheduling approaches have turned to utilize MaxWeight based techniques due to its theoretical guarantees and the ability to reduce the frequency of resource contention (Liu et al., 2020; Krishnasamy et al., 2018). For instance, the pessimistic-optimistic online dispatch approach, POND, is a variant of the MaxWeight approach (Liu et al., 2020). POND formulates the scheduling problem as a constrained optimization objective with unknown dispatch, arrival and reward distributions, such that each Fog node has a virtual queue to track violation counts. It uses an Upper-Confidence Bound (UCB) based exploration strategy (Auer et al., 2002) with the final decisions being made with the MaxWeight weights as the expected reward value of each scheduling decision. However, prior work has demonstrated that MaxWeight policies suffer from instability in dynamic workloads, high delays and inefficiency in modeling large-scale Fog networks (Bae et al., 2019; van de Ven et al., 2009, 2013). MaxWeight schedulers are also known to have high wait times due to their inability to adapt to volatile workload settings (Tuli et al., 2022e).

Surrogate Modeling. Most classical research in this area employs meta-heuristic algorithms with a DNN or regression model as a surrogate that approximates QoS of a given system state. This is due to their generic formulation and ease of implementation. For instance, prior works have shown that evolutionary-based methods, and generally gradient-free approaches, perform well in dynamic scenarios (Wang et al., 2019c; Han et al., 2018; Wang et al., 2020g; Tuli et al., 2020a). Some works use a combination of a DNN surrogate, and classical optimization techniques such as mixed-integer linear programming (MILP) (Jiang et al., 2019a). Evolutionary approaches such as genetic algorithms (GA) lie in the domain of gradient-free optimization methods. The GA method schedules workloads using a neural model to approximate the objective value and a genetic algorithm to reach the optimal decision (Han et al., 2018). Such methods use either analytical models (Wang et al., 2019c), Gaussian Mixture Model (GMM) or polynomial approximators (Tuli et al., 2020a) or neural networks (Han et al., 2018) to predict system QoS for a given scheduling decision and input Fog state. Typically, such approaches run a search scheme with non-local jumps, due to cross-over and mutation-like operations, to converge towards an optimum. However, gradient-free methods are known to take much longer to converge (Bogolubsky et al., 2016) and are not as scalable (Rios and Sahinidis, 2013) as gradient-based methods. Moreover, non-local jumps can significantly change the scheduling decision, leading to a high number of preemptive task migrations. This entails checkpointing the running task, migrating it to another Fog node and resuming its execution on the new node (Engelmann et al., 2009). This can give rise to high migration overheads, subsequently increasing the average task response times and SLA violation rates. Furthermore, prior work also establishes that neural approximators can precisely model the gradients of the objective function with respect to input using back-propagation (Nguyen-Thien and Tran-Cong, 1999). Now, although such works use these gradients with respect to input for solving differential equations, they can also be applied for gradient-based optimizations. However, even with the advantages of scalability and quick convergence to optima, few prior works use gradient-based methods as neural approximators are not consistent with the convexity/concavity requirements of such methods (Nandi et al., 2001). This problem is alleviated by momentum and annealing in schedulers like GOBI and GOSH (Tuli et al., 2022e,a). Such methods take the scheduling decision and state of the Fog system as resource utilization characteristics of workloads and Fog nodes and output a QoS estimate. Using backpropagation to input, *i.e.*, fixing the neural network parameters and updating the scheduling decision based on the gradient of DNN output, these methods find the optimal scheduling decisions. Other schedulers, like HUNTER (Tuli et al., 2021b), model the input scheduling decision as a graph and use Graph Neural Networks (GNNs), facilitating the inference by capturing the correlations across workloads and Fog nodes. However, with such models, as they run black-box optimization steps, the interpretability of their outputs is low. Further, continuous approximation of a discrete optimization problem is known to give sub-optimal decisions in some cases (Miranda-Varela and Mezura-Montes, 2018).

Stochastic Modeling: Another type of models that approximate system QoS is stochastic surrogate models. These include Heteroscedastic Gaussian Processes to approximate the distribution of the QoS metrics instead of giving only a static output for a given input state (Jamshidi and Casale, 2016; Bui et al., 2017; Panda et al., 2015). Similarly, prior works also predict the mean and variance estimates of system QoS based on historical data to perform robust and safe decision optimization (Panda et al., 2015; Jawad et al., 2018) or use error-based exploration (Jamshidi and Casale, 2016). Other methods use Deep Belief Networks (DBN) for response-time predictions, which are used to make prompt offloading decisions under mobility and fluctuating resource demands (Alelaiwi, 2019). Typically, due to the poor modeling accuracy of Gaussian Processes, they cannot perform well in complex environments like heterogeneous Fog environments. Hence, more sophisticated models like Bayesian Neural Networks (BNNs) to additionally model the stochasticity in the QoS metrics (Jawad et al., 2018; Wu et al., 2020). Recent state-of-the-art methods also rely on Natural Parameter Networks (NPNs) that allow using arbitrary exponential family of distributions to model the weights and parameters of a neural networks (Tuli et al., 2022a).

Reinforcement Learning models: Recently, reinforcement learning based methods have shown themselves to be robust and versatile to diverse workload characteristics and complex Fog setups (Tuli et al., 2020b; Tang et al., 2018; Basu et al., 2019; Gazori et al., 2019). Such methods use a Markovian assumption of state which is the scheduling decision at each interval. Based on new observations of reward signals, they explore or exploit their knowledge of the state-space to converge to an optimal decision. Recent methods, such as DQLCM (Tang et al., 2018) and DeepRM (Li et al., 2019b), model the container migration problem as a multi-dimensional Markov Decision Process (MDP) and use a deep-reinforcement learning strategy, namely deep Q-Learning to schedule workloads in a heterogeneous Fog environment. Another similar method, SDAEM-MMQ (Wang et al., 2021b) uses a stacked denoising autoencoder with minimax Q learning for accurate Q estimates and robust optimization. Policy gradient methods, such as Sheng et al. (2021), train a DNN to directly predict the optimal scheduling decision instead of Q values. A recent method, Asynchronous Advantage Actor–Critic (A3C), is a policy gradient method that schedules workloads using an actor–critic pair of DNN agents (Tuli et al., 2020b). This approach uses Residual Recurrent Neural Networks (R2N2) to predict the expected reward for each action *i.e.*, scheduling decision and tries to optimize the cumulative reward signal. Another similar method, Multi-Agent Deep Deterministic Policy Gradient (MADDPG) (Cheng et al., 2021) formulates the decision optimization problem as a stochastic game among multiple RL agents to reach to an optimal schedule. However, such methods are still slow to adapt to real-world application scenarios (Tuli et al., 2022e). This leads to higher wait times and subsequently high response times and SLA violations, leading to poor scalability with workload or the number of nodes in the Fog system.

Coupled Optimization. Finally, coupled or symbiotic simulation and model-based control have long been used in the modeling and optimization of distributed systems (Onggo et al., 2021; Bosmans et al., 2019; Onggo et al., 2018). Many prior works have used hybrid simulation models to optimize decision-making in dynamic systems (Mustafee et al., 2015; Onggo et al., 2018). To achieve this, they monitor, analyze, plan and execute decisions using previous knowledge-base corpora (MAPE-k) (Gill et al., 2019). However, such works use this to facilitate search methods and not generate additional data to aid the decision-making of an AI model. Recent methods, such as GOBI* (Tuli et al., 2022e), use an interleaved decision optimization and co-simulation to run an interactive dynamic between the different levels of fidelity, *i.e.*, simulation and surrogate, to optimize QoS. A similar work (Onggo et al., 2018), presents the notion of *symbiotic simulation* that aims to feed in the resource characteristics related data to a co-simulated model for optimizing resource management related decisions using ML techniques. Another similar work, EDSS (Onggo et al., 2021), uses a co-simulator to estimate the effects of various resource scheduling decisions from an ML model and choose the one with highest QoS. Running a co-simulator gives another estimate of system QoS, solving two problems at once: the problem of exposure bias to training data as well as the data saturation problem. The former arises due to the surrogate models being trained on a set of pre-collected execution traces, wherein the system characteristics might be different from those at test time. The latter arises due to the limited diversity in training data such that even with more datapoints, the performance of the DNN does not improve.

4.3.2. *AI augmented scheduling of workflows*

Workflow like applications typically have precedence constraints of the form of a DAG that must be adhered to when scheduling such applications. These workloads could be of the form of a layer or semantic split neural models (see Section 4.2.1) or other scientific workflow applications (Gill et al., 2019).

Meta-Heuristic methods. This class of methods leverages high-level problem independent algorithms to find the optimal scheduling decision for the workflows. Most state-of-the-art approaches belong to this category. Among these, many use variants of the PSO optimization technique (Wang et al., 2020e). One such technique is the immune-based particle swarm optimization (IMPSO) method (Wang et al., 2020e). It uses candidate affinity to prevent poor candidates from being discarded in subsequent iterations, allowing it to surpass other PSO-based methods in terms of execution costs and average response time. Other techniques, categorized commonly as list scheduling, use metrics like earliest finish time, critical path, and dynamic resource utilization levels (Adhikari et al., 2019). However, list scheduling performs poorly in settings with non-preemptable jobs and heterogeneous requirements or machines (Adhikari et al., 2019). Others include ACO, such as Huang et al. (2020a). Such a technique starts with several random or heuristically initialized candidate solutions. Each candidate is iteratively optimized, moving it slightly in the state-space where the optimization objective tends to increase. Such methods aim to reach a balance between makespan-service spread, makespan-energy and makespan-cost, respectively. Further, novel bio-inspired meta-heuristic algorithms are also introduced to solve Fog application scheduling simultaneously considering multiple objectives. For instance, the Ant Mating Optimization (AMO) (Ghanavati et al., 2020), aims to minimize the total system makespan and energy consumption for Fog task scheduling.

Surrogate Optimization. Other recent methods use genetic algorithms to optimize the scheduling decision, again using a DNN as a surrogate model (Ismayilov and Topcuoglu, 2020; Pham and Fahringer, 2020). Again, due to non-local jumps in the search space, such methods typically lead to better QoS estimates, at the cost of higher task migration overheads (Tuli et al., 2022e). Recent techniques, such as ESVR (Pham and Fahringer, 2020), initialize its candidate population using the Heterogeneous Earliest Finish Time (HEFT) heuristic and optimize using the crossover-mutation scheme. To account for volatility in the system, ESVR continuously fine-tunes the neural network surrogate using the latest workload traces and host characteristics (Pham and Fahringer, 2020). A similar technique is DNSGA (Ismayilov and Topcuoglu, 2020) that uses a multi-objective optimization method that uses a Pareto Optimal Front (POF) aware approach that prevents the set of candidates from converging to the same optima (Ismayilov and Topcuoglu, 2020). Prior work shows that these two methods outperform previously proposed genetic algorithms-based techniques (Ismayilov and Topcuoglu, 2020; Pham and Fahringer, 2020). However, in the case of long-running workflows, having a short-term QoS estimate is detrimental to the system performance as it leads to myopic optimization. To tackle this, recent methods, such as Monte-Carlo Deep Surrogate (MCDS) (Tuli et al., 2022b), trains a DNN to generate long-term QoS estimates by running multiple Monte-Carlo simulations on a co-simulator. This not only helps in long-term optimization, but also facilitates stable learning.

Game-Theoretic Modeling. Another recently proposed workflow scheduling model, namely Closure, uses an attack-defense game-theoretic formulation (Wang et al., 2020b). Unlike other schemes that assume mostly homogeneous resources, Closure has been shown to efficiently manage heterogeneous devices by calculating the Nash Equilibrium of the attack-defense game model. This is crucial in Edge-cloud environments where there are contrasting resource capacities of Edge and Cloud.

Reinforcement Learning. Some methods constrain the action space of the MDP formulation to exclude scheduling decisions that violate the precedence constraints set by the incoming workloads. These include removing infeasible actions from the action set at each state of the MDP (Wang et al., 2019b) for Deep Q Networks (DQN) or masking the policy likelihood scores in policy gradient methods (Hu et al., 2019; Ghosal et al., 2020). For instance, DQ-HEFT (Kaur et al., 2020) superimposes the task order over the reward function to ensure that the Q-learning model converges to an optimal scheduling decision.

Other. Many prior works utilize other augmentation strategies in tandem with AI. For instance, some works optimize the Fog network. Examples include (Jalali et al., 2019, 2017), which introduce cognitive Edge gateways that use machine learning (regression and ensemble models) to automatically learn the best allocation for each task based on the Fog environment status and performance requirements of the tasks. Similarly, other methods (Wang et al., 2020f) propose an intelligent task offloading algorithm to synergistically run them on Edge and Cloud platforms. Their dynamic switching algorithm groups applications using a support vector machine based approach to improve the performance in terms of delay and energy consumption. Other methods such as gradient descent search (Alsurdeh et al., 2021) has been adopted for hybrid workflow scheduling in Edge and cloud computing to optimize execution time and monetary cost.

4.4. AI augmented maintenance

In this work, we focus on the aspect of maintaining Fog systems using resource management techniques, particularly concerned with fault tolerance, resilience and remediation. Resilience is crucial when utilizing AI for resource management, as corrupted computation from failed nodes can lead to ML systems having erroneous behavior. Such errors can be fatal in some scenarios, such as autonomous driving and medical predictions. We measure system resilience with three metrics.

1. *Resource Contention:* Stressful workloads tend to overwhelm the resource capacities of the Edge or Cloud nodes, leading to competition among workloads for resources. This competition can cause failures due to inefficient resource scheduling, resulting in outages. The most common way of mitigating this is by proactive resource provisioning that ensures sufficient resources are available for incoming workloads before they arrive. However, it is crucial to eschew the over-provisioning of resources to avoid system under-utilization or resource wastage in Fog systems.
2. *Service Availability:* It is possible that the node performing a crucial computation fails due to hardware or software faults. This disrupts the service provided to the user and is a critical metric to measure system reliability. It is usually addressed by having multiple Fog nodes involved in the processing of the same application so that one can take over if the other fails. This resilience concept is known as hot-standby, where the backup resources are called fallback nodes (Zhao et al., 2020a). However, this metric trades off with energy and cost as application replication leads to redundant computations, leading to inefficiency.
3. *Security and Privacy:* Fog systems must also ensure data resilience to ensure that data is not compromised by malicious attacks (Zhang et al., 2019a). This includes data integrity, *i.e.*, resilience to data corruption, and data confidentiality, *i.e.*, sensitive data remains hidden from malicious entities. To avoid data corruption and stealing, technologies such as encryption (Bonawitz et al., 2017), differential privacy (Abadi et al., 2016), detection (Preuveneers et al., 2018) are used.

A summary of resilience methods for Fog systems in presented in Table 6.

4.4.1. AI augmented fault-detection and prediction

Several machine learning algorithms have been proposed for fault detection and prediction in Edge Cloud environments. They have proposed a framework that includes time-series data collection and data pre-processing components for the training of DNNs.

Unsupervised Reconstruction Models. Majority of prior work proposes reconstruction-based methods that predominantly aim to encapsulate the temporal trends and predict the time-series system data in an unsupervised fashion, then use the deviation of the prediction with the ground-truth data as anomaly scores. In such methods, the time-series system data may correspond to utilization characteristics of the running workloads in a Fog system. One such method, LSTM-NDT (Hundman et al., 2018), relies on an LSTM to forecast data for the next timestamp. This work also proposes a non-parametric dynamic error thresholding (NDT) strategy to set a threshold for anomaly labeling using moving averages of the error sequence. A similar work, Omnianomaly (Su et al., 2019), uses a stochastic recurrent neural network (similar to an LSTM-Variational Autoencoder Park et al., 2018) and a planar normalizing flow to generate reconstruction probabilities. It also proposes an adjusted Peak-Over-Threshold (POT) method for automated anomaly threshold selection that outperforms the previously used NDT approach. This work led to a significant performance leap compared to the prior art, but at the expense of high training times. The Multi-Scale Convectional Recursive Encoder Decoder (MS-CRED) (Zhang et al., 2019) converts an input sequence window into a normalized two-dimensional image and then passes it through a ConvLSTM layer. This method is able to capture more complex inter-modal correlations and temporal information, however is unable to generalize to settings with insufficient training data. The CAE-M (Zhang et al., 2021a) uses a convolutional autoencoding memory network, similar to MSCRED. It passes the time-series through a CNN with the output being processed by bidirectional LSTMs to capture long-term temporal trends. Such recurrent neural network-based models have been shown to have high computation costs and low scalability for high dimensional datasets (Audibert et al., 2020). The DAGMM (Zong et al., 2018) method uses a deep autoencoding Gaussian mixture model for dimension reduction in the feature space and recurrent networks for temporal modeling. This work predicts an output using a mixture of Gaussians, where the parameters of each Gaussian are given by a deep neural model. However, it still is slow and unable to explicitly utilize inter-modal correlations (Deng and Hooi, 2021). The Graph Deviation Network (GDN) approach learns a graph of relationships between data modes and uses attention-based forecasting and deviation scoring to output anomaly scores. MTAD-GAT (Zhao et al., 2020b) uses a graph-attention network to model both feature and temporal correlations and pass it through a lightweight Gated-Recurrent-Unit (GRU) network that aids detection without severe overheads. Traditionally, attention operations perform input compression using convex combinations where the weights are determined using neural networks.

Generative Models. More recent works such as USAD (Audibert et al., 2020), MAD-GAN (Li et al., 2019a) and openGauss (Li et al., 2021) do not use resource-hungry recurrent models, but only attention-based network architectures to improve training speeds. The USAD method uses an autoencoder with two decoders with an adversarial

Table 6
Summary of state-of-the-art methods for AI augmented maintenance. Color scheme as per Table 7.

Decision type	Category	Ref.	Method	Infra.	Benchmark	Framework/Simulator	Merits	Limitations
Fault detection and prediction	Unsupervised reconstruction models	Hundman et al. (2018), Su et al. (2019) and Park et al. (2018)	LSTM	E+C	SMD	S: Custom	Generalizable	Overhead
		Zhang et al. (2019) and Zhang et al. (2021a)	ConvLSTM	E+C	Custom	S: Custom	High accuracy	Scalability
		Zong et al. (2018)	DNN + GMM	E+C	Custom	S: Custom	High accuracy	Overhead
		Zhao et al. (2020b) and Deng and Hooi (2021)	Graph neural network	E+C	SWaT	S: Custom	High accuracy	Overhead
	Generative models	Li et al. (2019a) and Feng et al. (2021)	Generative adversarial nets	E+C	SWaT	–	Fast and Scalable	Exposure bias
		Audibert et al. (2020) and Gan et al. (2020)	VAE	C	–	F: Custom	Memory efficient	Scalability
		He et al. (2020) and Girish and Rao (2021)	GNN + LSTM + VAE	C	HDFS	F: Custom	High accuracy	Execution time
		Chouliaras and Sotiriadis (2021)	LSTM + VAE	C	–	–	Generalizable	Overhead
		Huang et al. (2020b)	Transformers	E+C	HDFS	F: Openstack	Fast execution	Generalizability
		Tuli et al. (2021c)	Neural design	E	SMD	F: COSCO	Memory efficient	Overhead
	Clustering methods	Won and Kim (2021)	Few shot learning	C	–	F: Openstack	Fast execution	Accuracy
		Li et al. (2018)	Fuzzy clustering	E+C	–	–	Fast execution	Accuracy
		Hu et al. (2021)	VAE + Fuzzy clustering	E+C	Custom	S: Custom	Fast execution	Accuracy
Fault remediation	Fault-tolerant scheduling	Talaat et al. (2020)	Q learning	E+C	MHealth	–	Generalizable	Scalability
		Talaat et al. (2019)	Deep surrogate optimization	E+C	MHealth	S: iFogSim	Cost efficient	Interpretability
		Liu et al. (2016)	Particle swarm optimization	C	–	S: Cloudsim*	Memory efficient	Scalability
		Satpathy et al. (2018)	Crow Search	C	Custom	S: Cloudsim	Energy efficient	Scalability
		Wang et al. (2021c)	Deep Q learning	E+C	–	S: Custom	High throughput	Scalability
		Tuli et al. (2022c)	Generative adversarial nets	E	DeFog	F: COSCO	High QoS	Data Hungry
	Load balancing	Jan et al. (2021)	Particle swarm optimization	E+C	–	S: iFogSim	Fast execution	Scalability
		Kaur and Aron (2021)	Grey wolf optimization	E+C	WfCommons	S: iFogSim*	Cost efficient	Scalability
		Marahatta et al. (2020)	DNN + Search	C	Google Cluster	S: Cloudsim	High Util. Ratio	Interpretability
		Arabnejad et al. (2017)	Fuzzy logic + Search	C	–	F: Openstack	Low overhead	Low QoS
	Scaling	Etemadi et al. (2021)	RNN	E+C	Custom	S: iFogSim	Low overhead	Scalability
		Abdullah et al. (2020)	Decision tree regression	E+C	Custom	F: Custom	Fast execution	Scalability
		Etemadi et al. (2020)	NAR network	E+C	T-Drive	S: iFogSim*	Cost efficient	Scalability
		Li et al. (2020a)	DNN + ARIMA	E+C	Custom	–	Cost efficient	Scalability
		Naha et al. (2021)	Fuzzy logic + Search	E+C	–	S: Cloudsim*	Fast and Scalable	Scalability
		Aral and Brandić (2021)	Dynamic Bayesian network	E	SETI	–	Fast execution	Interpretability
		Aral and Brandic (2017)	Bayesian neural network	E	Custom	–	Fast execution	Interpretability
	Straggler analysis	Aral and Brandic (2018)	Dynamic Bayesian network	E	SETI	–	Fast execution	Interpretability
		Tuli et al. (2021a)	VAE + LSTM	C	PlanetLab	S: Cloudsim	High QoS	Overhead

game-style training framework. This is one of the first works that focus on low overheads by using a simple autoencoder and can achieve a several-fold reduction in training times compared to the prior art. The MAD-GAN (Li et al., 2019a) uses an LSTM based GAN model to model the time-series distribution using generators. This work uses not only the prediction error, but also the discriminator loss in the anomaly scores. The openGauss approach uses a tree-based LSTM that has lower memory and computational footprint and allows capturing temporal trends even with noisy data. However, due to the small window as an input and the use of simple or no recurrent models, the latest models are unable to capture long-term dependencies effectively. A recently proposed HitAnomaly (Huang et al., 2020b) method uses vanilla transformers as encoder–decoder networks, but is only applicable to natural-language log data and not appropriate for generic continuous time-series data as inputs. Other methods, such as TopoMAD (He et al., 2020), use a topology-aware neural network that is composed of a Long-Short-Term-Memory (LSTM) and a variational autoencoder (VAE) to detect faults. However, the reconstruction error is only obtained for the last state, limiting them to using reactive fault recovery policies. Similar methods use slight variations of LSTM networks with either dropout layers (Girish and Rao, 2021), causal Bayesian networks (Gan et al., 2020) or recurrent autoencoders (Chouliaras and Sotiriadis, 2021). A GAN-based approach that uses a stepwise training process, StepGAN (Feng et al., 2021), converts the input time-series into matrices and executes convolution operations to capture temporal trends. However, such techniques are not agnostic to the number of hosts or workloads as they assume a maximum limit of the active tasks in the system. Moreover, even though more accurate than heuristic-based approaches, deep learning models such as deep autoencoders, GANs and recurrent networks are adaptive and accurate, but have a high memory footprint that adversely affects system performance. To resolve this, some works have been proposed that have low memory footprint, such as GONs (Tuli et al., 2021c).

Clustering Models. Very recent works also propose a few-shot learning method for fault detection (Won and Kim, 2021). Other recent methods utilize deep neural networks to execute fuzzy clustering (Hu et al., 2021; Li et al., 2018). For instance, the Adaptive Weighted Gath-Geva (AWGG) (Hu et al., 2021) clustering method is an unsupervised model that detects faults using stacked sparse autoencoders to reduce detection times. Such methods train using supervised labels and do not present a mechanism to recover from faults once detected, and hence cannot be used to develop end-to-end fault tolerance in Fog systems. Other methods, such as Isolation Forest (Liu et al., 2008) in an unsupervised method that is used for anomaly detection in systems (Tuli et al., 2022d).

4.4.2. *AI augmented fault remediation*

When Edge servers fail or are unavailable, optimal migration of the running tasks is crucial. However, it is also important to ensure that the task placement and scheduling procedures are fault-aware and aim to minimize system faults to minimize the overheads of running remediation strategies.

Fault-Aware Scheduling. Recently, several resilience models have been proposed that leverage AI methods like RL, surrogate or reconstruction modeling. Many of these methods run proactive scheduling and task placement steps to avoid faults in a future state. An RL based approach is Load Balancing and Optimization Strategy (LBOS) (Talaat et al., 2020) that allocates the resources using RL. The reward of the RL agent is calculated as a weighted average of multiple QoS metrics to avoid system contention by balancing the load across multiple compute nodes. The values of the weights are determined using genetic algorithms. LBOS observes the network traffic constantly, gathers the statistics about the load on each Edge server, manages the arriving user requests and uses dynamic resource allocation to assign them to available Edge nodes. However, RL approaches are known to be slow to adapt in dynamic settings (Tuli et al., 2022e). Most other approaches use neural networks as a surrogate model. For instance, Effective Load Balancing Strategy (ELBS) (Talaat et al., 2019) is a recent framework that offers an execution environment for IoT applications and creates an interconnect among Cloud and Edge servers. The ELBS method uses the priority scores to proactively allocate tasks to Edge nodes or worker nodes as brokers to avoid system failures. It uses a fuzzy inference system to calculate the priority scores of different tasks based on three fuzzy inputs: SLA deadline, user-defined priority, and estimated task processing time. The priority values are generated by a neural network acting in the capacity of a surrogate of QoS scores. The Proactive Coordinated Fault Tolerance (PCFT) (Liu et al., 2016) method uses Particle Swarm Optimization (PSO) to reduce the overall transmission overhead, network consumption and total execution time for a set of tasks. This method first predicts faults in the running host machines by anticipating resource deterioration and uses PSO to find target hosts for preemptive migration decisions. This approach mainly focuses on reducing transmission overheads in distributed Cloud setups but often fails to improve the I/O performance of the compute nodes. CSAVM (Satpathy et al., 2018) uses another evolutionary Crow Search scheme to take live migration decisions for the task queues. The method is used to optimize the power consumption of a compute setup by preventing unnecessary migrations. DDQP (Wang et al., 2021c) uses double deep Q-networks to place services on network nodes. However, such reinforcement learning schemes are known to be slow to adapt in volatile settings (Tuli et al., 2022e). Another such work is PreGAN (Tuli et al., 2022c), which uses a GAN to generate preemptive migration decisions and anomaly scores from an input Fog system state. It uses a co-simulator in tandem with a few-shot anomaly classifier to ensure robust model training and fine-tunes the model to adapt to dynamic scenarios.

Load Balancing for Tolerance. Load balancing is a concept that proactively aims to balance the load on different elements of a Fog infrastructure to avoid a faulty future system state. One such work, namely DPSO (Jan et al., 2021), proposes network gateways to host the load balancing logic where they monitor the load across Edge servers and balance the load using evolutionary algorithms. Furthermore, a migration mechanism is also incorporated where application modules are rearranged to achieve a balanced load across Edge servers. Migration is triggered based on a machine learning-based dynamic threshold. Similarly, FOCALB (Kaur and Aron, 2021) proposes a hybrid load balancing algorithm based on Grey Wolf Optimization (GWO) and Ant Colony Optimization (ACO), where energy consumption, execution time and implementation cost of scientific workflows are achieved by uniformly distributing the workload across Fog devices to optimize the Fog resource utilization. Similarly, some methdos (Arabnejad et al., 2017) propose a fuzzy logic based weighting scheme to run load-balancing task placement. Other methods, like PEFS (Marahatta et al., 2020), present a prediction-based energy aware fault-tolerant load balancing scheme that uses a neural network to predict faults in the system and run load-balancing strategies to ensure a high resource utilization ratio. Load balancing based approaches are proactive in terms of fault-tolerance and do not require to use additional compute infrastructure in case of failures, making them more suitable for resource constrained settings.

Automatic Scaling. Many methods aim to optimal decide how to scale the Fog infrastructure to avoid or recover from faults in the system. Similar to resource provisioning (Section 4.2.2), here too, it is vital to avoid over-provisioning and under-provisioning of limited Fog resources under dynamic workloads. An RNN based method (Etemadi et al., 2021) provides a deep learning based solution to utilize metrics such as resource requests (*i.e.*, CPU, RAM, etc.) and Fog resource status (e.g., CPU efficiency, storage utilization, network traffic, active/inactive resources) to make optimum auto-scaling decisions. Moreover, AI-augmented auto-scaling methods have the potential to support proactive auto-scaling of containers under dynamic workload fluctuations. Similarly, Abdullah et al. (2020) presents a predictive auto-scaling policy using decision tree regression (DTR) model where a reactive rule-based auto-scaling mechanism is employed to train the proactive model under multiple heterogeneous workloads. Existing works explore the use of AI-augmented workload forecasting (Li et al., 2020a; Etemadi et al., 2020) to make proactive auto-scaling decisions within Edge/ Fog environments. Another method, MADRP (Li et al., 2020a), uses a hybrid ARIMA and DNN model to forecast the workloads, whereas a nonlinear autoregressive (NAR) neural network is used by Etemadi et al. (2020) to predict the future demands for the Fog devices. Methods such as FLBFH (Naha et al., 2021), propose a fuzzy logic-based method to handle unpredicted and predicted failures in Fog environments. Such methods predict two failure scores to decide what actions to be undertaken to handle failures for unreliable Fog devices. The first failure score is based on device mobility, device response time and device power availability. This score determines the checkpointing interval as a proactive mechanism for unpredicted failures. The second score is based on CPU utilization, device mobility, device response time, device power availability, device communication. This score is used to decide about preemptive task migration. In some cases where the rate of unpredicted failure is high, the proposed fuzzy-logic mechanism will suggest an application replication. An extension to this work is the Dependency and Topology-aware Failure Resilience (DTFR) algorithm, which considers failure probability, response time and the number of replicas to schedule services on Edge servers (Aral and Brandić, 2021). DFTR explores the spatio-temporal failure dependency among Edge servers to develop a dynamic method with minimum redundancy to enhance the failure resilience of services. Other works (Aral and Brandic, 2017), continue this trend to estimate the availability level of VMs in Edge Data Centers (EDCs) based on Bayesian Networks. The probabilistic models consider dependencies between different failures such as hardware, software, or network failures, and power outage. This model is utilized to select VMs that can meet the availability requirements in SLA. Another similar method is the Fuzzy-based Real-Time Auto-scaling (FRAS) (Etemadi et al., 2021) technique that leverages a virtualized environment for the recovery of IoT applications that run on compromised or faulty Edge nodes. Here, FRAS executes each IoT application in a virtual machine (VM) and performs VM autoscaling to improve execution speed and reduce execution costs. The VM autoscaling decisions making involves inference of system QoS using a fuzzy recurrent neural network as a surrogate model.

Straggler Aware Models. Another common performance problem in Fog systems is dealing with straggler tasks that are slow running instances that increase the overall response time. Such tasks can significantly impact the system's QoS and the SLA. Methods, such as JFP (Aral and Brandic, 2018), exploit failure dependencies between Edge servers to predict the failure probability of given service deployment. JFP evaluates the use of replication in Edge servers based on analyzing historical failure logs of individual servers, modeling temporal dependencies as a Dynamic Bayesian Network (DBN), and predicting the probability at which a certain number of servers fail simultaneously. It also uses two replica scheduling algorithms to optimize failure probability and the cost of redundancy in an Edge computing environment. Similarly, other methods such as START (Tuli et al., 2021a), proactively predict the occurrence of straggler tasks to avoid adverse impact on system QoS.

5. Classification of state-of-the-art

Table 7 classifies the discussed state-of-the-art works in Section 4 as per the AI methods they use. This facilitates researchers in identifying the class of methods that have been used in the past and can be utilized to solve one of the scopes of deployment, scheduling and maintenance for resource management in Fog systems.

Classical AI. This category includes traditional AI schemes that do not utilize DNNs, such as local and evolutionary search, regression and meta-heuristic optimization schemes. We also include neural design, *i.e.*, the application specific design of neural models to achieve optimal performance or reduced overheads. We observe that the search and design based methods are quite popular in the case of DNN deployment. This is predominantly due to the search-driven DNN design specific improvements required to ensure optimal deployment of large-scale AI models on constrained Fog nodes. Nevertheless, we see some

Table 7
Classification of state-of-the-art techniques in terms of the used AI methods (Guan et al., 2021; Wan et al., 2012).

Category	Method	Deployment		Scheduling		Maintenance	
		DNN deployment	Resource provisioning	Bag-of-Tasks	Workflows	Detection and prediction	Tolerance
Classical AI	Informed and local search	Jiang et al. (2019b), Liu et al. (2017b), Zhou et al. (2021), Sattler et al. (2019), Luo et al. (2021b), Lane et al. (2016), Imani et al. (2019), Kang et al. (2017), Zhang et al. (2021b) and Kaplunovich and Yesha (2020)	.	Panda et al. (2015), Jawad et al. (2018) and Alelaiwi (2019)	Alsurdeh et al. (2021)	.	Satpathy et al. (2018) and Marahatta et al. (2020)
	Neural design	Shao and Zhang (2020a), Yu et al. (2020), Huang et al. (2020c), Reisizadeh et al. (2020), Hamer et al. (2020), Tran et al. (2019), Coelho et al. (2021), Jain et al. (2018), Matsubara et al. (2019), Teerapittayanon et al. (2017), Goli et al. (2020), Chen et al. (2018), Huang et al. (2020d) and Kim et al. (2017)	.	.	.	Tuli et al. (2021c)	.
	Maxweight, Linear/Dynamic Prog.	Callegaro et al. (2020)	Stuckey et al. (2020)	Liu et al. (2020)	Wan et al. (2012)	.	.
	Regression (Linear/Gaussian/SVM)	.	Hyndman and Athanasopoulos (2018), Zhu et al. (2016), Chen and Wang (2018), Luo et al. (2020), Luo et al. (2021a), Taylor and Letham (2018), Singh et al. (2019), Calheiros et al. (2014) and Xu et al. (2020a)	Jamshidi and Casale (2016), Bui et al. (2017), Onggo et al. (2021) and Onggo et al. (2018)	Wang et al. (2020f)	Zong et al. (2018)	Abdullah et al. (2020) and Li et al. (2020a)
	ACO/AMO/PSO/GA/BO/GWO	.	Aliyu et al. (2020), Luo et al. (2020), Luo et al. (2021a), Zhu et al. (2016), Chen et al. (2020b) and Asghari et al. (2021)	Han et al. (2018) and Tuli et al. (2020a)	Wang et al. (2020e), Huang et al. (2020a), Ghanavati et al. (2020), Ismayilov and Topcuoglu (2020) and Pham and Fahringer (2020)	.	Liu et al. (2016), Jan et al. (2021) and Kaur and Aron (2021)
Reinforcement learning	Tabular RL (SARSA/Q learning)	.	.	Wang et al. (2021b)	.	.	Talaat et al. (2020)
	Deep RL (DQN)	Tuli (2022)	Xu et al. (2020b), Bitsakos et al. (2018) and Sami et al. (2021)	Tang et al. (2018) and Li et al. (2019b)	Wang et al. (2019b), Kaur et al. (2020)	.	Wang et al. (2021c)
	Policy gradient learning	Xia et al. (2019), Li et al. (2019c), Zhao et al. (2021) and Yu et al. (2021)	Xu et al. (2020c), Chen et al. (2020a) and Chen et al. (2021)	Sheng et al. (2021),Tuli et al. (2020b) and Cheng et al. (2021)	Ghosal et al. (2020) and Hu et al. (2019)	.	.
Neural optimization	Deep surrogate optimization	.	Wilder et al. (2019)	Tuli et al. (2022e), Tuli et al. (2021b) and Tuli et al. (2022a)	Tuli et al. (2022b)	.	Talaat et al. (2019)
Neural approximation	Fully Connected Network (FCN)	.	Levy et al. (2020) and Chen et al. (2019)	Tuli et al. (2022e) and Tuli et al. (2022a)	.	.	.
	Convolutional Neural Network (CNN)	.	Bega et al. (2019) and Jeddi and Sharifian (2019)	.	.	Zhang et al. (2019) and Zhang et al. (2021a)	.
	Recurrent Neural Network (GRU/LSTM)	.	Ouhame et al. (2021) and Yazdanian and Sharifian (2021)	.	Feng et al. (2019)	Hundman et al. (2018), Su et al. (2019), Park et al. (2018), He et al. (2020), Girish and Rao (2021) and Chouliaras and Sotiriadis (2021)	Etemadi et al. (2021) and Tuli et al. (2021a)
	GAN/GNN/Transformers/Fuzzy-Logic	.	.	Tuli et al. (2022c) and Guan et al. (2021)	.	Zhao et al. (2020b), Deng and Hooi (2021), Li et al. (2019a), Feng et al. (2021), Won and Kim (2021) and Li et al. (2018), Hu et al. (2021)	Tuli et al. (2022c), Arabnejad et al. (2017), Naha et al. (2021), Aral and Brandić (2021), Aral and Brandic (2017) and Aral and Brandic (2018)

overlap across the three domains: deployment, scheduling and maintenance. For instance, risk-based and robust optimization has been seen in gradient pruning for DNN deployment (Sattler et al., 2019), task scheduling (Jawad et al., 2018; Panda et al., 2015) and load balancing (Marahatta et al., 2020). Similarly, neural design has been used for splitting DNNs for inference of resource-constrained Edge nodes (Kim et al., 2017) and memory-efficient anomaly detection (Tuli et al., 2021c). Regression models have been popular across all domains. All kinds of predictions are made using regression techniques, such as workload demand prediction for optimal resource provisioning (Zhu et al., 2016; Luo et al., 2020, 2021a), task QoS prediction (Jamshidi and Casale, 2016; Tuli et al., 2020a) and time-series reconstruction for anomaly detection (Zong et al., 2018). Similarly, meta-heuristic optimization strategies have frequently been in use in Fog research. For instance, PSO optimization has been used for decision optimization of VM provisioning in Cloud (Zhu et al., 2016; Chen et al., 2020b), workflow scheduling decision (Wang et al., 2020e) and load-balancing (Jan et al., 2021).

Reinforcement Learning. This category includes the various ways to solve MDP style problems using AI methods, such as tabular RL (Q and SARSA learning), deep Q learning and policy gradient methods (A3C, DDPG, etc.). Most state-of-the-art approaches do not utilize tabular RL due to its poor scalability of modeling real-life state–action spaces in physical Fog and Cloud systems with thousands of devices. Thus, researchers tend to rely on neural network-based approximation of the Q function, which estimates the long-term reward using a DNN. DQNs have been used to optimize the placement decisions of neural network-based tasks generated after DNN splitting (Tuli, 2022), QoS aware resource provisioning (Bitsakos et al., 2018), task and workflow scheduling (Tang et al., 2018; Li et al., 2019b; Wang et al., 2019b) and fault-tolerant scheduling (Wang et al., 2021c). However, in complex Fog scenarios, a richer action space might be required to ensure that RL agents do not get stuck in local optima. As tabular RL and DQNs have deterministic action policies, researchers have shifted to utilizing DNNs for stochastic action prediction, *i.e.*, policy gradient learning (PGL). These include REINFORCE, Actor–Critic and other forms of DNNs that predict action probabilities instead of Q values. For instance, some methods use PGL to decide the optimal placement of split neural models in a heterogeneous Edge-cloud setup (Yu et al., 2021). Other methods utilize PGL to optimize metrics such as energy (Tuli et al., 2020b) and cost (Cheng et al., 2021) by using them as reward signals.

Neural Optimization and Approximation. We consider another category of optimization that either uses DNNs to approximate optimization objects (unlike regression in classical AI) or gradient-optimization to generate optimal decisions (unlike ACO, PSO, etc. in classical AI). Recently, many methods use DNNs as QoS surrogates, to optimize provisioning decision (Wilder et al., 2019), scheduling decisions (Tuli et al., 2022e,a,b) or fault-tolerant scheduling (Talaat et al., 2019). Other methods utilize DNNs directly to take deployment, scheduling or maintenance decisions. For instance, FCNs (Levy et al., 2020; Chen et al., 2019), CNNs (Bega et al., 2019; Jeddi and Sharifian, 2019) and LSTMs (Ouhame et al., 2021; Yazdanian and Sharifian, 2021) are used for resource provisioning, FCN (Tuli et al., 2022e) and LSTM (Feng et al., 2019) are used for scheduling, and CNNs (Zhang et al., 2019, 2021a) and RNNs (Hundman et al., 2018; Su et al., 2019; Park et al., 2018; He et al., 2020; Girish and Rao, 2021; Chouliaras and Sotiriadis, 2021) are also used for reconstruction based fault detection. Fog maintenance related state-of-the-art methods also leverage other DNN types, including GNNs and GANs (Tuli et al., 2022c; Arabnejad et al., 2017; Naha et al., 2021; Aral and Brandić, 2021; Aral and Brandic, 2017, 2018).

6. Trends, challenges and future directions

Existing AI-driven resource management techniques cover a wide range of decision making problems. We now identify the key trends in the domain of AI based augmentation for resource management in the Fog continuum and elucidate them in Section 6.1. We also discuss in Section 6.2, the limitations of the current state-of-the-art works as per the classes identified in Section 5. Stemming from the identified limitations, we discuss emerging challenges in the field of AI-augmented Fog continuum systems and a series of open opportunities while briefly proposing new methods for future blue-sky research in application areas (Section 6.4) and AI methods (Section 6.5).

6.1. Trends

Shift to Deep Surrogate Models. Recently, there has been a shift from using regression models, such as linear regression, support vector regression, Gaussian process regression, to training a DNN. Regression models allow us to tune the parameters of a function using expectation maximization (Russell and Norvig, 2009). These models are typically used to generate an estimate of the system performance, usually a combination of QoS metrics, with respect to independent variables like resource management decision (Luo et al., 2021a, 2020). However, in practice, the data distributions that these models try to capture may have far more complex relationships with independent variables that such models can represent. To combat this, researchers now resort to DNNs as function approximators and surrogates of QoS metrics such as energy consumption, average response time and execution costs (Wilder et al., 2019; Tuli et al., 2022e, 2021b).

Shift to Co-Simulated Digital-Twins. AI models that rely on DNNs for function approximation, such as deep surrogate optimization, DQN, and policy gradient methods often face present issues characteristic of DNNs. For instance, when training a DNN to predict workload demands in a future state, it is often trained with historical trace data collected from a Fog system. However, when the model is applied in a setting with different workload traces, the model has poor demand prediction accuracy as it is never exposed to new data at training time. This problem is commonly referred to as the *exposure bias* problem in DNN training. To tackle this, recent methods now develop a co-simulated digital twin of the Fog system to solve three problems (Tuli et al., 2022e; Talaat et al., 2019). First, for data augmentation, *i.e.*, generate new traces by random perturbation of the environment or workload parameters to solve the exposure bias problem. Second, to solve the *data saturation* problem, *i.e.*, increasing the amount of data does not improve the model performance. Co-simulation allows us to run A/B tests to generate diverse scenarios to improve model performance. Third, co-simulations allow the generation of new datapoint for the latest system state, facilitating fine-tuning DNNs to adapt to non-stationary workload settings.

Shift to Transformers and Geometric Models. For series like data, researchers traditionally used recurrent models such as GRUs and LSTMs. However, training these models is time-consuming, giving rise to high training costs on public Cloud or local Edge nodes. The main bottleneck of such models is the requirement of providing sequential data one at a time (Vaswani et al., 2017; Tuli et al., 2022d). Recent models, like Transformers, use self-attention to infer on the complete sequence at once, allowing faster training and higher accuracy. Further, instead of FCN, CNNs and LSTMs, researchers are now resorting to composite AI neural networks that also model the system state as a geometric model, most often as a graph. These might be used to encode the network architecture (Deng and Hooi, 2021) or the input decision (Tuli et al., 2021b). GNNs over graph-like data allow capturing data correlations to be aware of the spatial structures.

Shift to Resource Efficient Management. Most AI-based resource management applications are heavy in terms of resource requirements. Thus, broker nodes are typically more powerful than a common Fog worker (Tuli et al., 2019b; Gill et al., 2019). As the number of devices in the worker layer of Fog architectures increases, the resource management AI models become more data and resource hungry. To scale and allow resilience in the broker layer of Fog systems, running resource management applications on worker-like resource-limited devices becomes inevitable. Thus, systems based AI research is now working to develop more pragmatic AI models that can be deployed in decentralized and constrained environments (Tuli et al., 2021c; Li et al., 2020b; Hou et al., 2021). Further, researchers are developing DNNs that have much lower training times than before, facilitating quick adaptability in volatile environments (Huang et al., 2020b). Another important trends is taking into account the energy consumption of DNNs while achieving high accuracy to have more sustainable machine learning models (Tuli et al., 2021b).

Shift to Unsupervised Models. Traditional methods mainly rely on manual labeling of important data characteristics, such as fault indication, through domain experts, which is infeasible in modern IoT solutions with enormous amounts of log data (Hu et al., 2021). Thus, for large-scale systems, researchers are now developing unsupervised and semi-supervised models that are as accurate as supervised models (Hu et al., 2021; Su et al., 2019). The advantage of unsupervised models is that we do not require labeled data, allowing us to scale resource management systems to systems with possibly millions of IoT devices and Fog nodes.

6.2. Limitations

Scalability. Most AI models suffer from the limitation of having poor scalability. Scalability in Fog systems refers to the ability to apply an AI model as the number of Fog nodes or workloads increases without a significant drop in system performance. As the number of IoT devices and users relying on Fog architectures increases, it makes it crucial to develop scalable AI models. This specifically requires developing on top of existing AI methods that are scalable. For instance, tabular reinforcement learning (Q/SARSA learning) saves the Q estimate of each state and action pair in an MDP formulation. On the contrary, DQN utilizes a DNN to capture the interdependence across states and actions to train a generic model that can provide accurate Q estimates without needing the same number of parameters as tabular RL methods. Similarly, other techniques such as geometric modeling (GNNs) and attention operation further improve scalability.

Generalizability. Generalizability is the ability of an AI model to perform successfully for unseen inputs. This depends on the stability, robustness and adaptability of the developed model (Xu and Mannor, 2012). Thus, limitations in the above areas result in limiting the generalizability of the model. To improve the generalizability of the traditional machine learning and deep learning algorithms within the context of distributed architectures such as Edge and Fog computing, federated learning is a viable option. It allows the models to be trained with a sufficient amount of data when the collection of data at a central location is not a possible option due to privacy, security or the sheer volume of data generated. However, conventional federated learning needs to be further improved to adapt to the challenges of network, computation and storage heterogeneity within Edge/Fog environments (Hosseinalipour et al., 2020).

Reliability. Reliability limitations of AI techniques can be analyzed under two main aspects: stability and robustness. The former indicates the ability of a model to yield consistent performance across similar yet diverse data inputs. The latter indicates the consistency of the output of the approach under new data (Xu and Mannor, 2012). IoT data-related issues such as missing data due to unreliable networks, limited access to sensitive data (for example, health data), noisy data and malicious data, cripple the stability and robustness of the machine learning and deep learning approaches. Susceptibility of machine learning models, especially deep learning models to adversarial examples, is critical within the context of latency and safety–critical IoT applications where accuracy is paramount (Qiu et al., 2021). Meta-heuristic algorithms also face limitations in stability and robustness. As meta-heuristics are designed to converge towards a near-optimal solution, stability and robustness limitations occur due to their tendency to converge to local optimum solutions, especially due to the dynamic changes in Fog environments.

Adaptability. Adaptability is the ability of the AI models to maintain accuracy when training and test data belong to different distributions. However, traditional machine learning and deep learning approaches operate under the assumption that both training and test data share the same distribution, which results in performance reduction in real-world deployments, for instance, in cases with exposure bias (Pan and Yang, 2009). Insufficient and biased data (*i.e.*, due to data privacy and security issues in smart healthcare, IIoT, etc.), outdated training data and inability to use large data sets in Fog environments due to resource limitations demand higher adaptability in such use cases (Sufian et al., 2020). To overcome this limitation, some machine learning and deep learning approaches leverage transfer learning, which is a learning framework that enables knowledge transfer between task domains (Pan and Yang, 2009). Some AI models utilize co-simulators to generate diverse datapoints and adapt to changing settings (Tuli et al., 2022e). However, fine-tuning the parameters of DNNs using co-simulators gives rise to high overheads.

Agility. Agility indicates the ability of a system to adapt and evolve rapidly with changing Fog environments. This becomes a prominent requirement in IoT applications, Fog environments that keep evolving rapidly require high agility not only within application development and deployment but also for algorithm development for resource provisioning, application scheduling and system maintenance. To keep up with this nature, AI models used within these contexts need to be able to undergo rapid updates as more data and data sources appear, more service requirements appear and the nature of the deployment environment and its technologies evolve (i.e., updates in communication technologies, availability of novel Edge/Fog computation resources and their hardware or architectural changes etc.) (Jackson et al., 2019). But, the data-centric nature of the lifecycle of an AI model makes the development, testing, deployment cycle highly experimental and repetitive, thus making agility a major limitation (Schleier-Smith, 2015).

6.3. Emerging challenges

Legacy Deployment. As research progresses and more accurate and better performing models are developed, it typically follows the adoption of advanced AI models by industry. However, more complex AI models are usually more resource-hungry and need more powerful systems to be deployed on. To deploy an enhanced AI model, technology based companies such as Meta, Amazon, Netflix and Google frequently upgrade their devices, raising many sustainability concerns (Gill et al., 2019). Stemming from the scalability limitations of state-of-the-art, the integration of large-scale DNN models within legacy edge or cloud machines has become a challenging problem. As research moves in the direction of the neural design of sophisticated AI models, it becomes crucial to ensure that these new models can be deployed on legacy infrastructures to bring down deployment costs and carbon footprint of AI.

Automated and Generic Modeling. Another challenge being faced by industrial AI adopters, related to the generalizability and adaptability of AI models, is the ability to tune AI models in settings different from the ones tested by researchers (Liaw et al., 2018). As the performance of AI models is highly dependent on the proper tuning of a large number of hyperparameters, these variables need to be re-tuned when deploying a pre-trained model in a new setting of scheduling or fault detection in Edge/Cloud. This problem stems from generalizability, but needs to be solved specifically for each application domain of deployment, scheduling or maintenance. In such cases, either the hyperparameter values of the models need to be decided in an automated fashion, or the neural design needs to be generic enough to accommodate new Fog settings, with possibly different number of nodes, workload characteristics and user demands.

Interpretability. Many state-of-the-art AI methods are being utilized today as black-box models that give rise to high QoS in Fog systems, but do not have any transparency on the process that led to the various resource management level decisions made by an AI agent. This mainly entails explaining the main reasons for choosing or not choosing certain management decisions and exploring the unknown state spaces to ensure exhaustive coverage of the decision space. For sensitive industrial segments, such as healthcare and autonomous vehicles, it is crucial to expose the underlying patterns and features in the decision-making process to gain credibility for the end-user. Building such white-box or explainable models is an emerging field of research for the use of trustworthy AI models. Many AI models, such as decision trees, regression algorithms and rule-based systems allow interpretability, but are not as accurate or scalable as deep-learning based counterparts.

6.4. Application areas

Healthcare. With the rapid increase in connected devices in hospitals, such as sensors, mobile phones and wearables, the amount of data generated and their rate of generation is snowballing. This results in a massive increase in the volume and variety of available health data, paving the way for the development of more reliable and robust AI models in the areas of proactive monitoring, disease prevention, and more in smart healthcare (Panesar, 2019). However, this results in challenges related to ensuring data quality and security, especially in the context of distributed EdgeAI, particularly in the case of handling sensitive healthcare information like patient records. To overcome these challenges, future research is focusing on the convergence of Blockchain and AI, where Blockchain is used in solving data quality and integration issues that enable AI to improve the accuracy of data analytics (Yaqoob et al., 2021). Moreover, low latency communication technologies like 5G/6G enable novel technologies like Augmented Reality, Virtual Reality and Tactile Internet, thus improving and expanding services such as robot-assisted surgery.

Next Generation Networking. Due to the high data rate, reliability, ultra-low latency, and ultra-low energy consumption provided by 5G/6G, these wireless communication technologies are identified as the key enablers of future IoT applications. However, to support the ever-evolving service requirements of the IoT services, 5G/6G technologies have to be able to observe environment variations and dynamically self organize the network accordingly (Li et al., 2017). This can be achieved through AI-empowered management and orchestration of cellular resources within 5G/6G networks. With the advancements in Software Defined Networking (SDN) and network function virtualization (NFV), 5G/6G technologies introduce Network Slicing (NS) to support this. NS is a mechanism for provisioning virtualized network resources intelligently based on the service performance metrics (Letaief et al., 2019). Learning algorithms such as deep learning and reinforcement learning can improve the dynamism of NS through prediction-based proactive resource allocation in Edge or Cloud architectures with dynamic slice creations for different applications (Wijethilaka and Liyanage, 2021). We also need to consider upcoming cases where the internet is provided through satellites (such as the Starlink network) in lieu of the conventional copper/fiber connections. In such cases, the latency and bandwidth characteristics might be significantly different, requiring re-tuning hyperparameters or adapting existing AI based resource management policies (Song et al., 2021; Wang et al., 2019d).

Production and Supply Chain. In the coming age of IIoT, most industrial pipelines are managed by smart devices. Such devices may utilize AI methods to self-monitor and predict potential problems in the supply chain to optimize the overall service efficiency. The COVID-19 pandemic is an example demonstrating the importance of automation in logistics to avoid service downtimes (Salehi-Amiri et al., 2021). AI based forecasting approaches, such as recurrent neural models and GANs can be used to predict stock shortage and proactively order additional stocks to prevent shortage (Salehi-Amiri et al., 2021). Similarly, large-scale ML models can aid the development of smart-manufacturing technologies that utilize several IoT and Fog devices to collaboratively monitor, control manufacturing and production related equipment.

Smart Cities. Smart cities aim to utilize IoT to deliver services that can enhance the living standards within cities. This includes a plethora of application domains such as smart governance, smart energy, smart transportation, and smart security (Nayak et al., 2021). Advancements in wireless communications such as 5G/6G enable a massive amount of data to be transmitted towards Edge nodes in real-time. This has resulted in the rise of novel technologies like crowd-sensing and crowd-sourcing (Kong et al., 2019). Distributed collections and processing of such massive volumes of data demands future research to focus on distributed and reliable AI, specifically in data-sensitive applications at the Edge. At the same time, data security becomes crucial in future smart city services, with the widespread use of crowd-sensing and crowd-sourcing for data collection. Moreover, ultra-low latency communication provided future radio access networks to support services related to hazard avoidance and safety (Rudd-Orthner and Mihaylova, 2020), which requires AI models with higher reliability, accuracy and lower latency.

6.5. Methods

Self-supervised AI. The self-supervised learning technique enables learning with unlabeled data by solving pretext tasks (Saeed et al., 2020). In contrast to this, supervised learning depends on the availability of labeled data. Even though a massive amount of data gets generated by the sensors, the lack of annotated data poses an obstacle for using supervised learning. This is specifically applicable in scenarios where fault-detection or workload scheduling is required for previously unseen workload or device characteristics in Edge or Cloud platforms. As generating expert-labeled fault labels or optimal scheduling decisions is infeasible for large-scale systems, self-supervised learning offers a possible solution for such scenarios. This approach is capable of generating a more generalizable model by removing the heavy dependency on labeled data by automatically generated data annotations, possibly using a co-simulator (Saeed et al., 2020). Moreover, self-supervised learning has the potential to achieve higher reliability due to its robustness to adversarial examples, label corruption, and input corruptions (Hendrycks et al., 2019).

Model Driven RL. As Fog environments and service demands are dynamic; algorithms should have the capability to adapt accordingly. IoT applications (*i.e.*, healthcare, smart cities, etc.) and their enabling telecommunication technologies (i.e., 5G, 6G) benefit from RL-based intelligence due to higher adaptability of RL techniques and their ability to learn without prior knowledge (Sami et al., 2021). However, exploration errors, long learning time and distributed learning within resource constraint devices in Edge environments are some of the challenges in utilizing RL techniques within resource-constrained and distributed Fog environments. A canonical case is the development of RL based methods for deploying large-scale application workflows on constrained Edge or Cloud clusters. Addressing these challenges in future research is vital for the RL-based approaches to reach their full potential within EdgeAI scenarios. In an attempt to address these challenges, EdgeAI research is exploring advanced RL approaches such as model-based RL (Sutton and Barto, 2018) and co-simulated RL (Amini et al., 2020).

Analog AI. The current implementation of AI is targeted for digital systems where the values are stored in a binary format. Herein, the major challenge posed by digital implementations of DNNs is the linear dependence of memory footprint with the number of parameters of the neural model. Upcoming analog memory-based chips present new ways to perform the same operations, but with orders of magnitude lower amount of memory requirement, computational load and energy consumed (Channamadhavuni et al., 2021). This is particularly useful in memory-constrained Edge and MEC devices where sophisticated DNNs need to be executed in *AI on Fog* setting or decentralized resource management is required in *AI for Fog* settings. However, there are some drawbacks in the loss of precision in computation across layers within a DNN. The tradeoff offered by such DNN implementations is similar to model pruning and splitting, but with possibly more extreme energy/compute benefits. This direction has been explored to a limited extent and requires further investigation and software development to efficiently harness the potential of analog hardware accelerators.

Decentralized Modeling. The success of distributed EdgeAI, by utilizing layer and semantic splitting strategies for AI deployment (discussed in Section 4), shows some promise in other domains of scheduling and maintenance. For scheduling applications and maintaining Fog systems at scale, it is possible to decentralize the training and inference procedures of the resource management level AI applications across multiple broker nodes. The decentralized fashion of resource management has a two-fold benefit. Firstly, there is no single point of failure in the system as the management steps are run in multiple broker nodes. Secondly, it allows the distribution of resource management load across multiple computing devices, facilitating the scalability of the model.

AI Driven Simulations. A major advantage of co-simulators in Fog systems is the ability to generate new data points for tuning AI models and resolving issues like exposure bias and data saturation (Renda et al., 2020). However, another important benefit of co-simulators is the ability to run multiple simulations concurrently and pick the best resource management decisions, allowing interpretable decision-making. This is due to the ability of co-simulators to generate a complete execution trace, possible for several future states of the systems and allowing developers or end-users to visualize the long-term effects of various decisions. They are able to do this much faster than executing the decisions on a physical infrastructure and waiting for several minutes to reflect changes in QoS. This is primarily due to the discrete event-driven execution style of modern simulators. This is applicable for all three types of resource management decisions, *i.e.*, deployment, scheduling and maintenance. Simulations can indicate changes in QoS scores for each model compression or splitting type, application placement or fault-remediation steps like preemptive migrations. These signals can facilitate decision making. However, co-simulators are mere approximations of the entire Fog system and typically fail to map the entire complexity of real infrastructure. However, the success of deep surrogate models hints us to build simulators with possibly millions of parameters and utilize DNNs to estimate optimal parameter values, such that our simulators resemble the real systems as closely as possible. An increased number of parameters could, in principle, give a higher representative capacity to our simulators, now being able to map complex real-life workloads and device characteristics. Thus, AI-driven simulators could help improve system performance with the added bonus of interpretable decision making.

AI Driven Co-Design. Currently, almost all resource management solutions for Fog systems solve only a specific problem from the three domains of deployment, scheduling or maintenance. However, for holistic performance enhancement, it is crucial to develop AI models that can concurrently take decisions across multiple facets of the management of Fog resources to efficiently exploit the synergy across these decision domains. Research in AI-based augmentation of Fog systems may benefit from other efforts in system co-design (Hao et al., 2021) to improve upon the existing management solutions. This is crucial in Fog systems particularly due to the constraints certain decision types impose on other resource management control knobs. For instance, provisioning decision constrains the devices on which incoming tasks can be scheduled on or the active tasks may be migrated to. There is a need to build end-to-end AI models that rely on multi-modal data and can take multiple decision types simultaneously for data privacy and improved system performance.

7. Conclusions

This work conducts an extensive literature review of the methods concerned with AI-based augmentation of Fog systems. We discuss diverse state-of-the-art techniques for Fog resource management, specifically for optimal AI deployment, workload scheduling and system maintenance. We consider two kinds of AI models: AI *on* and AI *for* Fog computing. We use taxonomy of AI methods and classify them broadly into classical methods, machine learning, reinforcement learning and deep learning. There is significant overlap across different decision domains in terms of the used AI models. This overlap suggests the importance of certain design decisions over others and hints at the possible gaps of current research. We have highlighted the importance of a more comprehensive research style that not only considers specific aspects of resource management but distills historical knowledge gathered from the myriad of AI-based decision-making methods to develop well informed AI models and eclectic management solutions. The various advances in the field of computing need to be considered in tandem to bolster AI research and build holistic AI-based methods for emerging application areas, future technologies and next-generation users.

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.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work is supported by the President’s Ph.D. Scholarship at Imperial College London, United Kingdom and Australian Research Council Discovery Project. We thank Shikhar Tuli, Zifeng Niu, Runan Wang, Matthew Sheldon and William Plumb for helpful discussions.

References

Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K., Zhang, L., 2016. Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. pp. 308–318.

Abdullah, M., Iqbal, W., Mahmood, A., Bukhari, F., Erradi, A., 2020. Predictive autoscaling of microservices hosted in fog microdata center. IEEE Syst. J. 15 (1), 1275–1286.

Abioye, S.O., Oyedele, L.O., Akanbi, L., Ajayi, A., Delgado, J.M.D., Bilal, M., Akinade, O.O., Ahmed, A., 2021. Artificial intelligence in the construction industry: A review of present status, opportunities and future challenges. J. Build. Eng. 44, 103299.

Adhikari, M., Amgoth, T., Srirama, S.N., 2019. A survey on scheduling strategies for workflows in cloud environment and emerging trends. ACM Comput. Surv. 52 (4), 1–36.

Alelaiwi, A., 2019. An efficient method of computation offloading in an edge cloud platform. J. Parallel Distrib. Comput. 127, 58–64.

Aliyu, M., Murali, M., Gital, A.Y., Boukari, S., 2020. Efficient metaheuristic population-based and deterministic algorithm for resource provisioning using ant colony optimization and spanning tree. Int. J. Cloud Appl. Comput. (IJCAC) 10 (2), 1–21.

Alsurdeh, R., Calheiros, R.N., Matawie, K.M., Javadi, B., 2021. Hybrid workflow scheduling on edge cloud computing systems. IEEE Access 9, 134783–134799.

Amini, A., Gilitschenski, I., Phillips, J., Moseyko, J., Banerjee, R., Karaman, S., Rus, D., 2020. Learning robust control policies for end-to-end autonomous driving from data-driven simulation. IEEE Robot. Autom. Lett. 5 (2), 1143–1150.

An, J., Li, W., Le Gall, F., Kovac, E., Kim, J., Taleb, T., Song, J., 2019. EiF: Toward an elastic IoT fog framework for AI services. IEEE Commun. Mag. 57 (5), 28–33.

Arabnejad, H., Pahl, C., Estrada, G., Samir, A., Fowley, F., 2017. A fuzzy load balancer for adaptive fault tolerance management in cloud platforms. In: European Conference on Service-Oriented and Cloud Computing. Springer, pp. 109–124.

Aral, A., Brandic, I., 2017. Quality of service channelling for latency sensitive edge applications. In: 2017 IEEE International Conference on Edge Computing. EDGE, IEEE, pp. 166–173.

Aral, A., Brandic, I., 2018. Dependency mining for service resilience at the edge. In: 2018 IEEE/ACM Symposium on Edge Computing. SEC, pp. 228–242. http://dx.doi.org/10.1109/SEC.2018.00024.

Aral, A., Brandić, I., 2021. Learning spatiotemporal failure dependencies for resilient edge computing services. IEEE Trans. Parallel Distrib. Syst. 32 (7), 1578–1590. http://dx.doi.org/10.1109/TPDS.2020.3046188.

Asghari, A., Sohrabi, M.K., Yaghmaee, F., 2021. Task scheduling, resource provisioning, and load balancing on scientific workflows using parallel SARSA reinforcement learning agents and genetic algorithm. J. Supercomput. 77 (3), 2800–2828.

Audibert, J., Michiardi, P., Guyard, F., Marti, S., Zuluaga, M.A., 2020. USAD: Unsupervised anomaly detection on multivariate time series. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 3395–3404.

Auer, P., Cesa-Bianchi, N., Fischer, P., 2002. Finite-time analysis of the multiarmed bandit problem. Mach. Learn. 47 (2–3), 235–256.

2021. IoT device simulator | Implementations | AWS solutions. https://aws.amazon.com/solutions/implementations/iot-device-simulator/, (Accessed on 12/25/2021).

Bae, J., Lee, J., Chong, S., 2019. Beyond max-weight scheduling: A reinforcement learning-based approach. In: 2019 International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT). IEEE, pp. 1–8.

Baek, J., Kaddoum, G., 2020. Heterogeneous task offloading and resource allocations via deep recurrent reinforcement learning in partial observable multifog networks. IEEE Internet Things J. 8 (2), 1041–1056.

Bagchi, S., Siddiqui, M.-B., Wood, P., Zhang, H., 2019. Dependability in edge computing. Commun. ACM 63 (1), 58–66.

Basu, D., Wang, X., Hong, Y., Chen, H., Bressan, S., 2019. Learn-as-you-go with Megh: Efficient live migration of virtual machines. IEEE Trans. Parallel Distrib. Syst. 30 (8), 1786–1801.

Bega, D., Gramaglia, M., Fiore, M., Banchs, A., Costa-Perez, X., 2019. DeepCog: Optimizing resource provisioning in network slicing with AI-based capacity forecasting. IEEE J. Sel. Areas Commun. 38 (2), 361–376.

Belcastro, L., Falcone, A., Garro, A., Marozzo, F., 2021. Evaluation of large scale RoI mining applications in edge computing environments. In: 2021 IEEE/ACM 25th International Symposium on Distributed Simulation and Real Time Applications (DS-RT). IEEE, pp. 1–8.

2021. System data | Capital bikeshare. https://www.capitalbikeshare.com/system-data, (Accessed on 12/26/2021).

Bitsakos, C., Konstantinou, I., Koziris, N., 2018. Derp: A deep reinforcement learning cloud system for elastic resource provisioning. In: 2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom). IEEE, pp. 21–29.

Bittencourt, L., Immich, R., Sakellariou, R., Fonseca, N., Madeira, E., Curado, M., Villas, L., DaSilva, L., Lee, C., Rana, O., 2018. The internet of things, fog and cloud continuum: Integration and challenges. Internet Things 3, 134–155.

Bogolubsky, L., Dvurechenskii, P., Gasnikov, A., Gusev, G., Nesterov, Y., Raigorodskii, A.M., Tikhonov, A., Zhukovskii, M., 2016. Learning supervised pagerank with gradient-based and gradient-free optimization methods. In: Advances in Neural Information Processing Systems. pp. 4914–4922.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A., Seth, K., 2017. Practical secure aggregation for privacy-preserving machine learning. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 1175–1191.

Bond-Taylor, S., Willcocks, C.G., 2021. Gradient origin networks. In: International Conference on Learning Representations.

Bonomi, F., Milito, R., Zhu, J., Addepalli, S., 2012. Fog computing and its role in the internet of things. In: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing. pp. 13–16.

Bosmans, S., Mercelis, S., Denil, J., Hellinckx, P., 2019. Testing IoT systems using a hybrid simulation based testing approach. Computing 101 (7), 857–872.

Boulos, M.N.K., Yassine, A., Shirmohammadi, S., Namahoot, C.S., Brückner, M., 2015. Towards an "Internet of Food": food ontologies for the internet of things. Future Internet 7 (4), 372–392.

Brogi, A., Forti, S., 2017. Qos-aware deployment of IoT applications through the fog. IEEE Internet Things J. 4 (5), 1185–1192.

Bui, D.-M., Yoon, Y., Huh, E.-N., Jun, S., Lee, S., 2017. Energy efficiency for cloud computing system based on predictive optimization. J. Parallel Distrib. Comput. 102, 103–114.

Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I., 2009. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Gener. Comput. Syst. 25 (6), 599–616.

Calheiros, R.N., Masoumi, E., Ranjan, R., Buyya, R., 2014. Workload prediction using ARIMA model and its impact on cloud applications' QoS. IEEE Trans. Cloud Comput. 3 (4), 449–458.

Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A., Buyya, R., 2011. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw. - Pract. Exp. 41 (1), 23–50.

Callegaro, D., Matsubara, Y., Levorato, M., 2020. Optimal task allocation for time-varying edge computing systems with split DNNs. In: GLOBECOM 2020-2020 IEEE Global Communications Conference. IEEE, pp. 1–6.

Carvalho, G., Cabral, B., Pereira, V., Bernardino, J., 2021. Edge computing: current trends, research challenges and future directions. Computing 103 (5), 993–1023.

Castro, P., Ishakian, V., Muthusamy, V., Slominski, A., 2019. The rise of serverless computing. Commun. ACM 62 (12), 44–54.

Celik, Z.B., Babun, L., Sikder, A.K., Aksu, H., Tan, G., McDaniel, P., Uluagac, A.S., 2018. Sensitive information tracking in commodity IoT. In: USENIX Security Symposium. Baltimore, MD.

Channamadhavuni, S., Thijssen, S., Jha, S.K., Ewetz, R., 2021. Accelerating AI applications using analog in-memory computing: Challenges and opportunities. In: Great Lakes Symposium on VLSI. pp. 379–384.

Chen, J., Chen, S., Wang, Q., Cao, B., Feng, G., Hu, J., 2019. iRAF: A deep reinforcement learning approach for collaborative mobile edge computing IoT networks. IEEE Internet Things J. 6 (4), 7011–7024.

Chen, Y., Liu, Z., Zhang, Y., Wu, Y., Chen, X., Zhao, L., 2020a. Deep reinforcement learning-based dynamic resource management for mobile edge computing in industrial internet of things. IEEE Trans. Ind. Inform. 17 (7), 4925–4934.

Chen, X., Shi, Q., Yang, L., Xu, J., 2018. ThriftyEdge: Resource-efficient edge computing for intelligent IoT applications. IEEE Netw. 32 (1), 61–65.

Chen, J., Wang, Y., 2018. A resource demand prediction method based on EEMD in cloud computing. Procedia Comput. Sci. 131, 116–123.

Chen, X., Wang, H., Ma, Y., Zheng, X., Guo, L., 2020b. Self-adaptive resource allocation for cloud-based software services based on iterative QoS prediction model. Future Gener. Comput. Syst. 105, 287–296.

Chen, M., Wang, T., Zhang, S., Liu, A., 2021. Deep reinforcement learning for computation offloading in mobile edge computing environment. Comput. Commun. 175, 1–12.

Cheng, Z., Min, M., Liwang, M., Huang, L., Gao, Z., 2021. Multi-agent DDPG-based joint task partitioning and power control in fog computing networks. IEEE Internet Things J..

Chouliaras, S., Sotiriadis, S., 2021. Detecting performance degradation in cloud systems using LSTM autoencoders. In: International Conference on Advanced Information Networking and Applications. Springer, pp. 472–481.

Cicconetti, C., Conti, M., Passarella, A., 2020. A decentralized framework for serverless edge computing in the internet of things. IEEE Trans. Netw. Serv. Manag. 18 (2), 2166–2180.

Coelho, C.N., Kuusela, A., Li, S., Zhuang, H., Ngadiuba, J., Aarrestad, T.K., Loncar, V., Pierini, M., Pol, A.A., Summers, S., 2021. Automatic heterogeneous quantization of deep neural networks for low-latency inference on the edge for particle detectors. Nat. Mach. Intell. 1–12.

Coleman, T., Casanova, H., Pottier, L., Kaushik, M., Deelman, E., da Silva, R.F., 2021. WfCommons: A framework for enabling scientific workflow research and development. arXiv preprint arXiv:2105.14352.

Cortez, E., Bonde, A., Muzio, A., Russinovich, M., Fontoura, M., Bianchini, R., 2017. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In: SOSP. pp. 153–167.

Costello, K., 2019. Gartner survey shows 37 percent of organizations have implemented AI in some form.

Das, A., Patterson, S., Wittie, M., 2018. Edgebench: Benchmarking edge computing platforms. In: 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion). IEEE, pp. 175–180.

Deng, A., Hooi, B., 2021. Graph neural network-based anomaly detection in multivariate time series. In: AAAI.

Deng, S., Xiang, Z., Taheri, J., Khoshkholghi, M.A., Yin, J., Zomaya, A.Y., Dustdar, S., 2020a. Optimal application deployment in resource constrained distributed edges. IEEE Trans. Mob. Comput. 20 (5), 1907–1923.

Deng, S., Zhao, H., Fang, W., Yin, J., Dustdar, S., Zomaya, A.Y., 2020b. Edge intelligence: The confluence of edge computing and artificial intelligence. IEEE Internet Things J. 7 (8), 7457–7469.

Du, W., Zhang, X., He, Q., Liu, W., Cui, G., Chen, F., Ji, Y., Cai, C., Yang, Y., 2020. Fault-tolerating edge computing with server redundancy based on a variant of group degree centrality. In: International Conference on Service-Oriented Computing. Springer, pp. 198–214.

Duc, T.L., Leiva, R.G., Casari, P., Östberg, P.-O., 2019. Machine learning methods for reliable resource provisioning in edge-cloud computing: A survey. ACM Comput. Surv. 52 (5), 1–39.

Durbin, J., Koopman, S.J., 2012. Time Series Analysis by State Space Methods. Oxford University Press.

Ebadifard, F., Babamir, S.M., 2021. Autonomic task scheduling algorithm for dynamic workloads through a load balancing technique for the cloud-computing environment. Cluster Comput. 24 (2), 1075–1101.

Engelmann, C., Vallee, G.R., Naughton, T., Scott, S.L., 2009. Proactive fault tolerance using preemptive migration. In: 2009 17th Euromicro International Conference on Parallel, Distributed and Network-Based Processing. IEEE, pp. 252–257.

Ergun, K., Yu, X., Nagesh, N., Cherkasova, L., Mercati, P., Ayoub, R., Rosing, T., 2020. RelIoT: Reliability simulator for IoT networks. In: International Conference on Internet of Things. Springer, pp. 63–81.

Etemadi, M., Ghobaei-Arani, M., Shahidinejad, A., 2020. A learning-based resource provisioning approach in the fog computing environment. J. Exp. Theor. Artif. Intell. 1–24.

Etemadi, M., Ghobaei-Arani, M., Shahidinejad, A., 2021. A cost-efficient auto-scaling mechanism for IoT applications in fog computing environment: a deep learning-based approach. Cluster Comput. 1–16.

Feng, H., Jiang, Y., Niyato, D., Zheng, F.-C., You, X., 2019. Content popularity prediction via deep learning in cache-enabled fog radio access networks. In: 2019 IEEE Global Communications Conference. GLOBECOM, IEEE, pp. 1–6.

Feng, Y., Liu, Z., Chen, J., Lv, H., Wang, J., Yuan, J., 2021. Make the rocket intelligent at IoT edge: Stepwise GAN for anomaly detection of LRE with multi-source fusion. IEEE Internet Things J..

Fox, G., Glazier, J., Kadupitiya, J., Jadhao, V., Kim, M., Qiu, J., Sluka, J.P., Somogy, E., Marathe, M., Adiga, A., et al., 2019. Learning everywhere: Pervasive machine learning for effective high-performance computation. In: 2019 IEEE International Parallel and Distributed Processing Symposium Workshops. IPDPSW, IEEE, pp. 422–429.

Gan, Y., Dev, S., Lo, D., Delimitrou, C., 2020. Sage: Leveraging ML to diagnose unpredictable performance in cloud microservices. ML Comput. Archit. Syst..

Gao, W., Luo, C., Wang, L., Xiong, X., Chen, J., Hao, T., Jiang, Z., Fan, F., Du, M., Huang, Y., et al., 2018. AIBench: towards scalable and comprehensive datacenter AI benchmarking. In: International Symposium on Benchmarking, Measuring and Optimization. Springer, pp. 3–9.

Gazori, P., Rahbari, D., Nickray, M., 2019. Saving time and cost on the scheduling of fog-based IoT applications using deep reinforcement learning approach. Future Gener. Comput. Syst..

Ge, L., Parhi, K.K., 2020. Classification using hyperdimensional computing: A review. IEEE Circuits Syst. Mag. 20 (2), 30–47.

Ghanavati, S., Abawajy, J.H., Izadi, D., 2020. An energy aware task scheduling model using ant-mating optimization in fog computing environment. IEEE Trans. Serv. Comput..

Ghosal, G.R., Ghosal, D., Sim, A., Thakur, A.V., Wu, K., 2020. A deep deterministic policy gradient based network scheduler for deadline-driven data transfers. In: 2020 IFIP Networking Conference (Networking). IEEE, pp. 253–261.

Gill, S.S., Tuli, S., Xu, M., Singh, I., Singh, K.V., Lindsay, D., Tuli, S., Smirnova, D., Singh, M., Jain, U., et al., 2019. Transformative effects of IoT, Blockchain and Artificial Intelligence on cloud computing: Evolution, vision, trends and open challenges. Internet Things 8, 100–118.

Girish, L., Rao, S.K., 2021. Anomaly detection in cloud environment using artificial intelligence techniques. Computing 1–14.

Goli, A., Hajihassani, O., Khazaei, H., Ardakanian, O., Rashidi, M., Dauphinee, T., 2020. Migrating from monolithic to serverless: A fintech case study. In: Companion of the ACM/SPEC International Conference on Performance Engineering. pp. 20–25.

Gong, C., Jiang, Z., Wang, D., Lin, Y., Liu, Q., Pan, D.Z., 2019. Mixed precision neural architecture search for energy efficient deep learning. In: 2019 IEEE/ACM International Conference on Computer-Aided Design. ICCAD, IEEE, pp. 1–7.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

Guan, J., Tang, H., Wang, J., Yao, J., Wang, K., Mao, W., 2021. A GAN-based fully model-free learning method for short-term scheduling of large power system. IEEE Trans. Power Syst..

Gunasekaran, J.R., Mishra, C.S., Thinakaran, P., Kandemir, M.T., Das, C.R., 2020. Implications of public cloud resource heterogeneity for inference serving. In: Proceedings of the 2020 Sixth International Workshop on Serverless Computing. pp. 7–12.

Gupta, H., Vahid Dastjerdi, A., Ghosh, S.K., Buyya, R., 2017. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. Softw. - Pract. Exp. 47 (9), 1275–1296.

Hamer, J., Mohri, M., Suresh, A.T., 2020. Fedboost: A communication-efficient algorithm for federated learning. In: International Conference on Machine Learning. PMLR, pp. 3973–3983.

Han, K., Xie, Z., LV, X., 2018. Fog computing task scheduling strategy based on improved genetic algorithm. Comput. Sci. 4, 22.

Hao, C., Dotzel, J., Xiong, J., Benini, L., Zhang, Z., Chen, D., 2021. Enabling design methodologies and future trends for edge AI: Specialization and co-design. IEEE Des. Test.

Hao, T., Huang, Y., Wen, X., Gao, W., Zhang, F., Zheng, C., Wang, L., Ye, H., Hwang, K., Ren, Z., et al., 2018. Edge AIBench: towards comprehensive end-to-end edge computing benchmarking. In: International Symposium on Benchmarking, Measuring and Optimization. Springer, pp. 23–30.

Hasan, M., Goraya, M.S., 2018. Fault tolerance in cloud computing environment: A systematic survey. Comput. Ind. 99, 156–172.

2021. [HDFS-448] An IOException thrown in the BlockReceiver file causes some tests to hang - ASF JIRA. https://issues.apache.org/jira/browse/HDFS-448, (Accessed on 12/28/2021).

He, Z., Chen, P., Li, X., Wang, Y., Yu, G., Chen, C., Li, X., Zheng, Z., 2020. A spatiotemporal deep learning approach for unsupervised anomaly detection in cloud systems. IEEE Trans. Neural Netw. Learn. Syst..

Hendrickson, S., Sturdevant, S., Harter, T., Venkataramani, V., Arpaci-Dusseau, A.C., Arpaci-Dusseau, R.H., 2016. Serverless computation with openlambda. In: 8th {USENIX} Workshop on Hot Topics in Cloud Computing (HotCloud 16).

Hendrycks, D., Mazeika, M., Kadavath, S., Song, D., 2019. Using Self-Supervised Learning Can Improve Model Robustness and Uncertainty. Curran Associates Inc., Red Hook, NY, USA.

Hosseinalipour, S., Brinton, C.G., Aggarwal, V., Dai, H., Chiang, M., 2020. From federated to fog learning: Distributed machine learning over heterogeneous wireless networks. IEEE Commun. Mag. 58 (12), 41–47.

Hou, L., Yuan, Z., Huang, L., Shen, H., Cheng, X., Wang, C., 2021. Slimmable generative adversarial networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35. pp. 7746–7753.

Hu, P., Dhelim, S., Ning, H., Qiu, T., 2017. Survey on fog computing: architecture, key technologies, applications and open issues. J. Netw. Comput. Appl. 98, 27–42.

Hu, Y., de Laat, C., Zhao, Z., 2019. Learning workflow scheduling on multi-resource clusters. In: 2019 IEEE International Conference on Networking, Architecture and Storage. NAS, IEEE, pp. 1–8.

Hu, X., Li, Y., Jia, L., Qiu, M., 2021. A novel two-stage unsupervised fault recognition framework combining feature extraction and fuzzy clustering for collaborative AIoT. IEEE Trans. Ind. Inform..

Huang, T., Lin, W., Xiong, C., Pan, R., Huang, J., 2020a. An ant colony optimization-based multiobjective service replicas placement strategy for fog computing. IEEE Trans. Cybern..

Huang, S., Liu, Y., Fung, C., He, R., Zhao, Y., Yang, H., Luan, Z., 2020b. HitAnomaly: Hierarchical transformers for anomaly detection in system log. IEEE Trans. Netw. Serv. Manag. 17 (4), 2064–2076.

Huang, Y., Qiao, X., Tang, J., Ren, P., Liu, L., Pu, C., Chen, J., 2020c. DeepAdapter: A Collaborative Deep Learning Framework for the Mobile Web Using Context-Aware Network Pruning. In: Proceedings - IEEE INFOCOM, Vol. 2020-July. Institute of Electrical and Electronics Engineers Inc., pp. 834–843. http://dx.doi.org/10.1109/INFOCOM41043.2020.9155379.

Huang, J., Samplawski, C., Ganesan, D., Marlin, B., Kwon, H., 2020d. CLIO: Enabling automatic compilation of deep learning pipelines across IoT and Cloud. In: Proceedings of the 26th Annual International Conference on Mobile Computing and Networking. pp. 1–12.

Hundman, K., Constantinou, V., Laporte, C., Colwell, I., Soderstrom, T., 2018. Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 387–395.

Hyndman, R.J., Athanasopoulos, G., 2018. Forecasting: Principles and Practice. OTexts.

Imani, M., Gupta, S., Kim, Y., Rosing, T., 2019. Floatpim: In-memory acceleration of deep neural network training with high precision. In: 2019 ACM/IEEE 46th Annual International Symposium on Computer Architecture. ISCA, IEEE, pp. 802–815.

Ismayilov, G., Topcuoglu, H.R., 2020. Neural network based multi-objective evolutionary algorithm for dynamic workflow scheduling in cloud computing. Future Gener. Comput. Syst. 102, 307–322.

Jackson, S., Yaqub, M., Li, C.-X., 2019. The agile deployment of machine learning models in healthcare. Front. Big Data 1, 7.

Jain, S., Venkataramani, S., Srinivasan, V., Choi, J., Chuang, P., Chang, L., 2018. Compensated-DNN: Energy efficient low-precision deep neural networks by compensating quantization errors. In: 2018 55th ACM/ESDA/IEEE Design Automation Conference. DAC, IEEE, pp. 1–6.

Jalali, F., Lynar, T., Smith, O.J., Kolluri, R.R., Hardgrove, C.V., Waywood, N., Suits, F., 2019. Dynamic edge fabric environment: Seamless and automatic switching among resources at the edge of IoT network and cloud. In: 2019 IEEE International Conference on Edge Computing. EDGE, IEEE, pp. 77–86.

Jalali, F., Smith, O.J., Lynar, T., Suits, F., 2017. Cognitive IoT gateways: automatic task sharing and switching between cloud and edge/fog computing. In: Proceedings of the SIGCOMM Posters and Demos. pp. 121–123.

Jamshidi, P., Casale, G., 2016. An uncertainty-aware approach to optimal configuration of stream processing systems. In: 2016 IEEE 24th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems. MASCOTS, IEEE, pp. 39–48.

Jan, M.A., Zakarya, M., Khan, M., Mastorakis, S., Menon, V.G., Balasubramanian, V., Rehman, A.U., 2021. An AI-enabled lightweight data fusion and load optimization approach for Internet of Things. Future Gener. Comput. Syst. 122, 40–51.

Javadi, B., Kondo, D., Vincent, J.-M., Anderson, D.P., 2009. Mining for statistical models of availability in large-scale distributed systems: An empirical study of seti@ home. In: 2009 IEEE International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems. IEEE, pp. 1–10.

Javadi, B., Sun, J., Ranjan, R., 2020. Serverless architecture for edge computing. In: Edge Computing: Models, Technologies and Applications. pp. 249–264.

Javaid, H., Saleem, S., Wajid, B., Khan, U.G., 2021. Diagnose a disease: A fog assisted disease diagnosis framework with bidirectional LSTM. In: 2021 International Conference on Digital Futures and Transformative Technologies (ICoDT2). IEEE, pp. 1–6.

Jawad, M., Qureshi, M.B., Khan, U., Ali, S.M., Mehmood, A., Khan, B., Wang, X., Khan, S.U., 2018. A robust optimization technique for energy cost minimization of cloud data centers. IEEE Trans. Cloud Comput..

Jeddi, S., Sharifian, S., 2019. A water cycle optimized wavelet neural network algorithm for demand prediction in cloud computing. Cluster Comput. 22 (4), 1397–1412.

Jha, D.N., Alwasel, K., Alshoshan, A., Huang, X., Naha, R.K., Battula, S.K., Garg, S., Puthal, D., James, P., Zomaya, A., et al., 2020. IoTSim-Edge: A simulation framework for modeling the behavior of Internet of Things and edge computing environments. Softw. - Pract. Exp. 50 (6), 844–867.

Jiang, F., Wang, K., Dong, L., Pan, C., Xu, W., Yang, K., 2019a. Deep-learning-based joint resource scheduling algorithms for hybrid MEC networks. IEEE Internet Things J. 7 (7), 6252–6265.

Jiang, Y., Wang, S., Valls, V., Ko, B.J., Lee, W.-H., Leung, K.K., Tassiulas, L., 2019b. Model pruning enables efficient federated learning on edge devices. arXiv preprint arXiv:1909.12326.

Kadota, I., Sinha, A., Uysal-Biyikoglu, E., Singh, R., Modiano, E., 2018. Scheduling policies for minimizing age of information in broadcast wireless networks. IEEE/ACM Trans. Netw. 26 (6), 2637–2650.

Kang, Y., Hauswald, J., Gao, C., Rovinski, A., Mudge, T., Mars, J., Tang, L., 2017. Neurosurgeon: Collaborative intelligence between the cloud and mobile edge. ACM SIGARCH Comput. Archit. News 45 (1), 615–629.

Kaplunovich, A., Yesha, Y., 2020. Automatic tuning of hyperparameters for neural networks in serverless cloud. In: 2020 IEEE International Conference on Big Data (Big Data). IEEE, pp. 2751–2756.

Kaur, M., Aron, R., 2021. FOCALB: Fog computing architecture of load balancing for scientific workflow applications. J. Grid Comput. 19 (4), 1–22.

Kaur, A., Singh, P., Singh Batth, R., Peng Lim, C., 2020. Deep-Q learning-based heterogeneous earliest finish time scheduling algorithm for scientific workflows in cloud. Softw. - Pract. Exp..

Khanna, A., Sah, A., Choudhury, T., 2020. Intelligent mobile edge computing: A deep learning based approach. In: International Conference on Advances in Computing and Data Sciences. Springer, pp. 107–116.

Kim, J., Park, Y., Kim, G., Hwang, S.J., 2017. SplitNet: Learning to semantically split deep networks for parameter reduction and model parallelization. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, pp. 1866–1874.

Kong, X., Liu, X., Jedari, B., Li, M., Wan, L., Xia, F., 2019. Mobile crowdsourcing in smart cities: Technologies, applications, and future challenges. IEEE Internet Things J. 6 (5), 8095–8113.

Kraemer, F.A., Braten, A.E., Tamkittikhun, N., Palma, D., 2017. Fog computing in healthcare–a review and discussion. IEEE Access 5, 9206–9222.

Krishnasamy, S., Akhil, P., Arapostathis, A., Sundaresan, R., Shakkottai, S., 2018. Augmenting max-weight with explicit learning for wireless scheduling with switching costs. IEEE/ACM Trans. Netw. 26 (6), 2501–2514.

Kristiani, E., Yang, C.-T., Wang, Y.T., Huang, C.-Y., 2018. Implementation of an edge computing architecture using openstack and kubernetes. In: International Conference on Information Science and Applications. Springer, pp. 675–685.

Kumari, A., Tanwar, S., Tyagi, S., Kumar, N., 2018. Fog computing for Healthcare 4.0 environment: Opportunities and challenges. Comput. Electr. Eng. 72, 1–13.

Lane, N.D., Bhattacharya, S., Georgiev, P., Forlivesi, C., Jiao, L., Qendro, L., Kawsar, F., 2016. Deepx: A software accelerator for low-power deep learning inference on mobile devices. In: 2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks. IPSN, IEEE, pp. 1–12.

Langroudi, H.F., Carmichael, Z., Kudithipudi, D., 2019a. Deep learning training on the edge with low-precision posits. arXiv preprint arXiv:1907.13216.

Langroudi, H.F., Carmichael, Z., Pastuch, D., Kudithipudi, D., 2019b. Cheetah: Mixed low-precision hardware & software co-design framework for dnns on the edge. arXiv preprint arXiv:1908.02386.

Laskaridis, S., Venieris, S.I., Almeida, M., Leontiadis, I., Lane, N.D., 2020. SPINN: synergistic progressive inference of neural networks over device and cloud. In: Proceedings of the 26th Annual International Conference on Mobile Computing and Networking. pp. 1–15.

Lera, I., Guerrero, C., Juiz, C., 2019. YAFS: A simulator for IoT scenarios in fog computing. IEEE Access 7, 91745–91758.

Letaief, K.B., Chen, W., Shi, Y., Zhang, J., Zhang, Y.-J.A., 2019. The roadmap to 6G: AI empowered wireless networks. IEEE Commun. Mag. 57 (8), 84–90.

Levy, S., Yao, R., Wu, Y., Dang, Y., Huang, P., Mu, Z., Zhao, P., Ramani, T., Govindaraju, N., Li, X., et al., 2020. Narya: Predictive and adaptive failure mitigation to avert production cloud vm interruptions. In: Symposium on Operating Systems Design and Implementation (OSDI'20).

Li, C., Bai, J., Ge, Y., Luo, Y., 2020a. Heterogeneity-aware elastic provisioning in cloud-assisted edge computing systems. Future Gener. Comput. Syst. 112, 1106–1121.

Li, C., Cerrada, M., Cabrera, D., Sanchez, R.V., Pacheco, F., Ulutagay, G., Valente de Oliveira, J., 2018. A comparison of fuzzy clustering algorithms for bearing fault diagnosis. J. Intell. Fuzzy Systems 34 (6), 3565–3580.

Li, D., Chen, D., Jin, B., Shi, L., Goh, J., Ng, S.-K., 2019a. MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks. In: International Conference on Artificial Neural Networks. Springer, pp. 703–716.

Li, H., Ota, K., Dong, M., 2019b. Deep reinforcement scheduling for mobile crowdsensing in fog computing. ACM Trans. Internet Technol. (TOIT) 19 (2), 1–18.

Li, T., Sahu, A.K., Talwalkar, A., Smith, V., 2020b. Federated learning: Challenges, methods, and future directions. IEEE Signal Process. Mag. 37 (3), 50–60.

Li, E., Zeng, L., Zhou, Z., Chen, X., 2019c. Edge AI: On-demand accelerating deep neural network inference via edge computing. IEEE Trans. Wireless Commun. 19 (1), 447–457.

Li, R., Zhao, Z., Zhou, X., Ding, G., Chen, Y., Wang, Z., Zhang, H., 2017. Intelligent 5G: When cellular networks meet artificial intelligence. IEEE Wirel. Commun. 24 (5), 175–183.

Li, G., Zhou, X., Sun, J., Yu, X., Han, Y., Jin, L., Li, W., Wang, T., Li, S., 2021. Opengauss: An autonomous database system. Proc. VLDB Endow. 14 (12), 3028–3042.

Liang, Q., Shenoy, P., Irwin, D., 2020. AI on the edge: Characterizing AI-based IoT applications using specialized edge architectures. In: 2020 IEEE International Symposium on Workload Characterization. IISWC, IEEE, pp. 145–156.

Liaw, R., Liang, E., Nishihara, R., Moritz, P., Gonzalez, J.E., Stoica, I., 2018. Tune: A research platform for distributed model selection and training. arXiv preprint arXiv:1807.05118.

Lim, W.Y.B., Luong, N.C., Hoang, D.T., Jiao, Y., Liang, Y.-C., Yang, Q., Niyato, D., Miao, C., 2020. Federated learning in mobile edge networks: A comprehensive survey. IEEE Commun. Surv. Tutor. 22 (3), 2031–2063.

Liu, K., Gurudutt, A., Kamaal, T., Divakara, C., Prabhakaran, P., 2017a. Edge computing framework for distributed smart applications. In: 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI). IEEE, pp. 1–8.

Liu, D., Kong, H., Luo, X., Liu, W., Subramaniam, R., 2021a. Bringing AI to edge: From deep learning's perspective. Neurocomputing.

Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C., 2017b. Learning efficient convolutional networks through network slimming. In: IEEE International Conference on Computer Vision. pp. 2736–2744.

Liu, X., Li, B., Shi, P., Ying, L., 2020. POND: Pessimistic-Optimistic oNline Dispatch. arXiv preprint arXiv:2010.09995.

Liu, F.T., Ting, K.M., Zhou, Z.-H., 2008. Isolation forest. In: 2008 Eighth Ieee International Conference on Data Mining. IEEE, pp. 413–422.

Liu, S., Wang, N., 2020. Collaborative optimization scheduling of cloud service resources based on improved genetic algorithm. IEEE Access 8, 150878–150890.

Liu, J., Wang, S., Zhou, A., Kumar, S.A., Yang, F., Buyya, R., 2016. Using proactive fault-tolerance approach to enhance cloud service reliability. IEEE Trans. Cloud Comput. 6 (4), 1191–1202.

Liu, X., Xu, Z., Qin, Y., Tian, J., 2021b. A discrete-event-based simulator for deep learning at edge. arXiv preprint arXiv:2112.00952.

Loshchilov, I., Hutter, F., 2016. Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983.

Luo, C., Qiao, B., Chen, X., Zhao, P., Yao, R., Zhang, H., Wu, W., Zhou, A., Lin, Q., 2020. Intelligent virtual machine provisioning in cloud computing. In: International Joint Conference on AI. pp. 1495–1502.

Luo, C., Qiao, B., Xing, W., Chen, X., Zhao, P., Du, C., Yao, R., Zhang, H., Wu, W., Cai, S., et al., 2021a. Correlation-aware heuristic search for intelligent virtual machine provisioning in cloud systems. In: AAAI Conference on AI, Vol. 35. pp. 12363–12372.

Luo, P., Yu, F.R., Chen, J., Li, J., Leung, V.C., 2021b. A novel adaptive gradient compression scheme: Reducing the communication overhead for distributed deep learning in the internet of things. IEEE Internet Things J..

Luo, C., Zhang, F., Huang, C., Xiong, X., Chen, J., Wang, L., Gao, W., Ye, H., Wu, T., Zhou, R., et al., 2018. AIoT bench: towards comprehensive benchmarking mobile and embedded device intelligence. In: International Symposium on Benchmarking, Measuring and Optimization. Springer, pp. 31–35.

Luping, W., Wei, W., Bo, L., 2019. CMFL: Mitigating communication overhead for federated learning. In: 2019 IEEE 39th International Conference on Distributed Computing Systems. ICDCS, IEEE, pp. 954–964.

Mahmoudi, N., Khazaei, H., 2021. SimFaaS: A performance simulator for serverless computing platforms. arXiv preprint arXiv:2102.08904.

Mahmud, R., Kotagiri, R., Buyya, R., 2018. Fog computing: A taxonomy, survey and future directions. In: Internet of Everything. Springer, pp. 103–130.

Mahmud, R., Ramamohanarao, K., Buyya, R., 2020a. Application management in fog computing environments: A taxonomy, review and future directions. ACM Comput. Surv. 53 (4), 1–43.

Mahmud, R., Srirama, S.N., Ramamohanarao, K., Buyya, R., 2020b. Profit-aware application placement for integrated fog–cloud computing environments. J. Parallel Distrib. Comput. 135, 177–190.

Mampage, A., Karunasekera, S., Buyya, R., 2021. A holistic view on resource management in serverless computing environments: Taxonomy, and future directions. arXiv preprint arXiv:2105.11592.

Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.B., 2017. Mobile edge computing: Survey and research outlook. arXiv preprint arXiv:1701.01090.

Mao, Y., Zhang, J., Letaief, K.B., 2016. Dynamic computation offloading for mobile-edge computing with energy harvesting devices. IEEE J. Sel. Areas Commun. 34 (12), 3590–3605.

Marahatta, A., Xin, Q., Chi, C., Zhang, F., Liu, Z., 2020. PEFS: AI-driven prediction based energy-aware fault-tolerant scheduling scheme for cloud data center. IEEE Trans. Sustain. Comput..

Matrouk, K., Alatoun, K., 2021. Scheduling algorithms in fog computing: A survey. Int. J. Netw. Distrib. Comput..

Matsubara, Y., Baidya, S., Callegaro, D., Levorato, M., Singh, S., 2019. Distilled split deep neural networks for edge-assisted real-time systems. In: Proceedings of the 2019 Workshop on Hot Topics in Video Analytics and Intelligent Edges. pp. 21–26.

Mayer, R., Graser, L., Gupta, H., Saurez, E., Ramachandran, U., 2017. Emufog: Extensible and scalable emulation of large-scale fog computing infrastructures. In: 2017 IEEE Fog World Congress. FWC, IEEE, pp. 1–6.

McChesney, J., Wang, N., Tanwer, A., de Lara, E., Varghese, B., 2019. Defog: fog computing benchmarks. In: Proceedings of the 4th ACM/IEEE Symposium on Edge Computing. pp. 47–58.

Mechalikh, C., Taktak, H., Moussa, F., 2019a. A Scalable and Adaptive Tasks Orchestration Platform for IoT. In: 2019 15th International Wireless Communications & Mobile Computing Conference. IWCMC, IEEE, pp. 1557–1563.

Mechalikh, C., Taktak, H., Moussa, F., 2019b. PureEdgeSim: A simulation toolkit for performance evaluation of cloud, fog, and pure edge computing environments. In: 2019 International Conference on High Performance Computing & Simulation. HPCS, IEEE, pp. 700–707.

Meisner, D., Wu, J., Wenisch, T.F., 2012. Bighouse: A simulation infrastructure for data center systems. In: 2012 IEEE International Symposium on Performance Analysis of Systems & Software. IEEE, pp. 35–45.

Metevier, B., Giguere, S., Brockman, S., Kobren, A., Brun, Y., Brunskill, E., Thomas, P., 2019. Offline contextual bandits with high probability fairness guarantees. Adv. Neural Inf. Process. Syst. 32.

2021. UCI machine learning repository: MHEALTH dataset data set. https://archive.ics.uci.edu/ml/datasets/MHEALTH%2bDataset, (Accessed on 12/28/2021).

Miao, H., Lin, F.X., 2021. Enabling large neural networks on tiny microcontrollers with swapping. arXiv preprint arXiv:2101.08744.

Miranda-Varela, M.-E., Mezura-Montes, E., 2018. Constraint-handling techniques in surrogate-assisted evolutionary optimization. An empirical study. Appl. Soft Comput. 73, 215–229.

Misra, N., Dixit, Y., Al-Mallahi, A., Bhullar, M.S., Upadhyay, R., Martynenko, A., 2020. IoT, big data and artificial intelligence in agriculture and food industry. IEEE Internet Things J..

Murshed, M.S., Murphy, C., Hou, D., Khan, N., Ananthanarayanan, G., Hussain, F., 2021. Machine learning at the network edge: A survey. ACM Comput. Surv. 54 (8), 1–37.

Mustafee, N., Powell, J., Brailsford, S.C., Diallo, S., Padilla, J., Tolk, A., 2015. Hybrid simulation studies and hybrid simulation systems: definitions, challenges, and benefits. In: 2015 Winter Simulation Conference. WSC, IEEE, pp. 1678–1692.

Naha, R.K., Garg, S.K., Amin, M.B., Ranjan, R., 2021. Fuzzy logic-based robust failure handling mechanism for fog computing. CoRR abs/2103.06381. arXiv:2103.06381. URL https://arxiv.org/abs/2103.06381.

Nandi, S., Ghosh, S., Tambe, S.S., Kulkarni, B.D., 2001. Artificial neural-network-assisted stochastic process optimization strategies. AIChE J. 47 (1), 126–141.

Nayak, J., Vakula, K., Dinesh, P., Naik, B., Mohapatra, S., Swarnkar, T., Mishra, M., 2021. Intelligent computing in IoT-enabled smart cities: A systematic review. In: Green Technology for Smart City and Society. Springer, pp. 1–21.

Nayeri, Z.M., Ghafarian, T., Javadi, B., 2021. Application placement in Fog computing with AI approach: Taxonomy and a state of the art survey. J. Netw. Comput. Appl. 103078.

Ndiaye, M., Oyewobi, S.S., Abu-Mahfouz, A.M., Hancke, G.P., Kurien, A.M., Djouani, K., 2020. IoT in the wake of COVID-19: A survey on contributions, challenges and evolution. IEEE Access 8, 186821–186839.

Nguyen, T.-D., Huh, E.-N., 2018. ECSim++: An INET-based simulation tool for modeling and control in edge cloud computing. In: 2018 IEEE International Conference on Edge Computing. EDGE, IEEE, pp. 80–86.

Nguyen, T., Le, T., Zhao, H., Tran, H.Q., Nguyen, T., Phung, D., 2021. TIDOT: A teacher imitation learning approach for domain adaptation with optimal transport. In: IJCAI.

Nguyen-Thien, T., Tran-Cong, T., 1999. Approximation of functions and their derivatives: A neural network implementation with applications. Appl. Math. Model. 23 (9), 687–704.

Nicoletti, B., 2013. Cloud computing. In: Cloud Computing in Financial Services. Springer, pp. 29–64.

Nikitas, A., Michalakopoulou, K., Njoya, E.T., Karampatzakis, D., 2020. Artificial intelligence, transport and the smart city: Definitions and dimensions of a new mobility era. Sustainability 12 (7), 2789.

Oma, R., Nakamura, S., Duolikun, D., Enokido, T., Takizawa, M., 2018. An energy-efficient model for fog computing in the Internet of Things (IoT). Internet Things 1, 14–26.

Onggo, B.S., Corlu, C.G., Juan, A.A., Monks, T., de la Torre, R., 2021. Combining symbiotic simulation systems with enterprise data storage systems for real-time decision-making. Enterp. Inf. Syst. 15 (2), 230–247.

Onggo, B.S., Mustafee, N., Smart, A., Juan, A.A., Molloy, O., 2018. Symbiotic simulation system: hybrid systems model meets big data analytics. In: 2018 Winter Simulation Conference. WSC, IEEE, pp. 1358–1369.

2021. Baetyl/baetyl: Extend cloud computing, data and service seamlessly to edge devices. https://github.com/baetyl/baetyl, (Accessed on 12/25/2021).

Ouhame, S., Hadi, Y., Ullah, A., 2021. An efficient forecasting approach for resource utilization in cloud data center using CNN-LSTM model. Neural Comput. Appl. 1–13.

Pacheco, R.G., Bochie, K., Gilbert, M.S., Couto, R.S., Campista, M.E.M., 2021. Towards edge computing using early-exit convolutional neural networks. Information 12 (10), 431.

Pan, S.J., Yang, Q., 2009. A survey on transfer learning. IEEE Trans. Knowl. Data Eng. 22 (10), 1345–1359.

Panda, S., Neha, B., Sathua, S., 2015. An uncertainty-based task scheduling for heterogeneous multi-cloud systems. Int. J. Inf. Process. 9 (2), 13–24.

Panesar, A., 2019. Machine Learning and AI for Healthcare. Springer.

Park, D., Hoshi, Y., Kemp, C.C., 2018. A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder. IEEE Robot. Autom. Lett. 3 (3), 1544–1551.

Pham, T.-P., Fahringer, T., 2020. Evolutionary multi-objective workflow scheduling for volatile resources in the cloud. IEEE Trans. Cloud Comput..

Piorkowski, M., Sarafijanovic-Djukic, N., Grossglauser, M., 2009. Dataset of mobility traces of taxi cabs in San Francisco, USA. In: CRAWDAD Dataset Epfl/Mobility, 2009-02-24.

2021. beloglazov/planetlab-workload-traces: A set of CPU utilization traces from PlanetLab VMs collected during 10 random days in March and April 2011. https://github.com/beloglazov/planetlab-workload-traces, (Accessed on 12/28/2021).

Preuveneers, D., Rimmer, V., Tsingenopoulos, I., Spooren, J., Joosen, W., Ilie-Zudor, E., 2018. Chained anomaly detection models for federated learning: An intrusion detection case study. Appl. Sci. 8 (12), 2663.

Qiu, M., Thuraisingham, B., Daneshmand, M., Ning, H., Barnaghi, P., 2021. Special issue on robustness and efficiency in the convergence of artificial intelligence and IoT. IEEE Internet Things J. 8 (12), 9460–9462.

Ray, P.P., 2021. A review on TinyML: State-of-the-art and prospects. J. King Saud Univ.-Comput. Inf. Sci..

Reisizadeh, A., Mokhtari, A., Hassani, H., Jadbabaie, A., Pedarsani, R., 2020. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In: International Conference on Artificial Intelligence and Statistics. PMLR, pp. 2021–2031.

Reiss, C., Wilkes, J., Hellerstein, J.L., 2011. Google Cluster-Usage Traces: Format+ Schema. White Paper, Google Inc., pp. 1–14.

Renda, A., Chen, Y., Mendis, C., Carbin, M., 2020. DiffTune: Optimizing CPU simulator parameters with learned differentiable surrogates. In: 2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture. MICRO, IEEE, pp. 442–455.

Rios, L.M., Sahinidis, N.V., 2013. Derivative-free optimization: a review of algorithms and comparison of software implementations. J. Global Optim. 56 (3), 1247–1293.

Rudd-Orthner, R.N., Mihaylova, L., 2020. Repeatable determinism using non-random weight initialisations in smart city applications of deep learning. J. Reliab. Intell. Environ. 6 (1), 31–49.

Russell, S., Norvig, P., 2009. Artificial Intelligence: A Modern Approach, third ed. Prentice Hall Press, USA.

Saeed, A., Salim, F.D., Ozcelebi, T., Lukkien, J., 2020. Federated self-supervised learning of multisensor representations for embedded intelligence. IEEE Internet Things J. 8 (2), 1030–1040.

Salehi-Amiri, A., Jabbarzadeh, A., Zahedi, A., Akbarpour, N., Hajiaghaei-Keshteli, M., 2021. Relief supply chain management using internet of things to address COVID-19 outbreak. Comput. Ind. Eng. 107429.

Sami, H., Otrok, H., Bentahar, J., Mourad, A., 2021. AI-based resource provisioning of IoE services in 6G: A deep reinforcement learning approach. IEEE Trans. Netw. Serv. Manag..

Satpathy, A., Addya, S.K., Turuk, A.K., Majhi, B., Sahoo, G., 2018. Crow search based virtual machine placement strategy in cloud data centers with live migration. Comput. Electr. Eng. 69, 334–350.

Sattler, F., Wiedemann, S., Müller, K.-R., Samek, W., 2019. Robust and communication-efficient federated learning from non-iid data. IEEE Trans. Neural Netw. Learn. Syst. 31 (9), 3400–3413.

Satyanarayanan, M., 2017. The emergence of edge computing. Computer 50 (1), 30–39.

Scarpiniti, M., Baccarelli, E., Momenzadeh, A., Sarv Ahrabi, S., 2021. DeepFogSim: A toolbox for execution and performance evaluation of the inference phase of conditional deep neural networks with early exits atop distributed Fog platforms. Appl. Sci. 11 (1), 377.

Schleier-Smith, J., 2015. An architecture for agile machine learning in real-time applications. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 2059–2068.

Sefraoui, O., Aissaoui, M., Eleuldj, M., 2012. OpenStack: toward an open-source solution for cloud computing. Int. J. Comput. Appl. 55 (3), 38–42.

Shahidinejad, A., Ghobaei-Arani, M., 2021. A metaheuristic-based computation offloading in edge-cloud environment. J. Ambient Intell. Humaniz. Comput. 1–10.

2021. tiny_shakespeare | TensorFlow datasets. https://www.tensorflow.org/datasets/catalog/tiny_shakespeare, (Accessed on 12/26/2021).

Shao, J., Zhang, J., 2020a. Bottlenet++: An end-to-end approach for feature compression in device-edge co-inference systems. In: 2020 IEEE International Conference on Communications Workshops (ICC Workshops). IEEE, pp. 1–6.

Shao, J., Zhang, J., 2020b. Communication-computation trade-off in resource-constrained edge inference. IEEE Commun. Mag. 58 (12), 20–26.

Sharma, Y., Javadi, B., Si, W., Sun, D., 2016. Reliability and energy efficiency in cloud computing systems: Survey and taxonomy. J. Netw. Comput. Appl. 74, 66–85.

Shen, S., van Beek, V., Iosup, A., 2015. Statistical characterization of business-critical workloads hosted in cloud datacenters. In: CCGrid. IEEE, pp. 465–474.

Sheng, S., Chen, P., Chen, Z., Wu, L., Yao, Y., 2021. Deep reinforcement learning-based task scheduling in IoT edge computing. Sensors 21 (5), 1666.

Shi, Y., Yang, K., Jiang, T., Zhang, J., Letaief, K.B., 2020. Communication-efficient edge AI: Algorithms and systems. IEEE Commun. Surv. Tutor. 22 (4), 2167–2191.

Shivakumar, S.K., 2015. Ensuring high availability for your enterprise web applications. In: Architecting High Performing, Scalable and Available Enterprise Web Applications. pp. 59–99.

Singh, P., Gupta, P., Jyoti, K., 2019. TASM: Technocrat ARIMA and SVR model for workload prediction of web applications in cloud. Cluster Comput. 22 (2), 619–633.

Singh, J., Singh, P., Gill, S.S., 2021. Fog computing: A taxonomy, systematic review, current trends and research challenges. J. Parallel Distrib. Comput. 157, 56–85.

Sivanathan, A., Gharakheili, H.H., Loi, F., Radford, A., Wijenayake, C., Vishwanath, A., Sivaraman, V., 2018. Classifying IoT devices in smart environments using network traffic characteristics. IEEE Trans. Mob. Comput. 18 (8), 1745–1759.

Song, Z., Hao, Y., Liu, Y., Sun, X., 2021. Energy-efficient multiaccess edge computing for terrestrial-satellite Internet of Things. IEEE Internet Things J. 8 (18), 14202–14218.

Soualhia, M., Fu, C., Khomh, F., 2019. Infrastructure fault detection and prediction in edge cloud environments. In: Proceedings of the 4th ACM/IEEE Symposium on Edge Computing. pp. 222–235.

Stackowiak, R., 2019. Azure IoT solutions overview. In: Azure Internet of Things Revealed. Springer, pp. 29–54.

Stuckey, P.J., Guns, T., Bailey, J., Leckie, C., Ramamohanarao, K., Chan, J., et al., 2020. Dynamic programming for predict+optimise. In: AAAI, Vol. 34. pp. 1444–1451.

Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., Pei, D., 2019. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2828–2837.

Sufian, A., Ghosh, A., Sadiq, A.S., Smarandache, F., 2020. A survey on deep transfer learning to edge computing for mitigating the COVID-19 pandemic. J. Syst. Archit. 108, 101830.

Suryadevara, N.K., 2021. Energy and Latency reductions at the Fog gateway using a Machine Learning classifier. Sustain. Comput.: Inform. Syst. 100582.

Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning: An Introduction. MIT Press.

Talaat, F.M., Ali, S.H., Saleh, A.I., Ali, H.A., 2019. Effective load balancing strategy (ELBS) for real-time fog computing environment using fuzzy and probabilistic neural networks. J. Netw. Syst. Manage. 27 (4), 883–929.

Talaat, F.M., Saraya, M.S., Saleh, A.I., Ali, H.A., Ali, S.H., 2020. A load balancing and optimization strategy (LBOS) using reinforcement learning in fog computing environment. J. Ambient Intell. Humaniz. Comput. 1–16.

Tang, Z., Zhou, X., Zhang, F., Jia, W., Zhao, W., 2018. Migration modeling and learning algorithms for containers in fog computing. IEEE Trans. Serv. Comput. 12 (5), 712–725.

Taylor, S.J., Letham, B., 2018. Forecasting at scale. Amer. Statist. 72 (1), 37–45.

Teerapittayanon, S., McDanel, B., Kung, H.-T., 2017. Distributed deep neural networks over the cloud, the edge and end devices. In: 2017 IEEE 37th International Conference on Distributed Computing Systems. ICDCS, IEEE, pp. 328–339.

Theis, T.N., Wong, H.-S.P., 2017. The end of moore's law: A new beginning for information technology. Comput. Sci. Eng. 19 (2), 41–50.

Tran, N.H., Bao, W., Zomaya, A., Nguyen, M.N., Hong, C.S., 2019. Federated learning over wireless networks: Optimization model design and analysis. In: IEEE INFOCOM 2019-IEEE Conference on Computer Communications. IEEE, pp. 1387–1395.

Tuli, S., 2022. SplitPlace: Intelligent placement of split neural nets in mobile edge environments. SIGMETRICS Perform. Eval. Rev. 49 (2), 63–65. http://dx.doi.org/10.1145/3512798.3512821.

Tuli, S., Basumatary, N., Buyya, R., 2019a. Edgelens: Deep learning based object detection in integrated iot, fog and cloud computing environments. In: 2019 4th International Conference on Information Systems and Computer Networks. ISCON, IEEE, pp. 496–502.

Tuli, S., Casale, G., Jennings, N.R., 2022a. GOSH: Task scheduling using deep surrogate models in fog computing environments. IEEE Trans. Parallel Distrib. Syst..

Tuli, S., Casale, G., Jennings, N.R., 2022b. MCDS: AI augmented workflow scheduling in mobile edge cloud computing systems. IEEE Trans. Parallel Distrib. Syst..

Tuli, S., Casale, G., Jennings, N.R., 2022c. PreGAN: Preemptive migration prediction network for proactive fault-tolerant edge computing. In: IEEE Conference on Computer Communications. INFOCOM, IEEE.

Tuli, S., Casale, G., Jennings, N.R., 2022d. TranAD: Deep transformer networks for anomaly detection in multivariate time series data. Proc. VLDB 15 (6), 1201–1214.

Tuli, S., Gill, S.S., Casale, G., Jennings, N.R., 2020a. iThermoFog: IoT-Fog based automatic thermal profile creation for cloud data centers using artificial intelligence techniques. Internet Technol. Lett. 3 (5), e198.

Tuli, S., Gill, S.S., Garraghan, P., Buyya, R., Casale, G., Jennings, N., 2021a. START: Straggler prediction and mitigation for cloud computing environments using encoder LSTM networks. IEEE Trans. Serv. Comput..

Tuli, S., Gill, S.S., Xu, M., Garraghan, P., Bahsoon, R., Dustdar, S., Sakellariou, R., Rana, O., Buyya, R., Casale, G., et al., 2021b. HUNTER: AI based holistic resource management for sustainable cloud computing. J. Syst. Softw. 111–124.

Tuli, S., Ilager, S., Ramamohanarao, K., Buyya, R., 2020b. Dynamic scheduling for stochastic edge-cloud computing environments using A3C learning and residual recurrent neural networks. IEEE Trans. Mob. Comput..

Tuli, S., Mahmud, R., Tuli, S., Buyya, R., 2019b. Fogbus: A blockchain-based lightweight framework for edge and fog computing. J. Syst. Softw. 154, 22–36.

Tuli, S., Poojara, S.R., Srirama, S.N., Casale, G., Jennings, N.R., 2022e. COSCO: Container orchestration using co-simulation and gradient based optimization for fog computing environments. IEEE Trans. Parallel Distrib. Syst. 33 (1), 101–116. http://dx.doi.org/10.1109/TPDS.2021.3087349.

Tuli, S., Tuli, S., Casale, G., Jennings, N.R., 2021c. Generative optimization networks for memory efficient data generation. In: Advances in Neural Information Processing Systems, Workshop on ML for Systems.

Ushakov, Y.A., Polezhaev, P.N., Shukhman, A.E., Ushakova, M.V., Nadezhda, M., 2018. Split neural networks for mobile devices. In: 2018 26th Telecommunications Forum. TELFOR, IEEE, pp. 420–425.

Varga, A., 2010. OMNeT++. In: Modeling and Tools for Network Simulation. Springer, pp. 35–59.

Varghese, B., Buyya, R., 2018. Next generation cloud computing: New trends and research directions. Future Gener. Comput. Syst. 79, 849–861.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. Adv. Neural Inf. Process. Syst. 30, 5998–6008.

van de Ven, P., Borst, S., Shneer, S., 2009. Instability of maxweight scheduling algorithms. In: IEEE INFOCOM 2009. IEEE, pp. 1701–1709.

van de Ven, P.M., Borst, S.C., Ying, L., 2013. Inefficiency of MaxWeight scheduling in spatial wireless networks. Comput. Commun. 36 (12), 1350–1359.

Wan, C., Wang, C., Pei, J., 2012. A QoS-awared scientific workflow scheduling schema in cloud computing. In: 2012 IEEE International Conference on Information Science and Technology. IEEE, pp. 634–639.

Wang, H., Cai, G., Huang, Z., Dong, F., 2019a. ADDA: Adaptive distributed DNN inference acceleration in edge computing environment. In: 2019 IEEE 25th International Conference on Parallel and Distributed Systems. ICPADS, IEEE, pp. 438–445.

Wang, H., Gui, S., Yang, H., Liu, J., Wang, Z., 2020a. GAN slimming: All-in-one GAN compression by a unified optimization framework. In: European Conference on Computer Vision. Springer, pp. 54–73.

Wang, Y., Guo, Y., Guo, Z., Baker, T., Liu, W., 2020b. CLOSURE: A cloud scientific workflow scheduling algorithm based on attack–defense game model. Future Gener. Comput. Syst. 111, 460–474.

Wang, X., Han, Y., Leung, V.C., Niyato, D., Yan, X., Chen, X., 2020c. Convergence of edge computing and deep learning: A comprehensive survey. IEEE Commun. Surv. Tutor. 22 (2), 869–904.

Wang, S., Hu, Y., Wu, J., 2020d. KubeEdge. AI: AI platform for edge devices. arXiv preprint arXiv:2007.09227.

Wang, P., Lei, Y., Agbedanu, P.R., Zhang, Z., 2020e. Makespan-driven workflow scheduling in clouds using immune-based PSO algorithm. IEEE Access 8, 29281–29290.

Wang, C., Li, R., Li, W., Qiu, C., Wang, X., 2021a. SimEdgeIntel: A open-source simulation platform for resource management in edge intelligence. J. Syst. Archit. 115, 102016.

Wang, T., Liang, Y., Zhang, Y., Zheng, X., Arif, M., Wang, J., Jin, Q., 2020f. An intelligent dynamic offloading from cloud to edge for smart IoT systems with big data. IEEE Trans. Netw. Sci. Eng. 7 (4), 2598–2607.

Wang, B., Liu, F., Lin, W., 2021b. Energy-efficient VM scheduling based on deep reinforcement learning. Future Gener. Comput. Syst. 125, 616–628.

Wang, Y., Liu, H., Zheng, W., Xia, Y., Li, Y., Chen, P., Guo, K., Xie, H., 2019b. Multi-objective workflow scheduling with deep-Q-network-based multi-agent reinforcement learning. IEEE Access 7, 39974–39982.

Wang, L., Mao, W., Zhao, J., Xu, Y., 2021c. DDQP: A double deep Q-learning approach to online fault-tolerant SFC placement. IEEE Trans. Netw. Serv. Manag. 18 (1), 118–132.

Wang, Y., Wang, S., Yang, B., Gao, B., Wang, S., 2020g. An effective adaptive adjustment method for service composition exception handling in cloud manufacturing. J. Intell. Manuf. 1–17.

Wang, J., Wu, N., Zhao, W.X., Peng, F., Lin, X., 2019c. Empowering A* search algorithms with neural networks for personalized route recommendation. In: The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 539–547.

Wang, Y., Yang, J., Guo, X., Qu, Z., 2019d. Satellite edge computing for the internet of things in aerospace. Sensors 19 (20), 4375.

Wijethilaka, S., Liyanage, M., 2021. Survey on network slicing for Internet of Things realization in 5G networks. IEEE Commun. Surv. Tutor. 23 (2), 957–994.

2021. Page view statistics for Wikimedia projects. https://dumps.wikimedia.org/other/pagecounts-raw/, (Accessed on 12/26/2021).

Wilder, B., Dilkina, B., Tambe, M., 2019. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In: AAAI, Vol. 33. pp. 1658–1665.

Won, H., Kim, Y., 2021. Performance analysis of machine learning based fault detection for cloud infrastructure. In: 2021 International Conference on Information Networking. ICOIN, IEEE, pp. 877–880.

Wong, A., Famouri, M., Shafiee, M.J., 2020. Attendnets: tiny deep image recognition neural networks for the edge via visual attention condensers. arXiv preprint arXiv:2009.14385.

Wu, W., He, L., Lin, W., Mao, R., 2020. Accelerating federated learning over reliability-agnostic clients in mobile edge computing systems. IEEE Trans. Parallel Distrib. Syst..

Xia, C., Zhao, J., Cui, H., Feng, X., Xue, J., 2019. Dnntune: Automatic benchmarking dnn models for mobile-cloud computing. ACM Trans. Archit. Code Optim. (TACO) 16 (4), 1–26.

Xu, T., Han, G., Qi, X., Du, J., Lin, C., Shu, L., 2020a. A hybrid machine learning model for demand prediction of edge-computing-based bike-sharing system using internet of things. IEEE Internet Things J. 7 (8), 7345–7356.

Xu, H., Mannor, S., 2012. Robustness and generalization. Mach. Learn. 86 (3), 391–423.

Xu, Z., Tang, J., Yin, C., Wang, Y., Xue, G., Wang, J., Gursoy, M.C., 2020b. ReCARL: Resource allocation in cloud RANs with deep reinforcement learning. IEEE Trans. Mob. Comput..

Xu, Y.-H., Yang, C.-C., Hua, M., Zhou, W., 2020c. Deep deterministic policy gradient (DDPG)-based resource allocation scheme for NOMA vehicular communications. IEEE Access 8, 18797–18807.

2021. Webscope | Yahoo labs. https://webscope.sandbox.yahoo.com/, (Accessed on 12/26/2021).

Yang, Z., Chen, M., Saad, W., Hong, C.S., Shikh-Bahaei, M., 2020a. Energy efficient federated learning over wireless communication networks. IEEE Trans. Wireless Commun. 20 (3), 1935–1949.

Yang, R., Yu, F.R., Si, P., Yang, Z., Zhang, Y., 2019. Integrated blockchain and edge computing systems: A survey, some research issues and challenges. IEEE Commun. Surv. Tutor. 21 (2), 1508–1532.

Yang, L., Zhong, C., Yang, Q., Zou, W., Fathalla, A., 2020b. Task offloading for directed acyclic graph applications based on edge computing in industrial internet. Inform. Sci. 540, 51–68.

Yaqoob, I., Salah, K., Jayaraman, R., Al-Hammadi, Y., 2021. Blockchain for healthcare data management: Opportunities, challenges, and future recommendations. Neural Comput. Appl. 1–16.

Yazdanian, P., Sharifian, S., 2021. E2LG: a multiscale ensemble of LSTM/GAN deep learning architecture for multistep-ahead cloud workload prediction. J. Supercomput. 1–31.

Ye, K., Liu, Y., Xu, G., Xu, C.-Z., 2018. Fault injection and detection for artificial intelligence applications in container-based clouds. In: International Conference on Cloud Computing. Springer, pp. 112–127.

Yu, F., Cui, L., Wang, P., Han, C., Huang, R., Huang, X., 2020. Easiedge: A novel global deep neural networks pruning method for efficient edge computing. IEEE Internet Things J. 8 (3), 1259–1271.

Yu, M., Jiang, Z., Ng, H.C., Wang, W., Chen, R., Li, B., 2021. Gillis: Serving large neural networks in serverless functions with automatic model partitioning. In: International Conference on Distributed Computing Systems.

Yuan, J., Zheng, Y., Zhang, C., Xie, W., Xie, X., Sun, G., Huang, Y., 2010. T-drive: driving directions based on taxi trajectories. In: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems. pp. 99–108.

Zhang, Y., Chen, Y., Wang, J., Pan, Z., 2021a. Unsupervised deep anomaly detection for multi-sensor time-series signals. IEEE Trans. Knowl. Data Eng..

Zhang, Y., Huang, H., Yang, L.-X., Xiang, Y., Li, M., 2019a. Serious challenges and potential solutions for the industrial Internet of Things with edge intelligence. IEEE Netw. 33 (5), 41–45.

Zhang, B., Mor, N., Kolb, J., Chan, D.S., Lutz, K., Allman, E., Wawrzynek, J., Lee, E., Kubiatowicz, J., 2015. The cloud is not enough: Saving iot from the cloud. In: 7th {USENIX} Workshop on Hot Topics in Cloud Computing (HotCloud 15).

Zhang, C., Song, D., Chen, Y., Feng, X., Lumezanu, C., Cheng, W., Ni, J., Zong, B., Chen, H., Chawla, N.V., 2019. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33. pp. 1409–1416.

Zhang, L., Zhang, L., 2017. Deep learning-based classification and reconstruction of residential scenes from large-scale point clouds. IEEE Trans. Geosci. Remote Sens. 56 (4), 1887–1897.

Zhang, S., Zhang, S., Qian, Z., Wu, J., Jin, Y., Lu, S., 2021b. Deepslicing: collaborative and adaptive CNN inference with low latency. IEEE Trans. Parallel Distrib. Syst. 32 (9), 2175–2187.

Zhao, H., Deng, S., Liu, Z., Yin, J., Dustdar, S., 2020a. Distributed redundancy scheduling for microservice-based applications at the edge. IEEE Trans. Serv. Comput..

Zhao, H., Wang, Y., Duan, J., Huang, C., Cao, D., Tong, Y., Xu, B., Bai, J., Tong, J., Zhang, Q., 2020b. Multivariate time-series anomaly detection via graph attention network. In: International Conference on Data Mining.

Zhao, Z., Wang, K., Ling, N., Xing, G., 2021. EdgeML: An AutoML framework for real-time deep learning on the edge. In: Proceedings of the International Conference on Internet-of-Things Design and Implementation. pp. 133–144.

Zhong, Z., Xu, M., Rodriguez, M.A., Xu, C., Buyya, R., 2021. Machine learning-based orchestration of containers: A taxonomy and future directions. arXiv preprint arXiv:2106.12739.

Zhou, Z., Chen, X., Li, E., Zeng, L., Luo, K., Zhang, J., 2019. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. Proc. IEEE 107 (8), 1738–1762.

Zhou, H., Zhang, W., Wang, C., Ma, X., Yu, H., 2021. BBNet: A novel convolutional neural network structure in edge-cloud collaborative inference. Sensors 21 (13), 4494.

Zhu, H., Akrout, M., Zheng, B., Pelegris, A., Jayarajan, A., Phanishayee, A., Schroeder, B., Pekhimenko, G., 2018. Benchmarking and analyzing deep neural network training. In: 2018 IEEE International Symposium on Workload Characterization. IISWC, IEEE, pp. 88–100.

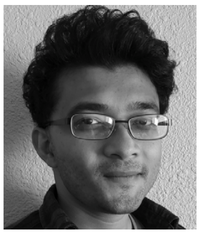
Zhu, Z., Peng, J., Zhou, Z., Zhang, X., Huang, Z., 2016. PSO-SVR-based resource demand prediction in cloud computing. J. Adv. Comput. Intell. Intell. Inform. 20 (2), 324–331.

Zong, B., Song, Q., Min, M.R., Cheng, W., Lumezanu, C., Cho, D., Chen, H., 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: International Conference on Learning Representations.

Shreshth Tuli is a President's Ph.D. Scholar at the Department of Computing, Imperial College London, UK. Prior to this he was an undergraduate student at the Department of Computer Science and Engineering at Indian Institute of Technology — Delhi, India. He has worked as a visiting research fellow at the CLOUDS Laboratory, School of Computing and Information Systems, the University of Melbourne, Australia. His research interests include Fog Computing and Deep Learning. For further information, visit https://shreshthtuli.github.io/.

Fatemeh Mirhakimi is a current Master of Research student at Western Sydney University, Australia. She received B.S. and M.S. in Computer Engineering from the Azad University of Arak, respectively. She has conducted a research project about power-consuming management in hybrid vehicles based on fuzzy-genetic methods. Her research interests include cloud and edge computing, the internet of things, and reliability and fault tolerance.

Samodha Pallewatta is a Ph.D. student at the Cloud Computing and Distributed Systems (CLOUDS) Laboratory, Department of Computing and Information Systems, The University of Melbourne, Australia. Her research interests encompass Fog/Edge Computing, Internet of Things (IoT), Microservice architecture and Distributed Systems. She is one of the contributors of the iFogSim simulator, used extensively for resource management research in Fog/Edge computing.



Syed Zawad received the B.Sc. degree in computer science and engineering from BRAC University, Dhaka, Bangladesh. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, University of Nevada, Reno, NV, USA. He has interned as a Researcher with Baidu, Sunnyvale, CA, USA. He also has three years of work experience as a Software Engineer for Web applications. His research area of interest is in high performance computing, deep learning, federated learning, and neural architecture search.

Giuliano Casale joined the Department of Computing at Imperial College London in 2010, where he is currently a Reader. Previously, he worked as a research scientist and consultant in the capacity planning industry. He teaches and does research in performance engineering and cloud computing, topics on which he has published more than 100 refereed papers. His research work has received multiple awards, recently the best paper award at ACM SIGMETRICS. He serves on the editorial boards of IEEE TNSM and ACM TOMPECS and as current chair of ACM SIGMETRICS.

Bahman Javadi is an Associate Professor in Networking and Cloud Computing at Western Sydney University, Australia. Prior to this appointment, he was a Research Fellow at the University of Melbourne and a Postdoctoral Fellow at the INRIA Rhone-Alpes, France. His research interests include Cloud computing, Edge computing, performance evaluation of large-scale distributed computing systems, and reliability and fault tolerance. He is a Senior Member of ACM, Senior Member of IEEE and Senior Fellow of Advance HE of UK. His website is: https://staff.cdms.westernsydney.edu.au/~bjavadi/.

Feng Yan is an Assistant Professor of Computer Science and Engineering at University of Nevada, Reno and the director of the Intelligent Data and Systems Lab. He received M.S. and Ph.D. degrees in Computer Science from the College of William and Mary and worked at Microsoft Research and HP Labs. His research bridges the fields of big data, machine learning, and systems. He is a recipient of the Best Student Paper Award of IEEE CLOUD 2018, the Best Paper Award of CLOUD 2019, and the Best Student Paper Award of ITNG 2021, as well as the NSF CAREER Award, the NSF CRII Award, and the Regents' Rising Researcher Award.

Rajkumar Buyya is a Redmond Barry Distinguished Professor and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He has authored over 625 publications and seven text books including "Mastering Cloud Computing" published by McGraw Hill, China Machine Press, and Morgan Kaufmann for Indian, Chinese and international markets respectively. He is one of the highly cited authors in computer science and software engineering worldwide (h-index=154, g-index=322, 124,800+ citations).

Nicholas R. Jennings is the Vice-Chancellor and President of Loughborough University. He is an internationally-recognized authority in the areas of AI, autonomous systems, cyber-security and agent-based computing. He is a member of the UK government's AI Council, the governing body of the Engineering and Physical Sciences Research Council, and chair of the Royal Academy of Engineering's Policy Committee. Before Loughborough, he was the Vice-Provost for Research and Enterprise and Professor of Artificial Intelligence at Imperial College London, the UK's first Regius Professor of Computer Science (a post bestowed by the monarch to recognize exceptionally high quality research) and the UK Government's first Chief Scientific Advisor for National Security.