Computing (2025) 107:100
https://doi.org/10.1007/s00607-025-01432-z

REGULAR ARTICLE

AI Techniques in the Microservices Life-Cycle: a Systematic Mapping Study

Sergio Moreschini[1,2] · Shahrzad Pour[3] · Ivan Lanese[4] · Daniel Balouek[5] · Justus Bogner[6] · Xiaozhou Li[2] · Fabiano Pecorelli[7] · Jacopo Soldani[8] · Eddy Truyen[9] · Davide Taibi[2]

Received: 11 July 2024 / Accepted: 28 January 2025

Abstract

The use of AI in microservices (MSs) is an emerging field as indicated by a substantial number of surveys. However these surveys focus on a specific problem using specific AI techniques, therefore not fully capturing the growth of research and the rise and disappearance of trends. In our systematic mapping study, we take an exhaustive approach to reveal all possible connections between the use of AI techniques for improving any quality attribute (QA) of MSs during the DevOps phases. Our results include 16 research themes that connect to the intersection of particular QAs, AI domains and DevOps phases. Moreover by mapping identified future research challenges and relevant industry domains, we can show that many studies aim to deliver

Daniel Balouek, Justus Bogner, Xiaozhou Li, Fabiano Pecorelli, Jacopo Soldani, and Eddy Truyen have contributed equally to this work.

✉ Sergio Moreschini
sergio.moreschini@tuni.fi

Shahrzad Pour
shmp@dtu.dk

Ivan Lanese
ivan.lanese@gmail.com

1 Tampere University, Tampere, Finland

2 University of Oulu, Oulu, Finland

3 DTU Compute, Technical University of Denmark, Lyngby, Denmark

4 OLAS team, University of Bologna/INRIA, Bologna, Italy

5 LS2N Laboratory, Inria, IMT Atlantique, Nantes, France

6 Vrije Universiteit Amsterdam, Amsterdam, The Netherlands

7 Pegaso Digital University, Naples, Italy

8 Department of Computer Science, University of Pisa, Pisa, Italy

9 DistriNet, KU Leuven, Leuven, Belgium

Published online: 29 March 2025

prototypes to be automated at a later stage, aiming at providing exploitable products in a number of key industry domains.

Keywords Microservices · AI · Machine learning

Mathematics Subject Classification 68T01

1 Introduction

Microservices (MSs) [1] is a popular architectural style for distributed applications, which originated from service-oriented computing and pushed the concept of modularity much further than its ancestors. As such, an MS-based system consists of small, loosely coupled, and possibly heterogeneous services, which can be deployed, updated, and scaled independently. This is often supported by executing individual services in containers, such as the ones provided by Docker [2]. Containerization ensures that services can be easily moved or duplicated. As a result, MSs can provide high flexibility, scalability, and evolvability. However, these advantages come at a price: an MS system often comprises many fine-grained services, which may interact according to complex patterns. Mastering this complexity is challenging.

In recent years, Artificial Intelligence (AI) in general and Machine Learning (ML) in particular have attracted considerable interest from research and practice [3]. As a result, AI techniques have been applied in various application areas, and software engineering is no exception [4]. In particular, AI has been applied in numerous works to support the development and operations of MSs. However, to the best of our knowledge, the role of AI for MSs is still unclear, and no holistic secondary studies analyzed the adoption of AI techniques for MSs in an exhaustive manner.

This paper focuses on the use of AI techniques to solve challenges or improve the quality of MS-based systems (AI4MS), e.g., regarding design, development, and operation. We would like to understand how and why AI techniques are used within MS Architecture (MSA) and its life-cycle, which AI approaches are used, in which industry domains, and which challenges are still open for future research. To this end, we report on what is being said in the literature on the topic, by providing a sort of"snapshot"of the state-of-the-art on AI4MS. We indeed performed a systematic mapping study (SMS) [5], investigating how the publication landscape evolved over the years and including 269 peer-reviewed papers published between 2017 and up to and including 2023. The aim of our SMS is to overview the *when*, *where*, *why*, and *how* of AI4MS, while also shedding light on open research challenges in the field. Among other things, we study how the number of AI4MS publications has evolved over the years, in which industry domains AI4MS is used, which quality attributes (QAs) it improves in which DevOps phases, and by means of which AI techniques.

To extract data according to taxonomies as uniform and unbiased as possible, we reused established classifications whenever possible. For instance, for AI techniques we used the classification in [6], and for improved QAs we referred to the ISO 25010:2011 (SQuaRE) standard [7]. For the phases of the software life-cycle in which an approach is used, we referred to the DevOps life-cycle [8].

The results of our work can inform researchers about the relationship between AI and MSs, with a focus on how modern AI techniques are used to improve MS systems. Such information can be used by researchers to take informed decisions on AI-based techniques to consider when designing future MSAs and to investigate valuable open challenges. Also, a refined understanding of which QAs are improved by using AI and in which DevOps phases can be useful to practitioners interested in enhancing their MS systems. The main contribution of this paper is a report on the state of the art concerning AI techniques to support MSs.

Paper structure: Section 2 introduces the background information on MS and the related works about AI4MS. Our research method is described in Sect. 3. Section 4 provides the results for the five research questions individually, while Sect. 5 refines the results of RQs 2-4 via a multidimensional analysis. Section 6 presents the main discussion points and implications originating from the analysis. The possible threats to the study's validity are in Sect. 7. Section 8 concludes the paper.

2 Background on microservices and related work

2.1 Microservices

As a reimagination of the service-oriented architectures (SOAs) [9] approach, MSA started to rise in popularity around 2014 [1, 10, 11]. However, the MSA architectural style was already used by several companies, such as Netflix [12]. Today, MSA is fairly popular in industry, e.g., 37% of developers surveyed by JetBrains in 2022 responded they were using MSs.[1] MSs are also a popular research topic today, with a substantial number of publications each year. Google Scholar[2] reports over 63,700 publications for the search term `"microservices"`. According to Fowler and Lewis [10], MSA is a service-based architectural style with characteristics like *componentization via services*, *organization around business capabilities*, *infrastructure automation*, and *evolutionary design*. Creating and operating MS-based systems can be challenging and expensive, and many companies even abandoned MSA for a monolithic architecture [13, 14]. However, a well-designed MSA is beneficial for many software quality attributes, such as maintainability, scalability, reliability, and portability [15–17].

Teams developing MSA-based systems usually follow a DevOps life-cycle to facilitate the management of a large number of small services [18].

DevOps is a software development paradigm trying to bring software faster and more reliably into production [19] by destroying barriers between the development and operation teams. Another objective of DevOps is to reduce cycle time [20], i.e., how long it takes from starting the development of a feature until its deployment in production. The software development life-cycle is composed of eight phases [19], with the last phase leading back into the first one (see also Fig. 1): *Plan*, *Code*, *Build*, *Test*, *Release*, *Deploy*, *Operate*, and *Monitor*. While MSAs can be developed following

[1] https://www.jetbrains.com/lp/devecosystem-2022/microservices/.

[2] https://scholar.google.com, queried on 2024-09-30.

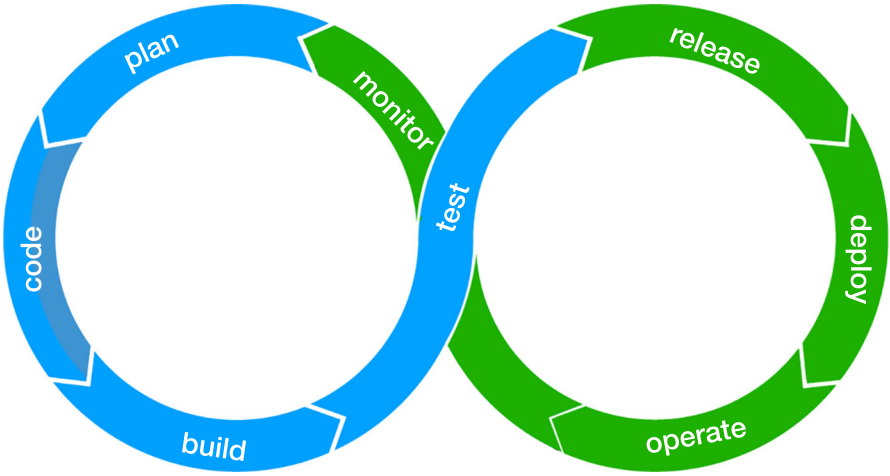

Fig. 1 DevOps life-cycle

other life-cycles as well, we will use the DevOps life-cycle as a reference, given that most other development models include a subset of its phases.

Lately, Machine Learning Operations (MLOps) has been proposed as a version of DevOps enhanced for ML-based applications. Multiple definitions of MLOps have been provided based on the different perspectives of the ML lifecycle. [21] provides a graphical representation of MLOps aimed at "maintaining the simple and iconic pipeline of DevOps, yet improving it by adding new circular steps for ML incorporation". Hence, we could also have used such a MLOps life-cycle representation [22] for our analysis. However, it would not have been possible to apply it to other AI techniques; hence, we preferred DevOps which can be applied to a larger set of works. The same applies for the concept of Artificial Intelligence for IT Operations (AIOps), which makes use of AI for automating the Operational side of DevOps pipelines.

2.2 Related work

Table 1 presents an overview of existing surveys on the use of AI in application architectures, container technologies, and infrastructures that are based on or support MSs. The found surveys belong to three categories, divided by a horizontal line in Table 1. The works in the first one focus on a set of AI techniques to solve specific problems in specific DevOps phases, while the second category studies the use of very specific AI models for MSs. Finally, the third category lists papers that focus on failure diagnosis and root-cause analysis (RCA) in MSAs by means of AIOps.

Most of the surveys in the first category focus on the use of AI to cost-effectively optimize performance by means of improved support for application placement, autoscaling, and monitoring in the later DevOps phases. Hilali et al. [23] present an SMS on the use of Machine Learning (ML) for the self-adaptation of MSs. Interestingly, the used methodology involves searching papers irrespective of whether they use

Table 1 Comparison of our SMS with the existing surveys

	Models	DevOps phase	Studied problem	Period	#SPs	Type
Ours	All	All	No restriction on the scope	2014-2023	269	SMS
[23]	ML DL	Operate monitor	Self-adaptation of MSs	2015-2021	62	SMS
[24]	ML RL	Deploy operate monitor	Performance efficiency of container orchestration	2016-2021	44	SLR
[25]	ML	Deploy operate monitor	Performance efficiency and reliability of edge-fog-cloud spectrum	n/a	70	SLR
[26]	ML EA	Deploy operate monitor	Performance efficiency of application placement (broader than MS) in fog computing	2017-2020	109	SLR
[27]	GNN	Monitor operate code	Anomaly detection, resource scheduling, monolith decomposition	2020-2022	10	SLR
[28]	Clustering	Plan code	Monolith decomposition	2015-2023	22	SMS
[29]	AIOps	Operate monitor	Reliability	n/a	n/a	survey
[30]	AIOps	Operate monitor	Reliability	2003-2024	94	SLR
[31]	AIOps	Operate monitor	Reliability	n/a	n/a	survey

DL, deep learning; RL, reinforcement learning; EA, Evolutionary algorithms; GNN, graph neural networks; GA, genetic algorithms; SPs, Selected papers

ML. An interesting finding is that only 40.3% of the collected papers on self-adaptation for MSs use either classical ML or Deep Learning (DL). Zhong et al. [24] present a taxonomy and future research directions for ML-based container orchestration via Reinforcement Learning (RL) with a focus on achieving improved performance efficiency. Duc et al. [25] present a survey of ML-based performance modeling and resource management techniques for distributed computing environments formed by the spectrum of edge, fog, and cloud computing. Nayeri et al. [26] present a taxonomy of AI-based application placement algorithms for optimizing performance metrics in fog computing environments. These algorithms are divided into three groups: Evolutionary Algorithms (EA), ML-based algorithms, and hybrid algorithms that combine different kinds of algorithms.

The second category of papers focuses on a particular AI technique. It includes two surveys. Nguyen et al. [27] present a survey on the use of Graph Neural Networks (GNN) in the field of MSs. Saucedo et al. [28] conducted a systematic mapping study on the use of AI for migrating monolithic application to MS-based applications. A striking finding is that a massive amount of papers have used clustering as the primary technique. In our survey, we also cover GNN papers, which are classified under the neural network keyword that belongs to the ML sub-class. For migration, we also found that clustering and unsupervised learning in general is an often used AI technique.

In the third category of improved reliability by means of AIOps, we have found 3 preprints on arXiv. The oldest work by Salesforce [31] studies AIOps for cloud computing with a partial focus on MSs, where it reviews the use of AI for incident detection, failure prediction and RCA. Moreover, it defines three different data sources for AIOps in cloud computing: (1) metric-based data, (2) heterogeneous log data and (3) traces, uncovering not only the dynamic topological structure of MSs but also generating multi-modal data by combining it with the two previous data sources. The two other surveys [29, 30] focus respectively on failure diagnosis and RCA of MSs. Similarly to [31], they consider metrics, logs, or traces as relevant data sources, but they also consider multi-modal approaches that combine metrics and logs [30]. Our survey pointed to 50 papers that use AI for improving reliability of MSs during the Ops stage (i.e., the"Deploy","Operate","Monitor" phases) and also identified better support for AIOps in MSs as a new trend (cfr. Sect. 5).

Unlike these surveys, we aim to understand the complete panorama of the use of AI in all DevOps phases, without restricting the survey to a specific problem. Thus our survey covers many more primary studies than the existing surveys. As a side effect, we need to remain at a more abstract level in the analysis.

3 Methodology

Our systematic mapping study is based on the guidelines defined by Petersen et al. [5]. We also applied the "snowballing" process defined by Wohlin [32]. In this section, we describe the goal and the research questions (Sect. 3.1), report our search strategy approach and outline the data extraction and the analysis of the corresponding data (Sect. 3.2). The list of selected papers (SPs) is provided as a supplementary material[3] due to space constraints.

3.1 Goal and research questions

As anticipated in the introduction, our goal is to analyze the use of AI techniques to solve the challenges posed by the design, development, and operation of MS systems. Based on it, we first conducted a preliminary study aiming to analyze the trend of the research on AI4MS by performing historical analysis. Specifically, we assessed how the number of AI4MS publications has evolved over the years. Then, we designed the remainder of our study around the following research questions (**RQs**):

[3] https://doi.org/10.6084/m9.figshare.26243993.

RQ$_1$	In which industry domains is AI used for MSs?
RQ$_2$	Which quality attributes are improved by AI4MS?
RQ$_3$	In which DevOps phases is AI4MS applied?
RQ$_4$	What AI techniques are used for realizing AI4MS?
RQ$_5$	What are the open challenges in AI4MS?

The defined research questions focus on the technical and scientific contents of the research. First, **RQ**$_1$ analyses which industry domain the approach targets. We then want to capture why and when AI has been applied, and the answer is typically in terms of improving some quality attribute in the context of some specific phase of the DevOps development life-cycle. Notably, even if the DevOps life-cycle is very common in MSs, we do not intend to disregard approaches based on different life-cycles. Still, they can normally be mapped into subsets of the phases of the DevOps cycle. **RQ**$_2$ focuses on the improved quality attributes, while **RQ**$_3$ discusses the DevOps phase where such improvement occurs. With **RQ**$_4$ we want to investigate what AI techniques have been used in the selected works. The answers to RQs 2-4 will first be discussed separately, and then they will be combined by means of *a multidimensional analysis* that identifies interesting connections between AI techniques (RQ4) (and the rationale for using them) and the combination of quality attributes (RQ2) and DevOps phases (RQ3), i.e., what AI technique is often applied to what quality attributes during which DevOps phases and why. Finally, **RQ**$_5$ highlights the challenges that need to be tackled in future research in the area.

3.2 Search strategy

The search strategy involves the outline of the most relevant bibliographic sources and search terms, the definition of the inclusion and exclusion criteria, and the selection process relevant to the inclusion decision. Our search strategy is depicted in Fig. 2.

Search terms Our search string consists of a bucket of different microservices spellings and a bucket of various AI-related keywords. We arrived at this search string by prototyping several queries and then iteratively refining the most promising candidate. We aimed for a broad coverage, while simultaneously trying to keep the number of false positives low. The concrete search string looks as follows:

("microservic"OR"micro-servic*"OR"micro servic*") AND ("AI"OR"artificial intelligence"OR"machine learning"OR "machine-learning"OR"ML"OR"deep learning"OR"deep-learning"OR "neural"OR"intelligen* learning*")*

Table 2 Inclusion and exclusion criteria

Criteria	Assessment criteria	Step
Inclusion	Papers discussing applications of AI to MSs	Both
Exclusion	Not fully written in English	T/A
	Non peer-reviewed	B
	Books	T/A
	Duplicated	T/A
	Full text inaccessible to us	F
	Out of topic	Both
	Published before [10] (i.e. older than 2014)	B

Bibliographic sources We selected the list of relevant bibliographic sources following the suggestions of Kitchenham and Charters [33] since these sources are recognized as the most representative in the software engineering domain and used in many secondary studies. The list includes: *ACM Digital Library, IEEEXplore Digital Library, Scopus, Google Scholar, Springer link*.

Inclusion and exclusion criteria We defined inclusion and exclusion criteria to be applied to the bibliographic information (B), to title and abstract (T/A), or to the full text (F), or to both the two last items (Both), as reported in Table 2. A main point is that the search, being keyword-based, naturally resulted in extracting both papers about applications of AI to MSs, relevant for our survey, and papers studying the use of MSs for supporting AI (mostly about using MS systems to support the execution of AI applications). We wanted to focus on the first class of papers; hence, the second class was discarded by our inclusion and exclusion criteria.

Search and selection process The search was conducted in September 2023 and included all the publications available until then. The application of the search terms returned 3,991 unique papers.

Testing the applicability of inclusion and exclusion criteria: Before applying the inclusion and exclusion criteria, we tested their applicability [34] on a subset of 50 retrieved papers (each assigned to two authors), randomly selected.

Applying inclusion and exclusion criteria to bibliographic information, title, and abstract: We applied the refined criteria to the remaining 3,941 papers. Two authors read each paper; in case of disagreement, at least one additional author was involved in the discussion to clear up any such disagreement. For 142 papers, we involved more than two authors. Out of the 3,991 initial papers, we included 814 papers based on titles and abstracts. We adopted *adaptive reading depth* [35] for initial inclusion: in case it was unclear from the title and abstract whether the paper was about the use of AI for MSs, we skimmed through the main text to get a more informed opinion. To measure the level of agreement among the authors at this stage, we computed Cohen's Kappa coefficient [36], which resulted in an almost perfect agreement (**0.889**).

Full reading: We fully read the 814 papers included by title and abstract, applying the same criteria defined in Table 2 and assigning each one to two authors. We involved

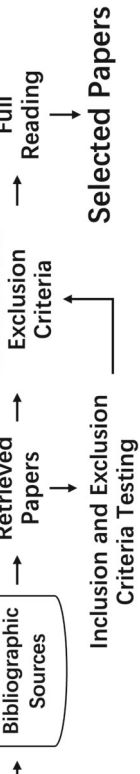

Fig. 2 The search and selection process

Table 3 Results of search and selection

Step	# Papers
Retrieval from bibliographic sources (unique)	3991
Reading by title and abstract (rejected)	3177
Full reading (rejected)	614
Backward and forward snowballing (accepted)	69
Primary studies	**269**

a third author for 55 papers to reach a final decision. Based on this step, we selected 200 papers as relevant contributions. The application of the inclusion and exclusion criteria resulted in an almost perfect agreement (Cohen's Kappa coefficient = **0.839**) [36].

Snowballing: We performed the snowballing process [32], considering all the references presented in the retrieved papers and evaluating all the papers referencing the retrieved ones. We applied the same process as for the retrieved papers. The snowballing search was conducted in January 2024, considering all papers published up to 2023 (papers after September 2023 were indeed only retrived by snowballing). We identified 158 potential papers but only 69 of these were included to compose the final set of publications.

Based on the search and selection process, we retrieved a total of 269 papers for the review, as reported in Table 3.

Quality Assessment: We decided not to perform any further quality assessment, as this is common for systematic mapping studies that want to provide an overview of the research landscape. The only quality control happened through the focus on peer-reviewed publications. Since AI4MS is a very young field, many approaches are also still preliminary, and a too-strict quality assessment may remove papers that are a first attempt towards a promising approach.

Data extraction and replicability: We extracted data from the selected Primary Studies (PSs). The data extraction form, together with the mapping of the information needed to answer each RQ, is summarized in Table 4.

To allow one to trace the data extraction process, we prepared a replication package[4] for this study with the complete results obtained. This would also allow replication and extension of our work by other researchers.

4 Results

4.1 Preliminary analysis on the number of AI4MS publications

Figure 3 shows the evolution of AI4MS publications from 2016 to 2023. We can observe that it took three years from the 2014 blog post by Martin Fowler and James Lewis [10] to start considering the use of AI techniques to support MSs in their

[4] https://doi.org/10.6084/m9.figshare.22663756.

Table 4 Data extraction

RQ	Info	Description
All	Title, authors, DOI, abstract, publication venue	Main information
Preliminary analysis	Year	
RQ1	Industry domain	Domain in which the work has been applied (divided in Level 1 and Level 2)
RQ2	Improved quality attributes	According to ISO25010
RQ3	Improved DevOps phases	
RQ4	AI Model	According to the taxonomy in [6]
RQ5	Future challenges	Future work and challenges

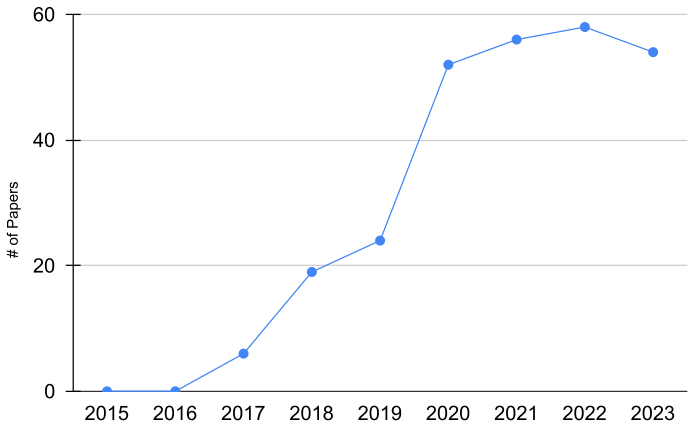

Fig. 3 Number of publications per year

development, deployment, and runtime management. Indeed, the first three years of MS-related research were mainly devoted to understanding the advantages, drawbacks, and potentials of MSs, as outlined, e.g., in [16].

In 2017, MSs were already widespread, with big IT players (e.g., Amazon, Netflix, and Spotify) using them to deliver their core businesses [SP113]. This raised interest in how to better support MSs, and researchers started using AI to realize such support. Since 2017, we indeed have had an ever-increasing trend of AI4MS publications, witnessing a wider and wider recognition of the potential of AI to support MSs.

However, starting in 2020, the trend of sharp increases in AI4MS slowed down, and indeed the numbers of publications through 2020 to 2023 are nearly identical (the small decrease in the last year should be considered with care, since it may be partially due to delays in publication and indexing of some papers).

4.2 RQ1. In which industry domains is AI used for MSs?

To classify the selected studies based on targeted industry domains, we started from the taxonomy of economic sectors defined by AIWatch [6]. The latter enables distinguishing the application of AI to different industry domains. Unsurprisingly, being MSs themselves part of the *information and communication* industry, 255 of the selected studies pertain to such an industry domain, with an ever-increasing trend since 2017 (in line with the results discussed in our preliminary analysis). We also observed a recent interest in using AI for MSs in the *manufacturing* field, with [SP113],[SP115],[SP136] showing that AI is now starting to get used to support MSs in realizing cyber-physical systems for the Industry 4.0 paradigm.

Given that the vast majority of the selected studies pertained to the *information and communication* industry domain, we mapped them to well-known sub-domains.

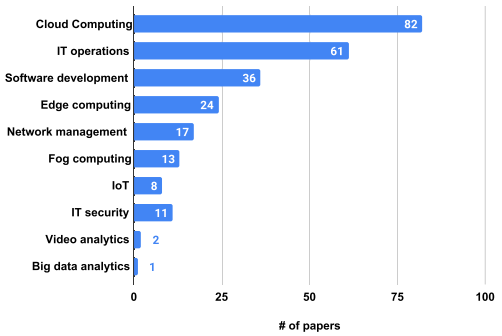

Fig. 4 Sub-domains of *information and communication* where AI is used for MSs

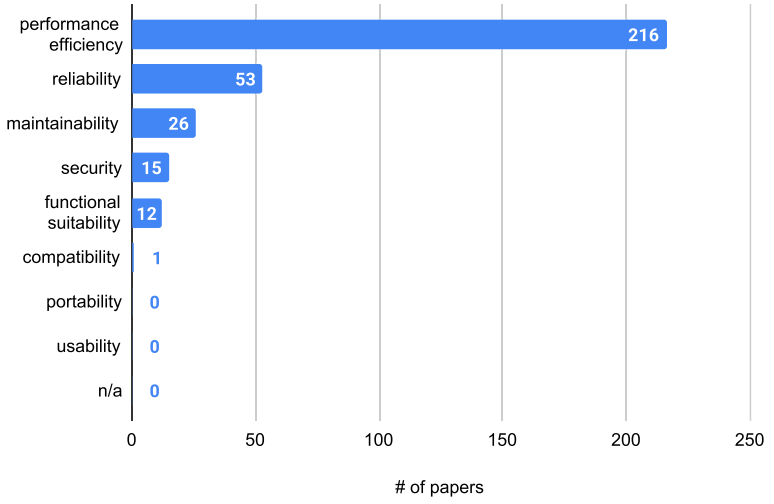

Fig. 5 Microservices quality attributes improved through the use of AI

The result is shown in Fig. 4, from which we observe that AI is mostly used to support MSs in *cloud computing*. This is somewhat expected, as one main advantage of MSs is to enable realizing cloud-native applications [16], which makes *cloud computing* their natural industry sub-domain.

The significant coverage of *edge computing* and *fog computing* aligns with the above considerations. Indeed, edge and fog computing are intended to enable computations to happen closer to IoT, either fully there or by creating a sort of computing continuum from cloud to IoT. This is done by distributing the services forming an application on computing devices that are physically close to Things, also exploiting virtualization, and similarly to what happens in-cloud; however, considering the locality of the computation and the fact that such devices have limited computing resources. For instance, 24 of the selected primary studies illustrate how MSs can be exploited to realize edge applications, as shown in Fig. 4. The figure points out that AI can support MSs in *edge computing* and *fog computing*, e.g., for resource provisioning [SP34],[SP60], MSs' scheduling [SP31], [SP49], or their runtime management [SP1], [SP38].

Another insight follows from the significant coverage of DevOps among selected studies pertaining to the *information and communication* industry domain. Indeed, *software development* and *IT operations* are targeted by 36 and 61 selected studies, respectively. On the *software development* side, AI is mostly used to automate the migration of existing applications to MSs, e.g., [SP30],[SP75], [SP93], [SP106]. On the *IT operations* side, AI is instead used for multiple tasks, e.g., auto-scaling [SP123],[SP129] or fault diagnosis [SP124],[SP132]. This showcases the potential of AI to support the DevOps activities for MSs, with an increasing trend since 2017, making this a promising research direction. Finally, in *IT security*, covered by 11 studies, AI is used to automate intrusion detection, typically based on detecting anomalously behaving MSs, e.g., [SP48],[SP68], [SP80],[SP135]. Despite low numbers, it started being considered only in 2018, with an overall increasing trend since then due to promising results. The use of AI for MSs in the domain of *IT security* hence deserves further investigation.

4.3 RQ2. Which quality attributes are improved by AI4MS?

To classify the papers according to the improved quality characteristics, we used the well-known ISO 25010:2011 standard "Systems and software Quality Requirements and Evaluation" (SQuaRE) [7]. It contains a software product quality model with eight different top-level quality attributes (QAs), i.e., functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, and portability. An attribute was assigned to a paper if the described use of AI was intended to improve this QA. No sub-QAs, e.g., time behavior or capacity for performance efficiency, or QAs outside of ISO 25010:2011, e.g., scalability or observability, were used.

Each of the 269 papers was assigned either 1 or 2 improved QAs, with no paper using AI to simultaneously improve 3 or more QAs. For two QAs, namely portability and usability, we did not find any papers. The distribution of the eight QAs is shown in Fig. 5. As visible, the sample is dominated by *performance efficiency* (216 papers,

∼80%). For most of these papers, the goal was to improve the scalability of MS-based systems, i.e., to increase throughput while simultaneously keeping response times small. AI-powered approaches to achieve this were, e.g., service auto-scaling techniques [SP5],[SP16],[SP54], sophisticated load-balancing [SP6],[SP34], [SP137], or dynamic service placement within a cloud-fog-edge-continuum [SP17],[SP19], [SP32]. Most of these 216 papers focused exclusively on performance efficiency (171, ∼80%). However, several papers also combined this QA with *reliability*: 34 of the 53 papers with reliability also improved performance efficiency (64%). Such papers either explicitly added availability as a targeted QA for their auto-scaling [SP15], scheduling [SP26], or load-balancing [SP35] approach or used AI to reduce service downtime by identifying anomalies and faults [SP22], [SP124], [SP133], [SP230], [SP132], [SP235].

Other quality attributes were less prominent in our sample. A total of 26 papers used AI to improve *maintainability*. These were usually approaches to help with architecting MSs, e.g., by using AI to propose how to decompose a monolithic application into microservices [SP134], [SP244], [SP268], suggesting detailed migration plans [SP30], or AI-powered approaches for architectural runtime adaptation [SP92], [SP95]. Maintainability papers were sometimes paired with performance efficiency, functional suitability, or reliability, but 11 papers also focused exclusively on maintainability. Similarly, 15 papers improved *security*, and all but 5 of these papers did so exclusively. Most of these approaches used AI to identify security-relevant anomalies and malicious behavior, e.g., by analyzing service communication traces [SP20], [SP80], [SP135]. Furthermore, 12 papers improved *functional suitability*. These were usually AI-based approaches for automatically recommending suitable services for composition [SP52], [SP75], [SP120]. Some papers also used natural language models as BERT [SP52] or GPT [SP232] to analyze natural language requirements and to propose suitable microservices based on them.

Lastly, a single paper used AI to improve compatibility [SP266]: in the context of wireless sensor networks, the authors propose a deep learning approach for microservice interoperability to allow dynamic service interactions. AI-based approaches to improve the *portability* or *usability* of microservices did not appear in our sample.

4.4 RQ3. In which DevOps phases is AI4MS applied?

To classify the papers according to the improved software engineering activities, we used the well-known DevOps life-cycle phases presented in Sect. 2.1. The analysis performed is depicted in Fig. 6. We can see that of the 269 papers, only 3 could not be traced to specific DevOps phases, while the rest improved at least a single phase. Unsurprisingly, we can see that none of the works contribute to the *Build* phase, as code compilation does not need AI.

Most of the papers focuses on the adoption of AI techniques for improving the *Operate* phase of MSs. Following the discussion in Sect. 2.1, it is not surprising that the rise of AIOps, pushed approaches on using AI to improve the *Operate* and *Monitor* phases [SP141] [SP225]. The majority of papers addressing the Operate phase focus on the concept of *Scaling* as a central theme. Such emphasis reflects an increasing

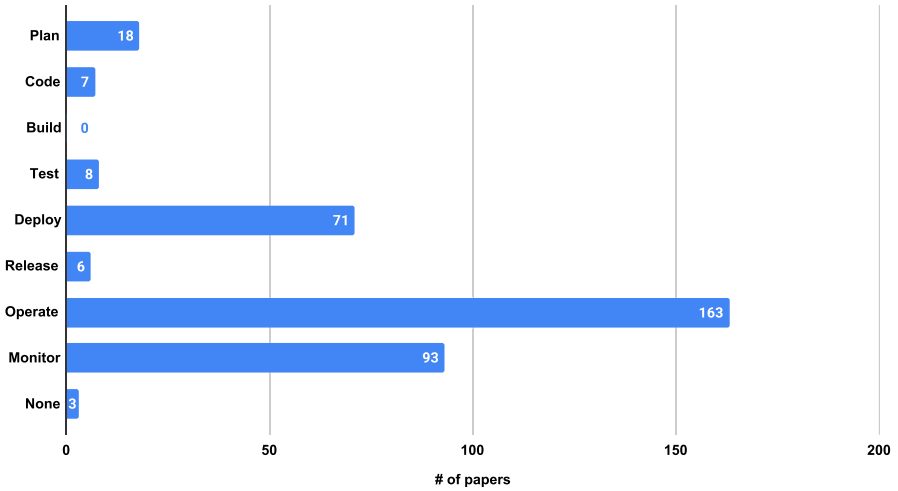

Fig. 6 DevOps phases improved by using AI

necessity to guarantee that systems and solutions can efficiently accommodate augmented workloads, user demands, or data volumes without jeopardising performance or reliability. The main goal of these projects is to use AI technologies to create and improve auto-scaling mechanisms, making the system more intelligent and adaptive. This would ensure the best use of resources while keeping the costs down and avoiding performance issues [SP15] [SP27] [SP35]. Among these techniques, Reinforcement Learning (RL) has gained significant attention in the context of managing scaling in unpredictable and highly variable workloads due to its flexibility and self-improving nature [SP43] [SP201].

Furthermore, we conducted an analysis on the trend of publications regarding the adoption of AI to improve each DevOps phase (shown in Fig. 7). Specifically concerning the three phases with the most publications, i.e., *Operate*, *Monitor* and *Deploy*, we can observe that from 2021 the increase of the number of papers on each of these three phases either decelerate or even decrease in numbers, compared to those of the previous years. Nonetheless, the publications concerning these three phases are still more than the ones considering the other phases.

4.5 RQ4. What AI techniques are used for realizing AI4MS?

To understand the AI technique used in the selected works, we classified them according to the taxonomy in [6]. We used both AI domains, for a coarse-grained analysis (useful, e.g., to understand the time evolution of the field), and keywords and AI subdomains together, for a more detailed analysis. We used AI subdomains together with keywords (and below, for conciseness, we will refer to both of them only as keywords) since it was not always possible to assign a specific keyword to an approach, e.g., since the work uses a family of related techniques. We remark that keywords are not orthogonal, and that a single approach may involve multiple keywords. Indeed, we

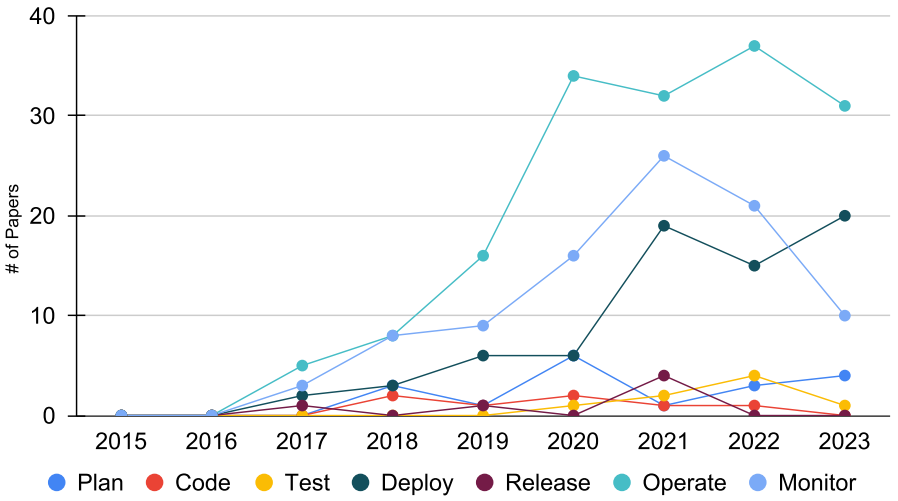

Fig. 7 DevOps phases improved by using AI by years

assigned from one to five keywords per paper. Also, some keywords are pretty general (e.g., neural networks), hence one could naturally expect higher frequencies. However, when multiple keywords would be compatible with an approach, we preferred specific keywords to more general ones.

Results are shown in Fig. 8 for AI domains and in Fig. 9 for keywords. Both the figures just show the frequency of each item. For domains, learning is by far the most frequent, followed by planning and reasoning, with all the other AI domains taking a marginal role. The relevance of learning is confirmed by looking at keywords, with many keywords in the domain having a high frequency. This is expected since such approaches are useful to tackle a number of software engineering problems, and MSs are no exception. Inside the domain, there is no clear winning approach, with various keywords scoring high, and the top places being taken by more general keywords (neural network, deep learning, reinforcement learning, etc.). Apart from these, the single keyword which scores highest is optimization from the planning domain, which finds obvious applications to find the best configurations to optimize relevant QAs. This is in line with the observation of Sect. 4.3 that the most considered QA is performance efficiency which, being quantitative, can benefit from optimization. Indeed, a number of works deal with optimization for various aspects of performance efficiency. This is for instance the case of [SP139] which tackles optimization of task scheduling in mobile Cloud computing, of [SP169] which considers application placement and migration in the Cloud-IoT continuum, and [SP263] which deals with deployment and startup of microservice instances in resource centres. However, optimization can also be used for other QAs, e.g., it is used in [SP156] to find microservice candidates in the refactoring of legacy systems into microservice architectures to optimize maintainability metrics such as feature modularization and reuse. Another frequent keyword is anomaly detection (from the learning domain), suitable for highlighting anomalous

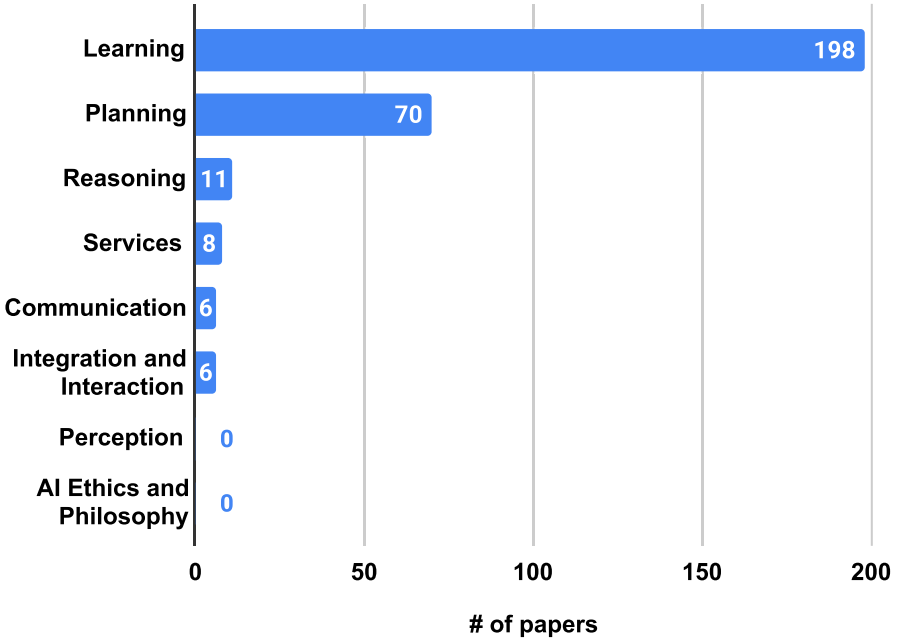

Fig. 8 AI domains of techniques applied to MSs

behaviors that need to be managed. Interestingly, anomalies are mostly related to performance efficiency, frequently paired with reliability [SP12], [SP22], but some of them are also related to security [SP158]. Indeed, [SP12] shows that by optimizing the performance of serverless systems by reducing the number of cold starts of functions, one is also able to reduce the number of failed calls. Instead, [SP158] looks for anomalies in logs of API invocations to highlight data breaches and DoS attacks. There are a few works in the domain of communication, which used to be focused on natural language understanding, such as [SP160] where it is used to extract information from user specifications. However, in 2023 the first work exploiting large language models (chatGPT in the specific case) for microservices [SP232] appeared. We expect such line of work to get considerable attention in the next years.

An analysis of the evolution of AI domains over the years is actually meaningful only for learning and planning, since the other domains have low frequencies. Learning had a relevant growth and is now essentially stable, hence the approach is probably reaching maturity. A similar trend is also visible for planning, albeit the growth ended up earlier. A deeper analysis of the involved works reveals that planning is only applied during Ops phases, while learning is applied to both Dev and Ops phases, therefore suggesting better compatibility of ML with the entire DevOps lifecycle.

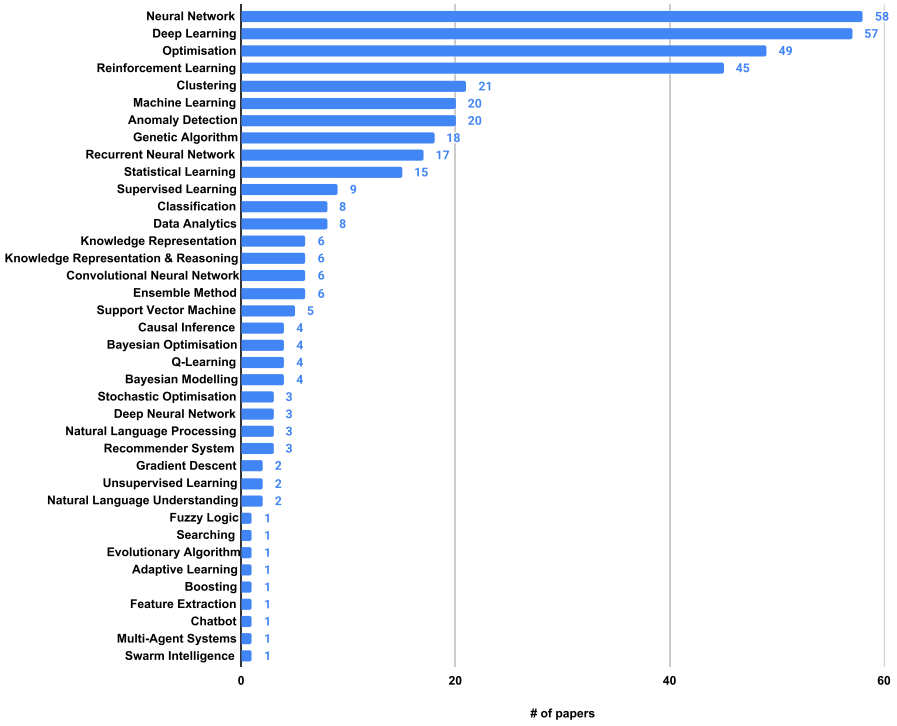

Fig. 9 AI keywords related to approaches applied to MSs

4.6 RQ5. What are the open challenges in AI4MS?

Among the 269 primary studies, 142 do not present a clear future challenge. When categorizing the challenges in the other studies, we identified different categories answering the questions *What next?* or *How?* We report them in Fig. 10.

Therein, *Resource Optimization* and *AI Precision* are the two aspects raising the most concern as future challenges in the AI4MS domain. The two aspects are mentioned in 25 and 21 papers, respectively. Combining such information shows that in most works, the authors are focused on presenting a minimum viable product (MVP) and leaving the optimization, mostly in the form of automation and resource optimization, for future challenges. The other important future challenges that have a relatively high number of mentions include *Automation*, *Efficiency*, *Validation*, *Adaptability*, and *AI models*. Summarizing, future directions focus on improving the AI models and introducing automation mechanisms.

Regarding the trend of the proposed future challenges (shown in Fig. 11), each of the top themes, e.g., *Resource Optimization* and *Automation*, have sharp increases in number of papers from 2019 to 2021. However, these topics decreased in numbers from 2021 to 2022, hence such challenges had received further investigation. Vice versa, other challenges have caught the attention of the researchers. This includes *AI precision*, *Validation*, and *Efficiency*, whose frequencies increased from 2021 to 2022.

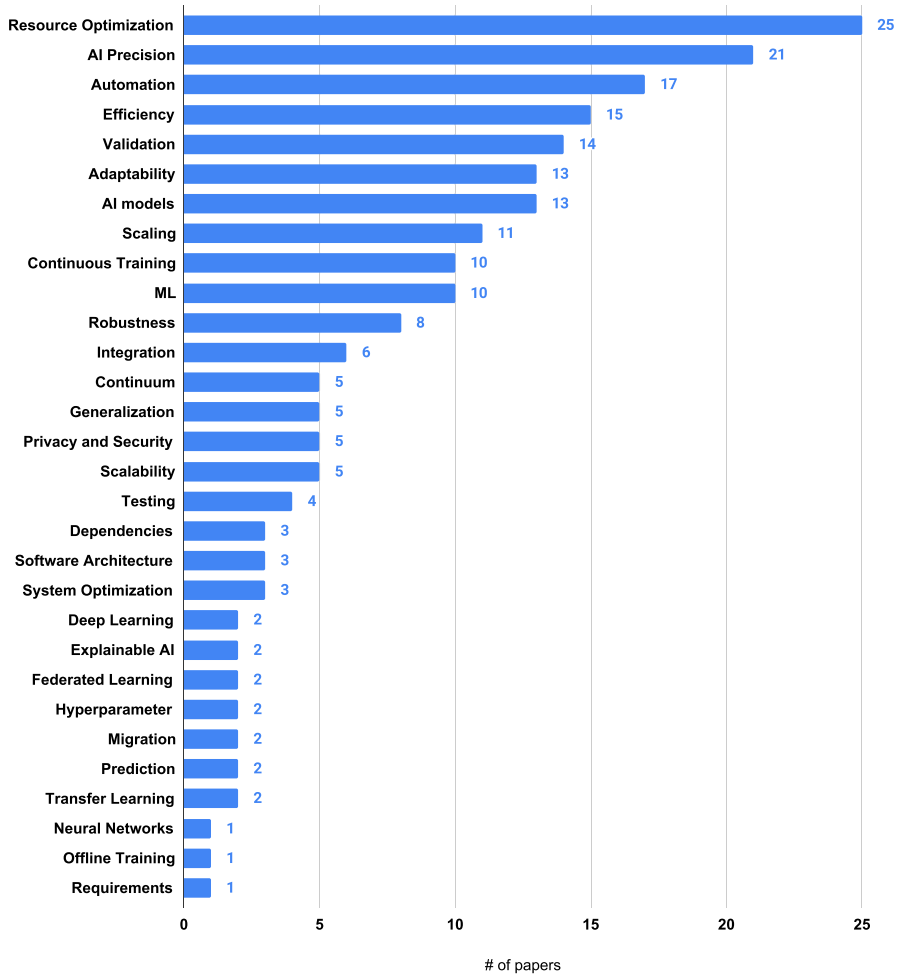

Fig. 10 Future challenges

5 Multidimensional analysis

While the data for each RQ can provide relevant insights individually, we also analyzed the combined data of several RQs for additional depth. Table 5 shows the weight of research for each combination of DevOps phase and QA. For each combination, we studied the extracted rationale and AI techniques to identify commonalities in research topics and approach. We found that distinct themes appear in the Dev stage (Plan, Code, Test, Release), Ops stage (Deploy, Operate, Monitor) and the full DevOps lifecycle. We identified 4 themes in the Dev stage (see Fig. 13), 10 themes in the Ops stage (see Fig. 14) and 2 performance efficiency themes in the full DevOps lifecycle (see Fig. 12).

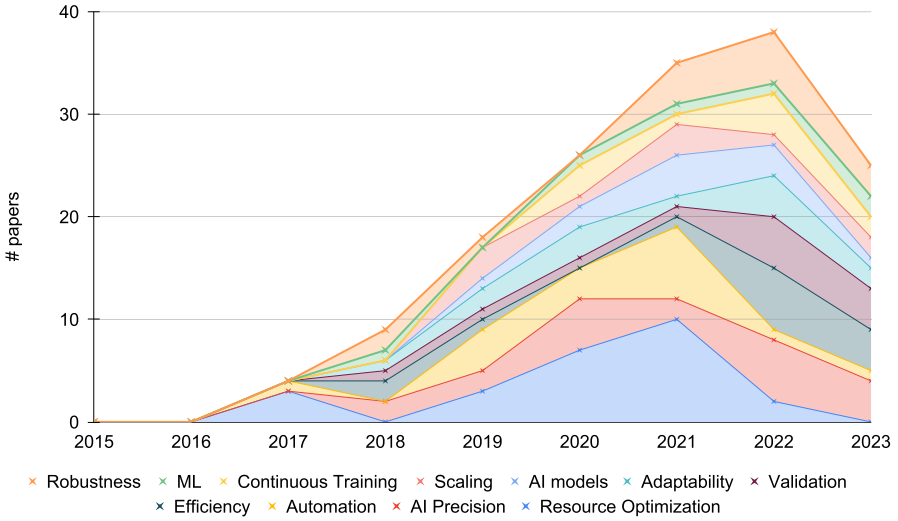

Fig. 11 Future challenges per year

Table 5 This table shows the amount of papers that cover a particular QA and DevOps phase. Papers that address multiple QAs or DevOps phases are counted multiple times

	Operate	Monitor	Deploy	Plan	Test	Code	Release
Performance efficiency	147	69	69	4	4	2	6
Reliability	26	25	9	0	3	0	0
Maintainability	6	7	3	12	1	3	0
Security	6	10	2	1	0	0	0
Functional suitability	2	2	0	3	2	4	0
Compatibility	1	0	0	0	0	0	0

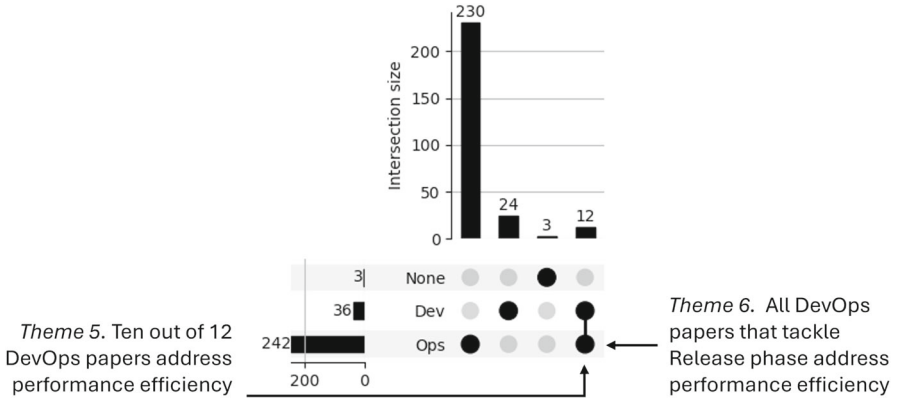

Fig. 12 An Upset plot of the amount of papers addressing the Dev stage, the Ops stage or both stages

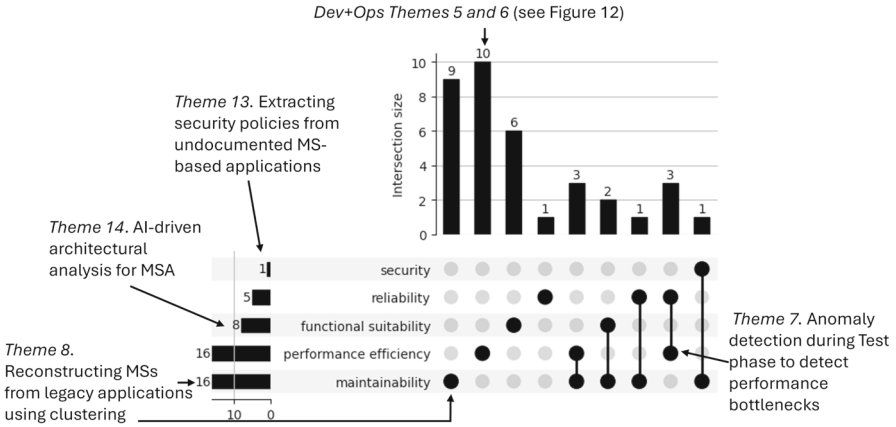

Fig. 13 An Upset plot of the amount of papers addressing a specific intersection of QAs during the Dev phases (Plan, Code, Test, Release)

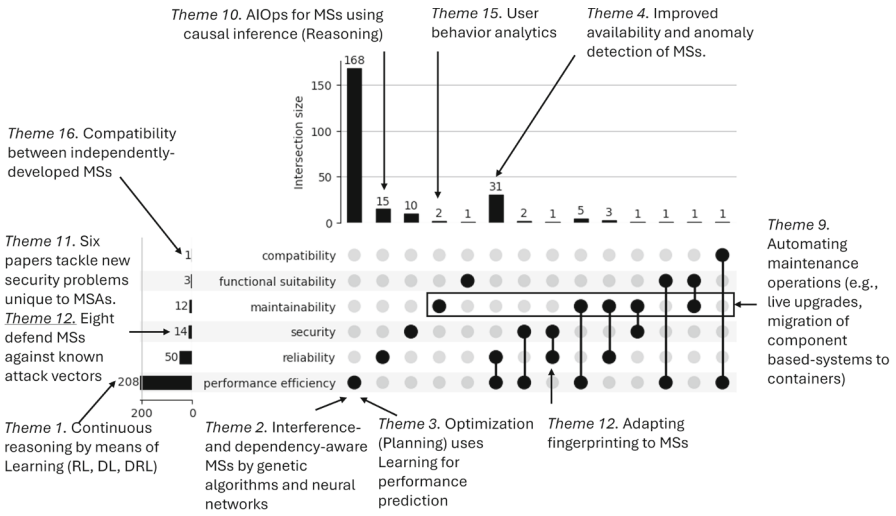

Fig. 14 An Upset plot of the amount of papers addressing a specific intersection of QAs during the Ops phases (Deploy, Monitor, Operate)

We present these 16 themes per QA and then per Ops (Deploy, Operate, Monitor) and Dev (Plan, Test, Code, Release) stage.

Performance efficiency: As shown by Table 5 and Fig. 14, the bulk of papers improve *performance efficiency during the Deploy, Operate and Monitor phases*. We identified several themes by studying the extracted rationale and AI techniques of these papers.

Theme 1: ML-based continuous reasoning. Some AI techniques are very suitable for continuous reasoning on vast data dimensions and data sizes (e.g. analysis of QoS

parameters, service and resource parameters), hereby also optimizing service selection, resource allocation and service placement strategies at run-time [SP141]. Moreover, [SP169], [SP174] and [SP193] underscore the necessity for continuous reasoning. As stated by [SP141], RL and DL are often used for this purpose, and combined into Deep Reinforcement Learning (DRL) because of their respective ability of dynamic decision-taking and automated feature acquisition.

Theme 2: Interference- and dependency-aware scheduling of MSs using genetic algorithms and neural networks. Many performance-only papers argue that during resource allocation and service placement, there is a need to account for (i) resource or availability interference between MSs [SP16], [SP49], [SP240], [SP263], very often using a genetic algorithm and (ii) inter-dependencies and associated call graphs [SP21], [SP128], [SP223], [SP206], typically using a neural network. E.g., [SP240] is a novel example of resource interference; it improves the performance efficiency of the Kubernetes scheduler with a genetic algorithm that places containers with shared library dependencies on the same node, hereby reducing resource usage due to the container library sharing mechanism. As an example of dependency management, [SP128] feeds into the aforementioned continuous reasoning capabilities a neural network to predict the performance impacts and back log pressures that different MS inter-dependencies may cause in a production cloud system.

Theme 3: Optimization uses ML for performance prediction. As stated in Sect. 4.5, optimization from the Planning AI domain is massively used for improving performance efficiency of MSs. Optimization typically relies on the AI Learning techniques such as neural networks to predict the performance of particular configurations and/or resource allocation parameters in the search space. Specific to MSs, the integration of optimization techniques with these ML models addresses load balancing, resource utilization, autoscaling, risk management, and energy efficiency, ultimately leading to more robust and efficient cloud and microservice architectures.

Theme 4: Improved availability and anomaly detection of MSs. As already elaborated in Sect. 4.3, 29 out of 34 papers that tackle both performance efficiency and reliability, improve availability either by using multi-objective optimization methods or reducing service downtime by means of anomaly detection, and do so exclusively during the Monitor or Operate phases.

With respect to *performance efficiency during the early Dev phases* (Plan, Code, Test, Release), the following themes could be identified.

Theme 5: Full DevOps lifecycle approaches mainly address performance efficiency. Out of 16 papers that address performance efficiency during the Dev stage, 10 papers address both Dev and Ops stages hereby representing 84% of all the 12 papers that cover both stages (cf. Figure 12).

Theme 6: DevOps papers that tackle the Release phase all address performance efficiency. No other QA than performance efficiency is addressed during the Release phase (cf. Table 5). We found 6 papers that focus on the Release phase while also addressing the entire Ops stage (e.g.,"Release, Deploy, Operate, Monitor"), i.e. representing 60% of all the performance efficiency papers covering both stages. This consistent grouping of Release phase and the entire Ops stage reflects a streamlining of the CI/CD process where the release of code is closely followed by its deployment to production systems, supporting faster delivery cycles. Then there are 10 performance

papers in the other Dev phases, 4 of which focus on performance efficiency only, 3 on the combination of *reliability and performance efficiency*, and 3 on *maintainability and performance efficiency*.

Theme 7: Anomaly detection during the Test phase. As a common rationale, we could identify that 2 out of 3 *reliability and performance efficiency* papers use anomaly detection during the Test phase to find performance bottlenecks in MS-based applications.

Maintainability: *Theme 8: Reconstructing MSs from legacy applications using clustering.* All the *maintainability and performance efficiency* papers focus on optimizing the re-partitioning of monolithic applications into MSs to achieve the best performance. [SP156] and [SP214] implement such re-partitioning during the Plan phase, whereas [SP134] provides a completely automated system for web applications that covers the Code, Deploy and Operate phases. It decomposes the application code into MSs, and deploys and auto-scales these with performance efficiency in mind. Similar findings can be drawn for *all maintainability papers that tackle the early Dev phases* (with or without a performance requirement). Out of 16 such papers, reconstructing MSs from legacy applications is the topic of 14 papers. However, similar work as the aforementioned [SP134] that covers Dev and Ops phase does not exist. There are only two other approaches [SP269], [SP120] that are executed during the Code phase, but they do not generate code artifacts. Finally, as already noted by Saucedo et al. [28], clustering and unsupervised ML in general is commonly used as the primary technique for supporting migration from monolithic applications to MSs. This is because these techniques allow inferring useful results from existing data sources without needing to label training data with explicit features based on prior or privileged knowledge [SP213]. Existing data sources include network metadata [SP213], syntactic and semantic properties of object-oriented programs and databases [SP269], [SP243], [SP245], and logs of non-functional metrics for determining appropriate units of resource allocation and service scaling [SP244], [SP134].

Theme 9. Automating maintenance operations during Ops stage. With respect to *maintainability during the later Ops phases*, 10 out of 12 papers *combine the maintainability QA with another QA*, hereby employing a wide range of AI techniques. The common rationale that binds this work is automating complex tasks and reducing manual intervention, with a particular focus on better performance efficiency (5 out of 10) or reliability (3 out of 10) of MSs.

Reliability: *Theme 10: AIOps for MSs using causal inference.* Out of 50 approaches that focus on *reliability* during the Ops phases, there are only 15 papers that focus exclusively on reliability, but for these papers, improving AIOps for MSs is a common trend in the extracted rationale, especially for papers published in 2023. AIOps is an approach to collect, analyze, and detect patterns in cloud and infrastructure data, thus predicting future usage, failures, and improving the management and resilience of complex IT environments [SP142],[SP125], [SP227]. Unique to MSs in AIOps is the use of neural networks, DL and causal inference to handle the complexity and dynamism of MSAs, which involve numerous interdependent services with complex spatial states and hundreds of metrics. Specific topics that are frequently handled include (1) log-based anomaly detection and fault localization [SP151], [SP157], [SP239], (2) selecting appropriate metrics as features in supervised ML [SP197],

(3) causal dependency learning to observe error propagation [SP142], [SP196], and (4) proactive and self-learning systems that become stronger upon faults rather than deteriorating [SP138], [SP195], [SP97]. Interestingly, causal inference from the Reasoning domain appears to be better than ML in this space [SP65]. As an example, both [SP196] and [SP142] use a specific technique called interventional causal learning.

Security: We identified two research themes for the papers that focus on security in the Ops stage. Note that for 12 out of 14 papers, the security improvements are implemented during *the Monitor or Operate phases*.

Theme 11: Tackling new security problems unique to MSAs. One trend is the detection and mitigation of new security anomalies and attacks that are due to unique properties of MSAs. There are 2 papers that use neural networks to profile the abnormal application-level behavior of MSs from highly distributed, heterogeneous and unstructured data [SP48], [SP165]. Other papers use AI to implement self-adaptive anomaly detection solutions that can cope with the dynamic and evolving nature of microservice environments [SP20], [SP158] or that can be automatically applied to different applications [SP160]. Finally, [SP148] presents a DRL-based scheduler to reduce lateral movement of attackers across the network of a Kubernetes cluster. The scheduler aims to determine subsets of applications that have similar microservice call chains and then exclusively place containers of the identified applications together on set of nodes in line with the identified chain patterns.

Theme 12: Defending MSs against known attack vectors. The bulk of the security papers focus on defending MSs against known attack vectors or adapting existing defenses to MSs. Attack vectors include data breaches and denial-of-service attacks [SP158], malicious threat patterns and zero-day vulnerabilities in containers [SP10], IoT network attacks [SP135], password guess attacks [SP231]. Adapted security defenses include fingerprinting [SP89].

Theme 13. Extracting security policies from undocumented MS-based applications. We only identified 1 paper that targets *security during the Dev stage* [SP213]. This work provides a three-fold mechanism operating during the *Plan* phase: first, a reconstruction of inter-MS interactions from undocumented MS-based applications is performed; then the extracted interactions are classified as normal or abnormal; and finally appropriate access control policies are defined accordingly.

Less Prevalent QAs: Although there are not so many *functional suitability* papers, we could distill a common research theme.

Theme 14: AI-driven architectural analysis for MSAs. For the *Dev* stage, all 8 functional suitability papers support AI-driven architectural analysis of MSAs. These papers contribute with approaches that help software designers identify suitable boundaries and granularity for MSs using various AI techniques such as Natural Language Processing [SP56] [SP120], clustering [SP247], generative AI [SP232], DL [SP83], and recommender systems [SP75]. Another work supports test case prioritization by quantifying the invocation weight of MSs using a recommender system based on Page Rank [SP183]. Finally, there is an agile approach that operates at the intersection of functional suitability and maintainability to quickly grasp the impact of new requirements on a code base using classification and neural networks [SP7].

Theme 15: User behavior analytics. There are only two pure functional suitability papers that tackle the *Ops* phase. Both papers support user behavior analytics by monitoring user interactions with MSs [SP168] [SP146].

Theme 16: Compatibility between independently developed MSs during Operate. The single paper addressing *compatibility* during the *Operate* phase improves the composition and co-existence of independently developed MSs [SP266].

6 Discussion

Figure 3 clearly shows the growing application of AI in the field of MSs. Significantly, most publications in AI4MS-despite having its roots in the industry-involve academics. Our decision to only consider peer-reviewed publications may be pertinent in this regard given that businesses typically present their outcomes through speeches and blog posts rather than peer-reviewed papers.

The most used AI techniques are ML and its different incarnations, but also more specific techniques such as optimization and anomaly detection are heavily used.

The primary use of AI until now has been to increase performance efficiency not just in the Operate phase but also during monitoring and deployment. Reliability is typically taken into account during the same phases as performance efficiency and has led to improved AIOps for MSs. Instead, security-related methods concentrate on the monitoring stage. Significantly, almost all of the techniques emphasize the Ops phases, with almost none focusing on the Dev phases (no approach at all considers Build). However, a notable exception to this concerns maintainability papers that implement automated migration of legacy applications to MSs during the Dev stages. Reference [SP134] is however the only automated tool for automatic refactoring of a monolithic artifact into code for MSs and Ops artifacts for service placement and resource allocation, but this approach is limited to web applications.

Our results highlight a few gaps in the literature. To put the list below into context, consider that the first peer-reviewed works on AI4MS were published only in 2017; therefore, research concentrated on the simplest problems. Also, the newest trends may currently be considered only in industry, hence not yet disseminated via refereed publications. The main identified gaps are described hereafter.

AI in Dev phases Most approaches focus on Deploy, Operate and Monitor (cf. Fig. 6). We expect AI to be able to play a major role also in the Plan (e.g., automatic requirement analysis), Code (e.g., automatic refactoring tools that also generate deployment artifacts) and Test (e.g., automatic test case generation) phases.

Portability, compatibility and usability As shown in Sect. 4.3, no research in our sample aims to improve these QAs, but for a single paper targeting compatibility. Several open research questions can be derived from this observation. First, while MSs are inherently more portable and interoperable (the 1st sub-characteristic of compatibility [7]), it is not known whether AI techniques such as natural language processing, expert systems and generative AI (see below for a more detailed discussion of generative AI) can improve the level of automation in vendor-agnostic model-driven configuration methods such as TOSCA [37]. Second, coexistence (the 2nd sub-characteristic of compatibility [7]), which can be interpreted as the desire to reduce dysfunctional emergent

behavior, caused by feature interactions between MSs, can definitively be improved by means of several AI techniques such as (a) genetic and evolutionary algorithms, (b) multi-agent systems, reinforcement learning, or a combination of both, (c) anomaly detection. Third, while AI-assisted selection of MSs for improving functional suitability has been marginally studied (e.g., [SP56]), there is a lack of understanding of to which extent AI-assisted user interface design must be done differently in the era of MSs.

AI for security While security is nowadays a main concern, it is clear from Fig. 5 that only a few works consider it, and they are concentrated in the monitoring and operate phases, to detect anomalies or ongoing attacks. We believe AI can contribute much more to tackling security issues in MSs, and such contributions can take place in most phases. Also, as discussed in Sect. 4.3 approaches for security do not improve other QAs, hence security calls for dedicated approaches and techniques.

Generative AI for DevOps A relevant instance of the use of AI in Dev phases, and in particular in the Code phase, concerns exploiting generative AI such as ChatGPT to assist programmers in code writing. Such approaches have born recently since ChatGPT was released towards the end of 2022, and are starting to be applied to MSs as well.[5] We believe such a research direction will gain interest in the future, hence we expect AI keywords such as Chatbot, occurring only in one of the primary studies we consider [SP232], will gain emphasis. Similar approaches could also be used to generate other artifacts, such as specifications, tests or documentation, and more in general to provide a natural language interface to tools. As for Ops, continuous monitoring, anomaly detection, and self-healing might be dominated by self-learning and generative AI tools in the near future.

Explainability Most of the surveyed AI techniques are not "explainable by design", meaning that, despite they support MSs in their DevOps life-cycle, they are not providing explanations of why this is the case. AI techniques can indeed be used to, e.g., determine performance/functional anomalies, identify the root causes of failures, or detect security leaks/intrusions. At the same time, associating identified issues to why they are considered so would help DevOps engineers in troubleshooting them and patching MSs to avoid such issues to happen again in the future, also focusing only on true positives. This hence calls for AI techniques that support MSs in their DevOps life-cycle while also being "explainable by design", much in the same way as the need for explainability is nowadays recognized in AI [38]. Among primary studies, [SP80] and [SP128] studies correspond to explainable AI.

7 Threats to validity

The results of an SMS may be subject to validity threats, mainly concerning the correctness and completeness of the survey. We follow the guidelines for identifying the threats to validity in secondary studies in the software engineering domain proposed by Ampatzoglou et al. [39]. We discuss them below.

[5] https://frends.com/video/creating-a-microservice-with-chatgpt-in-2-minutes.

Study selection validity In this study, we strictly follow the established and commonly accepted SMS guidelines in terms of the search strategy, review protocol, and the data extraction process [33]. By doing so, we significantly reduced the threats to the initial search and study filtering processes in the secondary study planning phase. To do so, the search string was formulated to include keywords identified from research questions and diversified using synonyms. However, though most of the publications are covered by the initial search, potential limitations on the search string may still evoke issues, which results in missing key studies. To mitigate the search limitations and extend the coverage of studies, we conducted snowballing, where we reviewed all the references listed in the selected studies and all the papers that reference the selected ones. Snowballing was recursively applied to papers coming from snowballing as well. As it was likely that the snowballing activity could continue for an excessively long period, the snowballing activity ceased at the end of January 2024. The inclusion and exclusion criteria were defined to assist the study selection. The criteria aligned with the paper's goal and research questions and the guidelines recommended by Petersen et al. [5]. The selection process prescribed that at least two authors conducted the study selection independently, with a third author involved in the discussion to resolve any disagreement.

Data validity The data extraction process is a similar procedure where two authors conducted an iterative analytic process driven by the open coding method to identify the classification schema. For certain categories, we adopted publicly available standards. For example, to answer RQ2, we adopt the ISO/IEC 25010 software quality model, which is commonly acknowledged as the cornerstone of a product quality evaluation system and determines the quality characteristics considered when evaluating software quality. By adopting such open standards, we shall avoid potential disagreement and bias, as well as guarantee the correctness of the collected data. For the data analysis process, thanks to the pre-defined categories, the extracted results can be easily summarized and displayed in the form of bar charts. On the other hand, publication bias is also a potential threat to data validity, where methods, techniques, and usage goals from companies are not included sufficiently due to the focus on peer-reviewed papers as well as confidential policies. Such a perspective can be further investigated by analyzing grey literature and industrial surveys in future studies.

8 Conclusion

In this paper, we conducted a systematic mapping study on the use of AI in the life-cycle of MS systems. Based on the selected 269 primary studies, we focus on understanding, in the area of MSs, which AI technologies are used, in which domain and according to which rationale, namely which software quality attributes the AI technologies aim to improve, and in which DevOps phases.

The results show that AI4MS is a trendy area, with increasing numbers of studies in many application areas. The main outcomes are: 1) while the main application area is, of course, IT, manufacturing is also starting to attract interest; 2) the main rationale is improving performance efficiency and reliability in Ops phases, while surprisingly Dev phases are rarely considered, and QAs such as Portability and Usability are not

considered at all; 3) current research focuses on building the minimum viable product showcasing some approach, with optimization and automation left for future work; 4) a multi-dimensional analysis identifies 16 research themes that include among others the use of deep reinforcement learning for performance efficiency, AIOps for MSs, tackling new security problems unique to MSAs, and adapting existing security techniques to MSs.

This paper provides insights on AI4MS, by keeping the discussion at a high level mainly due to the quite considerable amount of currently available/selected studies. Future work will include delving more into the details of sub-areas of AI4MS, which can be achieved by selecting subsets of the already selected studies based on additional selection criteria. In particular, we plan to analyse how AIOps and MLOps are currently used in the life-cycle of MSs. Other than going into the details, by narrowing the focus to AIOps and MLOps, it would become manageable to complement our analysis of peer-reviewed literature with grey literature, to shed light on both the state-of-the-art and state-of-practice on the topic. This is part of our future work.

On another front, we plan to complement the results presented in this study by analysing the dual situation, namely how MSs are used to support the design, development, and operation of AI systems.

Appendix A Primary Studies

SP1 S. Taherizadeh, V. Stankovski, M. Grobelnik, A capillary computing architecture for dynamic internet of things: Orchestration of microservices from edge devices to fog and cloud providers, Sensors 18 (2018).

SP2 J. Lv, M. Wei, Y. Yu, A container scheduling strategy based on machine learning in microservice architecture, in: 2019 IEEE International Conference on Services Computing (SCC).

SP3 Y. Li, L. Chen, D. Zeng, L. Gu, A customized reinforcement learning based binary offloading in edge cloud, in: 2020 IEEE 26th International Conference on Parallel and Distributed Sys- tems (ICPADS).

SP4 B. Magableh, M. Almiani, A deep recurrent q network towards self-adapting distributed microservice architecture, Software: Practice and Experience 50 (2020).

SP5 A. Goli., N. Mahmoudi., H. Khazaei., O. Ardakanian., A holistic machine learning-based autoscaling approach for microservice applications, in: Proceedings of the 11th International Conference on Cloud Computing and Services Science - CLOSER,, INSTICC, SciTePress, 2021.

SP6 J. Cui, P. Chen, G. Yu, A learning-based dynamic load balancing approach for microservice systems in multi-cloud environment, in: 2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPA-DS).

SP7 D. Russo, V. Lomonaco, P. Ciancarini, A machine learning approach for continuous development, in: P. Ciancarini, S. Litvinov, A. Messina, A. Sillitti, G. Succi (Eds.), Proceedings of 5th International Conference in Software Engineering for Defence Applications, Springer International Publishing, Cham, 2018.

SP8 M. Caporuscio, M. De Toma, H. Muccini, K. Vaidhyana-than, A machine learning approach to service discovery for microservice architectures, in: S. Biffl, E. Navarro, W. Löwe, M. Sirjani, R. Mirandola, D. Weyns (Eds.), Software Architecture, Springer International Publishing, Cham, 2021.

SP9 G. S. Siriwardhana, N. De Silva, L. S. Jayasinghe, L. Vithanage, D. Kasthurirathna, A network science-based approach for an optimal microservice governance, in: 2020 2nd International Conference on Advancements in Computing (ICAC), volume 1.

SP10 S. Kamthania, A novel deep learning rbm based algorithm for securing containers, in: 2019 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE).

SP11 M. G. Khan, J. Taheri, M. A. Khoshkholghi, A. Kassler, C. Cartwright, M. Darula, S. Deng, A performance modelling approach for sla-aware resource recommendation in cloud native network functions, in: 2020 6th IEEE Conference on Network Softwarization (NetSoft).

SP12 C. Liu, Z. Cai, B. Wang, Z. Tang, J. Liu, A protocol-independent container network observability analysis system based on ebpf, in: 26th IEEE International Conference on Parallel and Distributed Systems, ICPADS 2020, Hong Kong, December 2-4, 2020, IEEE, 2020.

SP13 S. Agarwal, M. A. Rodriguez, R. Buyya, A reinforcement learning approach to reduce serverless function cold start frequency, in: 2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid).

SP14 Z. He, P. Chen, X. Li, Y. Wang, G. Yu, C. Chen, X. Li, Z. Zheng, A spatiotemporal deep learning approach for unsupervised anomaly detection in cloud systems, IEEE Transactions on Neural Networks and Learning Systems (2020).

SP15 L. Toka, G. Dobreff, B. Fodor, B. Sonkoly, Adaptive AI-based auto-scaling for Kubernetes, in: 2020 20th IEEE/ ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID).

SP16 N. Cruz Coulson, S. Sotiriadis, N. Bessis, Adaptive microservice scaling for elastic applications, IEEE Internet of Things Journal 7 (2020).

SP17 K. Fu, W. Zhang, Q. Chen, D. Zeng, M. Guo, Adaptive resource efficient microservice deployment in cloud-edge continuum, IEEE Transactions on Parallel and Distributed Systems 33 (2022).

SP18 R. A. Addad, D. L. C. Dutra, T. Taleb, H. Flinck, Ai-based network-aware service function chain migration in 5 g and beyond networks, IEEE Transactions on Network and Service Management 19 (2022).

SP19 H. Sami, H. Otrok, J. Bentahar, A. Mourad, AI-Based Resource Provisioning of IoE Services in 6 G: A Deep Reinforcement Learning Approach, IEEE Transactions on Network and Service Management 18 (2021).

SP20 M. -O. Pahl, F. -X. Aubet, All eyes on you: Distributed multi-dimensional iot microservice anomaly detection, in: 2018 14th International Conference on Network and Service Management (CNSM), 2018.

SP21 X. Hou, C. Li, J. Liu, L. Zhang, S. Ren, J. Leng, Q. Chen, M. Guo, Alphar: Learning-powered resource management for irregular, dynamic microservice

graph, in: 2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS).

SP22 M. Jin, A. Lv, Y. Zhu, Z. Wen, Y. Zhong, Z. Zhao, J. Wu, H. Li, H. He, F. Chen, An anomaly detection algorithm for microservice architecture based on robust principal component analysis, IEEE Access 8 (2020).

SP23 Y. Zuo, Y. Wu, G. Min, C. Huang, K. Pei, An intelligent anomaly detection scheme for micro-services architectures with temporal and spatial data analysis, IEEE Transactions on Cognitive Communications and Networking 6 (2020).

SP24 S. Nedelkoski, J. Cardoso, O. Kao, Anomaly detection and classification using distributed tracing and deep learning, in: 2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID).

SP25 Q. Du, T. Xie, Y. He, Anomaly detection and diagnosis for container-based microservices with performance monitoring, in: J. Vaidya, J. Li (Eds.), Algorithms and Architectures for Parallel Processing, Springer International Publishing, Cham, 2018.

SP26 M. Lin, J. Xi, W. Bai, J. Wu, Ant colony algorithm for multi-objective optimization of container-based microservice scheduling in cloud, IEEE Access 7 (2019).

SP27 G. Baye, F. Hussain, A. Oracevic, R. Hussain, S. Ahsan Kazmi, Api security in large enterprises: Leveraging machine learning for anomaly detection, in: 2021 International Symposium on Networks, Computers and Communications (ISNCC).

SP28 A. U. Gias, G. Casale, M. Woodside, Atom: Model-driven autoscaling for microservices, in: 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS).

SP29 I. Prachitmutita, W. Aittinonmongkol, N. Pojjanasuksakul, M. Supattatham, P. Padungweang, Auto-scaling microservices on iaas under sla with cost-effective framework, in: 2018 Tenth International Conference on Advanced Computational Intelligence (ICACI).

SP30 N. Chondamrongkul, J. Sun, I. Warren, Automated planning for software architectural migration, in: 2020 25th International Conference on Engineering of Complex Computer Systems (ICECCS), 2020.

SP31 A. Samanta, Y. Li, F. Esposito, Battle of microservices: Towards latency-optimal heuristic scheduling for edge computing, in: 2019 IEEE Conference on Network Softwarization (NetSoft).

SP32 S. B. Nath, S. Chattopadhyay, R. Karmakar, S. K. Addya, S. Chakraborty, S. K. Ghosh, Containerized deployment of micro-services in fog devices: a reinforcement learning-based approach, The Journal of Supercomputing 78 (2021).

SP33 R. C. Chiang, Contention-aware container placement strategy for docker swarm with machine learning based clustering algorithms, Cluster Computing (2020).

SP34 M. De Sanctis, H. Muccini, and K. Vaidhyanathan. Data-driven adaptation in microservice-based iot architectures, in 2020 IEEE International Conference on Software Architecture Companion (ICSA-C). IEEE, 2020.

SP35 R. Yu, S.-Y. Lo, F. Zhou, G. Xue, Data-driven edge resource provisioning for inter-dependent microservices with dynamic load, in: 2021 IEEE Global Communications Conference (GLOBECOM).

SP36 N.-M. Dang-Quang, M. Yoo, Deep learning-based autoscaling using bidirectional long short-term memory for kubernetes, Applied Sciences 11 (2021).

SP37 R. Li, M. Du, H. Chang, S. Mukherjee, E. Eide, Deepstitch: Deep learning for cross-layer stitching in microservices, in: Proceedings of the 2020 6th International Workshop on Container Technologies and Container Clouds, WOC'20, Association for Computing Machinery, New York, NY, USA, 2021.

SP38 S. Wang, Y. Guo, N. Zhang, P. Yang, A. Zhou, X. Shen, Delay-aware microservice coordination in mobile edge computing: A reinforcement learning approach, IEEE Transactions on Mobile Computing 20 (2021).

SP39 S. Y. Shah, Z. Yuan, S. Lu, P. Zerfos, Dependency analysis of cloud applications for performance monitoring using recurrent neural networks, in: 2017 IEEE International Conference on Big Data (Big Data).

SP40 S. Wu, C. Denninnart, X. Li, Y. Wang, M. A. Salehi, Descriptive and predictive analysis of aggregating functions in server- less clouds: the case of video streaming, in: 2020 IEEE 22nd International Conference on High Performance Computing and Communications, IEEE 18th International Conference on Smart City, IEEE 6th International Conference on Data Science and Systems (HP- CC/SmartCity/DSS).

SP41 V. Cortellessa, L. Traini, Detecting latency degradation patterns in service-based systems, in: Proceedings of the ACM/SPEC International Conference on Performance Engineering, ICPE'20, Association for Computing Machinery, New York, NY, USA, 2020.

SP42 J. Grohmann, S. Eismann, S. Elflein, J. V. Kistowski, S. Kounev, M. Mazkatli, Detecting parametric dependencies for performance models using feature selection techniques, in: 2019 IEEE 27th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS).

SP43 J. Chou, E. Al-Masri, S. Kanzhelev, H. Fattah, Detecting security and privacy risks in microservices end-to-end communication using neural networks, in: 2021 IEEE 4th International Conference on Knowledge Innovation and Invention (ICKII).

SP44 A. A. Khaleq, I. Ra, Development of qos-aware agents with reinforcement learning for autoscaling of microservices on the cloud, in: 2021 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOSC).

SP45 C. Hou, T. Jia, Y. Wu, Y. Li, J. Han, Diagnosing performance issues in microservices with heterogeneous data source, in: 2021 IEEE Intl. Conf. on Parallel Distributed Processing with Applications, Big Data Cloud Computing, Sustainable Computing Communications, Social Computing Networking (ISPA/BDCloud/So-cialCom/Sustain- Com).

SP46 H. Tian, X. Xu, T. Lin, Y. Cheng, C. Qian, L. Ren, M. Bilal, Dima: Distributed cooperative microservice caching for internet of things in edge computing by deep reinforcement learning, World Wide Web 25 (2022).

SP47 M. Abdullah, W. Iqbal, F. Bukhari, A. Erradi, Diminishing returns and deep learning for adaptive cpu resource allocation of containers, IEEE Transactions on Network and Service Management 17 (2020).

SP48 M. Ghorbani, F. F. Moghaddam, M. Zhang, M. Pourzandi, K. K. Nguyen, M. Cheriet, Distappgaurd: Distributed application behaviour profiling in cloud-based environment, in: Annual Computer Security Applications Conference, ACSAC'21, Association for Computing Machinery, New York, NY, USA, 2021.

SP49 H. Zhao, S. Deng, Z. Liu, J. Yin, S. Dustdar, Distributed redundant placement for microservice-based applications at the edge, IEEE Transactions on Services Computing 15 (2022).

SP50 A. Samir, C. Pahl, Dla: Detecting and localizing anomalies in containerized microservice architectures using markov models, in: 2019 7th International Conference on Future Internet of Things and Cloud (FiCloud).

SP51 A. Samanta, J. Tang, Dyme: Dynamic microservice scheduling in edge computing enabled iot, IEEE Internet of Things Journal 7 (2020).

SP52 K. Zeng, I. Paik, Dynamic service recommendation using lightweight BERT-based service embedding in edge computing, in: 2021 IEEE 14th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC).

SP53 G. Orsini, W. Posdorfer, W. Lamersdorf, Efficient mobile clouds: Forecasting the future connectivity of mobile and iot devices to save energy and bandwidth, Procedia Computer Science 155 (2019) 121-128. The 16th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2019),The 14th International Conference on Future Networks and Communications (FNC-2019),The 9th International Conference on Sustainable Energy Information Technology.

SP54 S. Wang, Z. Ding, C. Jiang, Elastic scheduling for microservice applications in clouds, IEEE Transactions on Parallel and Distributed Systems 32 (2021).

SP55 H. Mohamed, O. El-Gayar, End-to-end latency prediction of microservices workflow on kubernetes: A comparative evaluation of machine learning models and resource metrics, volume 2020-January.

SP56 M. França, C. Werner, Evaluating cloud microservices with director, in: 2019 IEEE International Conference on Service-Oriented System Engineering (SOSE).

SP57 V. M. Mostofi, D. Krishnamurthy, M. Arlitt, Fast and efficient performance tuning of microservices, in: 2021 IEEE 14th International Conference on Cloud Computing (CLOUD).

SP58 H. Qiu, S. S. Banerjee, S. Jha, Z. T. Kalbarczyk, R. K. Iyer, Firm: An intelligent fine-grained resource management framework for slo-oriented microservices.

SP59 C. T. Joseph, J. P. Martin, K. Chandrasekaran, A. Kandasamy, Fuzzy reinforcement learning based microservice allocation in cloud computing environments, in: TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON).

SP60 D. C. Li, C.-T. Huang, C.-W. Tseng, L.-D. Chou, Fuzzy-based microservice resource management platform for edge computing in the internet of things, Sensors 21 (2021).
SP61 C. Guerrero, I. Lera, C. Juiz, Genetic algorithm for multi-objective optimization of container allocation in cloud architecture, Journal of Grid Computing 16 (2018).
SP62 J. Kim, S. Ullah, D.-H. Kim, Gpu-based embedded edge server configuration and offloading for a neural network service, The Journal of Supercomputing 77 (2021).
SP63 J. Park, B. Choi, C. Lee, D. Han, Graf: A graph neural network based proactive resource allocation framework for slo-oriented microservices, in: Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies, CoNEXT'21, Association for Computing Machinery, New York, NY, USA, 2021.
SP64 R. S. Kannan, L. Subramanian, A. Raju, J. Ahn, J. Mars, L. Tang, Grandslam: Guaranteeing slas for jobs in microservices execution frameworks, in: Proceedings of the Fourteenth EuroSys Conference 2019, EuroSys'19, Association for Computing Machinery, New York, NY, USA, 2019.
SP65 Álvaro Brandón, M. Solé, A. Huélamo, D. Solans, M. S. Pérez, V. Muntés-Mulero, Graph-based root cause analysis for service-oriented and microservice architectures, Journal of Systems and Software 159 (2020).
SP66 M. Yan, X. Liang, Z. Lu, J. Wu, W. Zhang, Hansel: Adaptive horizontal scaling of microservices using bi-lstm, Applied Soft Computing 105 (2021).
SP67 F. Rossi, V. Cardellini, F. L. Presti, Hierarchical scaling of microservices in kubernetes, in: 2020 IEEE International Con- ference on Autonomic Computing and Self-Organizing Systems (ACSOS).
SP68 J. Chen, H. Huang, H. Chen, Informer: Irregular traffic detection for containerized microservices rpc in the real world, High-Confidence Computing 2 (2022).
SP69 M. Zhu, H. Qu, J. Zhao, Instance expansion algorithm for micro-service with prediction, Electronics Letters 54 (2018).
SP70 A. A. Khaleq, I. Ra, Intelligent autoscaling of microservices in the cloud for real-time applications, IEEE Access 9 (2021).
SP71 D. Uzunidis, P. Karkazis, C. Roussou, C. Patrikakis, H. C. Leligou, Intelligent performance prediction: The use case of a hadoop cluster, Electronics 10 (2021).
SP72 L. Chen, Y. Xu, Z. Lu, J. Wu, K. Gai, P. C. K. Hung, M. Qiu, Iot microservice deployment in edge-cloud hybrid environment using reinforcement learning, IEEE Internet of Things Journal 8 (2021).
SP73 Y. Yu, J. Yang, C. Guo, H. Zheng, J. He, Joint optimization of service request routing and instance placement in the microservice system, Journal of Network and Computer Applications 147 (2019).
SP74 X. Zhou, X. Peng, T. Xie, J. Sun, C. Ji, D. Liu, Q. Xiang, C. He, Latent error prediction and fault localization for microservice applications by learning from system trace logs, in: Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on

the Foundations of Software Engineering, ESEC/FSE 2019, Association for Computing Machinery, New York, NY, USA, 2019.

SP75 I. Tsoumas, C. Symvoulidis, D. Kyriazis, Learning a generalized matrix from multi-graphs topologies towards microservices recommendations, in: K. Arai, S. Kapoor, R. Bhatia (Eds.), Intelligent Systems and Applications, Springer International Publishing, Cham, 2021.

SP76 M. Abdullah, W. Iqbal, A. Erradi, F. Bukhari, Learning predictive autoscaling policies for cloud-hosted microservices using trace-driven modeling, in: 2019 IEEE International Conference on Cloud Computing Technology and Science (CloudCom).

SP77 C. Streiffer, R. Raghavendra, T. Benson, M. Srivatsa, Learning to simplify distributed systems management, in: 2018 IEEE International Conference on Big Data (Big Data).

SP78 M. Imdoukh, I. Ahmad, M. G. Alfailakawi, Machine learning- based auto-scaling for containerized applications, Neural Com- puting and Applications 32 (2020).

SP79 G. Coviello, Y. Yang, K. Rao, S. Chakradhar, Magic-pipe: Self-optimizing video analytics pipelines, in: Proceedings of the 22nd International Middleware Conference, Middleware'21, Association for Computing Machinery, New York, NY, USA, 2021.

SP80 M. M. Ghorbani, F. F. Moghaddam, M. Zhang, M. Pourzandi, K. K. Nguyen, M. Cheriet, Malchain: Virtual application behaviour profiling by aggregated microservice data exchange graph, in: 2020 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), 2020.

SP81 A. Rai, R. Jagadeesh Kannan, Mathematical architecture of microservices for geographic information system based health management system, Asian Journal of Pharmaceutical and Clinical Research 10 (2017).

SP82 L. Wu, J. Tordsson, J. Bogatinovski, E. Elmroth, O. Kao, Microdiag: Fine-grained performance diagnosis for microservice systems, in: 2021 IEEE/ACM International Workshop on Cloud Intelligence (CloudIntelligence).

SP83 R. Oberhauser, S. Stigler, Microflows: Leveraging process mining and an automated constraint recommender for microflow modeling, in: B. Shishkov (Ed.), Business Modeling and Software Design, Springer International Publishing, Cham, 2018.

SP84 D. Liu, C. He, X. Peng, F. Lin, C. Zhang, S. Gong, Z. Li, J. Ou, Z. Wu, MicroHECL: High-efficient root cause localization in large-scale microservice systems, in: 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software En- gineering in Practice (ICSE-SEIP).

SP85 L. Wu, J. Tordsson, E. Elmroth, O. Kao, Microrca: Root cause localization of performance issues in microservices, in: NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium.

SP86 G. Yu, P. Chen, Z. Zheng, Microscaler: Cost-effective scaling for microservice applications in the cloud with an online learning approach, IEEE Transactions on Cloud Computing 10 (2022).

SP87 J. Yue, X. Wu, Y. Xue, Microservice aging and rejuvenation, in: 2020 World Conference on Computing and Communication Technologies (WCCCT).

SP88 M. Li, D. Tang, Z. Wen, Y. Cheng, Microservice anomaly detection based on tracing data using semi-supervised learning, in: 2021 4th International Conference on Artificial Intelligence and Big Data (ICAIBD).

SP89 H. Chang, M. Kodialam, T. Lakshman, S. Mukherjee, Microservice fingerprinting and classification using machine learning, in: 2019 IEEE 27th International Conference on Network Protocols (ICNP).

SP90 F. Guo, B. Tang, M. Tang, H. Zhao, W. Liang, Microservice selection in edge-cloud collaborative environment: A deep reinforcement learning approach, in: 2021 8th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud) /2021 7th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom).

SP91 Z. Lyu, H. Wei, X. Bai, C. Lian, Microservice-based architecture for an energy management system, IEEE Systems Journal 14 (2020).

SP92 D. Rodríguez-Gracia, J. A. Piedra-Fernández, L. Iribarne, J. Criado, R. Ayala, J. Alonso-Montesinos, C.-U. Maria de las Mercedes, Microservices and machine learning algorithms for adaptive green buildings, Sustainability 11 (2019).

SP93 F. H. Vera-Rivera, E. Puerto, H. Astudillo, C. M. Gaona, Microservices backlog-a genetic programming technique for identification and evaluation of microservices from user stories, IEEE Access 9 (2021).

SP94 Z. Yang, P. Nguyen, H. Jin, K. Nahrstedt, Miras: Model-based reinforcement learning for microservice resource allocation over scientific workflows, in: 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS).

SP95 P. Nguyen, K. Nahrstedt, Monad: Self-adaptive microservice infrastructure for heterogeneous scientific workflows, in: 2017 IEEE International Conference on Autonomic Computing (ICAC), 2017.

SP96 N. Parekh, S. Kurunji, A. Beck, Monitoring resources of machine learning engine in microservices architecture, in: 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON).

SP97 S. Ji, W. Wu, Y. Pu, Multi-indicators prediction in microservice using granger causality test and attention lstm, in: 2020 IEEE World Congress on Services (SERVICES).

SP98 R. Liu, P. Yang, H. Lv, W. Li, Multi-objective multi-factorial evolutionary algorithm for container placement, IEEE Transactions on Cloud Computing (2021).

SP99 D. Bhamare, M. Samaka, A. Erbad, R. Jain, L. Gupta, H. A. Chan, Multi-objective scheduling of micro-services for optimal service function chains, in: 2017 IEEE International Conference on Communications (ICC).

SP100 S. Horovitz, Y. Arian, M. Vaisbrot, N. Peretz, Non-intrusive cloud application transaction pattern discovery, in: 2019 IEEE 12th International Conference on Cloud Computing (CLOUD).

SP101 R. Filipe, J. Correia, F. Araujo, J. Cardoso, On black-box monitoring techniques for multi-component services, in: 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA).

SP102 H. Alipour, Y. Liu, Online machine learning for cloud resource provisioning of microservice backend systems, in: 2017 IEEE International Conference on Big Data (Big Data).

SP103 F. D. Pellegrini, F. Faticanti, M. Datar, E. Altman, D. Siracusa, Optimal blind and adaptive fog orchestration under local processor sharing, in: 2020 18th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT).

SP104 B. St évant, J.-L. Pazat, A. Blanc, Optimizing the performance of a microservice-based application deployed on user-provided devices, in: 2018 17th International Symposium on Parallel and Distributed Computing (ISPDC).

SP105 A. Mirhosseini, S. Elnikety, T. F. Wenisch, Parslo: A gradient descent-based approach for near-optimal partial slo allotment in microservices, in: Proceedings of the ACM Symposium on Cloud Computing, SoCC'21, Association for Computing Machinery, New York, NY, USA, 2021.

SP106 D. Bajaj, U. Bharti, A. Goel, S. C. Gupta, Partial migration for re-architecting a cloud native monolithic application into microservices and faas, in: C. Badica, P. Liatsis, L. Kharb, D. Chahal (Eds.), Information, Communication and Computing Technology, Springer Singapore, Singapore, 2020.

SP107 L. Wu, J. Bogatinovski, S. Nedelkoski, J. Tordsson, O. Kao, Performance diagnosis in cloud microservices using deep learning, in: H. Hacid, F. Outay, H.-y. Paik, A. Alloum, M. Petrocchi, M. R. Bouadjenek, A. Beheshti, X. Liu, A. Maaradji (Eds.), Service-Oriented Computing - ICSOC 2020 Workshops, Springer International Publishing, Cham, 2021.

SP108 B. Choi, J. Park, C. Lee, D. Han, Phpa: A proactive autoscaling framework for microservice chain, in: 5th Asia-Pacific Workshop on Networking (APNet 2021), APNet 2021, Association for Computing Machinery, New York, NY, USA, 2022.

SP109 J. Rahman, P. Lama, Predicting the End-to-End Tail Latency of Containerized Microservices in the Cloud, in: 2019 IEEE International Conference on Cloud Engineering (IC2E).

SP110 M. Abdullah, W. Iqbal, A. Mahmood, F. Bukhari, A. Erradi, Predictive autoscaling of microservices hosted in fog microdata center, IEEE Systems Journal 15 (2021).

SP111 N. Marie-Magdelaine, T. Ahmed, Proactive autoscaling for cloud-native applications using machine learning, in: GLOBECOM 2020 - 2020 IEEE Global Communications Conference.

SP112 K. Ray, A. Banerjee, N. C. Narendra, Proactive microservice placement and migration for mobile edge computing, in: 2020 IEEE/ACM Symposium on Edge Computing (SEC).

SP113 J. Bender, J. Ovtcharova, Prototyping machine-learning-supported lead time prediction using automl, Procedia Computer Science 180 (2021) 649-655. Proceedings of the 2nd International Conference on Industry 4.0 and Smart Manufacturing (ISM 2020).

SP114 Q. Li, B. Li, P. Mercati, R. Illikkal, C. Tai, M. Kishinevsky, C. Kozyrakis, Rambo: Resource allocation for microservices using bayesian optimization, IEEE Computer Architecture Letters 20 (2021).
SP115 A. Belhadi, Y. Djenouri, G. Srivastava, J. C.-W. Lin, Reinforcement learning multi-agent system for faults diagnosis of mircoservices in industrial settings, Computer Communications 177 (2021).
SP116 P. Li, J. Song, H. Xu, L. Dong, Y. Zhou, Resource scheduling optimisation algorithm for containerised microservice architecture in cloud computing, International Journal of High Performance Systems Architecture 8 (2018).
SP117 P. Kang, P. Lama, Robust resource scaling of containerized microservices with probabilistic machine learning, in: 2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC).
SP118 A. Bali, M. Al-Osta, S. Ben Dahsen, A. Gherbi, Rule based auto-scalability of iot services for efficient edge device resource utilization, Journal of Ambient Intelligence and Humanized Computing 11 (2020).
SP119 Y. Gan, M. Liang, S. Dev, D. Lo, C. Delimitrou, Sage: Practical and scalable ml-driven performance debugging in microservices, in: Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS'21, Association for Computing Machinery, New York, NY, USA, 2021.
SP120 S.-P. Ma, Y. Chuang, C.-W. Lan, H.-M. Chen, C.-Y. Huang, C.-Y. Li, Scenario-based microservice retrieval using word2vec, in: 2018 IEEE 15th International Conference on e-Business Engineering (ICEBE), 2018.
SP121 H. Gu, X. Li, M. Liu, S. Wang, Scheduling method with adaptive learning for microservice workflows with hybrid resource provisioning, International Journal of Machine Learning and Cybernetics 12 (2021).
SP122 Y. Gan, Y. Zhang, K. Hu, D. Cheng, Y. He, M. Pancholi, C. Delimitrou, Seer: Leveraging big data to navigate the complexity of performance debugging in cloud microservices, in: Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS'19, Association for Computing Machinery, New York, NY, USA, 2019.
SP123 F. Rossi, V. Cardellini, F. L. Presti, Self-adaptive threshold-based policy for microservices elasticity, in: 2020 28th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS).
SP124 J. Bogatinovski, S. Nedelkoski, J. Cardoso, O. Kao, Self-supervised anomaly detection from distributed traces, in: 2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC), 2020.
SP125 D. Petcu, Service deployment challenges in cloud-to-edge continuum, Scalable Computing: Practice and Experience 22 (2021).
SP126 P. Krämer, P. Diederich, C. Krämer, R. Pries, W. Kellerer, A. Blenk, sfc2cpu: Operating a service function chain platform with neural combinatorial optimization, in: 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM).

SP127 Y. Li, T. Li, P. Shen, L. Hao, W. Liu, S. Wang, Y. Song, L. Bao, Sim-drs: a similarity-based dynamic resource scheduling algorithm for microservice-based web systems, PeerJ Computer Science 7 (2021).
SP128 Y. Zhang, W. Hua, Z. Zhou, G. E. Suh, C. Delimitrou, Sinan: Ml-based and qos-aware resource management for cloud microservices, in: Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS'21, Association for Computing Machinery, New York, NY, USA, 2021.
SP129 J. Herrera, G. Molt ó, Toward bio-inspired auto-scaling algorithms: An elasticity approach for container orchestration platforms, IEEE Access 8 (2020).
SP130 P. Zhao, P. Wang, X. Yang, J. Lin, Towards cost-efficient edge intelligent computing with elastic deployment of container-based microservices, IEEE Access 8 (2020).
SP131 G. Somashekar, A. Gandhi, Towards optimal configuration of microservices, in: Proceedings of the 1st Workshop on Machine Learning and Systems, EuroMLSys'21, Association for Computing Machinery, New York, NY, USA, 2021.
SP132 H. Chen, K. Wei, A. Li, T. Wang, W. Zhang, Trace-based intelligent fault diagnosis for microservices with deep learning, in: 2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC).
SP133 P. Liu, H. Xu, Q. Ouyang, R. Jiao, Z. Chen, S. Zhang, J.Yang, L. Mo, J. Zeng, W. Xue, D. Pei, Unsupervised detection of microservice trace anomalies through service-level deep bayesian networks, in: 2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE), 2020.
SP134 M. Abdullah, W. Iqbal, A. Erradi, Unsupervised learning approach for web application auto-decomposition into microservices, Journal of Systems and Software 151 (2019).
SP135 W. Liang, Y. Hu, X. Zhou, Y. Pan, K. I.-K. Wang, Variational few-shot learning for microservice-oriented intrusion detection in distributed industrial iot, IEEE Transactions on Industrial Informatics 18 (2022).
SP136 H. Sami, A. Mourad, W. El-Hajj, Vehicular-obus-as-on-demand-fogs: Resource and context aware deployment of containerized micro-services, IEEE/ACM Transactions on Networking 28 (2020).
SP137 P. Petrou, S. Karagiorgou, D. Alexandrou, Weighted load balancing mechanisms over streaming big data for online machine learning, in: EDBT/ICDT Workshops, 2021.
SP138 H. Bangui, B. Rossi, B. Buhnova, A Conceptual Antifragile Microservice Framework for Reshaping Critical Infrastructures, in: 2022 IEEE International Conference on Software Maintenance and Evolution (ICSME), 2022.
SP139 A. Ali, M. M. Iqbal, A Cost and Energy Efficient Task Scheduling Technique to Offload Microservices Based Applications in Mobile Cloud Computing, IEEE Access 10 (2022).
SP140 Y. Qi, S. Shao, S. Wu, X. Qiu, S. Guo, S. Guo, A Distributed Intelligent Service Trusted Provision Approach for IoT, IEEE Internet of Things Journal 10 (2023).

SP141 H. Li, Y. Zhao, Z. Liu, W. Liu, A Distributed Microservice Scheduling Optimization Method, in: 2023 IEEE International Conference on Control, Electronics and Computer Technology (ICCECT), 2023.
SP142 Bagehorn F., Rios J., Jha S., Filepp R., Shwartz L., Abe N., Yang X., A fault injection platform for learning AIOps models, in: ACM International Conference Proceeding Series, 2022.
SP143 Hossein Ebrahimpour, Mehrdad Ashtiani, Fatemeh Bakhshi, Ghazaleh Bakhtiariazad, A heuristic-based package-aware function scheduling approach for creating a trade-off between cold start time and cost in FaaS computing environments, The Journal of Supercomputing 79 (2023).
SP144 Thiago Felipe da Silva Pinheiro, Paulo Pereira, Bruno Silva, Paulo Maciel, A performance modeling framework for microservices-based cloud infrastructures, The Journal of Supercomputing 79 (2023).
SP145 A. Heimerson, J. Eker, K. -E. Årzén, A Proactive Cloud Application Auto-Scaler using Reinforcement Learning, in: 2022 IEEE/ACM 15th International Conference on Utility and Cloud Computing (UCC), 2022.
SP146 C. H. Zhang, M. Omair Shafiq, A Real-time, Scalable Monitoring and User Analytics Solution for Microservices-based Software Applications, in: 2022 IEEE International Conference on Big Data (Big Data), 2022.
SP147 Sedigheh Khoshnevis, A search-based identification of variable microservices for enterprise SaaS, Frontiers of Computer Science (2022).
SP148 D. Zhou, H. Chen, G. Cheng, A Security Containers Placement Algorithm Based on DQN for Microservices to Defend Against Co-Resident Threat, in: 2023 8th International Conference on Computer and Communication Systems (ICCCS), 2023.
SP149 C. -A. Sun, T. Zeng, W. Zuo, H. Liu, A Trace-Log-Clusterings-Based Fault Localization Approach to Microservice Systems, in: 2023 IEEE International Conference on Web Services (ICWS), 2023.
SP150 Claudia Canali, Giuseppe Di Modica, Riccardo Lancellotti, Stefano Rossi, Domenico Scotece, A Validated Performance Model for Micro-services Placement in Fog Systems, SN Computer Science 4 (2023).
SP151 C. Duan, T. Jia, Y. Li, G. Huang, AcLog: An Approach to Detecting Anomalies from System Logs with Active Learning, in: 2023 IEEE International Conference on Web Services (ICWS), 2023.
SP152 Ruibo Chen, Yanjun Pu, Bowen Shi, Wenjun Wu, An automatic model management system and its implementation for AIOps on microservice platforms, The Journal of Supercomputing 79 (2023).
SP153 Mansoureh Zare, Yasser Elmi Sola, Hesam Hasanpour, An autonomous planning model for solving IoT service placement problem using the imperialist competitive algorithm, The Journal of Supercomputing 79 (2023).
SP154 Y. Sever, G. Ekinci, A. H. Dogan, B. Alparslan, A. S. Gurbuz, V. Jabrayilov, P. Angin, An Empirical Analysis of IDS Approaches in Container Security, in: 2022 International Workshop on Secure and Reliable Microservices and Containers (SRMC), 2022.

SP155 Wenliang LinYilie HeZhongliang DengKe WangBin JinXiaotian Zhou, An end-to-end software-defined network framework and optimal service development model for SAGN, Telecommunication Systems (2022).
SP156 Wesley K. G. Assunção, Thelma Elita Colanzi, Luiz Carvalho, Alessandro Garcia, Juliana Alves Pereira, Maria Julia de Lima, Carlos Lucena, Analysis of a many-objective optimization approach for identifying microservices from legacy systems, Empirical Software Engineering 27 (2022).
SP157 Iman Kohyarnejadfard, Daniel Aloise, Seyed Vahid Azhari, Michel R. Dagenais, Anomaly detection in microservice environments using distributed tracing data analysis and NLP, Journal of Cloud Computing 11 (2022).
SP158 M. Sowmya, A. J. Rai, V. Spoorthi, M. Irfan, P. B. Honnavalli, S. Nagasundari, API Traffic Anomaly Detection in Microservice Architecture, in: 2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing Workshops (CCGridW), 2023.
SP159 L. S. Hettiarachchi, S. V. Jayadeva, R. A. V. Bandara, D. Palliyaguruge, U. S. S. S. Arachchillage, D. Kasthurirathna, Artificial Intelligence-Based Centralized Resource Management Application for Distributed Systems, in: 2022 13th International Conference on Computing Communication and Networking Technologies (ICCCNT), 2022.
SP160 Rafael de Jesus Martins, Juliano Araújo Wickboldt, Lisandro Zambenedetti Granville, Assisted Monitoring and Security Provisioning for 5 G Microservices-Based Network Slices with SWEETEN, Journal of Network and Systems Management 31 (2023).
SP161 S. Choochotkaew, T. Chiba, S. Trent, T. Yoshimura, M. Amaral, AutoDECK: Automated Declarative Performance Evaluation and Tuning Framework on Kubernetes, in: 2022 IEEE 15th International Conference on Cloud Computing (CLOUD), 2022.
SP162 T. Miyazawa, M. Jibiki, V. P. Kafle, Automated Data Analytics and Resource Arbitration Scheduling for Containerized Network Functions, in: 2022 IEEE Future Networks World Forum (FNWF), 2022.
SP163 A. A. Pramesti, A. I. Kistijantoro, Autoscaling Based on Response Time Prediction for Microservice Application in Kubernetes, in: 2022 9th International Conference on Advanced Informatics: Concepts, Theory and Applications (ICAICTA), 2022.
SP164 Ahmet Vedat TokmakAkhan AkbulutCagatay Catal, Boosting the visibility of services in microservice architecture, Cluster Computing (2023).
SP165 A. Boukhtouta, T. Madi, M. Pourzandi, H. A. A., Cloud Native Applications Profiling using a Graph Neural Networks Approach, in: 2022 IEEE Future Networks World Forum (FNWF), 2022.
SP166 M. D. Hossain, T. Sultana, S. Akhter, M. I. Hossain, G. -W. Lee, C. S. Hong, E. -N. Huh, Computation Offloading Strategy Based on Multi-armed Bandit Learning in Microservice-enabled Vehicular Edge Computing Networks, in: 2023 International Conference on Information Networking (ICOIN), 2023.
SP167 D. Ou, C. Jiang, M. Zheng, Y. Ren, Container Power Consumption Prediction Based on GBRT-PL for Edge Servers in Smart City, IEEE Internet of Things Journal 10 (2023).

SP168 Z. Wang, C. -A. Sun, M. Aiello, Context-aware IoT Service Recommendation: A Deep Collaborative Filtering-based Approach, in: 2022 IEEE International Conference on Web Services (ICWS), 2022.
SP169 Juan Luis Herrera, Javier Berrocal, Stefano Forti, Antonio Brogi, Juan M. Murillo, Continuous QoS-aware adaptation of Cloud-IoT application placements, Computing 105 (2023).
SP170 M. Xu, C. Song, S. Ilager, S. S. Gill, J. Zhao, K. Ye, C. Xu, CoScal: Multifaceted Scaling of Microservices With Reinforcement Learning, IEEE Transactions on Network and Service Management 19 (2022).
SP171 Abdullah LakhanMuhammad Suleman MemonQurat-ul-ain MastoiMohamed ElhosenyMazin Abed MohammedMumtaz QabulioMohamed Abdel-Basset, Cost-efficient mobility offloading and task scheduling for microservices IoVT applications in container-based fog cloud network, Cluster Computing (2022).
SP172 P. Krämer, P. Diederich, C. Krämer, R. Pries, W. Kellerer, A. Blenk, D2A: Operating a Service Function Chain Platform With Data-Driven Scheduling Policies, IEEE Transactions on Network and Service Management 19 (2022).
SP173 Abdullah Alelyani, Amitava Datta, Ghulam Mubashar Hassan, DAScheduler: Dependency-Aware Scheduling Algorithm for Containerized Dependent Jobs, Journal of Grid Computing 21 (2023).
SP174 P. Benedetti, G. Coviello, K. Rao, S. Chakradhar, DataX Allocator: Dynamic resource management for stream analytics at the Edge, in: 2022 9th International Conference on Internet of Things: Systems, Management and Security (IOTSMS), 2022.
SP175 Y. Chen, M. Yan, D. Yang, X. Zhang, Z. Wang, Deep Attentive Anomaly Detection for Microservice Systems with Multimodal Time-Series Data, in: 2022 IEEE International Conference on Web Services (ICWS), 2022.
SP176 Z. Guo, K. Yu, Z. Lv, K. -K. R. Choo, P. Shi, J. J. P. C. Rodrigues, Deep Federated Learning Enhanced Secure POI Microservices for Cyber-Physical Systems, IEEE Wireless Communications 29 (2022).
SP177 C. Wang, B. Jia, H. Yu, X. Li, X. Wang, T. Taleb, Deep Reinforcement Learning for Dependency-aware Microservice Deployment in Edge Computing, in: GLOBECOM 2022 - 2022 IEEE Global Communications Conference, 2022.
SP178 Feiyan Guo, Bing Tang, Mingdong Tang, Wei Liang, Deep reinforcement learning-based microservice selection in mobile edge computing, Cluster Computing 26 (2023).
SP179 Q. Zhu, S. Wang, H. Huang, Y. Lei, W. Zhan, H. Duan, Deep-Reinforcement-Learning-Based Service Placement for Video Analysis in Edge Computing, in: 2023 8th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA), 2023.
SP180 C. Zhang, X. Peng, C. Sha, K. Zhang, Z. Fu, X. Wu, Q. Lin, D. Zhang, DeepTraLog: Trace-Log Combined Microservice Anomaly Detection through Graph-based Deep Learning, in: 2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE), 2022.

SP181 L. Traini, V. Cortellessa, DeLag: Using Multi-Objective Optimization to Enhance the Detection of Latency Degradation Patterns in Service-Based Systems, IEEE Transactions on Software Engineering (2023).
SP182 X. Yu, W. Wu, Y. Wang, Dependable Workflow Scheduling for Microservice QoS Based on Deep Q-Network, in: 2022 IEEE International Conference on Web Services (ICWS), 2022.
SP183 Lizhe ChenJi WuHaiyan YangKui Zhang, Does PageRank apply to service ranking in microservice regression testing?, Software Quality Journal (2022).
SP184 Sheuli Chakraborty, Debashis De, Kaushik Mazumdar, DoME: Dew computing based microservice execution in mobile edge using Q-learning, Applied Intelligence 53 (2023).
SP185 Z. Xiao, S. Hu, DScaler: A Horizontal Autoscaler of Microservice Based on Deep Reinforcement Learning, in: 2022 23rd Asia-Pacific Network Operations and Management Symposium (APNOMS), 2022.
SP186 Saravanan Muniswamy, Radhakrishnan Vignesh, DSTS: A hybrid optimal and deep learning for dynamic scalable task scheduling on container cloud environment, Journal of Cloud Computing 11 (2022).
SP187 S. B. Chetty, H. Ahmadi, M. Tornatore, A. Nag, Dynamic Decomposition of Service Function Chain Using a Deep Reinforcement Learning Approach, IEEE Access 10 (2022).
SP188 F. Rossi, V. Cardellini, F. L. Presti, M. Nardelli, Dynamic Multi-Metric Thresholds for Scaling Applications Using Reinforcement Learning, IEEE Transactions on Cloud Computing 11 (2023).
SP189 C. Lee, T. Yang, Z. Chen, Y. Su, M. R. Lyu, Eadro: An End-to-End Troubleshooting Framework for Microservices on Multi-source Data, in: 2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE), 2023.
SP190 J. Qi, H. Zhang, X. Li, H. Ji, X. Shao, Edge-edge Collaboration Based Microservice Deployment in Edge Computing Networks, in: 2023 IEEE Wireless Communications and Networking Conference (WCNC), 2023.
SP191 I. Syrigos, D. Kefalas, N. Makris, T. Korakis, EELAS: Energy Efficient and Latency Aware Scheduling of Cloud-Native ML Workloads, in: 2023 15th International Conference on COMmunication Systems & NETworkS (COMSNETS), 2023.
SP192 Mohamed Hedi Fourati, Soumaya Marzouk, Mohamed Jmaiel, EPMA: Elastic Platform for Microservices-based Applications: Towards Optimal Resource Elasticity, Journal of Grid Computing 20 (2022).
SP193 L. S. Hettiarachchi, S. V. Jayadeva, R. A. V. Bandara, D. Palliyaguruge, U. S. S. S. Arachchillage, D. Kasthurirathna, Expert System for Kubernetes Cluster Autoscaling and Resource Management, in: 2022 4th International Conference on Advancements in Computing (ICAC), 2022.
SP194 Mohammad Hadi Dehghani, Shekoufeh Kolahdouz-Rahimi, Massimo Tisi, Dalila Tamzalit, Facilitating the migration to the microservice architecture via model-driven reverse engineering and reinforcement learning, Software and Systems Modeling 21 (2022).

SP195 Kawasaki J., Koyama D., Miyasaka T., Otani T., Failure Prediction in Cloud Native 5 G Core With eBPF-based Observability, in: IEEE Vehicular Technology Conference, 2023.
SP196 Wang Q., Rios J., Jha S., Shanmugam K., Bagehorn F., Yang X., Filepp R., Abe N., Shwartz L., Fault Injection Based Interventional Causal Learning for Distributed Applications, in: Proceedings of the 37th AAAI Conference on Artificial Intelligence, AAAI 2023, 2023.
SP197 X. Yang, J. Wang, B. Zhou, W. Wang, W. Liu, Y. Dong, Fine-grained Spatiotemporal Features-Based for Anomaly Detection in Microservice Systems, in: 2022 IEEE 24th Int Conf on High Performance Computing & Communications, 8th Int Conf on Data Science & Systems, 20th Int Conf on Smart City, 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys), 2022.
SP198 R. Ren, Y. Wang, F. Liu, Z. Li, G. Tyson, T. Miao, G. Xie, Grace: Interpretable Root Cause Analysis by Graph Convolutional Network for Microservices, in: 2023 IEEE/ACM 31st International Symposium on Quality of Service (IWQoS), 2023.
SP199 W. Lv, P. Yang, T. Zheng, C. Lin, Z. Wang, M. Deng, Q. Wang, Graph Reinforcement Learning-based Dependency-Aware Microservice Deployment in Edge Computing, in: IEEE Internet of Things Journal, 2023.
SP200 H. X. Nguyen, S. Zhu, M. Liu, Graph-PHPA: Graph-based Proactive Horizontal Pod Autoscaling for Microservices using LSTM-GNN, in: 2022 IEEE 11th International Conference on Cloud Networking (CloudNet), 2022.
SP201 H. He, L. Su, K. Ye, GraphGRU: A Graph Neural Network Model for Resource Prediction in Microservice Cluster, in: 2022 IEEE 28th International Conference on Parallel and Distributed Systems (ICPADS), 2023.
SP202 T. Rathod, C. T. Joseph, J. P. Martin, Improving Industry 4.0 Readiness: Monolith Application Refactoring using Graph Attention Networks, in: 2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing Workshops (CCGridW), 2023.
SP203 Abeer Abdel Khaleq, Ilkyeun Ra, Intelligent microservices autoscaling module using reinforcement learning, Cluster Computing 26 (2023).
SP204 Y. Zhang, C. Li, N. Chen, P. Zhang, Intelligent Requests Orchestration for Microservice Management Based on Blockchain in Software Defined Networking: a Security Guarantee, in: 2022 IEEE International Conference on Communications Workshops (ICC Workshops), 2022.
SP205 Thijs Metsch, Magdalena Viktorsson, Adrian Hoban, Monica Vitali, Ravi Iyer, Erik Elmroth, Intent-Driven Orchestration: Enforcing Service Level Objectives for Cloud Native Deployments, SN Computer Science 4 (2023).
SP206 G. Pearce, A. Pflaum, D. A. Balasoiu, C. Szabo, Jeopardy Assessment for Dynamic Configuration of Collaborative Microservice Architectures, in: 2022 Winter Simulation Conference (WSC), 2022.
SP207 Feiyan Guo, Bing Tang, Mingdong Tang, Joint optimization of delay and cost for microservice composition in mobile edge computing, World Wide Web 25 (2022).

SP208 Javad Dogani, Farshad Khunjush, Mehdi Seydali, K-AGRUED: A Container Autoscaling Technique for Cloud-based Web Applications in Kubernetes Using Attention-based GRU Encoder-Decoder, Journal of Grid Computing 20 (2022).

SP209 S. Hirai, H. Baba, M. Matsumoto, T. Hamano, K. Noguchi, Machine Learning based Performance Prediction for Cloud-native 5 G Mobile Core Network, in: 2022 IEEE Wireless Communications and Networking Conference (WCNC), 2022.

SP210 Yang L., Li J., Shi K., Yang S., Yang Q., Sun J., MicroMILTS: Fault Location for Microservices Based Mutual Information and LSTM Autoencoder, in: APNOMS 2022 - 23rd Asia-Pacific Network Operations and Management Symposium: Data-Driven Intelligent Management in the Era of beyond 5 G, 2022.

SP211 W. Lv, Q. Wang, P. Yang, Y. Ding, B. Yi, Z. Wang, C. Lin, Microservice Deployment in Edge Computing Based on Deep Q Learning, IEEE Transactions on Parallel and Distributed Systems 33 (2022).

SP212 Y. Liu, B. Yang, X. Yang, Y. Wu, C. Li, Microservice Dynamic Migration based on Age of Service for Edge Computing, in: 2022 IEEE International Conference on Industrial Technology (ICIT), 2022.

SP213 W. Cruz, L. D. Michel, B. Drozdenko, S. Roodbeen, ML and Network Traces to M.A.R.S, in: 2023 IEEE International Conference on Cyber Security and Resilience (CSR), 2023.

SP214 K. Sooksatra, R. Maharjan, T. Cerny, Monolith to Microservices: VAE-Based GNN Approach with Duplication Consideration, in: 2022 IEEE International Conference on Service-Oriented System Engineering (SOSE), 2022.

SP215 Z. Li, H. Sun, Z. Xiong, Q. Huang, Z. Hu, D. Li, S. Ruan, H. Hong, J. Gui, J. He, Z. Xu, Y. Fang, Noah: Reinforcement-Learning-Based Rate Limiter for Microservices in Large-Scale E-Commerce Services, IEEE Transactions on Neural Networks and Learning Systems 34 (2023).

SP216 Y. Yu, J. Liu, J. Fang, Online Microservice Orchestration for IoT via Multi-objective Deep Reinforcement Learning, IEEE Internet of Things Journal 9 (2022).

SP217 A. Hrusto, E. Engström, P. Runeson, Optimization of Anomaly Detection in a Microservice System Through Continuous Feedback from Development, in: 2022 IEEE/ACM 10th International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems (SESoS), 2022.

SP218 X. Chen, Y. Wu, S. Xiao, Particle Swarm-Grey Wolf Cooperation Algorithm Based on Microservice Container Scheduling Problem, IEEE Access 11 (2023).

SP219 Y. Gan, M. Liang, S. Dev, D. Lo, C. Delimitrou, Practical and Scalable ML-Driven Cloud Performance Debugging With Sage, IEEE Micro 42 (2022).

SP220 Al Qassem L.M., Stouraitis T., Damiani E., Elfadel I.A.M., Proactive Random-Forest Autoscaler for Microservice Resource Allocation, IEEE Access 11 (2023).

SP221 B. Jeong, J. Jeon, Y. -S. Jeong, Proactive Resource Autoscaling Scheme based on SCINet for High-performance Cloud Computing, IEEE Transactions on Cloud Computing 11 (2023).
SP222 K. Zhang, C. Zhang, X. Peng, C. Sha, PUTraceAD: Trace Anomaly Detection with Partial Labels based on GNN and PU Learning, in: 2022 IEEE 33rd International Symposium on Software Reliability Engineering (ISSRE), 2022.
SP223 K. R. Sheshadri, J. Lakshmi, QoS aware FaaS for Heterogeneous Edge-Cloud continuum, in: 2022 IEEE 15th International Conference on Cloud Computing (CLOUD), 2022.
SP224 G. Somashekar, A. Suresh, S. Tyagi, V. Dhyani, K. Donkada, A. Pradhan, A. Gandhi, Reducing the Tail Latency of Microservices Applications via Optimal Configuration Tuning, in: 2022 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS), 2022.
SP225 J. E. Joyce, S. Sebastian, Reinforcement Learning based Autoscaling for Kafka-centric Microservices in Kubernetes, in: 2022 IEEE 4th PhD Colloquium on Emerging Domain Innovation and Technology for Society (PhD EDITS), 2022.
SP226 J. Zhao, C. Su, Y. Wang, Research on Microservice Coordination Technologies based on Deep Reinforcement Learning, in: 2022 2nd International Conference on Electronic Information Technology and Smart Agriculture (ICEITSA), 2022.
SP227 Park J., Son J., Kim D., Resource Metric Refining Module for AIOps Learning Data in Kubernetes Microservice, KSII Transactions on Internet and Information Systems 17 (2023).
SP228 S. M. Rajagopal, M. Supriya, R. Buyya, Resource Provisioning Using Meta-Heuristic Methods for IoT Microservices With Mobility Management, IEEE Access 11 (2023).
SP229 Zhengzhe XiangYuhang ZhengDongjing WangMengzhu HeCheng ZhangZengwei Zheng, Robust and Cost-effective Resource Allocation for Complex IoT Applications in Edge-Cloud Collaboration, Mobile Networks and Applications (2022).
SP230 S. Zhang, P. Jin, Z. Lin, Y. Sun, B. Zhang, S. Xia, Z. Li, Z. Zhong, M. Ma, W. Jin, D. Zhang, Z. Zhu, D. Pei, Robust Failure Diagnosis of Microservice System through Multimodal Data, IEEE Transactions on Services Computing 6 (2023).
SP231 S. P. Kadiyala, X. Li, W. Lee, A. Catlin, Securing Microservices Against Password Guess Attacks using Hardware Performance Counters, in: 2022 IEEE 35th International System-on-Chip Conference (SOCC), 2022.
SP232 T. Stojanovic, S. D. Lazarević, The Application of ChatGPT for Identification of Microservices, in: E-business technologies conference proceedings, 2023.
SP233 H. Zeng, T. Wang, A. Li, Y. Wu, H. Wu, W. Zhang, Topology-Aware Self-Adaptive Resource Provisioning for Microservices, in: 2023 IEEE International Conference on Web Services (ICWS), 2023.

SP234 Zeb S., Rathore M.A., Hassan S.A., Raza S., Dev K., Fortino G., Toward AI-Enabled NextG Networks with Edge Intelligence-Assisted Microservice Orchestration, IEEE Wireless Communications 30 (2023).
SP235 R. Ren, Y. Wang, F. Liu, Z. Li, G. Xie, Triple:The Interpretable Deep Learning Anomaly Detection Framework based on Trace-Metric-Log of Microservice, in: 2023 IEEE/ACM 31st International Symposium on Quality of Service (IWQoS), 2023.
SP236 F. Dressler, C. F. Chiasserini, F. H. P. Fitzek, H. Karl, R. L. Cigno, A. Capone, C. Casetti, F. Malandrino, V. Mancuso, F. Klingler, G. Rizzo, V-Edge: Virtual Edge Computing as an Enabler for Novel Microservices and Cooperative Computing, IEEE Network 36 (2022).
SP237 Mekki M., Brik B., Ksentini A., Verikoukis C., XAI-Enabled Fine Granular Vertical Resources Autoscaler, in: 2023 IEEE 9th International Conference on Network Softwarization: Boosting Future Networks through Advanced Softwarization, NetSoft 2023 - Proceedings, 2023.
SP238 Ruibo Chen, Jian Ren, Lingfeng Wang, Yanjun Pu, Kaiyuan Yang & Wenjun Wu, MicroEGRCL: An Edge-Attention-Based Graph Neural Network Approach for Root Cause Localization in Microservice Systems, in: Service-Oriented Computing. ICSOC 2022. Lecture Notes in Computer Science, vol 13740., 2022.
SP239 Lingzhi Wang, Nengwen Zhao, Junjie Chen, Pinnong Li, Wenchi Zhang, Kaixin Sui, Root-Cause Metric Location for Microservice Systems via Log Anomaly Detection, in: 2020 IEEE International Conference on Web Services (ICWS), 2020.
SP240 Zhijun Ding, Song Wang, Changjun Jiang, Kubernetes-Oriented Microservice Placement With Dynamic Resource Allocation, IEEE Transactions on Cloud Computing (2022).
SP241 Sasho Nedelkoski, Jorge Cardoso, Odej Kao, Anomaly Detection from System Tracing Data Using Multimodal Deep Learning, in: IEEE Cloud, 2019.
SP242 Rajsimman Ravichandiran, Hadi Bannazadeh, Alberto Leon-Garcia, Anomaly Detection using Resource Behaviour Analysis for Autoscaling systems, in: NetSoft, 2018.
SP243 Mohammad Javad Amiri, Object-Aware Identification of Microservices, in: IEEE SCC, 2018.
SP244 Yukun Zhang, Bo Liu, Liyun Dai, Kang Chen, Xuelian Cao, Automated Microservice Identification in Legacy Systems with Functional and Non-Functional Metrics, in: IEEE ICSA, 2020.
SP245 Sinan Eski, Feza Buzluca, An automatic extraction approach: transition to microservices architecture from monolithic application, in: XP 2018, 2018.
SP246 Luiz Carvalho Alessandro Garcia, Thelma Elita Colanzi, Wesley K. G. Assunção, Maria Julia Lima, Baldoino Fonseca Márcio Ribeiro, Carlos Lucena, Search-based many-criteria identification of microservices from legacy systems., in: GECCO, 2020.
SP247 Mohamed Daoud, Asmae El Mezouari, Noura Faci, Djamal Benslimane, Zakaria Maamar & Aziz El Fazziki, Automatic Microservices Identification from a Set of Business Processes, in: SADASC, 2020.

SP248 Malak Saidi, Anis Tissaoui, Sami Faiz, A DDD Approach Towards Automatic Migration To Microservices, in: ASET, 2023.
SP249 Chenghao Song, Minxian Xu, Kejiang Ye, Huaming Wu, Sukhpal Singh Gill, Rajkumar Buyya & Chengzhong Xu, ChainsFormer: A Chain Latency-Aware Resource Provisioning Approach for Microservices Cluster, in: International Conference on Service-Oriented Computing, 2023.
SP250 Mohit Kumar, Jitendra Kumar Samriya, Kalka Dubey, Sukhpal Singh Gill, QoS-aware resource scheduling using whale optimization algorithm for microservice applications, Wiley Software and Experience 54 (2023).
SP251 Chunyang Meng, Shijie Song, Haogang Tong, Maolin Pan, Yang Yu, DeepScaler: Holistic Autoscaling for Microservices Based on Spatiotemporal GNN with Adaptive Graph Learnin, in: 38th IEEE/ACM International Conference on Automated Software Engineering (ASE), 2023.
SP252 Mohamed Samir, Khaled T. Wassif, Soha H. Makady, Proactive Auto-Scaling Approach of Production Applications Using an Ensemble Model, IEEE Access 11 (2023).
SP253 Duc-Hung LUONG, Huu-Trung THIEU, Abdelkader OUTTAGARTS, Yacine GHAMRI-DOUDANE, Predictive Autoscaling Orchestration for Cloud-native Telecom Microservices, in: IEEE 5 G World Forum, 2018.
SP254 Bing Tang, Xiaoyuan Zhang, Qing Yang, Xin Qi, Fayez Alqahtani, Amr Tolba, Cost-optimized Internet of Things application deployment in edge computing environment, Wiley International Journal of Communication Systems (2023).
SP255 Yuan Meng, Shenglin Zhang, Yongqian Sun, Ruru Zhang, Zhilong Hu, Yiyin Zhang, Chenyang Jia, Zhaogang Wang, Dan Pei, Localizing Failure Root Causes in a Microservice through Causality Inference, in: 2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS), 2020.
SP256 Carlos Guerrero, Isaac Lera, Carlos Juiz, Resource optimization of container orchestration: a case study in multi-cloud microservices-based applications., The Journal of Supercomputing 74 (2018).
SP257 Yimeng Wang, Cong Zhao, Shusen Yang, Xuebin Ren, Luhui Wang, Peng Zhao, Xinyu Yang, MPCSM: Microservice Placement for Edge-Cloud Collaborative Smart Manufacturing, IEEE Transactions on Industrial Informatics 17 (2021).
SP258 Jinjin Lin, Pengfei Chen & Zibin Zheng, Microscope: Pinpoint performance issues with causal graphs in micro- service environments, in: Service-Oriented Computing. ICSOC 2018, 2018.
SP259 Adha Hrusto, Emelie Engström, Per Runeson, Towards optimization of anomaly detection in DevOps, Information and Software Technology 160 (2023).
SP260 Dacheng Zhou, Hongchang Chen, Ke Shang, Guozhen Cheng, Jianpeng Zhang, Hongchao Hu, Cushion: A proactive resource provisioning method to mitigate SLO violations for containerized microservices, IET Communications (2022).
SP261 Li, N., Tan, Y., Wang, X., Li, B., Luo, J., SCORE: A Resource-Efficient Microservice Orchestration Model Based on Spectral Clustering in Edge Computing, in: Service-Oriented Computing. ICSOC 2022, 2022.

SP262 Fan Guisheng, Chen Liang, Yu Huiqun, Qi Wei, Multi-objective optimization of container-based microservice scheduling in edge computing, Computer Science and Information Systems 18 (2021).
SP263 Ma W, Wang R, Gu Y, Meng Q, Huang H, Deng S, Wu Y, Multi-objective microservice deployment optimization via a knowledge-driven evolutionary algorithm, Complex & Intelligent Systems 7 (2021).
SP264 Guangba Yu, Pengfei Chen, et al, MicroRank: End-to-end latency issue localization with extended spectrum analysis in microservice environments, in: 2021 World Wide Web Conference, WWW 2021, 2021.
SP265 F Faticanti, M Savi, F De Pellegrini, Locality-aware deployment of application microservices for multi-domain fog computing, Computer Communications 203 (2023).
SP266 Ayoub Benayache, Azeddine Bilami, Sami Barkat, Pascal Lorenz, Hafnaoui Taleb, MsM: A microservice middleware for smart WSN-based IoT application, Journal of Network and Computer Applications 111 (2019).
SP267 CT Joseph, K Chandrasekaran, IntMA: Dynamic interaction-aware resource allocation for containerized microservices in cloud environments, Journal of Systems Architecture (2020).
SP268 AK Kalia, J Xiao, C Lin, S Sinha, J Rofrano, M Vukovic, D Banerjee, Mono2micro: an ai-based toolchain for evolving monolithic enterprise applications to a microservice architecture, in: ESEC/FSE 2020: Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2020.
SP269 De Alwis, A.A.C., Barros, A., Fidge, C., Polyvyanyy, A., Remodularization Analysis for Microservice Discovery Using Syntactic and Semantic Clustering, in: Advanced Information Systems Engineering. CAiSE 2020, 2020.

Author contributions Sergio Moreschini: Data extraction, Data analysis, Writing, Reviewing Shahrzad Pour: Data extraction, Data analysis, Writing, Reviewing Ivan Lanese: Data extraction, Data analysis, Writing, Reviewing Daniel Balouek: Data extraction, Data analysis, Writing, Reviewing Justus Bogner: Data extraction, Data analysis, Writing, Reviewing Xiaozhou Li: Data extraction, Data analysis, Writing, Reviewing Fabiano Pecorelli: Data extraction, Data analysis, Writing, Reviewing Jacopo Soldani: Data extraction, Data analysis, Writing, Reviewing Eddy Truyen: Data extraction, Data analysis, Writing, Reviewing Davide Taibi: Data extraction, Data analysis, Writing, Reviewing

Funding Open Access funding provided by University of Oulu (including Oulu University Hospital). Ivan Lanese has been partially supported by French ANR project SmartCloud ANR-23-CE25-0012. Shahrzad Pour has been partially supported by the European Commission fundings through DeployAI (Grant No:101146490) and BIPED (Grant No: 101139060) projects. Jacopo Soldani has been partly supported by the project FREEDA (CUP: I53D23003550006), funded by PRIN (MUR, Italy) and Next Generation EU. This research is partially funded by the Research Fund KU Leuven. This research was funded in part by the Mufano and 6GSoft projects (Business Finland).

Data availibility replication package: https://doi.org/10.6084/m9.figshare.22663756.v6 supplementary material:https://doi.org/10.6084/m9.figshare.26243993.

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. Dragoni N et al. (2017) Microservices: yesterday, today, and tomorrow. In: Present and ulterior software engineering, Springer, Cham, pp 195–216
2. Merkel D (2014) Docker: lightweight linux containers for consistent development and deployment. Linux J 2014(239):1–1
3. Liu J et al (2018) Artificial intelligence in the 21st century. IEEE Access 6:34403–34421
4. Kotti Z, Galanopoulou R, Spinellis D (2023) Machine learning for software engineering: a tertiary study. ACM Comput Surv 55(12):1–39
5. Petersen K et al (2015) Guidelines for conducting systematic mapping studies in software engineering: an update. Inf Softw Technol 64:1–18
6. AI Watch (2021) Defining Artificial Intelligence 2.0. Publications Office of the European Union, pp 1–125. https://doi.org/10.2760/019901
7. International Organization For Standardization (2011) ISO/IEC 25010-systems and software engineering-systems and software quality requirements and evaluation (SQuaRE)-system and software quality models, vol 2, pp 1–34
8. Yıldırım A (2019) DevOps Lifecycle: continuous integration and development. https://shorturl.at/RYkx0. Accessed 2023-03-23
9. Erl T (2005) Service-oriented architecture: concepts, technology, and design. Prentice Hall PTR, Upper Saddle River, pp 1–760
10. Fowler M, Lewis J (2014) Microservices. https://shorturl.at/7zOPl. Accessed 2023-03-16
11. Newman S (2015) Building microservices: designing fine-grained systems, 1st edn. O'Reilly Media, Sebastopol, pp 1–278
12. Wang A, Tonse S (2013) Announcing Ribbon: tying the Netflix mid-tier services together. https://shorturl.at/GK7pl. Accessed 2023-03-16
13. Noonan A (2018) Goodbye Microservices: from 100s of problem children to 1 superstar. https://shorturl.at/tiEfx. Access 2023-03-16
14. Mendonca NC et al (2021) The monolith strikes back: why Istio migrated from microservices to a monolithic architecture. IEEE Softw 38(05):17–22
15. Bogner J et al. (2019) Microservices in industry: insights into technologies, characteristics, and software quality. In: ICSA-C, IEEE, Hamburg, pp 187–195
16. Soldani J et al (2018) The pains and gains of microservices: a systematic grey literature review. J Syst Softw 146:215–232
17. Wang Y, Kadiyala H, Rubin J (2021) Promises and challenges of microservices: an exploratory study. Empirical Softw Eng 26(4):63
18. Chen L (2018) Microservices: architecting for continuous delivery and DevOps. In: ICSA, IEEE, Seattle, pp 39–397. https://doi.org/10.1109/ICSA.2018.00013
19. Bass L, Weber I, Zhu L (2015) DevOps: a software architect's perspective, 1st edn. Addison-Wesley Professional, Boston, pp 1–352
20. Callanan M, Spillane A (2016) DevOps: making it easy to do the right thing. IEEE Softw 33(3):53–59. https://doi.org/10.1109/MS.2016.66
21. Moreschini S et al. (2022) Mlops for evolvable ai intensive software systems. In: SANER 2022. https://doi.org/10.1109/SANER53432.2022.00155
22. Treveil M et al. (2020) Introducing MLOps. O'Reilly Media, ??? , pp 1–186

23. Hilali A, et al (2021) Microservices adaptation using machine learning: a systematic mapping study. In: ICSOFT 2021, SciTePress, online, pp 521–531
24. Zhong Z et al (2022) Machine learning-based orchestration of containers: a taxonomy and future directions. ACM Comput Surv 54(10s):1–35
25. Duc TL et al (2019) Machine learning methods for reliable resource provisioning in edge-cloud computing: a survey. ACM Comput Surv 52(5):1–39
26. Nayeri ZM et al (2021) Application placement in Fog computing with AI approach: taxonomy and a state of the art survey. JNCA 185:103078
27. Nguyen HX, Zhu S, Liu M (2022) A survey on graph neural networks for microservice-based cloud applications. Sensors 22(23):9492
28. Saucedo A, Rodríguez G (2024) Migration of monolithic systems to microservices using ai: a systematic mapping study. In: Anais do XXVII Congresso Ibero-Americano em Engenharia de Software, SBC, ???, pp 1–15. https://doi.org/10.5753/cibse.2024.28435
29. Wang T, Qi G (2024) A comprehensive survey on root cause analysis in (Micro) services: methodologies, challenges, and trends. https://arxiv.org/pdf/2408.00803
30. Zhang S, Pei D (2024) Failure diagnosis in microservice systems: a comprehensive survey and analysis. https://arxiv.org/abs/2407.01710
31. Cheng Q et al. (2023) AI for IT operations (AIOps) on cloud platforms: reviews, opportunities and challenges. https://arxiv.org/abs/2304.04661v1
32. Wohlin C (2014) Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: EASE 2014, pp 1–10
33. Kitchenham B, Charters S (2007) Guidelines for performing systematic literature reviews in software engineering
34. Kitchenham B, Brereton P (2013) A systematic review of systematic review process research in software engineering. IST 55(12):2049–2075
35. Petersen K, Feldt R, Mujtaba S, Mattsson M (2008) Systematic mapping studies in software engineering. EASE, pp 1–10
36. Cohen J (1960) A coefficient of agreement for nominal scales. Edu Psychol Meas 20(1):37–46
37. Soldani J et al (2021) The μTOSCA toolchain: mining, analyzing, and refactoring microservice-based architectures. Softw Pract Exp 51(7):1591–1621
38. Guidotti R, Monreale A et al (2018) A survey of methods for explaining black box models. ACM Comput Surv 51(5):1–42. https://doi.org/10.1145/3236009
39. Ampatzoglou A et al (2019) Identifying, categorizing and mitigating threats to validity in software engineering secondary studies. Inf. Softw Technol 106:201–230