electronics

MDPI

Article

A Survey of Energy Optimization Approaches for Computational Task Offloading and Resource Allocation in MEC Networks

Jinming Yang, Awais Aziz Shah and Dimitrios Pezaros *

School of Computing Science, University of Glasgow, Glasgow G12 8QQ, UK; j.yang.8@research.gla.ac.uk (J.Y.); awaisaziz.shah@glasgow.ac.uk (A.A.S.)
* Correspondence: dimitrios.pezaros@glasgow.ac.uk

Abstract: With the increased penetration of cloud computing and virtualization, a plethora of internet of things devices have been deployed globally. As a result, computationally intensive tasks are transmitted from the edge towards the centralized cloud for processing that leads to increased energy utilization in the cloud data centers while at the same increasing significant latency for critical applications. Recent years have witnessed a paradigm shift from centralized cloud computing towards mobile edge computing (MEC), where computational tasks are offloaded at the edge servers near user equipment (UE). This paradigm leads to lowering the energy utilization in the cloud data centers, along with low latency for UE and efficient resource utilization at the edge. In this context, the scale and complexity of the MEC networks is drastically increasing and, consequently, finding effective energy-efficient solutions for computational task offloading and resource allocation in MEC networks has become an ambitious task. To address the aforementioned challenges, this work surveys the state of the art in different categorizations of algorithm-based computational task offloading and resource allocation strategies focusing on energy utilization. It also provides a detailed cross-comparison of existing strategies in terms of their implementation specifications. Additionally, this paper also highlights open challenges and potential future research directions to facilitate efficient task offloading and resource allocation at the edge with reduced energy consumption at the centralized data centers. Our work also paves the way for the deployment of critical applications at the edge that require low latency and high service quality guarantees.

Keywords: MEC networks; computation offloading; resource allocation; energy efficiency

Citation: Yang, J.; Shah, A.A.; Pezaros, D. A Survey of Energy Optimization Approaches for Computational Task Offloading and Resource Allocation in MEC Networks. *Electronics* **2023**, *12*, 3548. https://doi.org/10.3390/electronics12173548

Academic Editor: Martin Reisslein

Received: 15 July 2023
Revised: 11 August 2023
Accepted: 16 August 2023
Published: 22 August 2023

1. Introduction

The last decade has witnessed rapid growth in the internet of things (IoT) paradigm. This includes not only growth in the number of heterogeneous devices but also the numerous types of sensors and use cases that require high data rates and low latency, e.g., remote surveillance, home automation devices, smart farming, intelligent mobile devices, and autonomous vehicles [1]. In this context, there is a rapid increase in industrial applications focusing on advancements in different sectors such as transportation, defense, healthcare, mobility, and so on. Typically, IoT applications are highly computationally intensive and require real-time processing with strict quality of service (QoS) requirements, i.e., high throughput and low latency [2,3]. User equipment (UE) has limited battery capacity and resources, which is often not well suited for complex computational tasks. Thanks to the advancements in cloud computing, mobile cloud computing (MCC) paved the way to transfering these computationally expensive tasks to the cloud for execution. For this purpose, large amounts of networking resources are required to ensure reliable data communication between the end device of the edge and the cloud. To this end, two main issues arise: low latency and increasing workloads transferred to the centralized cloud. A fundamental constraint of MCC, however, is the large transmission distance between the UE and the

remote cloud center. Apparently, the MCC is unsuitable for tasks that involve high-density data transfer and have stringent delay requirements. Moreover, processing large quantities of data at the MCC end can lead to increased computation and storage demands, which can result in overburdening the cloud servers [4].

Mobile edge computing (MEC) was proposed in 2014 to enable computations at the edge [5]. It represents a novel computing model that enables low-latency task computing and enhances the computing capabilities of UE. Since the MEC servers are close to the UE, tasks are processed at the edge with lower latency and less workload being transferred to the cloud [6]. MEC servers provide a limited degree of resources for task offloading at the edge which eventually leads to reduced energy consumption at the cloud and efficient resource utilization at the edge. However, unlike the cloud, MEC servers have limited resources to execute tasks. Therefore, firstly, there is a demand for optimized computation offloading strategies on the edge for executing a large number of concurrent tasks produced by the UE with QoS requirements. Secondly, efficient task offloading schemes are required to keep the computing load balanced between the MEC servers and the cloud.

Currently, the cost of energy utilization and worldwide carbon emissions is becoming a primary concern for governments [7]. The carbon footprint produced by cloud data centers can be substantially reduced with the advancements in distributed edge computing. Overall, the combination of efficient resource allocation strategies with task offloading schemes on MEC servers will provoke low energy costs, reduce the workload on the cloud data centers, and contribute towards net-zero goals [8].

In this context, this work surveys the state-of-the-art computational task offloading strategies and resource allocation schemes focusing on the minimization of energy consumption in MEC networks. To the best of our knowledge, there are no comprehensive survey papers that primarily focus on the minimization of energy consumption while considering both computation offloading and resource allocation issues. This includes covering offloading details, introducing exhaustive resources, employing relevant energy-saving techniques, and implementing algorithms in the MEC networks. The organization of this paper is given in Figure 1. The main contributions of this work are:

- A survey of the current state-of-the-art strategies for computational task offloading and resource allocation for energy minimization in MEC networks.
- A detailed description of the MEC architecture and its classification in terms of network topology coupled with various components implemented in the current literature.
- A thorough depiction of computational task offloading along with different emerging use cases.
- A category of multiple types of resources for executing computational tasks based on energy utilization.
- A reflective discussion of open challenges and future directions associated with energy-efficient MEC networks.

The rest of the paper is organized as follows: Section 2 summarizes the related research performed from the perspective of task offloading and resource allocation. Section 3 discusses the topology, components, and classification of MEC network architecture along with ETSI and 3GPP communication standards. A description, use cases, and categories of computational task offloading are given in Section 4. Resource allocation in MEC networks and existing strategies are presented in Section 5. Section 6 provides a cross-comparison between existing computational task offloading and resource allocation strategies in MEC networks, focusing on energy consumption and optimization. Finally, Section 7 highlights the open challenges and potential future directions.

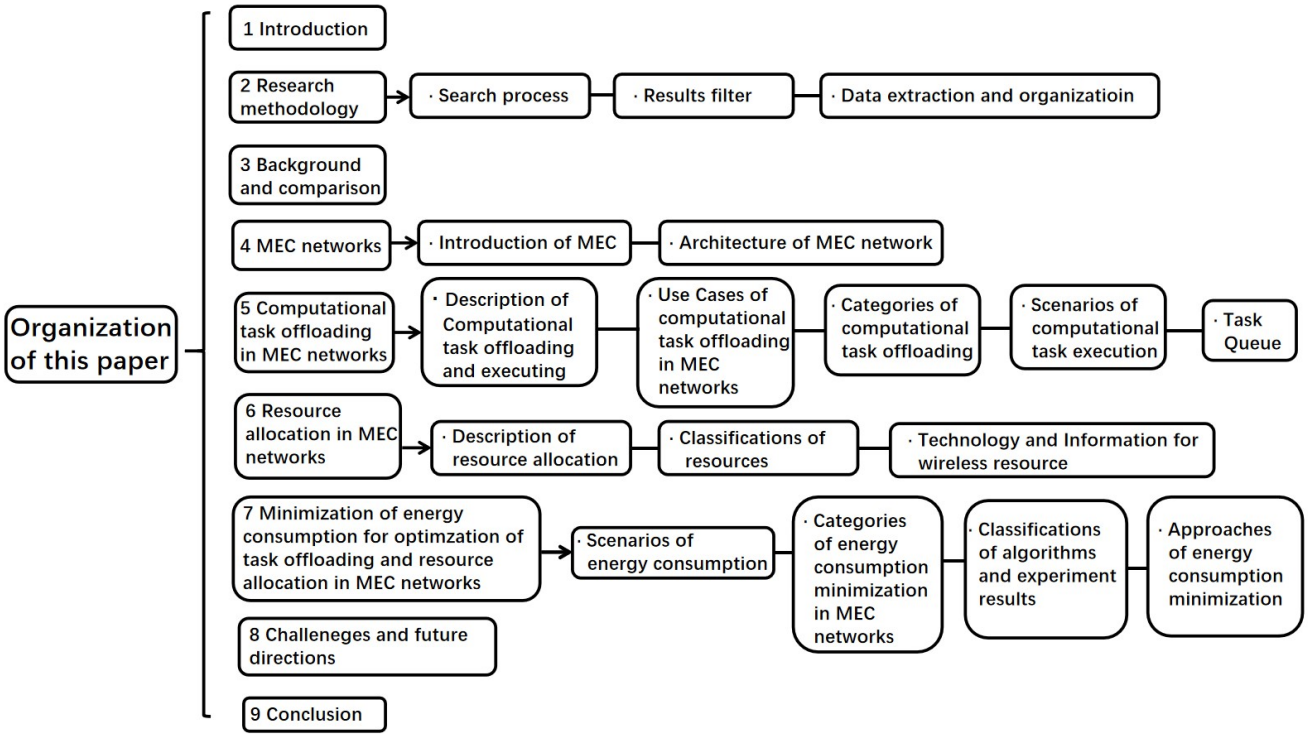

Figure 1. Organization of this paper.

2. Research Methodology

This section highlights the search, selection, and data extraction processes of this survey, as depicted in Figure 2.

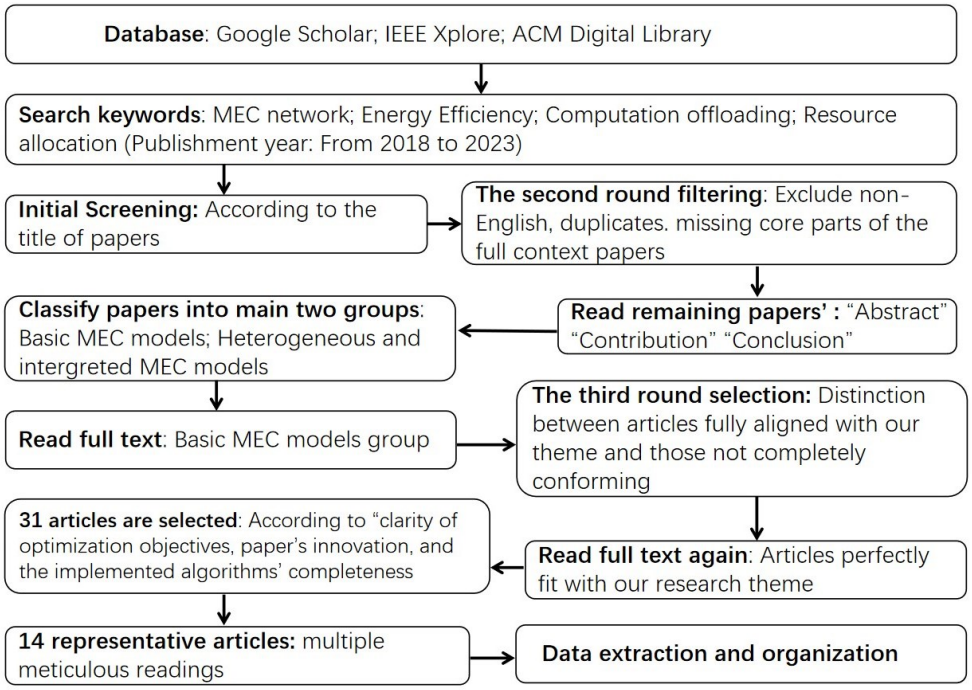

Figure 2. Research methodology.

2.1. Search Process

In order to study the existing research on energy-efficient computational task offloading and resource allocation in MEC networks, we initially selected three comprehensive and influential academic databases: Google Scholar, IEEE Xplore, and ACM Digital Library. Within these three databases, we searched for publications within the last five years (i.e., from 2018 to 2023) using the keywords "MEC network", "Energy efficiency", "Computation offloading", "Computational task offloading", and "Resource allocation". After the first round of filtering, a total of 332 articles were downloaded based on scrutiny of their titles.

2.2. Results Filter

In the next stage, papers that were non-English, duplicates, or missing core parts of the full text were excluded. The first author of this paper then read the "abstract", "contribution", and "conclusions" sections of the remaining papers, conducting an in-depth analysis of the specific research and main focus. Ultimately, a total of 95 associated articles were shortlisted. Then, the shortlisted papers were categorized into groups: basic MEC models without special communication scenarios (consisting of common UE end, access nodes (ANs), and MEC servers, etc.), as well as heterogeneous and integrated MEC models (e.g., unmanned aerial vehicle (UAV) scenarios in MEC networks) and the virtualized network function (VNF)-MEC network). Additionally, the remaining articles were tagged as supplementary references for our work (e.g., ETSI MEC and 3GPP standards [9,10]). The results of each stage of the selection process were discussed for feedback with the second author of this paper.

Upon meticulously reading the articles in the "basic MEC models" group, we realized that some of them did not comprehensively consider computation offloading combined with resource allocation to optimize network energy efficiency. For example, some focused only on resource allocation within the basic MEC network architecture, neglecting the subjects of network energy and computation offloading. Others were focused on computation offloading and resource allocation to optimize the energy efficiency of the UAV-MEC network, however, they did not pertain to the basic MEC network architecture. Therefore, in the third round of selection, we rigorously categorized the articles, distinguishing between those that strictly aligned with our research theme and those that did not fully conform to it. Following that, we carried out a deep reading and evaluation of these articles that perfectly fit with our research theme, considering factors such as the clarity of optimization objectives, innovation of the paper, and completeness of the implemented algorithms. After careful selection, 31 articles were chosen, and from those, we picked 14 highly representative main works for multiple meticulous readings. Through this process, we extracted relevant data, which formed the basis for the creation of the tables subsequently.

2.3. Data Extraction and Organization

We meticulously studied these articles through the third round of filtering, further selected relevant data, and organized them into a table. The outcome of the whole process is presented in Table 1. First, we conducted a thorough analysis of the basic MEC network architecture, focusing on specific details such as the types and quantities of AN, and the categories of wireless access technology (WAT). These sorting and recording details were used as the foundation to compile Table 2. It is worth mentioning that Table 2 also includes information about roadside units (RSUs) as the AN in the autonomous vehicle (AV) scenario. Although the AV-MEC scenario is not our main research field, as it does not belong to the basic MEC network model, we included it in Table 2 since RSUs are one of the important types of AN. Additionally, to illustrate the centralized and decentralized characteristics of the MEC network, we referenced two other articles [11,12] in Table 2 to supplement the completeness of the MEC architecture.

Next, the first and second authors attentively read the content related to computation offloading and resource allocation in these articles, and systematically categorized and recorded them. Important information includes the types of task offloading, the locations

of task execution, the optimization methods for computation offloading, as well as the optimization strategies for resource allocation, etc. Table 3 shows the important information and comprehensive record.

Finally, an exhaustive analysis of the implemented algorithms was presented for each article, categorizing them according to different types of algorithms (e.g., game theory, ML-based algorithms). Subsequently, we also provided a detailed description of each type of algorithm and analyzed their specific functions and roles. All this information is compiled and organized into Table 4.

3. Background and Comparison

Existing studies concentrate on the areas of computation offloading, resource allocation, and energy efficiency in MEC networks. The introduction of these existing efforts and illustration of our work is further explained in this section. In brief, Table 1 represents the research focus of previous surveys and this work.

Table 1. Main research areas of previous surveys vs. our work.

Ref.	Computation Offloading	Resource Allocation	Maximization of Energy Efficiency
[13]	✓	×	×
[14]	✓	×	×
[15]	✓	×	×
[16]	×	×	✓
[17]	×	✓	×
[18]	✓	✓	×
[19]	✓	✓	×
Our work	✓	✓	✓

The existing survey papers mainly focus on the area of computation offloading. For example, ref. [13] focuses on task offloading strategies according to the computational model and decision-making model in different MEC architectures and grouped the offloading algorithms as well as provided some use cases. Analogously, computation offloading, offloading objectives, and methodologies were presented in [14]. The objectives are organized into four categories: the minimization of latency and energy consumption and the maximization of revenue and system utility. Moreover, they classified and summarized the related implementing algorithms. Furthermore, the survey in [15] discussed computation offloading issues in general MEC network models, and provided six types of optimization methodologies to achieve different objectives.

In addition, energy-efficient MEC networks have attracted much attention in recent years. As mentioned in [16], the authors researched energy-aware edge networks, which included the hardware design, edge architecture design, edge operation system (OS), edge middleware between applications and OS, edge services, applications in the application layer, and energy-efficient computation offloading. Moreover, there are several examples of resource management and scheduling considering energy efficiency in the area of edge OSs.

From another point of view, efficient resource allocation approaches play a pivotal role in MEC networks. Several technologies for resource allocation and some related resources in an IoT environment were described by [17]; moreover, this paper illustrated the impact of IoT integration in enterprise architecture on resource allocation, task management, network architecture, and information system integration.

Meanwhile, many researchers have conducted surveys on the direction of computation offloading in conjunction with resource allocation. For instance, ref. [18] provides a detailed illustration of various MEC architectures according to MEC concepts, such as

small cell cloud (SCC), mobile micro cloud (MMC), European Telecommunication Standards Institute (ETSI) MEC, etc. The study also explores issues of computation offloading, resource allocation, and mobility management in MEC networks. The main elements of the objectives in this work involve energy efficiency, latency, and load balancing. A more recent survey [19] provides a three-tier edge network architecture (the edge layer, cloud layer, and "thing" layer), and discusses computational task offloading, resource allocation, and resource provisioning in basic network models. Furthermore, different categories of algorithms and some resource scheduling application scenarios were demonstrated.

These surveys have been published approximately within the last five years. However, an increasing number of surveyed new papers featuring novel contributions have emerged in recent years. For instance, ref. [20] proposed that due to the dense deployment of base stations (BSs) and MEC servers, the issues of task offloading and resource scheduling in ultra-dense networks (UDNs) are facing severe challenges. Therefore, we mention the relevant content of the UDN-MEC in our study. On the other hand, caching in MEC networks has emerged as a significant area of research interest, due to its ability to minimize electric usage from computational offloading, while also enhancing data security [21]. Hence, caching technology is also discussed in this work.

From the perspective of the novelty, this survey distinguishes itself from previous works in several ways. Firstly, we introduce unique perspectives by focusing on computation offloading and resource allocation within MEC networks, with a special emphasis on energy efficiency, an aspect not extensively explored in the existing surveys, as shown in Table 1. Secondly, we start with a basic introduction of MEC as well as the ETSI MEC and 3GPP standards and provide a comprehensive exploration of MEC networks, clearly defining network topology, and diverse network components. We provide insights into the development of standards and specifications for future networks, as well as our perspectives on different classifications of MEC architecture, aiming to facilitate the accurate assessment of MEC network models under various scenarios. This approach will equip the readers and research community with a comprehensive understanding of MEC networks and architectures, energy-efficient computational task offloading and resource allocation in MEC networks, and relevant standardization knowledge. This work offers consideration of the comprehensiveness and sufficiency of the computation offloading procedures under various use cases. This establishes a standard for task offloading and resource allocation in dynamic and complex real-world MEC networks, enabling readers to gain a comprehensive understanding in this field. Furthermore, a close examination and discussion have been performed, focusing on computational task offloading and resource allocation strategies to minimize network energy consumption. We describe the relationship between the energy consumption of different network components (e.g., UE, MEC end, transmission) and computation offloading and resource allocation, providing detailed information. We have extensively analyzed the energy consumption criteria with various objectives, such as minimizing energy consumption at the UE, MEC end, or network cost, etc. This understanding helps readers to deeply grasp the interconnection between energy consumption, computation offloading, and resource allocation within MEC networks and to identify strategies and structures aimed at minimizing energy consumption within the network. Additionally, task offloading, resource allocation, and various energy consumption minimization approaches have been discussed and analyzed in order to offer guidance to readers in choosing appropriate algorithms and adopting different energy consumption minimization techniques. Lastly, this work provides perspective future directions and open challenges paving the way for further research on energy-efficient computational task offloading and resource allocation in MEC networks in the future.

4. MEC Networks

This section provides a comprehensive overview of MEC networks, standardization approaches, and sheds light on the topology, components, and classifications of the MEC network architecture.

4.1. Introduction of MEC

MEC has garnered wide interest as an innovative computing framework to alleviate the intensive computation strains from the conventional cloud end. It was initially introduced by the ETSI in 2014 [5]. The ETSI MEC architecture partitions the MEC system into two tiers: the system level and the host level. The system level orchestrates different hosts within the MEC networks, with a single MEC system governing multiple active hosts. Regarding the host level, it includes multiple servers along with the management infrastructure. Firstly, the management infrastructure has MEC platform management and virtualization infrastructure management (VIM). Secondly, the servers consist of two sections: the virtualization infrastructure, which can receive resource requests from VIM, offer virtual resources, and deploy and run applications or services in servers; and the MEC platform, which can provide and instantiate applications and services in the virtualization infrastructure [22].

4.2. Architecture of MEC Network

This sub-section explores the architecture of the MEC network, analyzing its associated standards, topology, components, and various classifications. First, we begin with the ETSI MEC and 3GPP standards, introducing their specifications and deployment for MEC applications in the network architecture. Then, we present the basic topology structure of the MEC network, analyzing the relationships and functionalities across different levels. Next, there are details of the diverse components of the network architecture, exploring their roles and significance within the MEC network. Finally, we classify and describe the six different categories based on MEC architecture.

4.2.1. ETSI MEC and 3GPP Standards

ETSI ISG (industry specification group) MEC is the origin of edge computing standards and has released a series of specifications. It primarily focuses on the management and orchestration (MANO) of MEC applications, application enablement API, service APIs, and UE application API. Service environments that are enabled benefit from MANO and application enablement functions, while service APIs expose underlying network information and functions to applications. Hence, a key characteristic of MEC specifications is that applications can obtain context-aware information through these standardized APIs, achieving real-time perception of the local service environment. This local service environment provides a flexible and expandable framework that can introduce new services to create new service APIs. Finally, the UE application API allows client applications in the UE to interact with the MEC system to manage the application's lifecycle [9].

According to 3GPP next-generation network specifications, 5G and 6G networks are the key future target environments for MEC deployments. Responding to this, 3GPP has recently initiated several new measures aimed at integrating MEC into these next-generation network systems. Regarding standardization activities in 3GPP, working group (WG) SA1 is standardizing service requirements. These service requirements are regarded as "stage 1" (high-level requirements) and have triggered related research from downstream WGs (such as system architecture, SA2, network management, SA5, security, SA3, and application architecture, SA6). Among them, two service requirements (i.e., Rel-15 TS 22.261) laid the foundation for edge computing to support next-generation services, which are resource efficiency and an efficient user plane. In addition, some other standard development organizations (SDOs) have also worked on edge computing from different angles, such as the Internet Engineering Task Force (IETF) [10].

In summary, ETSI allows MEC applications to be flexibly deployed in network architectures through standardized APIs, greatly enhancing the efficiency and flexibility of the MEC system. Moreover, the next-generation network specifications of 3GPP provide a key environment for MEC deployment. As 5G and 6G networks further develop and are applied, MEC deployment will become more widespread, while various standard organizations will continue to play an important role in standard setting and optimization.

It is expected that more standards and specifications will be proposed to meet the new challenges and demands faced by MEC in more application scenarios.

4.2.2. Complete Basic Topology

The basic MEC architecture involves multiple computational tasks that can be processed locally on the UE side or offloaded to the MEC or cloud end. The complete MEC network topology is depicted in Figure 3.

From the UE to the edge layer, the UE side contains diverse terminal equipment, such as mobile phones, autonomous vehicles, and UAVs, which can offload computational tasks to nearby ANs through WAT. The widely used WATs indicate non-orthogonal multiple access (NOMA), frequency division multiple access (FDMA), orthogonal frequency division multiple access (OFDMA), time division multiple access (TDMA), etc. The edge layer includes MEC servers and access networks (ANs) comprised of BSs, access points (APs), and RSUs. More information about ANs and WAT is described in Section 4.2.3 and Section 6.3.1, respectively.

For the edge layer, a MEC server often consists of multiple MEC hosts. It directly connects to the AN via an optical fiber, and the transmission delay between them can be ignored [23].

With regard to the edge layer and core layer, the connection between ANs and core network devices (e.g., switches/routers/middle-box devices) can be wired or wireless [24]. The back-haul link is from the UE side to the core network layer. The virtual private network (VPN) serves as a pathway between the cloud end and the core network layer.

Additionally, ANs feature parameter collection and specific modules for calculating the number of items of UE linked to them. With this information, ANs can allocate wireless resources such as bandwidth, and balance the workload among themselves [12]. However, the MEC system can monitor the state of wireless resources via the interface between the MEC server and BS, which is crucial to optimize the resource allocation scheme in the whole MEC network [25].

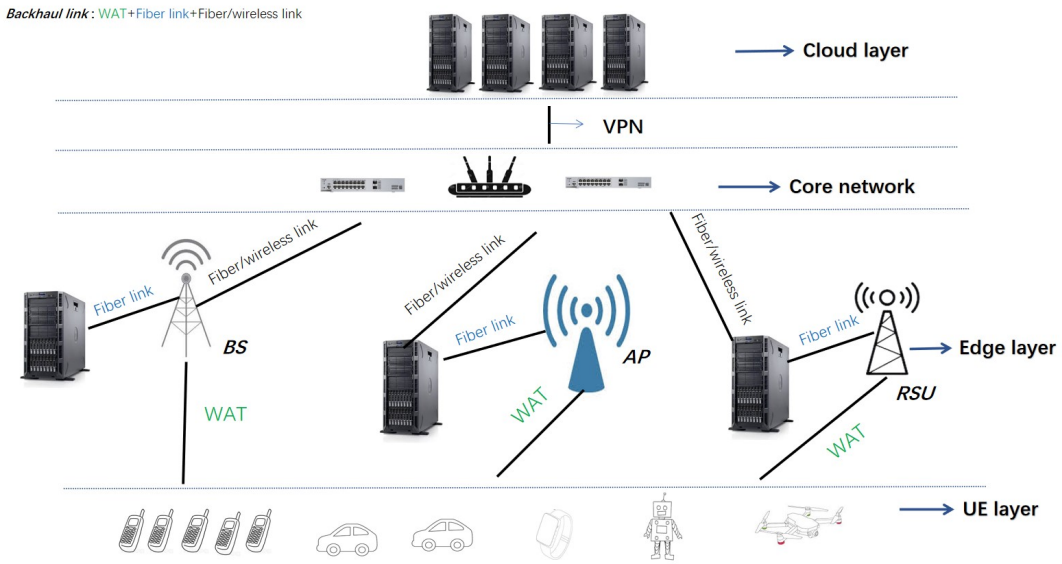

Figure 3. Complete basic topology.

4.2.3. Components of Network Architecture

This paper uses multiple terms. Firstly, MEC nodes involve edge network components, e.g., MEC servers, cloud end, edge routers, or switches. Secondly, we consider the MEC servers and management infrastructure are comprised of the MEC system. Next, the MEC

network encompasses the entire network, which includes UE, MEC systems, and ANs. Finally, the MEC end refers to either a single MEC server or multiple MEC servers.

There are often multiple UE devices and a large amount of computational tasks in MEC networks. The number of ANs and MEC servers depends on diverse network models. Basically, in the scenarios of multiple MEC servers, MEC networks are divided into decentralized and centralized layouts. In a centralized system, a controller is responsible for managing the MEC network globally. On the contrary, in decentralized networks, each server can make offloading decisions and resource allocation actions independently, rather than relying on a controller. Particularly, some MEC models may not explicitly mention the inclusion of MEC servers or ANs. Since each BS is equipped with computational resources to serve offloaded tasks, we can either ignore the description of the MEC end [26], or we can consider the MEC server and AN as an edge layer while neglecting the information about ANs [27,28]. Alternatively, ref. [29] focuses primarily on the perspective of the MEC end, thus there is no description of ANs.

Specifically, ref. [30] restructures the MEC system by integrating remote radio heads (RRHs) into BSs, and designating the baseband unit (BBU) in the MEC system to receive offloaded computational tasks from UE via uplink and sending the results back via downlink. Upon receiving the offloading request, the MEC system allocates virtual machines (VMs) from the BBU pool to provide computational resources for UE. Therefore, the MEC system is split into RRHs and corresponding VMs, effectively replacing the traditional concept of BSs and MEC servers.

In regards to ANs, a BS is the most common type of AN. Several classic BSs are categorized by coverage area, such as macro cell base station (MBS) and small cell base station (SBS). SBSs encompass femto cell BSs, pico cell BSs, and micro cell BSs (mBSs). For instance, multiple low-power SBSs are deployed in an MBS, taking the MBS as a manager that is in charge of collecting network information and controlling admission of offloading and resource allocation. The MBS is equipped with one MEC server to execute computational tasks. UE can offload tasks to the MEC server through SBSs to the MBS [31]. The connection between the SBS and MBS can be wired fibers [32], or wireless links [31]. Alternatively, only one MBS maintains a cloud server, and the SBSs host multiple MEC servers [33].

From the perspective of the WATs, e.g., 3G,4G,5G, NOMA, BSs can be of different types such as NodeB, eNB, and gNB. For example, UE can connect to the MEC in two approaches, one is via 5G access to the gNB, and the other is via WiFi access to an AP [34]. In addition, depending on the number of antennas, the categories of BS can contain single-antenna BS [35] or multi-antenna BS [36].

AP is another category of AN. In the WiFi access approach, AN typically refers to an AP, which could be installed in the user's homes, businesses, or public places [22]. In vehicular networks, vehicles can offload their computational tasks to the network through RSUs [37]. The RSUs integrated with MEC servers' capabilities are deployed along roads and can serve multiple vehicles within their coverage area [38].

Considering the load balancing problem in MEC networks, optimizing the computation offloading strategy can be achieved by balancing the workload among APs [39]. Furthermore, computational task migration among multiple MEC servers can keep the workload balanced [11,12,35]. Particularly in networks involving mBSs and an MBS, the mBSs can manage the queue of arriving computational tasks, controlling and forwarding them to the MBS, thus maintaining workload balance [40].

4.2.4. The Classifications of MEC Network Architecture

In this section, we subdivide the MEC network architecture into the following six categories. The details of network architecture categories and other related information are listed in Table 2.

1. One AN, one MEC server
   There is one AN connected to a single MEC server in this model; UE can offload computational tasks to this AN based on offloading decisions. The server provides resources for all computational tasks. However, this scenario is typically applicable in limited environments, such as small offices, where one MEC server cooperates with one AN to establish a high-speed and low-latency MEC network. This model is shown in Figure 4a.
2. Multiple ANs, one MEC server
   As shown in Figure 4b, this network design includes one server and multiple ANs. Computational tasks can be offloaded to the appropriate ANs according to the optimized offloading strategy. In an industrial internet of things (IIoT) environment, industrial end units (IEUs) are equivalent to UE. They are capable of offloading computational tasks to the MEC server deployed in edge data centers via multiple ANs. The industrial app server also housed in edge data centers can generate industrial applications instances to serve the MEC server, enabling the offloaded tasks to be executed within this server [34]. Nevertheless, it could potentially overload the MEC server when large amounts of computational tasks need to be offloaded simultaneously.
3. One AN, multiple MEC servers
   In particular, multiple servers can be hosted within a single AN. Once computational tasks are offloaded to the AN, the MEC system determines the specific server for task execution. Apparently, this network layout can balance the workload among servers. This configuration could appear in some specific application scenarios. For example, a BS can provide network interfaces, and multiple servers can execute computational tasks (e.g., scientific simulation or image processing) in parallel to improving processing efficiency. This model is shown in Figure 4c.
4. Multiple ANs, multiple MEC servers
   This category of MEC architecture includes multiple ANs and servers. However, within a single AN, there can be deployed one or multiple servers. Furthermore, the UDN is a classic scenario of this network layout. The UDN aims to significantly increase the number of ANs, in order to enhance network coverage and capacity [41]. Consequently, the model of multiple ANs and servers can better simulate real-world MEC networks. In other words, computational tasks can be offloaded to the appropriate AN and servers, depending on the offloading strategies. This model not only avoids overloading a single server but also benefits from the cooperation of multiple servers. For instance, in the context of video streaming cached by the MEC end, UE can send a video request to the local BS. If the video stream has already been cached in the local server, the video will be directly transmitted to the UE via the local BS. If not, the local MEC server will send the request to the cooperative servers deployed in other BSs to share the video or download the video stream from the cloud end [42]. This configuration is depicted in Figure 4d.
5. One MBS, multiple SBSs/other types of ANs, one MEC server
   This is a special design of the MEC network model involving MBS and SBS (or other types of ANs), as shown in Figure 4e. For example, one MBS covers multiple SBSs in a certain region. UE can offload computational tasks through appropriate SBSs, which are then forwarded to the MEC server deployed in the MBS. Once the task execution is completed, the data are transmitted back to the SBS and then to the UE side. This MBS covering multiple SBSs can effectively balance the network load and reduce the pressure on a single node [40]. To this end, this model is well suited for large facilities or densely populated areas, e.g., shopping malls and gymnasiums, where multiple SBSs can manage computational task offloading in high-traffic areas, and the MBS can provide computational resources for task execution.
6. One MBS, multiple SBSs/other types of ANs, multiple MEC servers
   Figure 4f presents a scenario where each of multiple SBSs (or other types of ANs) is integrated with an MEC server. Additionally, the MBS is equipped with a cloud

server that can provide more resources. With the aid of the cloud server, this network model better regulates offloading strategies and resource allocation. Hence, it is highly suitable for scenarios that require a large number of computational resources. In vehicle networks, a single MEC server hosted in the RSU can provide low-latency services for computational tasks offloaded by AVs, and the cloud server in the MBS can be used to reduce the computational burden of the edge layer [43].

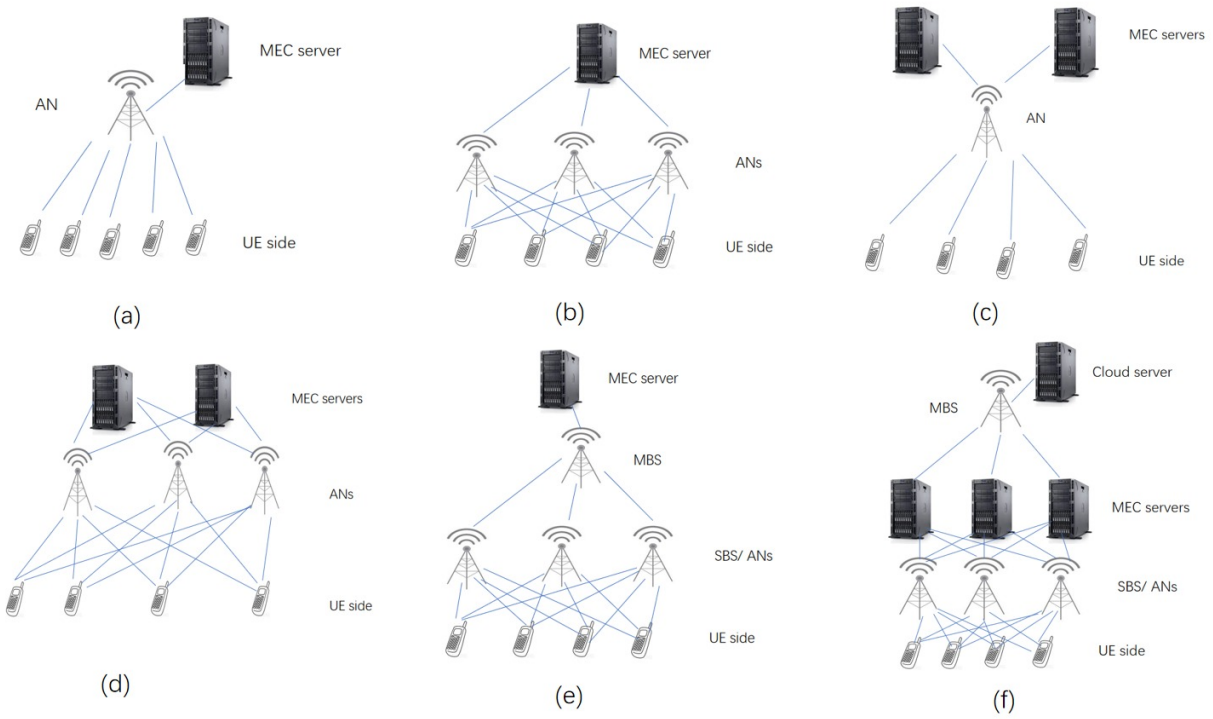

Figure 4. (**a**) One AN, one MEC server; (**b**) multiple ANs, one MEC server; (**c**) one AN, multiple MEC servers; (**d**) multiple ANs, multiple MEC servers; (**e**) one MBS, multiple SBS / other types of ANs, one MEC server; (**f**) one MBS, multiple SBS / other types of ANs, multiple MEC server.

Table 2. Implementation of MEC architecture in the literature.

Ref.	Type of AN	Number of ANs (One/Multiple)	WAT of Computation Offloading	Number of MEC Servers (One/Multiple)	Cloud Participation (Yes/No)	Decentralized MEC Layout (Yes/No)	Classification of MEC Architecture
[35]	BS	Multiple	TDMA	Multiple	×	✓	Multiple ANs, multiple MEC servers
[20]	BS	Multiple	OFDMA	Multiple	×	✓	Multiple ANs, multiple MEC servers
[36]	BS	One	-	One	×	✓	One AN, one MEC server
[33]	MBS SBS	One Multiple	OFDMA	Multiple	✓	✓	One MBS, multiple SBSs, multiple MEC servers
[26]	BS	Multiple	-	-	×	✓	Multiple ANs, multiple MEC servers
[28]	-	Multiple	NOMA and FDMA	Multiple	✓	✓	Multiple ANs, multiple MEC servers
[39]	AP	Multiple	-	One	×	✓	Multiple ANs, one MEC server
[44]	BS	One		One	×	✓	One AN, one MEC server
[41]	BS	One	TDMA	One	×	✓	One AN, one MEC server
[34]	gNB (5G); WiFi AP (WiFi 6)	Multiple Multiple	-	One	×	✓	Multiple ANs, one MEC server
[27]	-	Multiple	-	Multiple	✓	✓	Multiple ANs, multiple MEC servers
[11]	SBS	Multiple	-	Multiple	✓	×	Multiple ANs, multiple MEC servers
[12]	BS AP	Multiple Multiple	-	Multiple	×	×	Multiple ANs, multiple MEC servers

Table 2. *Cont.*

Ref.	Type of AN	Number of ANs (One/Multiple)	WAT of Computation Offloading	Number of MEC Servers (One/Multiple)	Cloud Participation (Yes/No)	Decentralized MEC Layout (Yes/No)	Classification of MEC Architecture
[43]	RSU	Multiple	-	Multiple	✓	×	One MBS, multiple SBSs/other types of ANs, multiple MEC servers
[40]	mBS MBS	Multiple One	-	One	×	✓	One MBS, multiple SBSs/other types of ANs, one MEC server
[30]	BS	Multiple	-	Multiple	×	✓	Multiple ANs, multiple MEC servers

AP: access point; AN: access node; BS: base station; FDMA: frequency division multiple access; gNB: gNodeB; MBS: macro cell base station; MEC: mobile edge computing; mBS: micro base station; NOMA: non-orthogonal multiple access; OFDMA: orthogonal frequency division multiple access; RSU: road side unit; SBS: small cell base station; TDMA: time division multiple access; WAT: wireless access technology.

5. Computational Task Offloading in MEC Networks

In MEC networks, selecting the appropriate resource/location for computational task execution is an ambitious task. This section first provides a detailed introduction related to the task offloading process. Subsequently, we present a variety of real-world use cases associated with computation offloading, demonstrating the different categories and scenarios of task offloading. Finally, detailed information pertinent to the task queue during the computation offloading is explained.

5.1. Description of Computational Task Offloading and Executing

In MEC networks, once the UE generates some computational tasks, the OS on the UE side determines offloading decisions based on the latency, offloading strategy, and the state of the UE side (e.g., the battery status and available resources).

In order to execute the tasks at the UE device's end, three possible conditions exist: execute the task at UE, offload the tasks to the MEC end, or offload the tasks to the cloud end. When the OS decides to offload computational tasks to the MEC end, the UE sends offloading requests and resource requests to it via WL. Subsequently, the MEC system might continue to offload the computational tasks to the cloud end due to insufficient resources or specific strategies. Alternatively, offloading requests might be rejected for various reasons, such as a lack of resources at servers, problems with the wireless channel during offloading, or certain computational tasks reaching their time limit.

When computational tasks are approved for offloading to the MEC end, they either begin being offloaded via WL immediately or wait for offloading in a queue managed by the UE side. In some cases, the ANs that receive the offloaded computational tasks from the UE are not close to the servers, hence the ANs need to maintain queues and forward tasks to the MEC end for execution. Furthermore, the MEC system is responsible for allocating relevant communication and computational resources, e.g., transmission power and CPU frequency, to these offloading tasks.

Once the offloaded computational tasks arrive at the MEC end, it processes them or places them in a queue to await execution. After execution, the tasks are returned to the UE, or else they remain in a queue on the MEC end. When the computational results arrive at the UE side, they are downloaded or stored in a queue managed by the UE side, waiting for an appropriate time to be downloaded. The summary of queues is discussed in Section 5.5.

Additionally, based on the analysis of the respective status and resource-related information from the UE side and the MEC end, they can jointly decide on offloading computational tasks to the cloud end. In a centralized MEC network, the controller is able to formulate a reasonable offloading strategy. For simplicity, we will not reiterate the process of task offloading to the cloud end, as it is analogous to the MEC end. The entire process of task offloading is illustrated in Figure 5.

Note that offloading computational tasks to servers does not always guarantee energy efficiency, as it depends on factors such as the device and server type, and task execution locations. In addition, energy consumption varies between the UE, MEC, and cloud servers due to dynamic networks. Therefore, inefficient offloading strategies can lead to lower energy efficiency, higher electricity costs, and larger carbon footprints [16].

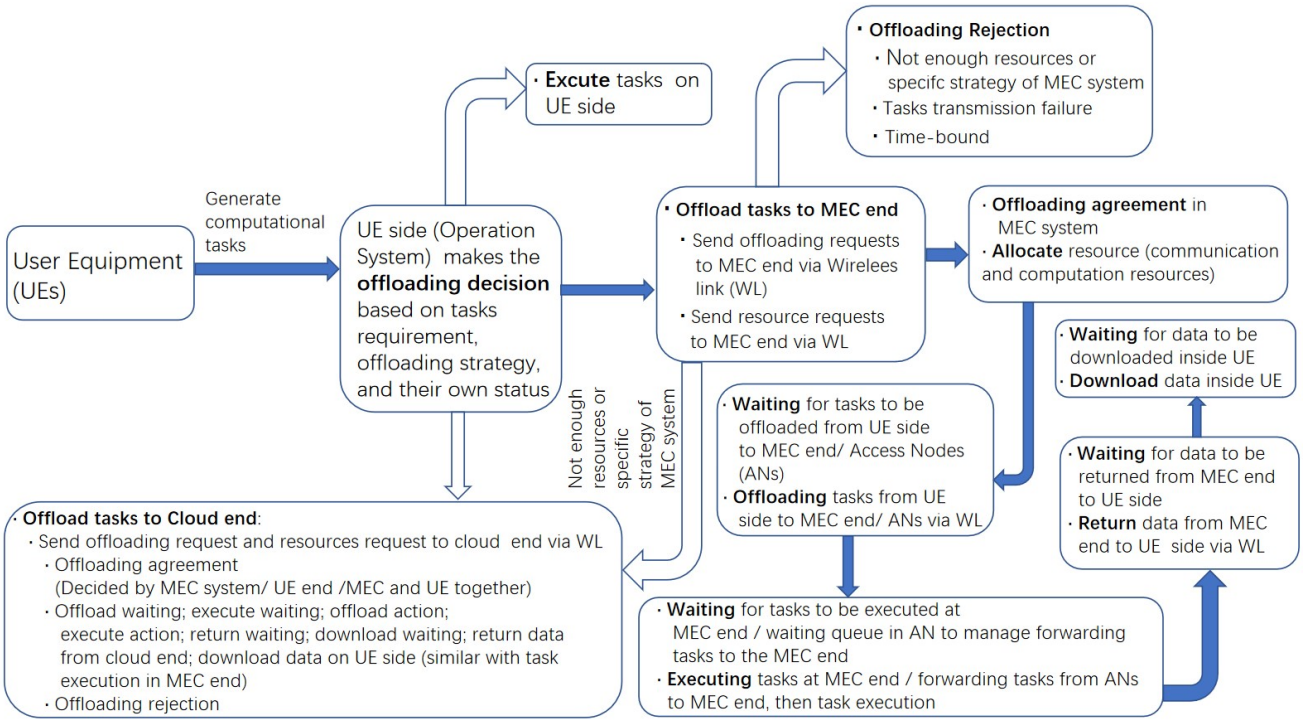

Figure 5. The process of computation offloading and resource allocation.

5.2. Use Cases of Computational Task Offloading in MEC Networks

The demand for computational task offloading in MEC networks has increased in recent years thanks to the advancements in communication and technology. Following are some prominent use cases that have a huge demand for offloading computationally intensive tasks in MEC networks.

1. Unmanned Aerial Vehicle (UAV)
   The flexibility of UAVs makes them ideal for auxiliary deployment in terrestrial networks to manage large-scale global internet traffic. UAV-supported MEC networks can serve as aerial platforms providing offloading computing services to energy-limited UE and enhancing the network's QoS. When UE cannot offload computational tasks directly to the ground BSs due to WL quality issues, UAVs can relay these computational tasks and offer processing and relaying services [45]. Furthermore, due to the high cost of construction of MEC platforms, some remote industrial production sites might lack adequate infrastructure, potentially leading to weakened or interrupted transmission signals due to obstructing buildings. Such conditions may increase the failure rate of wireless task offloading for industrial vehicles (IVs). UAVs can be deployed on a large scale to provide stable computing services for IVs because of their easy deployment and cost effectiveness, thus making UAVs an ideal solution to this problem [46]. On the other hand, if terrestrial signals are blocked, which prevents the tasks of UE from reaching ground-based MEC servers, UE can offload computational tasks to MEC nodes outside their range using UAVs with processing capabilities. This fully leverages the collaboration between UAVs and edge computing systems [47]. Notably, the implementation of digital twin (DT) technology contributes to optimizing resource scheduling within UAV-assisted MEC networks, enabling comprehensive monitoring of the state of the network. This approach addresses the stringent requirements for accurate network state perception and real-time scheduling decisions due to network dynamism [48].
2. Autonomous Vehicle (AV)
   Rapid advancements in AV technology have led to an increase in computational

needs. Their limited computational power often falls short of handling these intensive computational tasks. One of the vital challenges in vehicular networks is ensuring reliable service for applications that require low latency. For most vehicular applications, especially those related to traffic control and safety enhancement, real-time responsiveness is critical in the fast-changing environment of vehicular networks [49]. To this end, AVs can access resources from MEC servers via vehicle-to-infrastructure (V2I) links and offload tasks for fast response [50]. Additionally, a k-hop-limited data offloading strategy is designed for vehicles beyond the signal range of an RSU. This strategy leverages multi-hop vehicle-to-vehicle (V2V) paths, aiming to enhance data offloading efficiency through optimized V2V2I paths [51].

3. Industry Internet of Things (IIoT)
   The accelerated growth of industry 4.0 is driving the emergence of IIoT services, e.g., disaster relief robots and cloud robots within smart manufacturing settings [52]. The growing sophistication of IIoT systems has led to the rise of delay sensitive and computationally intensive (DSCI) services that demand substantial computational resources and pose challenges to IIoT MDs in meeting stringent QoS requirements. However, MEC offers distributed edge computational resources for IIoT services, facilitating real-time intelligent execution of industrial IoT services at MEC nodes [41]. Despite its efficiency and low latency, the limited number of MEC servers can lead to offloading rejections in high-traffic periods. Hence, it is necessary to develop efficient offloading mechanisms for time-sensitive computational tasks and simulate high-traffic MEC systems, thereby reducing latency in IIoT-MEC networks [53].
4. Augmented Reality/Mobile Augmented Reality (AR/MAR)
   AR technology can combine real and virtual environments, and imposes significant challenges such as low-latency requirements and high energy consumption on current communication systems and the UE's power consumption. To address this, MEC's integration with AR offers a solution [54]. Concurrently, the popularity of MAR applications on UE is growing. These applications provide UE with mobility and cost effectiveness, although their development is limited by the UE's constrained battery life and processing power. MEC emerges as a promising strategy for managing MAR applications. It enables the offloading of computation-intensive tasks to the MEC end, which can provide low-latency computation and improve the energy efficiency of UE [55].
5. Smart Cities
   A smart city leverages IoT systems to enhance urban life services and infrastructure, improving citizen experiences in aspects of accommodation, shared infrastructure, etc. However, with the rapid growth of IoT systems and varied QoS demands, servers grapple with efficient resource allocation in vast networks. The cloud in smart city IoT systems can lead to high energy usage and latency. However, MEC networks can effectively manage energy use and maintain latency and QoS requirements by bringing resources closer to device ends [56].
   Take a typical life scenario, MEC networks can fulfill the complex and diverse UE requirements that often need a structured compound task comprised of multiple sub-tasks. These sub-tasks may be independent and run in parallel, or dependent and run sequentially. Therefore, an offloading strategy based on edge computing that handles structured tasks is essential to guarantee task execution with high efficiency and low latency. For instance, in the context of business travel, a compound task might comprise three structured sub-tasks: weather forecasting (t1), flight booking (t2), and hotel reservation (t3). While t2 and t3 can be executed in parallel, given their independence, they still rely on the results provided by t1 [57].
   In the education sector, vocal music education systems are always centered on cloud platforms. With the development of edge computing, computational and storage resources can cater to the operational needs of related vocal systems, satisfying real-time, high-reliability, and low-cost requirements in the vocal educational field [58].

6. Health care
   IoT technology has promoted the continuous development of electronic medical care. Wireless body area networks (WBANs) allow individuals to accomplish health monitoring, receive targeted medical services, and seamlessly exchange medical data through smart wearable devices. MEC has become a key technology for the improvement of electronic medical service platforms. By offloading smart medical services to MEC servers, it can greatly meet the requirements of smart wearable devices for portability and low latency [59]. Moreover, patients can use smart devices such as body sensors and wearable devices to monitor their health conditions without physical contact with doctors. However, these smart devices often have limited battery power. By offloading computational tasks to the MEC system, we can alleviate the drain on the batteries of these devices [60].
7. Virtual Reality/Internet of Video Things (VR/IoVT)
   IoT systems with vision sensors (e.g., cameras in smartphones, vehicles, and buildings) have arisen as a new sub-field of the IoT called the internet of video things (IoVT). The role of IoVT is to process the sensed and related visual data; therefore, IoVT devices have enhanced local computing power for vision processing. Nevertheless, vision processing task offloading is still required in MEC networks because IoVT devices cannot fulfill the immense computational requirements of vision processing [61]. Another IoT visual aspect involves VR. UE can enjoy ultra-high-resolution immersive VR video through a head-mounted display (HMD). However, this requires ultra-low-latency viewport rendering and fast data transmission, necessitating substantial bandwidth and exceptional processing capabilities. Furthermore, the high energy consumption of the HMD could hinder the rapid development of wireless panoramic VR video. Ref. [62] discusses minimizing the energy consumption of MEC networks by optimizing the viewport rendering offload strategy to the MEC server and supporting the delivery of high-quality immersive VR video services.

5.3. Categories of Computational Task Offloading

Computational task offloading can be implemented through two strategies: partial offloading and full offloading. However, if the paper does not explicitly define their offloading strategy, we assume full offloading is the default for this study. Furthermore, some papers such as [63] discuss a mixed offloading approach, where both full offloading and partial offloading scenarios are considered together.

1. Partial offloading
   Essentially, partial offloading posits that a task can be divided into multiple sub-tasks, which are independent entities that can be offloaded to the MEC or cloud end, or executed locally on the UE side. Determining the proportion of sub-tasks to offload or retain locally is crucial. The subdivision of a complex computational task is significant for optimizing QoS. Furthermore, the dependency relationships among the sub-tasks during execution should also be taken into consideration.
2. Full offloading (binary offloading)
   Full offloading and binary offloading both follow the same principle in that computational tasks cannot be divided into sub-tasks. That is, each task must be offloaded to the MEC end in its entirety.

5.4. Scenarios of Computational Task Execution

The circumstances surrounding the execution of the computational tasks are different. Basically, the common scenarios of task execution are combined with the UE side and MEC end. when the resources of the MEC nodes are insufficient to satisfy the requirements of task execution or other situations, the cloud end can help resolve computational task offloading issues [64].

Furthermore, in the device to device (D2D) underlying communication model integrated into MEC networks, computational tasks can be executed locally, offloaded to

the MEC end, and offloaded to a nearby UE [65]. Especially, the authors of [66] consider scenarios where tasks can be discarded. Figure 6 presents a summary of computational task offloading categories and scenarios.

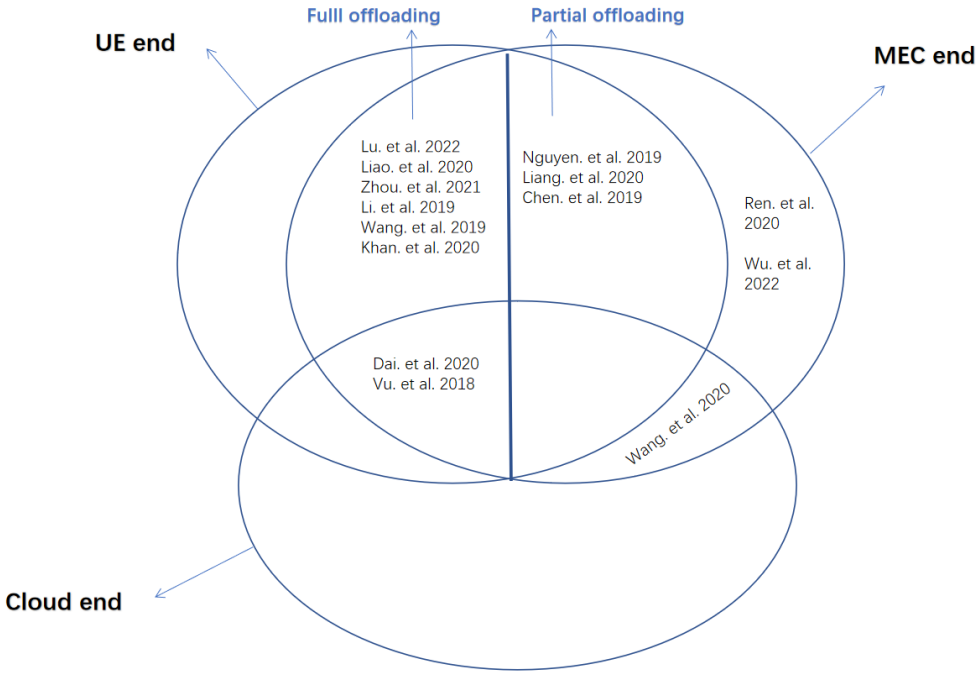

Figure 6. Scenarios and categories of task offloading [20,27,29,30,33–37,40,41,44,65,66].

5.5. Task Queue

Task queues store computational tasks awaiting offloading to servers, execution on UE or servers, return to the UE side, or download within UE. It is important to maintain network performance under stable task queue length constraints [31]. Such stable queues enable efficient task scheduling and processing, thus enhancing network performance.

Computational task queues in the MEC network are typically divided into local and remote queues. Local queues come into play when a large number of computational tasks are generated on UE. These tasks must wait in local queues for execution on the UE side. Alternatively, they are stored in a local queue, waiting for upload to ANs or servers via WL. Another local queue is the downloading queue inside the UE, which receives numerous processed results from servers before downloading them. The remote queue includes three types: first, a queue at the AN that receives offloading tasks from the UE and holds them for further upload to servers; second, a queue maintained by the server for computational task execution; finally, a return queue that delivers results either directly from MEC to the UE or via AN.

Additionally, there are some extended types of queues, such as virtual queues, task-priority-related queues, and energy queues. In [41], the authors propose a detailed queuing model, including an actual queue and a delay-aware virtual queue. As computational tasks arrive, the actual queue is updated. The virtual queue is designed based on the actual queue, with awareness of application latency to help sense the backlog of the actual task queue. Moreover, ref. [64] implements a priority-based resource allocation strategy, in which resource request queues are divided into a normal queue and a priority queue. Importantly, energy queues are maintained at both the UE side and the MEC end to monitor the state of energy consumption [11].

6. Resource Allocation in MEC Networks

Once the offloading decisions for computational tasks have been made (i.e., task execution on the UE side or offloading to servers), the next step is to allocate the appropriate

resources for processing these tasks. We initially provide an explanation of resource allocation. Following that, a detailed introduction to the complete set of resources involved in computation offloading and resource allocation is discussed. Lastly, to further enrich the content on resource allocation, one technique and one piece of information related to resources are described.

6.1. Description of Resource Allocation

When a computational task needs execution on the UE side or MEC end, allocating sufficient resources is necessary. For the local computation on the UE side, enough resources must be provided, considering computationally intensive tasks that need to be offloaded to the MEC end. The process usually involves the UE sending resource requests to the MEC end, which then receives the requests and allocates resources accordingly [64]. All in all, once the computational task has been approved for offloading to the MEC end, the next step is allocating resources for execution. These resources can be allocated in various ways by launching VMs, docker containers, or any other technologies [13].

Unfortunately, the resource-limited MEC end cannot provide unlimited computing services and processing for all UE tasks. It is important to determine how to effectively manage and allocate resources to the MEC end to achieve the optimization aim. The detailed sequence of task offloading and resource allocation is shown in Figure 5.

6.2. Classifications of Resources

Various physical resources can be virtualized. Resource virtualization is a key technology for improving performance in MEC networks. Firstly, by VMs or container deployment of virtualized resources, the MEC end can quickly respond to and process requests for computational task offloading, without the need to modify traditional hardware equipment. Secondly, this technology can dynamically allocate and release computational resources according to computational task requirements, thus raising resource utilization. In addition, virtualization technology can run multiple virtual instances in a single MEC node, which can make more efficient use of network hardware resources. This enhanced resource utilization can reduce equipment costs and energy consumption, thereby improving the overall network efficiency. However, this technology requires powerful physical devices to operate efficiently since virtual resources need to pay some initial costs and incur start-up latency, thus some small enterprises prefer not to resource virtualization to save costs [67]. As a result, whether there are visualizing resources or not depends on the actual situation. In this context, we further introduce two types of physical resources involved in the process of computational task offloading and resource allocation: device resources and network communication resources.

6.2.1. Device Resources

In this paper, the device resources mainly involve three categories, that are computational resources (i.e., CPU, GPU, and FPGA), memory and storage, and caching.

1. CPU
   During computation offloading, the key computational resource is the central processing unit (CPU) that is in charge of executing computational tasks. The performance of the CPU is critical to meet the task requirements, e.g., low latency and high throughput. It is necessary to introduce CPU frequency (also known as clock speed [35] or clock frequency). A higher CPU frequency indicates a faster speed of processing. In addition, it should be noted that the CPU processing capacity is also related to some chip parameters [44] and parameters of operating modes [34].
2. GPU
   The graphics processing unit (GPU) is also an important computational resource that is particularly well suited for processing graphics, images, and data-intensive tasks. It plays an important role in AR/MAR applications. Typically, such detection algorithms require sufficient GPU processing capacities to enhance the performance

of MAR services. However, most MAR devices have difficulties supporting these algorithms in real time. The MEC servers can provide GPU resources for AR/MAR services to implement relevant algorithms under low latency.

3. FPGA
   A field-programmable gate array (FPGA) can be used as a flexible and efficient computational resource. It can complement the CPU and GPU to increase computing performance for specific tasks. FPGA computing has the advantages of both a low cost and low power consumption, so it is a promising candidate for MEC servers. In addition, FPGA can handle timing-critical jobs by accepting requests from edge devices directly through its I/O and executing tasks with sustained performance through hard-wired logic. Therefore, multi-FPGA systems are particularly advantageous as they can handle multiple requests from multiple edge devices [68].
4. Memory and Storage
   Processing computational tasks involves not only computational resources but also other resources, involving storage for computing task data and data preprocessing procedures, e.g., data compression. Additionally, task queues can be stored in memory and storage space, waiting for execution. Storage can permanently store information such as applications, data, and intermediate results. Memory is used for temporary storage of information. Data in storage and memory can be obtained by UE and MEC nodes quickly, thereby enhancing the performance and efficiency of MEC networks. For instance, a realistic network executes tasks that require computational, memory, and storage resources. Here, RAM serves as one type of memory resource, and the hard disk is regarded as the storage resource. Furthermore, the required amounts of RAM and hard disk space for tasks in MEC networks are considered [69].
5. Caching
   Caching is a technique that can improve network retrieval performance by temporarily storing frequently accessed data in a fast storage system (e.g., RAM), thereby reducing access to slower storage devices (e.g., hard disks). When a program is cached at BSs, it is common practice to offload related computational tasks to the BSs for processing by the program. Therefore, caching a particular program can save costs, but only if the program is frequently reused to perform future offloaded tasks [70]. Additionally, caching relevant content in BSs associated with the UE can be beneficial. If the content has already been cached in the BS, it can be delivered directly to the UE side. The caching policy is updated after each content delivery, ensuring that the most popular content is always cached [32]. Mobile UE can also cache within the UE to reduce the number of communications with the MEC end, thereby reducing energy consumption and service delays [23]. Moreover, a reliable and secure MEC system can use active content caching to solve the problem related to the high frame loss rate during multi-hop transmission in wireless networks [21].

6.2.2. Networks Communication Resources

Optimizing wireless resource allocation is crucial, which includes the reasonable scheduling of limited resources such as channel, bandwidth, and the selection of AN. These measures enable the successful and efficient transmission of numerous offloaded computational tasks.

1. Channel allocation and bandwidth allocation
   Channel allocation refers to the selection of the spectrum range for computational task offloading, and bandwidth allocation is the distribution of bandwidth within that spectrum range, aiming to meet the transmission needs of different data. Rationally allocating channels can help ensure the high quality of wireless communication, and satisfy QoS requirements of computational tasks. The MEC system needs to allocate wireless channels rationally for offloading computational tasks [27]. Preventing interference among UE through channel pre-allocation and improving channel utilization efficiency are significantly important aspects [30]. For example, in an MBS

where many SBSs are deployed, and by reusing the spectrum between the small cells and the macro cell, the channel utilization efficiency is significantly enhanced [31]. However, channel interference caused by channel reuse can affect task transmission and communication efficiency [20]. Therefore, a balance must be struck between different QoS requirements and the channel allocation issue.
On the other hand, reasonable bandwidth allocation affects the MEC network's performance. When a UE offloads computational tasks to the MEC server through AP, the partition of the maximum available bandwidth of the AP should be orthogonally allocated to the UE, thus reducing channel interference and improving channel resource utilization [39]. Notably, dynamic adjustment of bandwidth allocation can better enhance network performance [30].

2. AN selection, and transmission power
With the extensive deployment of wireless local area networks (WLAN), each UE can offload computational tasks to servers via multiple ANs. However, if all items of UE select the same AP to offload their tasks to, it may incur higher system costs [39]. Therefore, choosing an appropriate AP to offload computational tasks to is very important. Secondly, offloading tasks in wireless networks involves using transmission power to transmit data [36]. Selecting a reasonable transmission power for offloading computational tasks can reduce the MEC network costs [39].
3. Data rate in wireless link
The data rate can directly affect the QoS and overall network performance. An increase in data transmission speed can lead to faster transmissions, thereby improving network efficiency. Furthermore, the data rate can be influenced by the transmission power, interference, and bandwidth between devices and BSs in a dynamic wireless environment [33]. For the different WATs, the data rate for task offloading is determined by the channel conditions and the selected network interface (e.g., 5G, WiFi) [34].

6.2.3. Time

In realistic scenarios, time can be viewed as a resource required by the UE. However, many papers have overlooked the time required for resource allocation [69]. Considering the time (i.e., task computing time, task transmission time) as a type of resource, optimizing these resources indicated can speed up computational task execution, network services, and connections, thereby reducing latency and energy consumption. For instance, it is necessary to optimize the time slots assigned to UE for offloading tasks in order to minimize the transmission energy consumption [41].

6.3. Technology and Information for Wireless Resource

In relation to wireless resource allocation, there are two main aspects of wireless resource management.

6.3.1. Wireless Access Technology (WAT)

There are many different types of WATs that can be picked to offload computational tasks. The commonly used WATs in MEC networks are NOMA, TDMA, FDMA, OFDMA, and low-power wide-area network (LPWAN). Every WAT comes with its own features, and we should select an appropriate WAT depending on the specific applications and situations in order to fully leverage its performance advantages.

For example, using TDMA to transmit computational tasks between the UE and MEC end, an interval would be separated into different time slots, and each of them can be allocated so as for the UE to offload the task to the MEC end. In other words, UE uses and shares the same channel because of limited channel resources but sends and receives information at different time intervals [35]. Moreover, TDMA allows UE to access a sub-channel at different times as well as permits a UE device to access different sub-

channels [41]. As a result, TDMA technology can improve spectrum utilization and reduce the possibility of channel conflicts.

Furthermore, both NOMA and FDMA are used for computational offloading. NOMA allows multiple items of UE to share the same time–frequency domain resources, which improves spectrum utilization. This can enhance the computing performance and UE connectivity of UDNs. FDMA channels are orthogonal, and the interference between different channels is minimal, contributing to system stability. FDMA assigns fixed frequency bandwidth, hence it is particularly suited to support constant rate services. The NOMA system dynamically adjusts resource allocation based on channel conditions [28].

Utilizing OFDMA technologies, each item of UE occupies a channel to offload computational tasks. The channels are allocated in such a way that the UE in different channels do not interfere with each other. This is different from OFDM where the same channel can be reused by multiple UE devices, causing interference between them. As a result, OFDMA can achieve higher utilization of the channel [20].

Another attractive WAT is LP-WAN; solutions based on this communication mode are expected to achieve a long coverage of about 10 km and improved power efficiency (lower power consumption), enabling the lifetime of the UE to reach about 10 years [71].

The heterogeneity of the WATs will be a prominent feature of the next-generation communication scenarios. Managing these diverse WATs is a complex task, especially in ultra-dense network scenarios. Therefore, technologies such as vertical sectorization and network slicing have emerged to optimize the allocation of resources, such as the radio spectrum [71].

6.3.2. Channel State Information (CSI)

In a wireless communication system, CSI plays an important role in accomplishing efficient and reliable communication. The communication system can learn the information of the wireless channel from the CSI to make corresponding adjustments to adapt to the dynamic channel environment. For example, the MBS can obtain useful information, including CSI, then make offloading admission control and computational resource allocation decisions [31]. Moreover, the consideration of imperfect CSI with random error is an extra factor [44].

7. Minimization of Energy Consumption for Optimization of Task Offloading and Resource Allocation in MEC Networks

Computing offloading and resource allocation strategies profoundly affect a network's energy consumption. This section thoroughly analyzes the issue of minimizing energy consumption in MEC networks, with a primary focus on key areas such as energy consumption scenarios, categories, implemented algorithms, and energy consumption minimization approaches. Specifically, we outline the objective functions that aim to minimize energy consumption in MEC networks. This is achieved by jointly optimizing computation offloading and resource allocation. The objectives can be categorized into several main areas: minimizing energy consumption on the local UE side, the MEC end, the cloud end, during uplink transmission, and the BS side. They also include maximizing network utility, which takes into account energy consumption. Increasing the number of computational tasks offloaded to the MEC end, which involves reducing the energy consumption of the UE. Table 3 provides details on offloading type, sites of task execution, optimization objectives, algorithms, and experiment results.

Table 3. Cross-comparison of existing works on optimization of computation offloading and resource allocation.

Ref.	Offloading Type	Execution Site	To Optimize Computation Offloading	To Optimize Resource Allocation	Objective	Implemented Algorithms	Objective's Increase/Reduction Percentage vs. Other Algorithms
[35]	Partial	UE and MEC end	Offloading proportion and locations	(a) CPU utilization of UE and MEC server; (b) transmission time slots; (c) total transmission time	Minimize energy consumption of UE	(a) Duality-based optimization algorithm; (b) a greedy algorithm	(Duality-based algorithm) Reduction of up to around 41%
[20]	Full	UE and MEC end	Decisions of offloading locations	(a) CPU utilization of MEC server; (b) BS selection; (c) channel allocation	Minimize system cost	(a) NICRA; (b) GABSRS	(GABSRS) Varying UE amount, task computing amount, task size: reduction of up to 22.82%, 32.20%, 38.11%
[36]	Partial	UE and MEC end	Offloading proportion and locations	(a) Transmission power; (b) computation resource utilization of UE	Minimize system total cost	(a) DQN; (b) DDPG	For multiple UE devices, DDPG: reduction of up to about 70% (DQN: slightly lower than 70%) when w = 0.5
[33]	Full	UE, MEC end, and cloud end	Decisions of offloading locations	(a) CPU utilization of MEC and cloud server; (b) transmission rate	Minimize total energy consumption	(a) MDP; (b) DDPG (dataset: a real-world dataset)	(DDPG) Reduction of up to about 92% with varying UE amount
[26]	Partial	UE and MEC end	Offloading proportion, locations, and the number of offloaded UE devices	Transmission power	Minimize the energy consumption of UE devices	SVM-based FL (dataset: a real city traffic data from OMNILab at Northeastern University)	Reduction of up to 20.1%
[28]	Full	MEC and cloud ends	Decisions of offloading locations	(a) CPU utilization of MEC server; (b) transmission power; (c) the state of MEC's PU (active and idle)	Minimize system total cost	(a) LSTM (dataset: a real traffic dataset [72]); (b) Lyapunov optimization and SCALE	Algorithm (b) with NOMA offloading: reduction of up to 51% (with FDMA: slightly lower than 51%)
[39]	Full	UE and MEC end	Decisions of offloading locations	(a) Bandwidth allocation; (b) computing resource utilization of MEC server; (c) transmission power	Minimize the system cost	CSAO scheme: (a) potential game theory; (b) Lagrange multiplier; (c) bisection algorithm	(CSAO) Different UE amount, task required computing resource, task size: reduction of up to around 42%, 15%, 7.5%
[44]	Full	UE and MEC end	Decisions of offloading locations	(a) Transmission power; (b) bandwidth allocation; (c) CPU utilization of MEC server	Minimize the energy consumption of UE	(a) CSSCA; (b) dual decomposition theory; (c) RODRA; (d) RBORA	(RBORA) Not the lowest, but with lower complexity, and considers channel estimation error
[41]	Full	MEC end	Queuing-based task offloading scheme in MEC end	Transmission time (offloading duration)	Minimize the energy consumption of offloading tasks to MEC end	(a) Perturbed Lyapunov optimization; (b) DAEE online offloading algorithm	(The entire algorithm) for different task arrival rates: reduction of about 87% and 75%
[34]	Full	UE and MEC end	Decision of task processing and offloading locations	(a) AN selection; (b) CPU utilization of UE; (c) transmission power	Minimize the total energy consumption and system total cost	(a) Online multi-agent RL together with a game-theory-based algorithm; (b) JTSRA	(The entire proposed algorithm) increases energy efficiency by up to roughly 65%
[27]	Full	UE, MEC end, and cloud end	Decisions of offloading locations	(a) CPU utilization of MEC server; (b) channel allocation	Minimize the energy consumption of UE	(a) ROP based on the interior point method; (b) IBBA	(a) (IBBA, ROP) Varying task complexity: reduce by up to around 63%; (b) (ROP) various delay constraints: reduction of up to around 7.3%

Table 3. *Cont.*

Ref.	Offloading Type	Execution Site	To Optimize Computation Offloading	To Optimize Resource Allocation	Objective	Implemented Algorithms	Objective's Increase/Reduction Percentage vs. Other Algorithms
[29]	Partial	UE and MEC end	Offloading proportion and locations	(a) Transmission power; (b) CPU utilization of UE and MEC server; (c) bandwidth allocation	Minimize the delay and energy consumption	(a) CCCP algorithm; (b) IBCD algorithm	(CCCP and IBCD) Different weight value between delay and energy consumption: reduction of up to around 80% and 76%
[40]	Full	MEC end	Queuing-based offloading model in MBS (MEC) and mBS	(a) The rate and volumes of task offloading and forwarding; (b) upload and download channel allocation	Maximize network utility	JCRM algorithm	(JCRM) With increasing weight parameter and varying channel numbers: increase of up to roughly 31% and 80%
[30]	Full	UE and MEC end	Decisions of offloading locations	(a) Transmission power; (b) bandwidth allocation	Maximize the total number of offloaded tasks	A low-complexity heuristic algorithm	Varying CPU cycles of MEC end and UE amounts: increase of up to about 70% and doubles

AP: access point; AF: amplify-and-forward; BS: base station; CPU: central processing unit; CCCP: concave–convex procedure; CSSCA: constrained stochastic successive convex approximation-based; CSAO: computation offloading strategy and resource allocation; DDAE: delay-aware energy-efficient; DF: decode-and-forward; DQN: deep Q-network; DDPG: deep deterministic policy gradient; EOA: equal opportunity allocation; FL: federated learning; GABSRS: genetic-algorithm-based BS selection and resource schedule; IBBA: improved branch and bound algorithm; IBCD: inexact block coordinate descent; JCRM: joint channel allocation and resource management; LSTM: long short-term memory; MBS: macro base station; MDP: Markov decision process; MEC: mobile edge computing; NICRA: Newton-IPM-based computing resource allocation; PU: processing unit; ROP: relaxing optimization policy; RODRA: relaxed offloading decision and resource allocation; RBORA: ranking-based binary offloading and resource allocation; SCALE: successive convex approximation for the low-complexity; SVM: support vector machine; UE: user equipment; VS: versus.

7.1. Scenarios of Energy Consumption

7.1.1. Energy Consumption on UE Side

The energy consumption on the UE side is mainly decided by processing computational tasks. It primarily relates to the number of computational tasks or sub-tasks computed by the CPU, local CPU frequency, and operating parameters and modes of CPU, which depends on the chip architecture [20,26,29,33,35,36,39,44]. Therefore, when computational tasks are executed on the UE side, the energy consumption mainly relies on CPU operation [11]. In addition, the consumed energy on the UE side also includes the received data energy consumption from the MEC and cloud end [27]. In certain scenarios, the UE's power consumption involves not only the energy used for scanning available ANs but also the maintenance power required to keep the UE in an idle state [39]. Particularly, all computational tasks need to be offloaded to the MEC end for execution, rather than being computed locally. Therefore, in this scenario, there is no energy usage on the UE side [28,40,41].

7.1.2. Energy Consumption of MEC End

When computational tasks are executed in MEC servers, the energy consumption is primarily from computational task execution. In the same way, the consumed energy is determined by the number of computational tasks, the CPU frequency, and some parameters of the chip architecture [20,29,40]. In addition, in IoT applications, IoT sensor devices continuously collect tasks and upload all of them to relevant MEC servers for processing. It is assumed that the MEC server can possess numerous PUs, which can be turned on or off individually. Therefore, the consumed energy includes the active PUs' energy expenditure for processing computational tasks, along with the idle power of PUs at the MEC end [28]. To simplify the calculation of task execution on the MEC end, the computation energy consumption evaluated the power consumption per CPU cycle and the required computational resources that are measured in CPU cycles [33]. In particular, if we assume there are sufficient computational resources to process computational tasks on the MEC end, ignoring the CPU execution energy consumption on the MEC end [41].

7.1.3. Energy Consumption on Cloud End

Concerning the cloud end participating in the computational task offloading, the energy consumption of the cloud end is similar to the MEC end. It is tied to the consumed power per CPU cycle and the required computation resources in CPU cycles [33]. However, some scenarios might overlook the power consumed by the cloud end [28].

7.1.4. Energy Consumption of BSs

In regards to the AN's energy consumption, the BS is considered the most significant AN; we focus on the power consumed by BSs here. BSs can be powered in various ways involving renewable energy generation systems, energy batteries, and power grids. The energy generated by the photovoltaic panels is used to power the BS and its cache, and when the output exceeds the energy consumed by the BS, the unused energy is saved in the battery. If there is no renewable energy or battery energy available for network supply, the required energy is drawn from the power grid [32]. Regarding the type of energy consumption, there are two parts: constant and dynamic electrical usage. However, dynamic energy usage can be ignored when the results are returned via downlink transmission [20]. On the other hand, many papers deem it unnecessary to compute the power consumption of BSs, such as when BSs integrated with the MEC end can have a continuous power supply, and the BSs' energy consumption can be negligible [26]. Alternatively, when a BS is in sleep mode, its power consumption can be ignored [32].

7.1.5. Energy Consumption of Task Transmission

Generally, the transmission energy consumption between the UE and the servers refers to the power consumption in uplink offloading computational tasks. This factor plays a

crucial role during the computation offloading and resource allocation, as such, the transmission energy consumption should not be overlooked [73]. The size of the downloaded data returned to the UE is usually much smaller than the uploaded computational task, so the downlink transmission time is not considered [41]. Hence, in most cases, the energy consumption for returning data can be ignored [20]. An exception is the transmission energy in both the uplink and downlink because there are many mBSs that receive offloaded computational tasks from UE and an MEC server deployed in an MBS can receive the forward tasks from the mBSs. For this reason, we can divide the energy consumed during transmission into transmitting power consumption (from MBS to mBS), by contrast, the energy consumed during reception is called receiving power consumption (from mBS to MBS) [40].

The power consumption of transmission consists of offloading time and transmission power, and relevant factors also include the volume of data transmitted and the transmission rate [20,33,35,36,39,41,44]. Notably, some factors are interrelated, for example, the required transmission time equals the size of the offloaded tasks divided by the offloading rate, and bandwidth is related to the transmission rate [39]. In order to simplify the calculation of wireless transmission, the energy consumed during transmission is often considered as a composite of the energy consumed to offload a single data unit and the total size of the data transmitted [27].

In particular, wireless transmission operates in 5G and WiFi modes. In the 5G network, the transmission power consumption is equal to the energy consumed in the data transmission, in WiFi mode, the consumed energy includes both scanning the available spectrum and data transmission power [34]. In addition, FOMA and NOMA are used to offload computational tasks from MEC to the cloud end. The relevant elements of the transmission energy include not only the transmission power and time but also the amplifier coefficient and constant circuit power. The consumed transmission energy is different in these two transmission models. In a heterogeneous MEC network, the hybrid relaying (HR) architecture is utilized in the MEC network, which combines the advantages of amplify-and-forward (AF) and decode-and-forward (DF) relaying schemes. In a two-UE game scenario, computational tasks can be divided to improve processing efficiency. One UE can locally process certain computational tasks and share the results with the other UE via an amplify-and-forward (AF) relay through the MEC server. The remaining tasks are offloaded to the MEC server, and their results are shared with the other UE using a decode-and-forward (DF) relay scheme. Therefore, these AF and DF relay schemes influence the transmission energy consumption [29].

7.2. Categories of Energy Consumption Minimization in MEC Networks

7.2.1. Minimize Energy Consumption on UE Side

Some papers focus on minimizing energy consumption on the UE side. In these studies, the energy expenditure associated with task offloading is often classified as part of the total electrical usage of the UE side. For example, in [35], the authors propose an optimization strategy for partial task offloading (i.e., the proportion of a task to be offloaded to the MEC end). They also optimize the CPU utilization on both the UE and MEC ends for task execution, as well as the offloading transmission time. Similarly, the minimization of energy consumption on the UE side in [26] involves the computational task local execution, task transmission, and a fixed value for UE operation, and relevant strategies are proposed to optimize partial computational task offloading and transmission power. Likewise, ref. [44] aims to minimize the consumed energy of the UE by optimizing offloading locations (i.e., UE side and MEC end), transmission power, uplink bandwidth allocation, which is related to the transmission rate, and CPU frequency allocation at the MEC end. Although ref. [27] also focuses on minimizing energy consumption on the UE side, it considers not only the energy associated with task execution locations, local task computation, and uplink data transmission but also the power consumed when receiving results from the MEC end. Therefore, to effectively reduce energy consumption, it is

necessary to optimize the CPU utilization for computing tasks at the MEC end, and the spectrum channel allocation for computational task offloading.

7.2.2. Minimize Energy Consumption of Transmission

The authors in [41] assume that the CPU computing energy consumption at the MEC end is negligible due to abundant computational resources, and all computational tasks need to be offloaded to the MEC end for processing. As a result, the total power consumption primarily stems from the transmission process. Within this queue-based task offloading framework, they focus mainly on optimizing the offloading transmission time.

7.2.3. Maximize Network Utility

The maximization of the MEC network utility presented in [40] comprises three components: first, maximizing the data utility function related to forward offloading tasks; second, minimizing computation energy consumption at the MEC end; third, minimizing the total consumed energy during transmission. The associated resource factors that need optimization include the offloading rate, the volume of offloaded and forwarded computational tasks from the UE to the MBS via the mBS, and the uplink and downlink bandwidth allocation between the UE side and the mBS.

7.2.4. Maximize the Number of Offloading Tasks

In fact, maximizing the number of computational tasks offloaded to the MEC end does not directly enhance the energy efficiency on the UE side. However, it can save the energy consumption used for local computational task execution. Optimized elements include transmission power and bandwidth allocation for offloading tasks.

7.2.5. Minimize System Cost of MEC Networks

Minimizing the total network system cost is a comprehensive concept. Ref. [39] defines the system cost as encompassing the energy consumption in two sections. The first one is data transmission, and the second one involves the UE side, where the consumed energy stems from local task computation, scanning the available spectrum, and maintaining energy consumption when the UE is idle. The minimization aim can be achieved by providing optimal bandwidth allocation and transmission power to offload computational tasks, as well as the computing frequency allocated by the MEC server for local task execution.

Accounting for the total energy consumption and other network metrics, such as delay, it is important to establish a balance among them. The authors in ref. [20] aim to minimize the system cost that includes the total latency (i.e., computation and transmission latency) and consumed energy (across UE, MEC, task transmission, and BSs). They optimize the CPU utilization of the MEC end for computational task execution and the selection of BSs and channels for the task offloading strategy. Differently, ref. [36] includes a penalty for failed task execution in their system cost, in addition to transmission and computation power consumption. The optimized items encompass the transmission power and local computation resources utilization. Furthermore, ref. [34] further considers UE local computation, task transmission, the commercial cost of MEC execution, and cost of uplink offloading. Moreover, the consumed energy for scanning networks in the WiFi offloading model is part of the total transmission power consumption. To minimize the system cost, optimization is needed in AN selection, transmission power, and the UE's CPU utilization. The binary parameters, such as task processing and offloading decisions, also need to be optimized within the offloading strategy. Lastly, the study [28] aims to maximize computational throughput and minimize energy consumption at the MEC end to optimize the overall system cost. Their focus is on MEC's CPU utilization, transmission power, and the state of PUs (working or idle).

In particular, minimizing the total delay and energy consumption of task execution at the UE and MEC ends, along with transmission power consumption, can be seen as minimizing the system cost. Optimization targets include transmission power and

bandwidth allocation, applicable to both AF and DF relaying schemes, as well as CPU utilization [29]. Meanwhile, in ref. [33], the authors minimize the total energy consumption, encompassing task execution and transmission power, by optimizing CPU utilization for task execution in the MEC and cloud server, along with the transmission rate.

7.3. Classifications of Algorithms and Experiment Results

The implemented approaches that jointly optimize computation offloading and resource allocation issues can be classified into several categories: convex transformation/relaxation, convex optimization algorithms, heuristic algorithms, ML-based algorithms, Lyapunov optimization, game theory, branch and bound, and others. More details about these categories of algorithms are listed in Table 4.

Table 4. Introduction of implemented algorithms

Ref.	Classification of Algorithm	Algorithm	Function
[35]	(a) Convex transformation/relaxation; (b) Convex optimization; (c) Heuristic algorithm	(a) Penalty method; (b) Duality-based optimization algorithm; (c) A greedy algorithm	(a) Transform the non-convex objective (minimization energy consumption of UE) to a convex problem; (b) Optimize the convex problem to obtain the optimal solution; (c) Comparison with duality-based optimization algorithm
[20]	(a) Convex optimization; (b) Heuristic algorithm	(a) Newton-IPM-based computing resource allocation (NICRA); (b) Genetic-algorithm-based BS selection and resource scheduling (GABSRS)	(a) Computing resource allocation algorithm; (b) BS selection (offloading issue) and channel allocation
[36]	ML-based algorithm	(a) DQN; (b) DDPG	(a) Task offloading and resource allocation algorithm; (b) Extend DQN to continuous action spaces (improve learning stability)
[33]	ML-based algorithm	(a) MDP; (b) DDPG	(a) Transform the objective to MDP form (minimization of total energy consumption); (b) Address the problem of MDP
[26]	ML-based algorithm	SVM-based FL	To ensure the association between UE and BSs, optimize computation offloading and resource allocation to achieve the goal of minimization energy consumption of UE
[28]	(a) ML-based algorithm; (b) Lyapunov optimization	(a) LSTM; (b) Lyapunov optimization and successive convex approximation for the low-complexity (SCALE) approach	(a) Predict the long-term workload in networks, optimize the number of active PUs; (b) Dynamic resource allocation and computation offloading algorithm with the low-complexity in small timescale
[39]	(a) Game theory; (b) Convex optimization; (c) Convex optimization	(a) Potential game theory; (b) Bisection algorithm; (c) Lagrange multiplier	(a) Computation offloading algorithm (APs selection); (b) Wireless radio resource allocation algorithm; (c) Computation resource allocation algorithm
[44]	(a) Convex transformation/relaxation; (b) Convex optimization; (c) Convex optimization; (d) Others	(a) Constrained stochastic successive convex approximation-based (CSSCA) algorithm; (b) A relaxed offloading decision and resource allocation (RODRA) algorithm based on IPM; (c) A Lagrange dual decomposition algorithm; (d) A ranking-based binary offloading and resource allocation (RBORA) algorithm	(a) Relax the original optimization problem; (b) Computation offloading and resource allocation algorithm; (c) Resolve optimization problem with low complexity (IPM with high complexity); (d) Consider offloading priority then resource allocation
[41]	(a) Lyapunov optimization; (b) Lyapunov optimization	(a) Perturbed Lyapunov optimization; (b) Delay-aware energy-efficient (DAEE) online offloading algorithm	(a) Construct a virtual queue, and transform the problem of ensuring task execution deadlines into a stable control of virtual queue; (b) Adaptive offloading strategy (offload more tasks in good quality of network while keeping a low queue backlog)
[34]	(a) ML-based algorithm; (b) Lyapunov optimization	(a) Online multi-agent RL together with a game theory-based algorithm; (b) Joint task scheduling and resource allocation (JTSRA)	(a) Offloading link selection and transmission power allocation; (b) Local task execution scheduling algorithm and local CPU resource allocation
[27]	(a) Convex optimization; (b) Branch and bound	(a) Relaxing optimization policy (ROP) based on IPM; (b) Improved branch and bound algorithm (IBBA)	(a) Offloading strategy and resource allocation algorithm; (b) A low-complexity computation offloading and resource allocation algorithm

Table 4. *Cont.*

Ref.	Classification of Algorithm	Algorithm	Function
[29]	(a) Convex optimization; (b) Convex optimization	(a) Concave–convex procedure (CCCP) optimization algorithm; (b) Inexact block coordinate descent (IBCD) algorithm	(a) Computation offloading and resource allocation algorithm; (b) A low-complexity computation offloading and resource allocation algorithm
[40]	Lyapunov optimization	Joint channel allocation and resource management (JCRM)	Task offloading and wireless radio resource allocation
[30]	Heuristic algorithm	A low-complexity heuristic algorithm	An uplink channel pre-allocation method based on hypergraph techniques (avoid interference among UE), and offloading strategy and resource allocation

AP: access point; BS: base station; CPU: central processing unit; DQN: deep Q-network; DDPG: deep deterministic policy gradient; FL: federated learning; IPM: interior point method; LSTM: long short-term memory; ML: machine learning; MDP: Markov decision process; PU: processing unit; RL: reinforcement learning; SVM: support vector machine; UE: user equipment.

7.3.1. Convex Transformation/Relaxation

Convex transformation, also known as convex relaxation, simplifies non-convex optimization problems by turning them into convex ones or reducing constraint strength. In ref. [35], the non-convex computation offloading and resource allocation problem involves complex coupling among variables. A penalty algorithm is applied to achieve convexity by investigating these variable relationships. Similarly, a constrained stochastic successive convex approximation-based algorithm (CSSCA) simplifies the same problem by converting it into an iterative one, enabling easier acquisition of a near-optimal solution [44].

7.3.2. Convex Optimization

Convex optimization is a subset of optimization problems where both the objective function and constraints are convex. This attribute simplifies the problem compared to other optimization types, making convex optimization algorithms widely used.

After the convex transformation, the problem becomes convex. For example, in ref. [35], there is a duality-based algorithm that optimizes the optimization convex problem, demonstrating a reduction in energy consumption by up to about 41% compared to other methods. Furthermore, ref. [27] proposes a relaxed offloading decision and resource allocation algorithm (RODRA) based on the interior point method (IPM) to optimize the computation offloading and resource allocation issues. However, due to the high complexity of convex problems, a Lagrange dual decomposition algorithm is introduced to reduce the calculation complexity.

In some cases, the optimization problem can be divided into several sub-tasks (i.e., sub-algorithms). For instance, in ref. [20], the Newton-IPM-based computing resource allocation algorithm (NICRA) optimizes the computing resource allocation. It alternately uses the bisection algorithm and the Lagrange multiplier algorithm to optimize wireless radio resource and computational resource allocations in MEC networks. In addition, compared to other baselines, the computation offloading strategy and resource allocation optimization (CSAO) in ref. [39], which consists of different sub-tasks, reduces the energy usage of the network by up to around 42% with varying numbers of UE devices, 15% for different required computational resources for task execution, and 7.5% in scenarios with various task data sizes. As for the convex problems, a relaxed offloading decision and resource allocation algorithm (RODRA) and Lagrange dual decomposition algorithm with low complexity are proposed to optimize computation offloading and resource allocation [44]. Similarly, the authors in [27] suggest a relaxing optimization policy (ROP) based on IPM to optimize the offloading strategy and resource allocation. This leads to a reduction in consumed energy by up to 63% for varying task complexity and 7.3% for varying task delay requirements.

Note that some algorithms integrate convex transformation/relaxation with convex optimization. For example, the authors in ref. [29] initially employ the concave–convex

procedure (CCCP) algorithm to approximate the original optimization problem as a convex one, which is then solved. However, due to the high computational complexity of the CCCP algorithm, they propose the inexact block coordinate descent (IBCD) algorithm to reduce the algorithm complexity. Simulation results demonstrate that the CCCP and IBCD algorithms can decrease the system cost by up to approximately 80% and 76%, respectively.

While convex optimization ensures a globally optimal solution, it may require significant computational resources and convergence time in large-scale models. Moreover, converting a non-convex problem into a convex problem may introduce some approximation errors.

7.3.3. Heuristic Algorithms

Heuristic algorithms with their high flexibility can be used to solve optimization problems. As an example, to solve the NP-hard mixed-integer nonlinear programming problem, the authors in ref. [30] propose a low-complexity heuristic algorithm to increase the number of offloaded tasks, thereby reducing the UE side's energy consumption. Therefore, the experimental results show that this algorithm can increase the number of offloaded tasks by up to around 70% for different CPU cycles at the MEC end, and double the number of offloaded tasks in the MEC network with varying numbers of UE devices. In detail, the authors employ hypergraph methods to pre-allocate uplink channels for the offloaded tasks, which helps to reduce channel interference among UE devices. Subsequently, they optimize offloading decisions and resource allocation. Moreover, in ref. [20], the genetic-algorithm-based BS selection and resource scheduling (GABSRS) is presented to address the problem of selection of an appropriate BS and channel allocation for computation offloading. GABSRS bridges the gap between the NICRA solution for computational resource allocation and the optimal result. As such, compared to other algorithms, GABSRS is employed to optimize this second sub-task (i.e., BS selection and channel allocation). It is shown that GABSRS decreases energy usage by up to 38.11% when considering diverse task data sizes, 32.2% with the varying task-required computational resource, and 22.82% for diverse amounts of UE devices. For comparison, a greedy algorithm is presented in ref. [35].

However, heuristic algorithms are not always guaranteed to find the global optimal solution. Hence, they are not well suited for addressing all problems.

7.3.4. ML-Based Algorithms

Concerning relevant ML algorithms, they are ideally suited for utilization in dynamic MEC networks, providing robust solutions to accommodate real-time-varying wireless environments, etc. For instance, ref. [33] employs the deep reinforcement learning (DRL) algorithm to address the issue of computation offloading and resource allocation by transforming the strategy to a Markov decision process (MDP) problem, because DRL is based on MDP, therefore, transforming the objective function to MDP is necessary. Next, the deep deterministic policy gradient (DDPG) finds the solution to the MDP and then processes some action refinement to transfer the continuous offloading decision made by DDPG into the integer values. With the increasing number of UE devices, the DDPG algorithm can decrease energy usage by up to around 92% compared to other algorithms. Based on the model of RL, a study [36] proposes a deep Q-network (DQN) algorithm to optimize task offloading and resource allocation. However, the DQN method struggles with continuous action space in high dimensions. The DDPG algorithm can resolve this problem of DQN, offering smooth target updates to enhance learning stability. Taking into account multiple UE devices in MEC networks, DDPG reduces energy consumption by up to roughly 70%, and the result of the DQN is slightly lower than that of DDPG when the weight parameter of computation delay is equal to 0.5. Furthermore, in ref. [26], the authors propose a support vector machine (SVM)-based federated learning method to actively determine UE associations. This algorithm allows BS and UE to cooperatively train a global support vector machine (SVM) model that can predict the best associations, thus optimizing the offloading

strategy and resource allocation to minimize the power consumption of task execution and task transmission. Therefore, the SVM-based federated learning (FL) approach can reduce the consumed energy by up to 20.1% compared to the traditional centralized SVM algorithm. Additionally, to minimize the energy use at the MEC end, there is a long short-term memory (LSTM)-based algorithm to predict the long-term workload while controlling the number of active PUs in the MEC server [28]. In the particular case of [34], the entire optimization problem is divided into two sub-tasks. For the first one, an online multi-agent RL is combined with a game-theory-based algorithm. The WoLF-PHC algorithm, which is a multi-agent RL algorithm, is used to develop the offloading strategy (i.e., the selection of ANs) due to its low complexity and high scalability. Then, a game-theory-based algorithm is utilized to calculate the power allocation from the UE side to the MEC end. The results show that this entire algorithm can increase energy efficiency by up to approximately 65% versus other approaches.

On the other hand, due to the increasing complexity of dynamic networks, adjusting the hyperparameters of machine learning algorithms and controlling the convergence time present significant challenges.

7.3.5. Lyapunov Optimization

Lyapunov algorithms are widely used in network optimization. Lyapunov optimization optimally controls dynamic systems using Lyapunov functions. Thus, it is appropriate to employ low-complexity Lyapunov optimization algorithms in dynamic and stochastic IoT networks. For instance, ref. [28] utilizes a combination of Lyapunov optimization and successive convex approximation to devise an optimal strategy for task offloading and resource allocation, considering the dynamic workload arrival at the MEC end. This proposed algorithm accounts for both FDMA and NOMA offloading plans and is designed to adapt to dynamic networks. The algorithm using NOMA offloading can reduce the total system cost by up to 51% compared to other methods when the system is stable. However, the algorithm using FDMA offloading consumes slightly more energy than when using NOMA offloading. Correspondingly, the authors in ref. [34] propose the joint task scheduling and resource allocation (JTSRA) algorithm to optimize the local computational task scheduling and resource allocation from the UE to the MEC end. Furthermore, ref. [40] presents a joint channel allocation and resource management (JCRM) algorithm to optimize the offloading solutions and the data rate during the process from the UE side to the MEC end. For the different channel numbers, JCRM boosts the network utility by up to 80% for different channel quantities and 31% with the increasing weight parameter. Furthermore, a perturbed Lyapunov optimization approach is used to manage a virtual queue. By controlling the stability of the virtual queue, it guarantees the deadline of tasks and handles the backlog of the actual task queue, thereby designing an efficient offloading strategy. Hence, a delay-aware energy-efficient (DAEE) algorithm is proposed to develop an adaptive offloading strategy with resource allocation of transmission time. At 3000 and 6000 bits/s task arrival rates, DAEE with perturbed Lyapunov optimization cuts the energy consumption by approximately 87% and 75%, respectively, versus other algorithms [41].

Note that in a large-scale complex dynamic network environment, applying Lyapunov optimization algorithms may be challenging due to issues such as extended convergence time and difficulty in finding a clear optimal solution.

7.3.6. Game Theory

Game theory is an effective way to deal with the interactions between incentive structures of a competitive nature, particularly, it describes a reactive decision-ahead scenario where other competitors react to their own decisions, and then adjust the initial decision to make a better final decision [74]. In ref. [39], the authors employ potential games to design an optimal computation offloading strategy, which effectively balances the load across multiple WAPs.

Although game theory can provide some theoretical optimal solutions, these solutions may be challenging to implement in real scenarios.

7.3.7. Branch and Bound

The branch and bound algorithm is a solution space search algorithm used in integer programming or optimization problems and is a global optimization method. In this regard, it can find the global optimal solution to integer optimization problems. At the same time, branch and bound is a common method for solving combinatorial optimization problems. For instance, ref. [27] proposes an improved branch and bound algorithm (IBBA) to create a low-complexity computation offloading and resource allocation algorithm. It can reduce the consumed energy by up to approximately 63% in the case of varying task complexity. While the energy consumption of ROP is lower than IBBA with the relaxation of delay requirements, the tasks processed by ROP may not meet the specified constraints. Nevertheless, when tackling large problems, the branch and bound method might be challenging due to excessive complexity and extended time to find the optimal solution or excessive storage requirements.

7.3.8. Others

In ref. [44], a ranking-based binary offloading and resource allocation algorithm (RBORA) is proposed that takes into account the offloading priorities on the UE side. Despite various factors (e.g., the varying number of UE devices, and the maximum CPU frequency of the MEC server), this method can reduce power consumption. Although there are two algorithms that perform better than RBORA, it has the advantage of low complexity and also takes into account the channel estimation error.

7.4. Approaches of Energy Consumption Minimization

In general, there are multiple technologies that can be employed to reduce the overall energy consumption in MEC networks.

7.4.1. Optimal Computation Offloading and Resource Allocation Strategy

From the preceding discussion, it is clear that designing optimized computation offloading and resource allocation strategies to minimize energy consumption in MEC networks is of great significance. Such strategies often take into account various factors such as task characteristics, device capabilities, network conditions, and latency requirements. Effective strategies can be designed for computation offloading and task allocation using various implemented algorithms to minimize energy consumption. This will efficiently address the issue of energy minimization in MEC networks.

7.4.2. Sleep Mode Technology

To reduce the overall energy consumption of the network, we can set up an energy-saving sleep mode within the MEC servers. That is, the working status of servers can be optimized according to real-time network demands. For instance, the MEC end can be adjusted to sleep mode when the demand for task processing is low. In ref. [28], the authors assume that there are many PUs within the MEC node. They can independently process data, and each of them can be turned on or off based on the processing demand. Therefore, by optimizing the number of activated PUs, we can not only lower the processing energy consumption at the MEC end but also improve the overall energy efficiency of the network.

7.4.3. Demand Forecasting

The rapid development of ML plays a key role in predicting MEC network environments, especially under certain circumstances. For example, by utilizing relevant ML-based algorithms, we can anticipate future network conditions and task volumes, etc. This enables us to design adaptive strategies that accommodate network dynamic changes, enhance the system's responsiveness, and lead to improved network energy efficiency. The authors

in [75] propose a distributed adaptive task offloading algorithm, which can predict the task queue length in future drones, obtains related information, and then uses this information in the load adaptation strategy, thereby achieving the minimization of task completion time and energy consumption.

7.4.4. Data Compression

Effective compression layers reduce the size of offloaded computational task data in low-bandwidth states [76]. Essentially, a smaller task data size requires less transmission time and processing time. This not only optimizes network resource utilization but also reduces the energy consumption associated with data transmission and processing, thereby leading to energy savings in the operation of the network. However, it is noted that data compression does not always reduce energy usage because the process of data compression and decompression can consume a certain amount of energy. Therefore, designing a reasonable compression strategy is necessary. For example, ref. [77] proposes a data compression scheme used in multi-user drone edge computing to reduce energy consumption on the device side, and further optimizes the offloading strategy for compression tasks to enhance the overall computational performance of the network.

7.4.5. Caching Technology

Caching technology is also an efficient strategy to reduce energy consumption. For instance, ref. [23] proposes a mobile cache system on the UE side, which means the computational tasks do not need to be executed locally or offloaded to the MEC end if the required content has been cached on the UE end, thereby reducing energy consumption to some degree. In ref. [78], the authors construct a multi-user MEC architecture for interactive AR/VR games based on multimodal semantic communication, and they create a two-way cache task model (i.e., considering input data generated from both local devices and the internet) for cache-enhanced computation. In this model, the UE end can actively cache map information that each player frequently accesses, thus reducing the latency and energy consumption required for local data upload or map information download, and achieving system cost minimization. Additionally, caching can be deployed not only at the UE end but also in BSs or MEC servers. So that the UE side can inquire whether the required content exists within the MEC or BS end, if it does exist, the content can be directly returned without the need for the UE to offload computational tasks, or for the MEC or BS side to carry out related processing and other operations, thereby reducing unnecessary energy consumption. Therefore, formulating reasonable caching architectures and strategies plays a crucial role in improving the energy efficiency of MEC networks.

7.4.6. Network Slices

Moreover, network slices can improve the allocation strategies for resources and also offer the possibility to perform optimizations in energy efficiency aspects [24]. Firstly, by adopting network slicing, we can utilize the resources of MEC networks more efficiently to meet the requirements of various types of tasks. Specifically, network slicing can group tasks with similar needs together and achieve more effective utilization of network resources [79]. Secondly, efficient resource utilization is closely linked to the maximization of network energy efficiency in MEC networks. Improved resource allocation strategies and higher resource utilization can contribute to energy consumption optimization to a certain extent. By more precisely allocating resources to the tasks that require them, and by reducing idle or inefficient resource usage, the network system can operate more efficiently. Such optimization can both reduce wasted energy and potentially meet task requirements across a broader scope, thereby having the potential to enhance the overall energy efficiency of the network.

7.4.7. Containers

As a lightweight virtualization technology, containers allow multiple applications to run in isolation on a single system, without the need to allocate full operating system resources for each application. This efficient resource utilization can significantly reduce the cost of device hardware and energy. Furthermore, we can utilize docker commands to instantiate and manage a processing container within an MEC server. Using the docker start command, a container can be transitioned to a running state. Conversely, docker stop commands can be used to transition the container to an idle state. Setting containers' different statuses can decrease the unnecessary energy consumption in MEC networks [11]. Furthermore, the lightweight characteristics of containers enable rapid application migration, meaning that containers running on one device can easily be moved to another. Due to the lightweight nature of the migration process, such an operation can significantly reduce the energy consumption associated with migration, thereby optimizing deployment strategies to minimize the overall energy consumption of the network. In summary, container technology plays a crucial role in MEC networks and directly impacts the maximization of energy efficiency.

7.4.8. Some Energy-Relevant Techniques

Finally, the application of some energy-related technologies is able to optimize energy efficiency in MEC networks. The first is energy harvesting (EH), which can capture green energy, such as solar energy, wind energy, and solar radiation, from the surrounding environment to provide constant power to UE devices' batteries, thus extending their lifespan and reducing reliance on external power sources [80]. Moreover, EH devices (e.g., solar panels) can be installed within BSs to provide renewable energy for the MEC servers to improve network power flexibility. In addition, surplus energy can be stored in local energy storage devices for future use. Simultaneously, different servers can share their leftover energy through a microgrid connected to the BSs, consequently reducing the amount of electrical energy obtained from the power grid and optimizing overall energy utilization in MEC networks [81]. As a result, the additional energy provided by EH to the MEC network enables the implementation of more aggressive energy-saving strategies, thereby further reducing energy consumption.

The second strategy is dynamic voltage and frequency scaling (DVFS). DVFS enhances energy efficiency by dynamically adjusting the CPU frequency according to the load requirements. This technique can effectively balance performance and energy utilization [36]. Specifically, to improve battery efficiency, the system-on-chip (SoC) in the UE can incorporate DVFS features. For instance, the cpufreq and devfreq modules in the Android Linux kernel can dynamically set the CPU frequencies and other components with DVFS capabilities. The governor adjusts various frequency algorithms to balance power consumption and performance, i.e., it increases the CPU frequency under a high workload and decreases it otherwise [16]. Furthermore, DVFS technology also can be integrated into MEC and BSs to adjust the working voltage and frequency [82]. It should be noted that the configuration of DVFS needs to take into account the balance between QoS and energy efficiency in the network, in order to meet the targeted objectives.

Lastly, a wireless power communication network (WPCN), a form of wireless energy transfer (WET), focuses on how to efficiently perform wireless energy transmission, and then utilize the collected energy for communication. WPCN enables wireless devices to receive power through the air, reducing the energy needed for wired connections and traditional charging as well as lowering the energy consumption and operational costs of devices. By integrating WPCN with MEC, the network can benefit from optimized resource allocation, thereby minimizing the overall network energy consumption [83]. In addition, simultaneous wireless information and power transfer (SWIPT) technology is another application of WET, which can supplement the battery capacity of electric vehicles [84]. Compared to traditional charging methods, SWIPT receives power wirelessly, allowing charging whether the vehicle is moving or parked. This reduces the reliance on

the UE-limited battery, enhancing the energy efficiency and flexibility of the MEC network. By alleviating energy consumption stress, it contributes to more effective task offloading and processing within the network. However, even though SWIPT technology offers a new direction for the energy-efficient charging of a moving UE end, the practical application and widespread adoption of SWIPT still face challenges in wireless power transfer efficiency, transmission distance, and economic costs. Therefore, continuous optimization and research are key to realizing its potential for energy savings.

8. Challenges and Future Directions

Computational task offloading strategies and resource allocation schemes in MEC networks along with minimizing energy consumption have received significant attention from the research community in recent years. However, there remain a number of challenges and potential future directions in these areas, which will be discussed in this section. Figure 7 shows the architecture of this section.

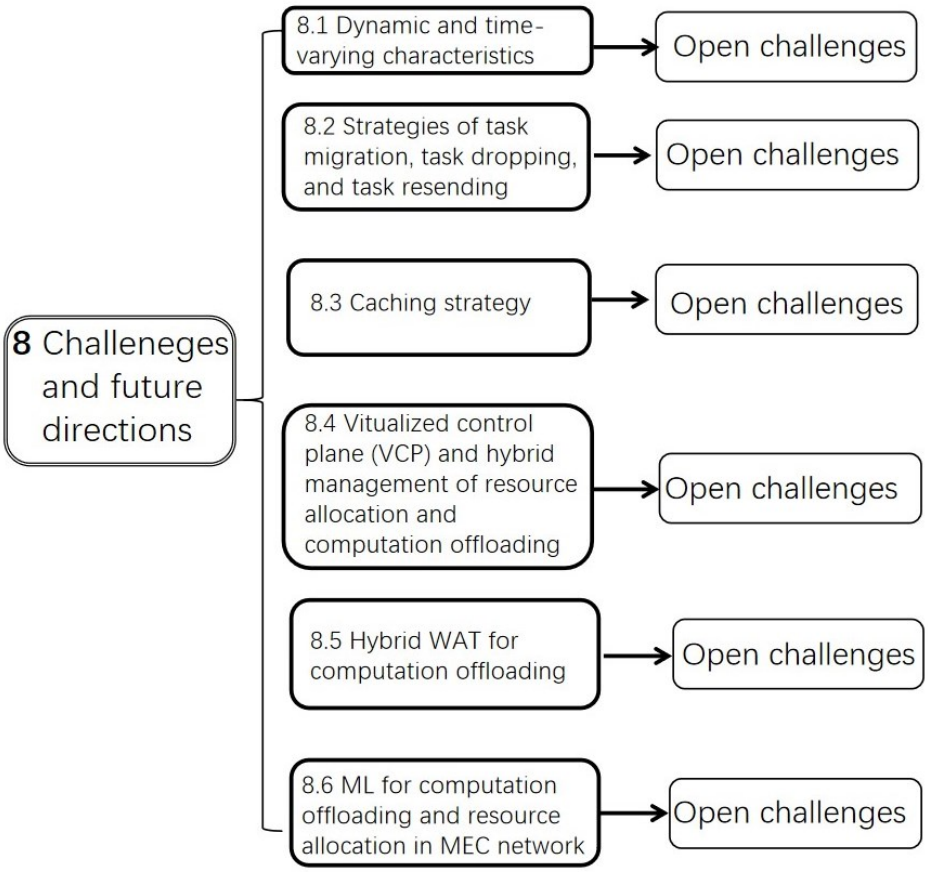

Figure 7. Architecture of Section 8.

8.1. Dynamic and Time-Varying Characteristics

Dynamics in MEC networks describe state or action changes over time. The dynamic nature of network environments significantly influences the offloading strategy, resource allocation optimization, and thus energy utilization maximization in MEC networks. Specifically, dynamic factors may render the original offloading scheme and resource allocation strategy unsuitable for changing scenarios. Thus, updating the optimal computation offloading and resource allocation strategy is necessary to achieve energy consumption minimization.

As a result, monitoring these dynamic factors with time-variant variables is beneficial, as the network status fluctuates across time slots. This approach better simulates real network conditions and improves experimental alignment with real networks. Understanding

these dynamics enhances the prediction of network trends and future behavior, aiding the optimization of computation offloading, resource allocation, and energy minimization in MEC networks.

The dynamic factors mentioned here refer to elements that change dynamically in the external environment of the network, e.g., mobility of UE, the status of the UE's battery, wireless path loss, available channel numbers, channel gain, and CSI. They also include dynamic characteristics during computation offloading and resource allocation such as the dynamic processes of task generation, offloading, execution, and offloading request. More typical dynamic elements, settings, and open challenges in MEC networks are shown in Table 5.

Open Challenges for Dynamic and Time-Varying Characteristics

1. No or limited consideration for dynamic elements
   Many existing studies either overlook the dynamic characteristics or consider too few dynamic factors in networks. Therefore, we believe the current challenge lies in deeply and comprehensively evaluating the impact of network dynamics and other corresponding time-varying elements on energy minimization in the optimization of computation offloading and resource allocation.
2. Insufficient consideration and simple settings for dynamic factors
   Some studies account for the dynamics but often oversimplify their randomness, compromising the realism of scenarios. For example, the UE joins and leaves in a region periodically [82]. It is also essential to consider how dynamic elements interact. For example, the battery level of the UE can affect the rate and quantity of task generation, offloading, and local execution. During the dynamic process of battery level changes, a low battery status may slow down the rate and volume of task generation and offloading (low transmission power), etc., in order to conserve battery life. On the other hand, the dynamic change in the number of tasks offloaded to the MEC server can impact the CPU's processing capacity and speed. If the server's task queue becomes overly backlogged, it could result in an overloaded server, thereby reducing the CPU's processing capacity. Moreover, if the MEC server processes a large number of tasks and returns a substantial amount of result data to the UE end, this might require frequent data reception and download operations on the UE side. These operations could significantly affect the UE's battery life, potentially leading to a rapid depletion in the battery level, thereby impacting task generation, etc. As such, considering the interplay of various dynamic factors, we should consider dynamic elements comprehensively and design appropriate computation offloading and resource allocation strategies with the aim of minimizing network energy consumption.
   The existing literature offers basic and simple dynamic configurations. For instance, ref. [69] mentions dynamic task generation, yet lacks details on task size and computation requirements, overlooking MEC processing procedures and times [41]. The authors of refs. [11,36] consider the UE battery status, but ignore its influence on task generation and offloading. Additionally, ref. [40] overlooks the correlation between offloading speed and volume, and task processing time. More challenges are outlined in Table 5.
3. The impact of dynamics
   While some studies touch on network dynamics, their impact is not sufficiently examined. For example, ref. [41] models UE mobility, but with only one AN and MEC server, it fails to optimize the offloading strategy for moving UE. Despite a perfect dynamic setting in ref. [85], it neglects the influence of UE mobility on dynamic MEC networks. Similarly, in ref. [64], while prioritizing offloading requests and reflecting network dynamics, it could also jointly consider minimizing energy consumption. Consequently, when designing and optimizing computation offloading and resource allocation strategies within MEC networks to minimize energy consumption, it is critically important to fully acknowledge the impact of various dynamic factors on the

network. For instance, the mobility of UE may influence offloading decisions, which is necessary to establish multiple BSs and wireless channels in the model to optimize UE device's offloading strategies. Likewise, considering the dynamic and random nature of wireless channels, we need to investigate the potential impact of a dynamic wireless environment on aspects such as maximum bandwidth and the number of available channels, dynamic channel attenuation parameters, etc. It is essential to understand and effectively manage these influences in order to establish a robust network model and set appropriate optimization objectives. To facilitate this, we can design flexible, efficient offloading and resource allocation strategies that can function effectively in various, complex, and real-world environments.

4. Multidimensional dynamic factor consideration
For highly mobile application scenarios (e.g., UAVs and AVs), we should additionally consider the strategies aimed at optimizing trajectory, speed, acceleration, etc., to minimize energy consumption. For instance, in UAV-assisted MEC networks, when ground UE needs to offload tasks to the MEC with UAV assistance, or directly to the UAV [47]. We need to consider optimizing parameters such as UAVs' flight trajectory and speed, etc. This optimal strategy not only allows for the carriage of more additional offloading tasks when necessary, assisting MEC with offloading but also contributes to network energy minimization by optimizing energy consumption based on actions such as the flight trajectory with parameters and task processing. Similarly, in the case of AV mobility, it is necessary to consider dynamically optimizing factors such as the driving trajectory, speed, and turning angles to efficiently implement task offloading and resource allocation strategies in complex vehicular network environments, thereby enhancing the network's energy efficiency. In addition, related MEC-UE interactive applications can be designed to monitor the real-time optimization analysis of a UAV's or AV's trajectory.

Table 5. Dynamic characteristic, settings, and open challenges.

Dynamic Characteristic	Ref.	Settings	Open Challenges
The mobility of UE	[82]	UE periodically join and leave, impacting the distance between UE and CAPs (time-varying variable)	Too simple settings to reflect the real network conditions
	[41]	Independent identical distribution for UE device's locations on the x- and y-axes (time-varying variables)	One single AN and MEC server, and all tasks need to be offloaded, no consideration for changing of offloading strategy
	[69]	Uniform distribution of the distance between UE and MEC servers (time-varying variable)	
	[85]	Random walking with arbitrary speed and direction	No full reflection the impact of UE mobility on dynamic network
Task generation	[41]	The number of task generations/arrival follows the Poisson distribution (time-varying variable)	No other information about tasks and MEC processing procedure (too simple setting)
	[11]	A task includes data input size, processing density, and the execution time limit following the Poisson distribution (time-varying variables)	No consideration for generation speed/amount, and the impact from UE battery level
Task processing time of MEC server	[11]	Exponential distribution (time-varying variable)	No consideration for the impact between task offloading speed/amount and task processing time, UE receiving data battery energy consumption, etc.
Battery status of UE	[85]	Unknown distribution (need to be predicted to the future energy level; time-varying variable)	No consideration for the relationship between UE
	[36]	Poisson distribution (time-varying variable)	Battery level and the speed/amount of task generation, offloading, and local execution, etc.
Task offloading	[40]	The stochasticity of the communication from UE to mBSs (time-varying variable)	No consideration for the
	[40]	The stochasticity of the communication from mBSs to MBS (time-varying variable)	Relationship between task offloading and processing, and UE battery level

Table 5. *Cont.*

Dynamic Characteristic	Ref.	Settings	Open Challenges
Offloading request	[69]	The number of incoming offloading requests (including the requested resources matrix and the distance matrix) follows the Poisson distribution (time-varying variables)	
	[64]	Based on the priority of arriving requests, if there is resource scarcity, the resources that were initially allocated for normal requests are dynamically reassigned to high-priority requests	No consideration for minimization energy consumption
Path loss	[82]	The small-scale Rayleigh fading coefficient follows a complex Normal distribution (time-varying variable)	
Uplinks availability	[41]	Uniform distribution for available sub-channels number (time-varying variable)	
Maximum bandwidth availability			(a) Consider the relationship between maximum bandwidth availability and the offloading tasks amount/speed; (b) consider the impact on maximum data transmission rate
Channel power gain	[41]	Exponential distribution (time-varying variable)	
	[86]	Independent identical distribution (time-varying variable)	
Channel state information (CSI)	[85]	Unknown distribution (can be obtained by MEC server at the start of each time slot; time-varying variable)	

CAP: computational access point; CSI: channel state information; mBS: micro cell base station; MBS: macro cell base station; MEC: mobile edge computing; UE: user equipment.

8.2. Strategies of Task Migration, Task Dropping, and Task Resending

Due to the dynamic nature of MEC networks, the problems such as network congestion, UE mobility, or insufficient MEC server resources need dynamic task migration schemes to ensure network QoS. Task migration can take place at any stage before task processing completion. Changes in the offloading strategy because of network dynamics or task time constraints may also require task dropping or resending. Therefore, the optimal strategy for task migration, dropping, and resending, which is subjected to network conditions or offloading strategies, requires a thorough assessment to minimize energy consumption while maintaining network QoS.

Open Challenges for Task Migration, Task Dropping, and Task Resending

1. No consideration for task migration, dropping, or resending
   Many studies neglect task migration, dropping, or resending under dynamic network conditions. Therefore, we suggest considering these actions under necessary circumstances to better adapt to the complex and dynamic MEC networks, thereby optimizing energy consumption minimization.
2. Multiple actions occur simultaneously
   While some papers discuss task migration, dropping, or resending, they often ignore scenarios where multiple actions might occur simultaneously. Hence, to optimize computation offloading and resource allocation in dynamic and complex MEC networks, we should consider these actions with the ultimate goal of minimizing energy consumption.
3. The impact of dynamic factors and optimal strategy
   In dynamic MEC networks, the optimal strategies for task migration, dropping, and retransmission often overlook their impact on energy consumption minimization. For instance, when deciding to perform task migration (or dropping, resending, or multiple operations combinations), we should formulate the optimal strategy to determine the best time, location, and content for the migration to minimize energy consumption while ensuring network QoS.
   Regarding the migration time, we need to assess when it can achieve both QoS assurance and network energy minimization. If network performance is predictable, advanced strategies for predicting the optimal migration time can be developed using RL. Otherwise, the key issue lies in making the best choice between migration and

waiting for migration, to find the optimal migration time that can balance both QoS and energy consumption minimization.

As for the location of migration, we need to evaluate the dynamic conditions of the target migration server (e.g., available resources and migration bandwidth) and the migration cost (e.g., energy consumption during migration) to determine the optimal migration location. There may be instances when the network experiences dynamic changes during the migration process, thus tasks may need to be migrated more than once. Consequently, in such complex dynamic networks, choosing the best migration time and the appropriate server presents a significant challenge.

When considering specific migration tasks (or sub-tasks), we need to understand the relationship between them. For instance, certain task execution needs to follow a specific sequence, which indicates a clear dependency relationship among them [87]. As a result, operations such as task migration must take this inter-dependency into account as a key constraint to ensure effective task execution and system stability. On the other hand, if a task is subject to partial offloading, the impact of migrating, dropping, or resending sub-tasks on the overall task needs to be carefully evaluated. The considerations for operations for task dropping and resending are similar to those for task migration, taking into account factors such as the time of occurrence, location, and specific tasks/sub-tasks. Particularly when multiple operations are considered to be happening simultaneously, the elements to be evaluated and optimization strategies in the dynamic and complex MEC network can become complicated. For example, within the same time frame, multiple operations might occur simultaneously on the same server, potentially making the dynamics of the network more complex, especially in terms of available resources, etc. This poses certain challenges to optimizing computation offloading and resource allocation aiming to minimize network energy consumption.

8.3. Caching Strategy

Caching is a very promising technology in MEC networks. By caching data in advance, we can reduce the communication between the UE and MEC end, thus saving system costs, increasing energy efficiency, and improving network reliability. Generally, the type of caching can be categorized on the basis of data context into several types, such as content caching (e.g., tasks computation results, web record), program caching (i.e., MEC application). From the perspective of the network location of the cache, the cache is divided into the UE, BS, or MEC end cache, etc.

Open Challenges for Caching Strategy

1. Elasticity of cache space

   The main challenge is that due to the limited resources of the MEC end and the dynamic nature of the MEC networks, caching policies need to adapt to the network's dynamic characteristics and ensure the resilience of the cache space. To improve the utilization of caching space, we need to consider optimizing the caching content update strategy within the flexible cache space to ensure minimizing the energy consumption in networks. In other words, we need to comprehensively optimize the caching space size, cached content, caching time, and the time and strategy for updating the cached content in a time-varying dynamic network, in order to ensure that while meeting the QoS requirements for task offloading and resource allocation assisted by caching, we also maximize network energy efficiency. For instance, in an MEC-assisted VR video service application scenario, ref. [88] takes into account the dynamic content popularity and employs a mixed strategy of deterministic task offloading and dynamic cache replacement to minimize service delays and system energy consumption. At each time slot, once the local VR device and the MEC server receive a new tile and compute it, they simultaneously update their caching content to make full use of the caching resources. However, this overlooks the impact that the

dynamic nature has on the cache space and strategy. For example, in a real multi-user scenario, there may be an increase in offloaded tasks received by the MEC server in a time slot, thus requiring dynamic adjustment of the cache space at the MEC end. Moreover, a thorough evaluation of the content popularity at this time may also be necessary.

2. Dynamic collaborative caching management
   When large amounts of data or code need simultaneous caching in the network, it is crucial to develop a coordinated caching policy between the UE and MEC servers, or among MEC servers. Simultaneously, to achieve the minimization of energy consumption in the computation offloading and resource allocation within the collaborative caching network model, the optimization of communication management in cooperative caching interaction interfaces (i.e., network protocols and APIs) between different devices must also be considered. That is to say, optimizing the strategy of different cached content along with considering the interaction among them to improve overall network performance and efficiency is an issue that requires our focused consideration.
   Furthermore, due to dynamic factors, this collaborative caching strategy may change over time. For instance, specific cached content might migrate to other servers. The target servers receiving this migration with an appropriate migration time must evaluate the ability to satisfy specific objectives but also consider the impact of the dynamic network environment on the cache migration, such as potential congestion in the migration channel.
   However, during cache content migration, caches may not update promptly, potentially leading to unnecessary computation offloading and resource allocation on the UE device's side, which reduces energy efficiency and impacts task computation. Therefore, a potential challenge is how to handle incoming task offloading and resource allocation requests during the process of cache migration and execution. This must be performed in order to avoid unnecessary task offloading and resource allocation, and thus ensure minimal energy consumption of the network. Ref. [89] considers UE mobility and uses a cooperative caching strategy in MEC systems. Before offloading a task from the UE side, the system queries whether the computation result exists in the MEC system. However, while this model accounts for the impact of cooperative caching and UE mobility, it does not fully address the potential issues of dynamic caching migration, and other dynamic influences (e.g., dynamic wireless channel in caching).

8.4. Virtualized Control Plane (VCP) and Hybrid Management of Resource Allocation and Computation Offloading

Most studies adopt either a centralized or a decentralized model for MEC in computing offloading and resource allocation, each with unique strengths and weaknesses. Highly scalable decentralized networks may lack global optimization goals such as overall network energy consumption and delay, which leads to inefficient resource allocation and task scheduling. On the other hand, centralized networks, such as those managed by an SDN controller, offer service functions such as surveillance, traffic regulation, and cluster administration [90], which enables global optimization. However, they might lack real-time responsiveness and adaptability due to centralized decision-making, which can increase processing delay, energy consumption, and bandwidth wastage. Therefore, the challenge is to develop a cooperative model that leverages the advantages of both centralized and decentralized models, focusing on efficient computation offloading, resource allocation, and minimizing network energy consumption. Figure 8 shows the hybrid layout of computation offloading and resource allocation.

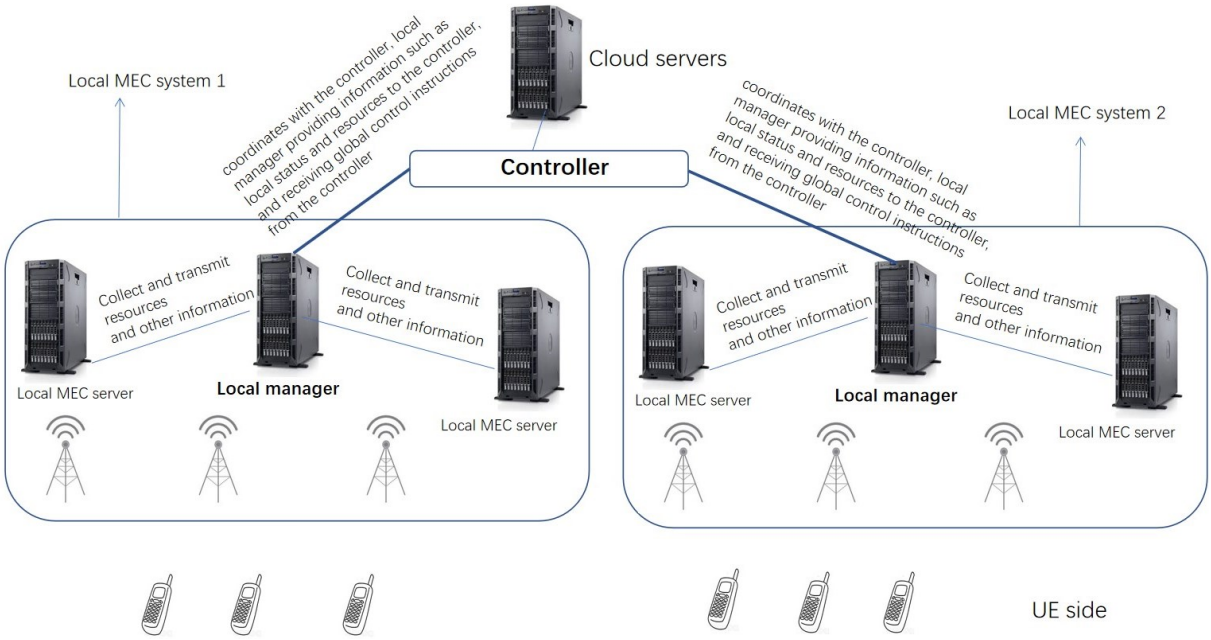

Figure 8. The hybrid layout of computation offloading and resource allocation.

Open Challenges for VCP and Hybrid Management of Resource Allocation and Computation Offloading

1. Single centralized or decentralized MEC layout for computation offloading and resource allocation
   Most current studies on computation offloading and task allocation utilize a single network management layout, i.e., either centralized or distributed. However, they often fail to maximize the benefits of both models in network management. For example, ref. [91] discusses that centralized learning or scheduling methods might face communication overheads as network size increases. Consequently, they propose a decentralized resource allocation framework for UAV scenarios where both the UAV and the ground-based MEC offer computing services, greatly improving the computational efficiency of the integrated aerial–ground MEC network. However, they do not consider the role of centralized global control from a global perspective to optimize energy efficiency.
2. Classification of strategy sets between the centralized controller and MEC system's VCP
   Combining the analyses from Sections 7.1–7.3, it is obvious to realize that in optimization of computation offloading and resource allocation, many strategies come into play. From a global network perspective, the strategy set managed by the SDN controller should consider global performance optimization, and it should supervise rather than interfere with the strategies in local MEC networks in order to efficiently manage MEC networks and optimize edge computing.
   In such a hybrid management mode, reasonably determining the scope of the strategy set ownership poses a challenge to optimize network energy utilization in dynamic MEC networks. In general, it is necessary to consider that the SDN controller monitors global information by collecting relevant data filtered by the local MEC server manager, to fully understand the workload and resource utilization, etc., for different MEC system regions. Then, the SDN controller can formulate relevant optimal global policies, update the local manager MEC server, and dispatch these policies to the other MEC servers. Meanwhile, within local MEC networks, we could adopt a distributed management model and optimize associated local strategies for computation offloading and resource allocation algorithms. A similar network layout is applied in [92], this model comprises different regional MEC systems and a master SDN controller, and aims to optimize computation offloading and resource allocation to minimize

the system cost. While this model shares similarities with the network design we have been discussing, the paper still adopts SDN-style centralized management and control to make decisions regarding computation offloading and resource allocation. Moreover, the paper does not consider the dynamism of MEC networks, thereby significantly lowering the workload of the SDN controller.

3. Network interruption between different MEC systems
As tasks migrate across multiple MEC system regions or when task processing results are migrated due to the mobility of UE, network connections may be interrupted. This situation can trigger unnecessary data retransmissions and signal interference, leading to additional network energy consumption. Therefore, effectively managing network connections within different MEC systems is a pressing issue. Particularly in situations where a large amount of UE traverses MEC system regions, it is critical to coordinate SDN and multiple MEC systems to optimize computation offloading and resource allocation, minimizing network energy consumption. If all UE behaviors across MEC system domains are supervised and managed solely by the SDN, this could lead to an excessive workload for the SDN controller. Conversely, if these behaviors are managed only within the MEC systems, there may be a lack of a global perspective, leading to sub-optimal computation offloading and task migration, and increased energy consumption. Thus, future research must consider strategies for coordinating management spaces and strategies between the SDN and MEC systems to formulate an excellent network connection management scheme and achieve superior energy efficiency and network performance.

8.5. Hybrid WAT for Computation Offloading

With the rapid expansion of the scale of MEC networks in the future, the number of offloaded tasks from UE to the MEC end will also significantly increase. In this process, wireless transmission energy consumption will occupy a substantial part of the network's total energy consumption. Therefore, designing and managing lower-energy-consumption wireless communication technologies while ensuring QoS will become a major challenge to be faced in the future.

Open Challenges for Hybrid WAT for Computation Offloading

Due to the dynamism and complexity of MEC networks, the application of various hybrid WATs in the process of computation offloading has become a major trend for the future. However, the rational selection of hybrid WATs in highly heterogeneous wireless networks to assist in computation offloading and resource allocation, and thus minimize energy consumption, is an area that will need particular focus in the future.

First, we need to comprehend the integration of different hybrid WAT technologies to leverage their respective strengths, thus ensuring QoS and optimizing energy consumption in various network environments. For instance, NOMA technology is an effective solution to enhance user connectivity and spectral efficiency, and it has received widespread attention in recent years. Applying NOMA in MEC can further improve the computation performance and user connectivity of ultra-dense IoT networks [28]. Furthermore, as UE is expected to be equipped with multiple antennas, the combination of MIMO and NOMA has been verified to enhance the spectral efficiency of communication systems [93].

Furthermore, ref. [94] mentions hybrid NOMA-MEC networks, where pure NOMA and OMA are combined for task-offloading strategies. As such, we can set different WAT combinations according to the distinct features of different network environments to enhance overall network energy efficiency. For example, applying OFDMA-MIMO instead of NOMA-MIMO can reduce certain decoding latency and interference, but this solution may be more applicable in scenarios where UE is evenly distributed.

In this regard, the design of hybrid WATs presents certain challenges, especially in multi-user and multi-antenna hybrid WAT network models, where interference might increase. Consequently, we need to design suitable hybrid WAT combinations and appro-

priate interference elimination or suppression techniques, also taking into consideration the potential additional energy consumption optimization by WAT combinations.

Therefore, in scenarios involving highly mobile communication models, the challenges are multifaceted. The joint optimization of parameters within hybrid WAT, computation offloading, and resource allocation is one key area. When aimed at minimizing energy consumption, these factors add complexity to the design and present significant challenges for model optimization. Moreover, as hybrid WAT technology is not yet widely standardized, further research and industry collaboration may be needed to ensure the interoperability and compatibility of the technology.

8.6. ML for Computation Offloading and Resource Allocation in MEC Network

In the dynamic environment of MEC networks, DRL serves as an ML technique that can adapt to the constant changes in the network. Through continuous data training and learning, DRL can gradually formulate optimized strategies related to computation offloading and resource allocation, thereby adapting to the high complexity of MEC networks. However, the training of DRL not only requires the utilization of certain computing resources but also consumes additional energy. In situations where the resources of both UE and MEC servers are limited, the task of carefully designing a reasonable DRL strategy and choosing the most suitable execution location becomes a critically important challenge.

Open Challenges for ML-Based Computation Offloading and Resource Allocation in MEC Networks

As is well known, DRL is well suited to adapt to dynamic networks, enabling the creation of optimized energy-efficient strategies for computation offloading and resource allocation. However, training a DRL model often requires a significant amount of resources and energy. In MEC systems, where resources are limited, training a DRL model that takes into account a variety of complex dynamic factors can demand even more resources and energy consumption. Therefore, as MEC network models become increasingly dynamic, determining the optimal location for DRL training emerges as a potential challenge.

Firstly, the location for model training emerges as a crucial consideration. DRL training is typically not conducted on the UE side due to the resource and battery constraints [95]. Instead, training usually takes place in the cloud or on idle MEC servers. While the cloud generally boasts more robust computing and storage capabilities, enabling rapid DRL model training, it may be hindered by latency and bandwidth restrictions with MEC servers. On the other hand, utilizing idle MEC servers can minimize network latency, allowing for more real-time responses, but they may lack the sufficient computing resources of the cloud. Furthermore, real-time and continuous assessments of the idle server's computation offloading and resource allocation are necessary, involving the optimized allocation of resources required for executing tasks and model training. Particularly, on a shared server, the computational resources required for training a DRL model and those needed for processing other tasks may involve the same system storage resources. Therefore, if the server must handle a large number of computing tasks simultaneously during the DRL training process, this could have a certain impact on the processing capability and speed of the computational offloading tasks. Therefore, an optimized mechanism could be designed to dynamically allocate training computational resources among MEC servers or between MEC servers and the cloud end, adapting to changes in network resource availability.

Secondly, given the complex dynamics of MEC networks, the frequency of policy updates must also be carefully considered. Specifically, meticulous planning is required for when to perform policy updates to ensure timely responses. At the same time, the necessary bandwidth for transmission must be taken into account, ensuring that it does not adversely affect other vital services.

In summary, determining the appropriate DRL training locations and devising DRL policy update strategies to guarantee QoS and improve energy efficiency within a dynamic and complicated MEC network is an ambitious task and an open research challenge.

9. Conclusions

Through computation offloading and resource allocation, the study of realizing energy efficiency in the MEC architecture is gaining significantly. In view of the current progress in related fields, in this paper, we distinguish our work from other surveys and articulate its unique contributions. Then, we survey the concept of MEC and the architecture of MEC networks. Next, we describe the basic process of computation offloading and task execution. In addition, we also exemplify the application scenarios involving computational task offloading in the current popular MEC networks. Subsequently, the concept of resource allocation and the types and introduction of resources in MEC networks are explained. Later, we investigate the optimization problems involving minimizing network energy consumption, which includes the details of computation offloading and resource allocation, and also illustrate and analyze some energy-saving technologies and implemented algorithms. Finally, we discuss existing challenges and future research directions, aiming to accelerate the development in this field further.

Author Contributions: Conceptualization, A.A.S.; Methodology, J.Y. and A.A.S.; Formal analysis, J.Y.; Investigation, J.Y. and D.P.; Data curation, A.A.S.; Writing—original draft, J.Y.; Writing—review & editing, A.A.S. and D.P.; Visualization, D.P.; Supervision, A.A.S. and D.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the PETRAS National Centre of Excellence for IoT Systems Cybersecurity which has been funded by the UK EPSRC under grant number EP/S035362/1 and the Royal Academy of Engineering under award reference code RCSRF2223-1645.

Data Availability Statement: No associated data to be shared related to this work.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AF	Amplify-and-forward
AN	Access node
AP	Access point
AR	Augmented reality
BBU	Baseband unit
BS	Base station
CAP	Computational access points
CCCP	Concave–convex procedure
CPU	Central processing unit
CSI	Channel state information
CSAO	Computation offloading strategy and resource allocation optimization
CSCO	Consolidated stochastic computation offloading
CSSCA	Constrained stochastic successive convex approximation-based
D2D	Device to device
DAEE	Delay-aware energy-efficient
DF	Decode-and-forward
DDPG	Deep deterministic policy gradient
DQN	Deep Q-network
DRL	Deep reinforcement learning
DSCI	Delay sensitive and computational intensive
DT	Digital twin
DVFS	Dynamic voltage and frequency scaling
EH	Energy harvesting
ETSI	European Telecommunication Standards Institute
FDMA	Frequency division multiple access
FIFO	First-in-first-out
FL	Federated learning

FOV	Field of view
FPGA	Field-programmable gate array
GABSRS	Genetic-algorithm-based BS Selection and resource scheduling
GPU	Graphics processing unit
HMD	Head-mounted display
HR	Hybrid relaying
IBBA	Improved branch and bound algorithm
IBCD	Inexact block coordinate descent
IETF	Internet Engineering Task Force
IEU	Industrial end units
IIoT	Industrial internet of things
IoT	Internet of things
IoVT	Internet of video things
IPM	Interior point method
ISG	Industry specification group
IV	Industrial vehicle
JCRM	Joint channel allocation and resource management
JTSRA	Joint task scheduling and resource allocation
LP-WAN	Low-power wide-area network
LSTM	Long short-term memory
MAR	Mobile augmented reality
MANO	Management and orchestration
MBS	Macro cell base station
mBS	Micro cell base station
MCC	Mobile cloud computing
MD	Mobile devices
MDP	Markov decision process
MEC	Mobile edge computing
MIMO	Multiple input multiple output
MMC	Mobile micro cloud
MNO	Mobile network operators
NICRA	Newton-IPM-based computing resource allocation
NOMA	Non-orthogonal multiple access
OFDMA	Orthogonal frequency division multiple access
OS	Operating system
PU	Processing unit
QoS	Quality of service
RAM	Random access memory
RBORA	Ranking-based binary offloading and resource allocation
RODRA	Relaxed offloading decision and resource allocation
ROP	Relaxing optimization policy
RRH	Remote radio head
RSU	Roadside unit
SBS	Small cell base station
SCALE	Successive convex approximation for the low-complexity
SCC	Small cell cloud
SDN	Software-defined networking
SDO	Standards developing organizations
SNR	Signal-to-noise ratio
SoC	System-on-chip
SVM	Support vector machine
SWIPT	Simultaneous wireless information and power transfer
TDMA	Time division multiple access
UAV	Unmanned aerial vehicle
UDN	Ultra-dense networks
UE	User equipment
VM	Virtual machine
V2I	Vehicle-to-infrastructure

V2V	Vehicle-to-vehicle
V2V2I	Vehicle-to-vehicle-to-infrastructure
VCP	Virtualized control plane
VNF	Virtualized network function
VIM	Virtualization infrastructure management
VPN	Virtual private network
VR	Virtual reality
VS	Versus
WAP	Wireless access point
WAT	Wireless access technology
WBAN	Wireless body area networks
WET	Wireless energy transfer
WG	Working group
WLAN	Wireless local area networks
WL	Wireless link
WPCN	Wireless power communication network

References

1. Bolettieri, S.; Bruno, R.; Mingozzi, E. Application-aware resource allocation and data management for MEC-assisted IoT service providers. *J. Netw. Comput. Appl.* **2021**, *181*, 103020. [CrossRef]
2. Li, X.; Zhao, L.; Yu, K.; Aloqaily, M.; Jararweh, Y. A cooperative resource allocation model for IoT applications in mobile edge computing. *Comput. Commun.* **2021**, *173*, 183–191. [CrossRef]
3. Wang, Q.; Guo, S.; Liu, J.; Yang, Y. Energy-efficient computation offloading and resource allocation for delay-sensitive mobile edge computing. *Sustain. Comput. Inform. Syst.* **2019**, *21*, 154–164. [CrossRef]
4. Wang, Z.; Lv, T.; Chang, Z. Computation offloading and resource allocation based on distributed deep learning and software defined mobile edge computing. *Comput. Netw.* **2022**, *205*, 108732. [CrossRef]
5. Zhu, M.; Gao, S.; Tu, G.; Chen, D. Multi-Access Edge Computing (MEC) Based on MIMO: A Survey. *Sensors* **2023**, *23*, 3883. [CrossRef]
6. Kuang, Z.; Ma, Z.; Li, Z.; Deng, X. Cooperative computation offloading and resource allocation for delay minimization in mobile edge computing. *J. Syst. Arch.* **2021**, *118*, 102167. [CrossRef]
7. Wu, G.; Yang, C.; Li, S.; Li, G.Y. Recent advances in energy-efficient networks and their application in 5G systems. *IEEE Wirel. Commun.* **2015**, *22*, 145–151. [CrossRef]
8. Bishoyi, P.K.; Misra, S. Enabling green mobile-edge computing for 5G-based healthcare applications. *IEEE Trans. Green Commun. Netw.* **2021**, *5*, 1623–1631. [CrossRef]
9. Kekki, S.; Featherstone, W.; Fang, Y.; Kuure, P.; Li, A.; Ranjan, A.; Purkayastha, D.; Jiangping, F.; Frydman, D.; Verin, G.; et al. *MEC in 5G Networks*; ETSI White Paper No. 28; ETSI: Valbonne, France, 2018, *28*, pp. 1–28.
10. Sabella, D.; Alleman, A.; Liao, E.; Filippou, M.; Ding, Z.; Baltar, L.G.; Srikanteswara, S.; Bhuyan, K.; Oyman, O.; Schatzberg, G.; et al. *Edge Computing: From Standard to Actual Infrastructure Deployment and Software Development*; ETSI White Paper; ETSI: Valbonne, France, 2019; pp. 1–41.
11. Xu, S.; Liu, Q.; Gong, B.; Qi, F.; Guo, S.; Qiu, X.; Yang, C. RJCC: Reinforcement-learning-based joint communicational-and-computational resource allocation mechanism for smart city IoT. *IEEE Internet Things J.* **2020**, *7*, 8059–8076. [CrossRef]
12. Li, C.; Zhang, Y.; Luo, Y. Adaptive handover based on traffic balancing and multi-dimensional collaborative resource management in MEC environment. *J. Supercomput.* **2022**, *78*, 6752–6787. [CrossRef]
13. Islam, A.; Debnath, A.; Ghose, M.; Chakraborty, S. A survey on task offloading in multi-access edge computing. *J. Syst. Arch.* **2021**, *118*, 102225. [CrossRef]
14. Feng, C.; Han, P.; Zhang, X.; Yang, B.; Liu, Y.; Guo, L. Computation offloading in mobile edge computing networks: A survey. *J. Netw. Comput. Appl.* **2022**, *202*, 103366. [CrossRef]
15. Sadatdiynov, K.; Cui, L.; Zhang, L.; Huang, J.Z.; Salloum, S.; Mahmud, M.S. A review of optimization methods for computation offloading in edge computing networks. *Digit. Commun. Netw.* **2022**, *9*, 450–461. [CrossRef]
16. Jiang, C.; Fan, T.; Gao, H.; Shi, W.; Liu, L.; Cérin, C.; Wan, J. Energy aware edge computing: A survey. *Comput. Commun.* **2020**, *151*, 556–580. [CrossRef]
17. Li, X.; Da Xu, L. A review of Internet of Things—Resource allocation. *IEEE Internet Things J.* **2020**, *8*, 8657–8666. [CrossRef]
18. Mach, P.; Becvar, Z. Mobile edge computing: A survey on architecture and computation offloading. *IEEE Commun. Surv. Tutorials* **2017**, *19*, 1628–1656. [CrossRef]
19. Luo, Q.; Hu, S.; Li, C.; Li, G.; Shi, W. Resource scheduling in edge computing: A survey. *IEEE Commun. Surv. Tutorials* **2021**, *23*, 2131–2165. [CrossRef]
20. Lu, Y.; Chen, X.; Zhang, Y.; Chen, Y. Cost-efficient resources scheduling for mobile edge computing in ultra-dense networks. *IEEE Trans. Netw. Serv. Manag.* **2022**, *19*, 3163–3173. [CrossRef]

21. Li, X.; Lan, X.; Mirzaei, A.; Bonab, M.J.A. Reliability and robust resource allocation for Cache-enabled HetNets: QoS-aware mobile edge computing. *Reliab. Eng. Syst. Saf.* **2022**, *220*, 108272. [CrossRef]
22. Cruz, P.; Achir, N.; Viana, A.C. On the edge of the deployment: A survey on multi-access edge computing. *ACM Comput. Surv.* **2022**, *55*, 1–34. [CrossRef]
23. Seo, Y.J.; Lee, J.; Hwang, J.; Niyato, D.; Park, H.S.; Choi, J.K. A novel joint mobile cache and power management scheme for energy-efficient mobile augmented reality service in mobile edge computing. *IEEE Wirel. Commun. Lett.* **2021**, *10*, 1061–1065. [CrossRef]
24. Gatzianas, M.; Mesodiakaki, A.; Kalfas, G.; Pleros, N.; Moscatelli, F.; Landi, G.; Ciulli, N.; Lossi, L. Offline Joint Network and Computational Resource Allocation for Energy-Efficient 5G and beyond Networks. *Appl. Sci.* **2021**, *11*, 10547. [CrossRef]
25. Trinh, B.; Muntean, G.M. A deep reinforcement learning-based resource management scheme for SDN-MEC-supported XR applications. In Proceedings of the 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 8–11 January 2022; IEEE: Piscataway, NJ, USA, 2022, pp. 790–795.
26. Wang, S.; Chen, M.; Saad, W.; Yin, C. Federated learning for energy-efficient task computing in wireless networks. In Proceedings of the ICC 2020–2020 IEEE International Conference on Communications (ICC), Dublin, Ireland, 7–11 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–6.
27. Vu, T.T.; Van Huynh, N.; Hoang, D.T.; Nguyen, D.N.; Dutkiewicz, E. Offloading energy efficiency with delay constraint for cooperative mobile edge computing networks. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–6.
28. Wang, Q.; Tan, L.T.; Hu, R.Q.; Qian, Y. Hierarchical energy-efficient mobile-edge computing in IoT networks. *IEEE Internet Things J.* **2020**, *7*, 11626–11639. [CrossRef]
29. Chen, X.; Cai, Y.; Shi, Q.; Zhao, M.; Champagne, B.; Hanzo, L. Efficient resource allocation for relay-assisted computation offloading in mobile-edge computing. *IEEE Internet Things J.* **2019**, *7*, 2452–2468. [CrossRef]
30. Liao, Y.; Shou, L.; Yu, Q.; Ai, Q.; Liu, Q. Joint offloading decision and resource allocation for mobile edge computing enabled networks. *Comput. Commun.* **2020**, *154*, 361–369. [CrossRef]
31. Huang, J.; Lv, B.; Wu, Y.; Chen, Y.; Shen, X. Dynamic admission control and resource allocation for mobile edge computing enabled small cell network. *IEEE Trans. Veh. Technol.* **2021**, *71*, 1964–1973. [CrossRef]
32. Vallero, G.; Deruyck, M.; Meo, M.; Joseph, W. Base Station switching and edge caching optimisation in high energy-efficiency wireless access network. *Comput. Netw.* **2021**, *192*, 108100. [CrossRef]
33. Dai, Y.; Zhang, K.; Maharjan, S.; Zhang, Y. Edge intelligence for energy-efficient computation offloading and resource allocation in 5G beyond. *IEEE Trans. Veh. Technol.* **2020**, *69*, 12175–12186. [CrossRef]
34. Zhou, F.; Feng, L.; Kadoch, M.; Yu, P.; Li, W.; Wang, Z. Multiagent RL aided task offloading and resource management in wi-fi 6 and 5G coexisting industrial wireless environment. *IEEE Trans. Ind. Inform.* **2021**, *18*, 2923–2933. [CrossRef]
35. Nguyen, P.D.; Ha, V.N.; Le, L.B. Computation offloading and resource allocation for backhaul limited cooperative MEC systems. In Proceedings of the 2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall), Honolulu, HI, USA, 22–25 September 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6.
36. Liang, Y.; He, Y.; Zhong, X. Decentralized computation offloading and resource allocation in mec by deep reinforcement learning. In Proceedings of the 2020 IEEE/CIC International Conference on Communications in China (ICCC), Chongqing, China, 9–11 August 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 244–249.
37. Li, Z.; Liu, J.; Chen, Y. An energy-efficient resource allocation strategy for vehicular networks. In Proceedings of the ACM Turing Celebration Conference-China, Chengdu, China, 17–19 May 2019; pp. 1–8.
38. Zhou, Z.; Yu, H.; Xu, C.; Chang, Z.; Mumtaz, S.; Rodriguez, J. BEGIN: Big data enabled energy-efficient vehicular edge computing. *IEEE Commun. Mag.* **2018**, *56*, 82–89. [CrossRef]
39. Li, Q.; Zhao, J.; Gong, Y. Computation offloading and resource allocation for mobile edge computing with multiple access points. *IET Commun.* **2019**, *13*, 2668–2677. [CrossRef]
40. Ren, J.; Mahfujul, K.M.; Lyu, F.; Yue, S.; Zhang, Y. Joint channel allocation and resource management for stochastic computation offloading in MEC. *IEEE Trans. Veh. Technol.* **2020**, *69*, 8900–8913. [CrossRef]
41. Wu, H.; Chen, J.; Nguyen, T.N.; Tang, H. Lyapunov-Guided Delay-Aware Energy Efficient Offloading in IIoT-MEC Systems. *IEEE Trans. Ind. Inform.* **2022**, *19*, 2117–2128. [CrossRef]
42. Huang, X.; He, L.; Chen, X.; Liu, G.; Li, F. A more refined mobile edge cache replacement scheme for adaptive video streaming with mutual cooperation in multi-mec servers. In Proceedings of the 2020 IEEE International Conference on Multimedia and Expo (ICME), London, UK, 6–10 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–6.
43. Huang, J.; Wan, J.; Lv, B.; Ye, Q.; Chen, Y. Joint computation offloading and resource allocation for edge-cloud collaboration in internet of vehicles via deep reinforcement learning. *IEEE Syst. J.* **2023**, *17*, 2500–2511. [CrossRef]
44. Wang, J.; Feng, D.; Zhang, S.; Liu, A.; Xia, X.G. Joint computation offloading and resource allocation for MEC-enabled IoT systems with imperfect CSI. *IEEE Internet Things J.* **2020**, *8*, 3462–3475. [CrossRef]
45. Ei, N.N.; Alsenwi, M.; Tun, Y.K.; Han, Z.; Hong, C.S. Energy-efficient resource allocation in multi-UAV-assisted two-stage edge computing for beyond 5G networks. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 16421–16432. [CrossRef]
46. Zhao, L.; Yang, K.; Tan, Z.; Song, H.; Al-Dubai, A.; Zomaya, A.Y.; Li, X. Vehicular computation offloading for industrial mobile edge computing. *IEEE Trans. Ind. Inform.* **2021**, *17*, 7871–7881. [CrossRef]

47. Yu, Z.; Gong, Y.; Gong, S.; Guo, Y. Joint task offloading and resource allocation in UAV-enabled mobile edge computing. *IEEE Internet Things J.* **2020**, *7*, 3147–3159. [CrossRef]
48. Liu, W.; Li, B.; Xie, W.; Dai, Y.; Fei, Z. Energy Efficient Computation Offloading in Aerial Edge Networks With Multi-Agent Cooperation. *IEEE Trans. Wirel. Commun.* **2023**. [CrossRef]
49. Lee, S.s.; Lee, S. Resource allocation for vehicular fog computing using reinforcement learning combined with heuristic information. *IEEE Internet Things J.* **2020**, *7*, 10450–10464. [CrossRef]
50. Gu, X.; Zhang, G. Energy-efficient computation offloading for vehicular edge computing networks. *Comput. Commun.* **2021**, *166*, 244–253. [CrossRef]
51. Huang, C.M.; Lai, C.F. The delay-constrained and network-situation-aware V2V2I VANET data offloading based on the multi-access edge computing (MEC) architecture. *IEEE Open J. Veh. Technol.* **2020**, *1*, 331–347. [CrossRef]
52. Liao, Y.; Shou, L.; Yu, Q.; Ai, Q.; Liu, Q. An intelligent computation demand response framework for IIoT-MEC interactive networks. *IEEE Netw. Lett.* **2020**, *2*, 154–158. [CrossRef]
53. Bebortta, S.; Senapati, D.; Panigrahi, C.R.; Pati, B. Adaptive Performance Modeling Framework for QoS-Aware Offloading in MEC-Based IIoT Systems. *IEEE Internet Things J.* **2021**, *9*, 10162–10171. [CrossRef]
54. Chen, X.; Liu, G. Energy-efficient task offloading and resource allocation via deep reinforcement learning for augmented reality in mobile edge networks. *IEEE Internet Things J.* **2021**, *8*, 10843–10856. [CrossRef]
55. Siriwardhana, Y.; Porambage, P.; Liyanage, M.; Ylianttila, M. A survey on mobile augmented reality with 5G mobile edge computing: architectures, applications, and technical aspects. *IEEE Commun. Surv. Tutorials* **2021**, *23*, 1160–1192. [CrossRef]
56. Mahmood, O.A.; Abdellah, A.R.; Muthanna, A.; Koucheryavy, A. Distributed Edge Computing for Resource Allocation in Smart Cities Based on the IoT. *Information* **2022**, *13*, 328. [CrossRef]
57. Kuang, L.; Gong, T.; OuYang, S.; Gao, H.; Deng, S. Offloading decision methods for multiple users with structured tasks in edge computing for smart cities. *Future Gener. Comput. Syst.* **2020**, *105*, 717–729. [CrossRef]
58. Sun, J. Research on resource allocation of vocal music teaching system based on mobile edge computing. *Comput. Commun.* **2020**, *160*, 342–350. [CrossRef]
59. Zhang, L.; Yuan, X.; Luo, J.; Feng, C.; Yang, G.; Zhang, N. An Adaptive Resource Allocation Approach Based on User Demand Forecasting for E-Healthcare Systems. In Proceedings of the 2022 IEEE International Conference on Communications Workshops (ICC Workshops), Seoul, Republic of Korea, 16–20 May 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 349–354.
60. Sigwele, T.; Hu, Y.F.; Ali, M.; Hou, J.; Susanto, M.; Fitriawan, H. Intelligent and energy efficient mobile smartphone gateway for healthcare smart devices based on 5G. In Proceedings of the 2018 IEEE global communications conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018; IEEE: Piscataway, NJ, USA; 2018, pp. 1–7.
61. Guo, F.; Lu, H.; Li, B.; Li, D.; Chen, C.W. NOMA-assisted multi-MEC offloading for IoVT networks. *IEEE Wirel. Commun.* **2021**, *28*, 26–33. [CrossRef]
62. Du, J.; Yu, F.R.; Lu, G.; Wang, J.; Jiang, J.; Chu, X. MEC-assisted immersive VR video streaming over terahertz wireless networks: A deep reinforcement learning approach. *IEEE Internet Things J.* **2020**, *7*, 9517–9529. [CrossRef]
63. Yang, T.; Chai, R.; Zhang, L. Latency optimization-based joint task offloading and scheduling for multi-user MEC system. In Proceedings of the 2020 29th Wireless and Optical Communications Conference (WOCC), Newark, NJ, USA, 1–2 May 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–6.
64. Sharif, Z.; Jung, L.T.; Ayaz, M. Priority-based Resource Allocation Scheme for Mobile Edge Computing. In Proceedings of the 2022 2nd International Conference on Computing and Information Technology (ICCIT), Tabuk, Saudi Arabia, 25–27 January 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 138–143.
65. Wang, C.; Qin, J.; Yang, X.; Wen, W. Energy-efficient offloading policy in D2D underlay communication integrated with MEC service. In Proceedings of the 3rd International Conference on High Performance Compilation, Computing and Communications, Xi'an China, 8–10 March 2019; pp. 159–164.
66. Khan, P.W.; Abbas, K.; Shaiba, H.; Muthanna, A.; Abuarqoub, A.; Khayyat, M. Energy efficient computation offloading mechanism in multi-server mobile edge computing—An integer linear optimization approach. *Electronics* **2020**, *9*, 1010. [CrossRef]
67. Shakarami, A.; Shakarami, H.; Ghobaei-Arani, M.; Nikougoftar, E.; Faraji-Mehmandar, M. Resource provisioning in edge/fog computing: A comprehensive and systematic review. *J. Syst. Arch.* **2022**, *122*, 102362. [CrossRef]
68. Fukushima, Y.; Iizuka, K.; Amano, H. Parallel Implementation of CNN on Multi-FPGA Cluster. In Proceedings of the 2021 IEEE 14th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC), Singapore, 20–23 December 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 77–83.
69. Ali, Z.; Khaf, S.; Abbas, Z.H.; Abbas, G.; Muhammad, F.; Kim, S. A deep learning approach for mobility-aware and energy-efficient resource allocation in MEC. *IEEE Access* **2020**, *8*, 179530–179546. [CrossRef]
70. Zhong, X.; Wang, X.; Li, L.; Yang, Y.; Qin, Y.; Yang, T.; Zhang, B.; Zhang, W. CL-ADMM: A cooperative-learning-based optimization framework for resource management in MEC. *IEEE Internet Things J.* **2020**, *8*, 8191–8209. [CrossRef]
71. Sanchez-Iborra, R.; Sanchez-Gomez, J.; Skarmeta, A. Evolving IoT networks by the confluence of MEC and LP-WAN paradigms. *Future Gener. Comput. Syst.* **2018**, *88*, 199–208. [CrossRef]
72. Lee, Y.T.; Chen, K.T.; Cheng, Y.M.; Lei, C.L. World of Warcraft avatar history dataset. In Proceedings of the Second Annual ACM conference on Multimedia Systems, San Jose CA USA, 23–25 February 2011; pp. 123–128.

73. Cong, P.; Zhou, J.; Li, L.; Cao, K.; Wei, T.; Li, K. A survey of hierarchical energy optimization for mobile edge computing: A perspective from end devices to the cloud. *ACM Comput. Surv.* **2020**, *53*, 1–44. [CrossRef]
74. Peng, K.; Huang, H.; Liu, P.; Xu, X.; Leung, V.C. Joint optimization of energy conservation and privacy preservation for intelligent task offloading in mec-enabled smart cities. *IEEE Trans. Green Commun. Netw.* **2022**, *6*, 1671–1682. [CrossRef]
75. Sacco, A.; Esposito, F.; Marchetto, G. Resource inference for sustainable and responsive task offloading in challenged edge networks. *IEEE Trans. Green Commun. Netw.* **2021**, *5*, 1114–1127. [CrossRef]
76. Elgendy, I.A.; Zhang, W.; Tian, Y.C.; Li, K. Resource allocation and computation offloading with data security for mobile edge computing. *Future Gener. Comput. Syst.* **2019**, *100*, 531–541. [CrossRef]
77. Cheng, K.; Fang, X.; Wang, X. Energy Efficient Edge Computing and Data Compression Collaboration Scheme for UAV-assisted Network. *IEEE Trans. Veh. Technol.* **2023**. [CrossRef]
78. Wang, C.; Yu, X.; Xu, L.; Wang, Z.; Wang, W. Multimodal semantic communication accelerated bidirectional caching for 6G MEC. *Future Gener. Comput. Syst.* **2023**, *140*, 225–237. [CrossRef]
79. Hossain, M.A.; Ansari, N. Hybrid multiple access for network slicing aware mobile edge computing. *IEEE Trans. Cloud Comput.* **2023**. [CrossRef]
80. Zhang, G.; Zhang, W.; Cao, Y.; Li, D.; Wang, L. Energy-delay tradeoff for dynamic offloading in mobile-edge computing system with energy harvesting devices. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4642–4655. [CrossRef]
81. Shalavi, N.; Perin, G.; Zanella, A.; Rossi, M. Energy efficient deployment and orchestration of computing resources at the network edge: A survey on algorithms, trends and open challenges. *arXiv* **2022**, arXiv:2209.14141.
82. Wu, Y.C.; Lin, C.; Quek, T.Q. A robust distributed hierarchical online learning approach for dynamic MEC networks. *IEEE J. Sel. Areas Commun.* **2021**, *40*, 641–656. [CrossRef]
83. Li, C.; Tang, J.; Zhang, Y.; Yan, X.; Luo, Y. Energy efficient computation offloading for nonorthogonal multiple access assisted mobile edge computing with energy harvesting devices. *Comput. Netw.* **2019**, *164*, 106890. [CrossRef]
84. Fu, J.; Zhu, P.; Hua, J.; Li, J.; Wen, J. Optimization of the energy efficiency in Smart Internet of Vehicles assisted by MEC. *EURASIP J. Adv. Signal Process.* **2022**, *2022*, 13. [CrossRef]
85. Xu, L.; Qin, M.; Yang, Q.; Kwak, K.S. Learning-aided dynamic access control in MEC-enabled green IoT networks: A convolutional reinforcement learning approach. *IEEE Trans. Veh. Technol.* **2021**, *71*, 2098–2109. [CrossRef]
86. Guo, K.; Gao, R.; Xia, W.; Quek, T.Q. Online learning based computation offloading in MEC systems with communication and computation dynamics. *IEEE Trans. Commun.* **2020**, *69*, 1147–1162. [CrossRef]
87. Teng, M.; Li, X.; Zhu, K. Joint Optimization of Sequential Task Offloading and Service Deployment in End-Edge-Cloud System for Energy Efficiency. *IEEE Trans. Sustain. Comput.* **2023**. [CrossRef]
88. Zheng, C.; Liu, S.; Huang, Y.; Yang, L. Hybrid policy learning for energy-latency tradeoff in MEC-assisted VR video service. *IEEE Trans. Veh. Technol.* **2021**, *70*, 9006–9021. [CrossRef]
89. Yang, S.; Liu, J.; Zhang, F.; Li, F.; Chen, X.; Fu, X. Caching-Enabled Computation Offloading in Multi-Region MEC Network via Deep Reinforcement Learning. *IEEE Internet Things J.* **2022**, *9*, 21086–21098. [CrossRef]
90. Duo, R.; Wu, C.; Yoshinaga, T.; Zhang, J.; Ji, Y. SDN-based handover scheme in cellular/IEEE 802.11 p hybrid vehicular networks. *Sensors* **2020**, *20*, 1082. [CrossRef]
91. Lin, W.; Ma, H.; Li, L.; Han, Z. Computing Assistance From the Sky: Decentralized Computation Efficiency Optimization for Air-Ground Integrated MEC Networks. *IEEE Wirel. Commun. Lett.* **2022**, *11*, 2420–2424. [CrossRef]
92. Cao, S.; Chen, S.; Chen, H.; Zhang, H.; Zhan, Z.; Zhang, W. HCOME: Research on hybrid computation offloading strategy for MEC based on DDPG. *Electronics* **2023**, *12*, 562. [CrossRef]
93. Wang, M.; Shi, S.; Zhang, D.; Wu, C.; Wang, Y. Joint Computation Offloading and Resource Allocation for MIMO-NOMA Assisted Multi-User MEC Systems. *IEEE Trans. Commun.* **2023**, *71*, 4360–4376. [CrossRef]
94. Ding, Z.; Xu, D.; Schober, R.; Poor, H.V. Hybrid NOMA offloading in multi-user MEC networks. *IEEE Trans. Wirel. Commun.* **2022**, *21*, 5377–5391. [CrossRef]
95. Alghamdi, I.; Anagnostopoulos, C.; Pezaros, D.P. Optimized Contextual Data Offloading in Mobile Edge Computing. In Proceedings of the 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM), Bordeaux, France, 18–20 May 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 473–479.