electronics

MDPI

Review

Edge Network Optimization Based on AI Techniques: A Survey

Mitra Pooyandeh and Insoo Sohn *

Division of Electronics & Electrical Engineering, Dongguk University, Seoul 04620, Korea; mitra.p@dgu.ac.kr
* Correspondence: isohn@dongguk.edu

Abstract: The network edge is becoming a new solution for reducing latency and saving bandwidth in the Internet of Things (IoT) network. The goal of the network edge is to move computation from cloud servers to the edge of the network near the IoT devices. The network edge, which needs to make smart decisions with a high level of response time, needs intelligence processing based on artificial intelligence (AI). AI is becoming a key component in many edge devices, including cars, drones, robots, and smart IoT devices. This paper describes the role of AI in a network edge. Moreover, this paper elaborates and discusses the optimization methods for an edge network based on AI techniques. Finally, the paper considers the security issue as a major concern and prospective approaches to solving this issue in an edge network.

Keywords: network edge; IoT; artificial intelligence; security

Citation: Pooyandeh, M.; Sohn, I. Edge Network Optimization Based on AI Techniques: A Survey. *Electronics* **2021**, *10*, 2830. https://doi.org/10.3390/electronics10222830

Academic Editor: George A. Tsihrintzis

Received: 27 October 2021
Accepted: 16 November 2021
Published: 18 November 2021

1. Introduction

The network edge is a new paradigm in which the storage resources and computing processes are located close to the devices. The huge amount of data that are produced by devices that are connected to the IoT network and the issue of transferring this amount of data to the cloud are causing the advent of edge networking. Obviously, moving the computing and storage close to the data-created source through the hardware or software infrastructure at the network edge can help to optimize the network specifically in terms of real-time delivery and reduce bandwidth. Optimizing the network edge in order to minimize using the infrastructure becomes more important. One of the best ways to extract information and make decisions from this huge amount of data is to empower the network edge with intelligence. In addition, integrating IoT and AI [1] at the edge permits smart services such as anomaly data detection and IoT data analysis. The AI technology at the network edge should be designed appropriately, which can perform efficiently in devices, because of these two issues: limitation of memory and computation power in the edge devices. Deploying AI at the network edge comes with challenges. One of these challenges is the heterogeneous environments created by various types of sensors used in IoT devices. Security threats are another challenge. It is observed that edge devices hold on to most of the data. Keeping data at the edge, however, provides a higher level of security. With increasing physical access come more threats, and this causes IoT devices and networks to be compromised [2]. This paper reviews the comprehensive research on the use of AI techniques for network edge optimization. The aim of this paper is to review the state-of-the-art technologies to optimize the edge network by the use of AI mechanisms. This is the first survey paper that concentrates on intelligence study on edge networks, which is in contrast to past surveys, which focused on general edge network technologies. For instance, in [3], the authors have reviewed the pros and cons of edge computing in IoT. They have categorized edge computing architectures into several groups: Front-end, Near-end, and Far-end. Furthermore, they have compared the performance of each cate-gory in terms of response time, computation capacity, and storage capacity. In [4], the authors have provided an overview of fundamental technologies supporting the AIoT, including an overview of the general architecture of the IoT, and edge computing

paradigms, along with corresponding hardware and systems. The focus of this paper is the convergence of AI and the IoT. Finally, several challenges and future directions for constructively integrating AI with IoT are presented. While there are many surveys that have discussed the general intelligence techniques used at a network edge, there is a lack of comprehensive research focusing on AI techniques that optimize the network edge intelligently. Hence, the motivation was to present wide-ranging research that takes into all aspects of intelligence optimization for the network edge. This paper is organized as follows. In Section 2, we introduce the concepts of the network. Moreover, we focus on important components that characterize the edge-based network and the edge-based IoT network. In Section 3, we introduce the most important optimization concepts that are used for AI-based network edge. In Section 4, we study the most recent research work on the use of AI optimization at the network edge. In addition, we present our comprehensive analysis of existing works in a table and discuss them in Section 5, with a brief discussion on security issues. This section is followed by Section 6, where we conclude with recommendations and future directions.

2. Network Concept

2.1. Cloud-Based Network

Today, data from social networks, governments, municipalities, airports, banks, and large insurance companies are generated and transmitted over the cloud. Data related to IoT also move and accumulate on this platform. A cloud-based network is where most of the important data of an institution are hosted in either a public cloud or a private cloud that gives access to users based on third-party server systems.

An important cloud-based network concept is the Domain Name System (DNS). It is often an integral part of a cloud-based web application. Services such as Google Cloud DNS translate user requests delivered over the Internet and link these to the suitable cloud services inserted into an application. An example of a network service is the Content Delivery Network (CDN). A public cloud provides CDN services to move data, APIs, and applications. These services can speed up mobile Internet service response times for end users located far from the virtual machines and storage [5,6]. A load balancer distributes user traffic across multiple models of your applications. With load distribution, application efficiency is increased. This reduces the volumes of data that must be moved, the consequent traffic, and the distance the data must travel, given lower latency and reduced transmission costs [7].

Important cloud-based network basic hardware are switches, routers, firewalls, storage arrays, and backup devices [8]. Virtualization software makes data storage and computing power somewhere far from the original hardware possible, and the cloud infrastructure is supported by the user interface system. Cloud storage is a data storage platform that helps users to upload files to virtual storage over the Internet without any physical storage device, and at any time, by returning to the desired server via logging on to the Internet and using files. Storage management ensures that the data are being correctly backed up; old backups are regularly removed; and if any storage component fails, the data are listed for retrieval [9]. Cloud computing is another important technology that is used in a cloud-based network. Cloud computing is a method for data computation, and it is similar to a server cluster architecture, which processes and stores massive data that is received from the users. Cloud computing techniques consist of data management, data storage, and data virtualization. Cloud computing can be defined as a virtualized resource that includes servers, storage, databases, networking, software, analytics, and intelligence, and it is available on demand [10,11].

2.2. Edge-Based Network

An edge-based network is a network located at the edge of a centralized network that brings data storage and computation as near the required point as possible, pushing applications, data, and computing power away from the centralized data center in order to

deliver low latency and save bandwidth, as shown in Figure 1. The edge infrastructure is the same as the cloud infrastructure, but it does its tasks at the edge of the network.

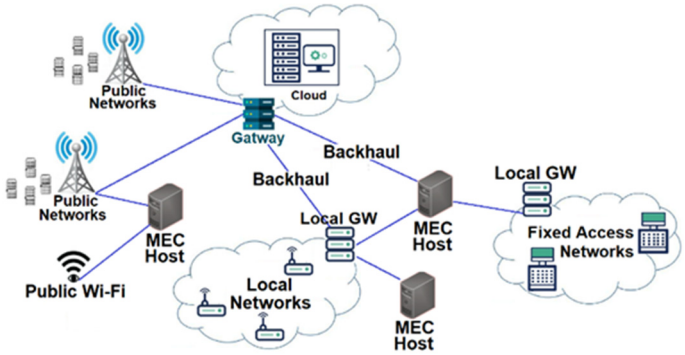

Figure 1. Mobile edge computing.

Edge computing is an edge paradigm that is defined as the part of a distributed computational topology, and in edge computing, data processing is done close to users. Edge computing places the computation resources at the network edge. Placing the data at the edge is done for real-time data communications without any delay because a delay affects the application performance. In addition, companies can reduce costs by locally processing and decreasing the amount of data that need to be sent to the cloud-based location [12]. Edge computing is an enabling technology that allows computation at the edge of the network [13].

There are two models of edge devices. The first model is named task offloading, and it can acquisition some tasks that should be managed by the user's device locally [14]. The second model is parts of the jobs that are being achieved with a cloud data center that can be done with edge devices, supporting cloud computing. For example, an edge device can monitor manufacturing equipment in a factory and another edge device is a video camera that sends the real-time image from a remote place. With a growing number of devices, the quality will decrease because of latency, but the bandwidth's cost can be massive. Edge computing solves these issues using local processing and storage. An edge gateway can process data from an edge device and then transmit only the related and real-time data back through the cloud and decrease bandwidth requirements. The edge devices are IoT sensors, a smartphone, a security camera, or a laptop. Edge gateways are defined as edge devices in an edge network system. The importance of edge computing over a private cloud can be illustrated with an example from healthcare systems. Though cloud-based distributed smart surveillance systems have the ability to aggregate and analyze video information, managing them presents a major challenge. In remote patient monitoring and elderly care, smart surveillance systems are important. Monitoring requires a robust response and real-time alerts from surveillance systems within the available bandwidth. In [15], a Cloud-based Object Tracking and Behavior Identification System (COTBIS) is introduced that employs the edge computing framework at the cloud level. A gateway is also presented. This is an emerging research area of the Internet of Things (IoT) that can enhance robustness and intelligence. Distributed video surveillance systems minimize network bandwidth and response time between wireless cameras and the cloud.

Important reasons for using an edge-based network are as follows [16]:

- There is not enough or reliable network bandwidth to send data to the cloud.
- There are security and privacy concerns about sending information over public networks or storing it in the cloud. With edge computation, data are stored locally.
- Some applications require fast data sampling or must calculate results with minimal delay.

In contrast, cloud computing may be a better option due to the following [17,18]:

- Cloud computing power is almost unlimited. Any tool can be used at any time for analysis.

- Due to environmental constraints, some applications can increase the cost of edge computing and make cloud computing more cost effective.
- The dataset may be large. The large number of applications in the cloud and the availability of other data can help applications start self-learning, which can lead to better results.
- Results may need to be widely distributed and viewed across different operating systems. The cloud space can be accessed from several points and from several devices.

However, cloud-based networks and edge-based networks can complement each other. It is observed that, for instance, vehicle controls must be safe, secure, and responsive, so this is done at the "edge", or inside the vehicle. However, a fleet monitoring program that collects performance data to plan navigation, maintenance, and routing must be implemented in the cloud, where large amounts of data received from multiple vehicles can be accessed and analyzed.

2.3. Edge-Based IoT Network

IoT systems refer to a wide range of sensors or devices that are connected and share information and data. IoT sensors generate a large amount of data that require high computation and efficient storage. Clouds tackle this problem. To send these huge amounts of data from IoT devices to the cloud that processes and shares the data, you need a lot of channel bandwidth. Due to the limited bandwidth, there is a delay in data transmission, which is a big problem in IoT systems and causes traffic problems in the network. Another important issue is the energy consumption by and battery savings for the IoT end devices. With the increasing amount of data processed in these devices, data processing consumes a large amount of energy and is another big problem in IoT systems [19].

To solve these problems of a cloud-based IoT network, edge technology is essential. Edge technology enables the information and data from IoT sensors to be collected in the best possible way and be ready for analysis. In an IoT sensor's data collection, the important point is to classify so that the types of information can be efficiently processed by edge computing. The benefits of IoT edge computing include:

- Improved performance: IoT performance is easily improved because higher volumes of data in more IoT devices can be collected quickly and without any delay.
- Greater compliance: IoT compliance with existing standards and protocols is complex. Network edge infrastructures are able to do this adaptation, making it much easier to integrate IoT devices.
- Data privacy and security: Edge technology distributes, calculates, and executes data across a wide range of devices and data centers, making it difficult to disable the entire network. A big concern about IoT edge processing devices is that they can easily become a gateway for cyberattacks using malware and other network attack methods from a weak point. Although this is a real danger, the distributed nature of the edge processing structure facilitates the implementation of security protocols that can quarantine infected parts without disabling the entire network. Because most data are processed on local devices instead of being transferred to a central database, edge processing reduces the amount of data that are compromised. It is also possible for eavesdropping while the data are in transmission, and even if a device is attacked, only locally collected information will be exposed [20].
- Reduced operating costs: Using edge technology for IoT and perform computation and processing at the edge network as shown in Figure 2, users do not require to incur extra costs for data transfer and processing in cloud service. Not only is edge technology a way to collect data for transfer to the cloud system but it will also process, analyze, and act on data collected at the edge in milliseconds because of the mitigation of the distance between IoT devices and the data processing location. There are many edge-based IoT applications, such as agricultural applications, wearables, smart homes, energy applications, healthcare applications, transportation applications, industrial

automation, surveillance cameras, and smart grids. All of these IoT applications require minimum delay processing and analyzing data.

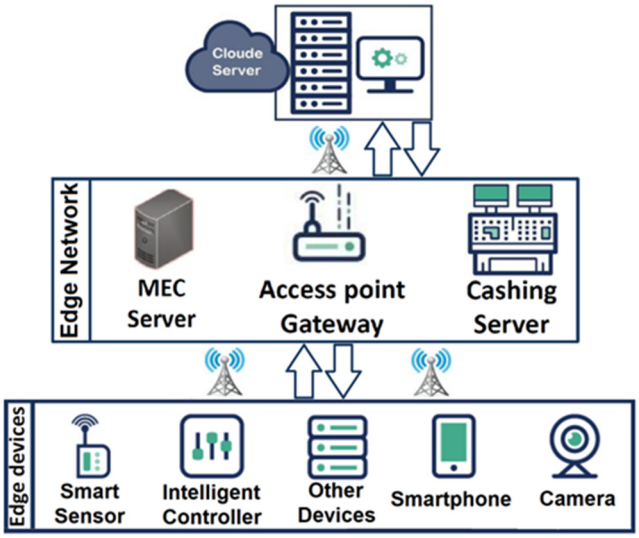

Figure 2. IoT edge network.

3. Optimization Concept

3.1. Machine Learning

Machine learning (ML) and artificial intelligence (AI) are important tools that imitate human learning using data and algorithms. These algorithms are trained within statistical methods and then make predictions or classifications. Machine learning, due to its nature, covers a range of disciplines and has a critical role in many fields, such as medicine, computer science, social media, and transportation. Machine learning algorithms adapt their algorithms with iteration. In the training process, samples of input data and desired results are prepared. After that, the algorithm can produce the desired results even though the new data have not been seen previously [21].

Machine learning algorithms are widely classified into supervised algorithms, unsupervised algorithms, semi-supervised algorithms, and reinforcement algorithms. Supervised learning models require data to make the algorithm and are tested on labeled data sets, but unsupervised machine learning algorithms do not need training data. On the contrary, they use deep learning through sets of unlabeled training data. Unsupervised learning models do not receive information about data features to examine. Reinforcement learning algorithms learn through trial and error. The model for receiving a selected goal and getting maximum reward uses the limited information and learns from its previous moves [22]. In turn, each one of the algorithms mentioned is divided into a subset of algorithms based on the performance, as shown in Figure 3. This diagram depicts the taxonomy of artificial intelligence techniques based on machine learning algorithms. This taxonomy was compiled by many different types of categorized ML algorithms and is intended to include all of the current algorithms. The main machine learning algorithms described in the diagram are supervised algorithms, unsupervised algorithms, semi-supervised algorithms, and reinforcement algorithms.

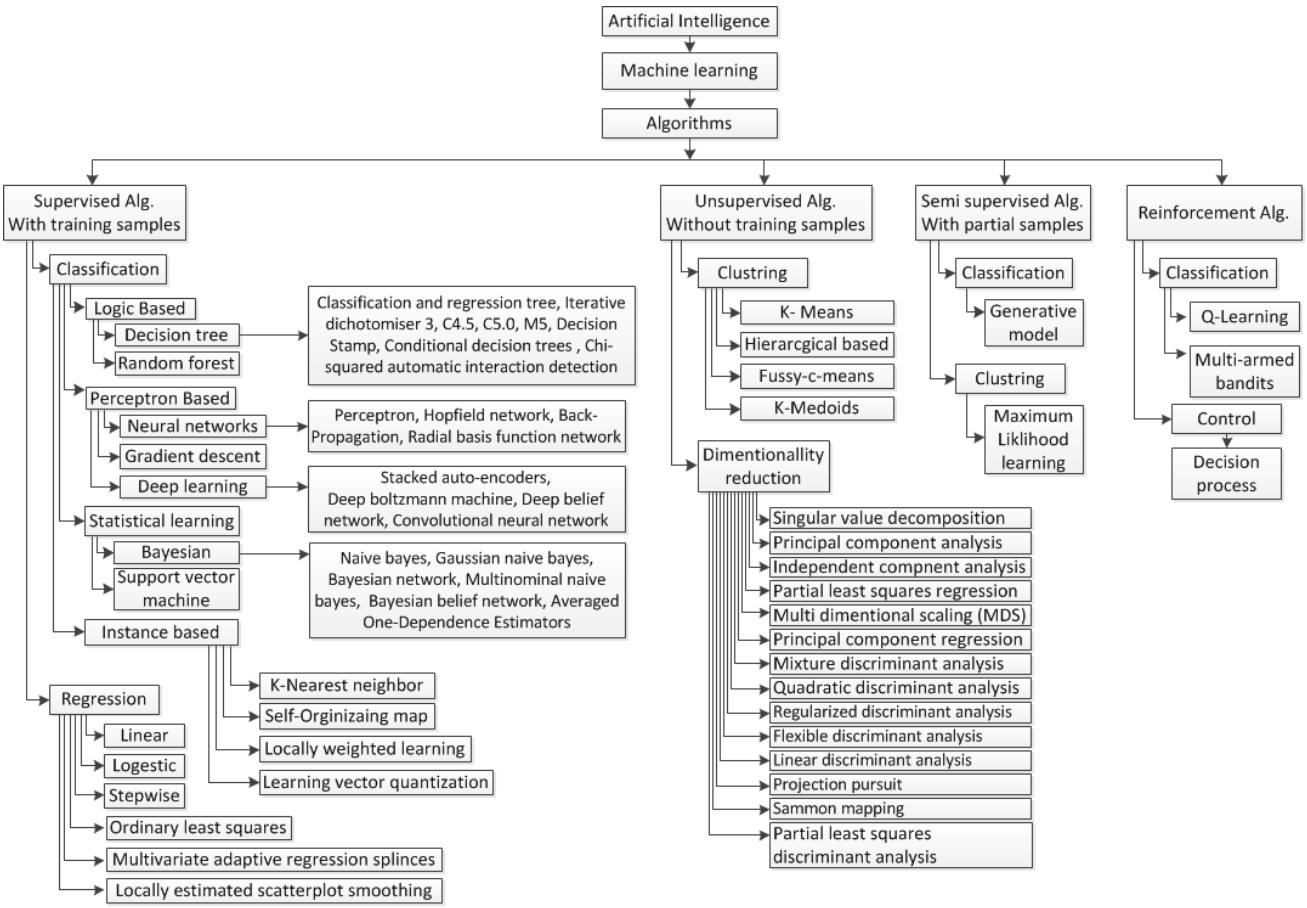

Figure 3. Machine learning algorithms.

3.2. Artificial Neural Network

An artificial neural network (ANN) is a machine learning technique that contains one or more hidden layers between input and output layers. An ANN consists of some neurons that are interconnected so that it can be defined as a graph. In this graph, each neuron has a transfer function, as follows:

$$y_i = f_i(\sum_{j=1}^{n} w_{ij}x_i - \theta_i) \quad (1)$$

where y_i is output for node i, x_j is the input number j, w_{ij} is the weight between output and input nodes, and θ_i is the threshold [23].

The data are processed through hidden layers and then transferred to the next layer with connections named weights. In fact, ANNs are an optimal method for classifying, clustering, and pattern recognition. ANNs mimic how the human brain processes information and decision. It should be noted that an ANN should be trained before deployment. This means that the human information about the desired results must be compared with the results that a machine gets. Finally, if these results do not match, the machine uses backpropagation algorithm to modify the weight. There are several types of ANNs, as follows: based on the connection pattern, such as the feedforward and recurrent; based on the number of hidden layers, such as single layer and multi-layer; based on the weights, such as fixed and adaptive; and based on the memory, such as static and dynamic.

Moreover, there are varying architecture types of ANNs, as shown in Figure 4. These types of ANNs are categorized based on different layers and weight structures. Artificial neural networks are applied to a wide spectrum of applications, such as process modeling and control, machine diagnostics, medical diagnosis, voice recognition, and fraud detection.

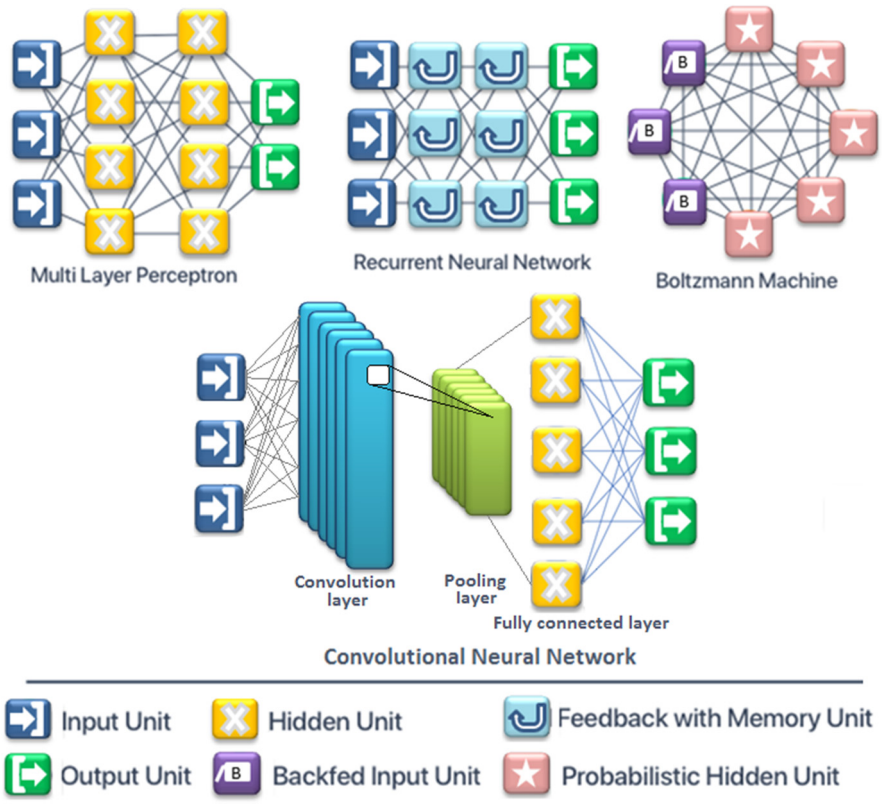

Figure 4. Architecture types of ANNs.

3.3. Federated Learning

Federated learning (FL) is a decentralized form of machine learning. Generally, in IoT networks, the data that are gathered from edge devices, such as mobile phones and laptops, are sent to a centralized server in the cloud or at the edge. Machine learning approaches require training data in a centralized way, while federated learning makes the devices learn a model collaboratively and require the training data to be kept on the device. Under this assumption, the device after downloading the current model learns from the data on the device and improves the model, with updates [24]. This update is sent to the cloud with encryption. In this way, all of the data remain on the device and updates are not stored in the cloud. This improves data security. In addition, given the user's demands, the communication time between the central cloud and the user device should be as short as possible. Therefore, to overcome this, it is better to place the model in the end-user device [25].

As a result, the advantages of FL are data security, heterogeneous data usage, real-time continuously learning, and light hardware. Federated machine learning is categorized as federated supervised learning, federated unsupervised learning, and federated semi-supervised learning. Potential use cases of FL are mobiles phone, healthcare, autonomous car, and retail.

3.4. Reinforcement Leaning

Reinforcement learning (RL) involves training machine learning models to make a series of decisions. Agents will find ways to accomplish their goals in uncertain, potentially complex environments. Artificial intelligence faces a game-like scenario in reinforcement learning. RL uses trial and error to resolve the problem. For the AI to perform what the programmer wants, it receives rewards or penalties for the actions it takes. The model must figure out how to maximize the reward, beginning with totally random trials and finishing with sophisticated tactics and superhuman abilities [26]. A potential application of reinforcement learning is to train the models that control autonomous cars. Ideally, the

computer should not get any instructions on driving the car and it would only require the reward function.

4. Artificial Intelligence at the Network Edge

One of the interesting features of AI is that it can be empowered to make smart decisions. AI is becoming an important component of the network edge to enable edge intelligence to reduce latency and enhance real-time analytic, scalability, information security and privacy, and automated decision-making. Thus, AI can optimize the network edge with technologies such as deep learning, reinforcement learning, and federated learning. IoT devices generate the main dataset, which is stored in the cloud database. These datasets are used for AI model training. An AI model in the cloud needs to be retrained and updated with newly generated data from IoT devices. When the AI model in the cloud is stable enough, then it is sent to the edge network to enable edge intelligence, as shown in Figure 5.

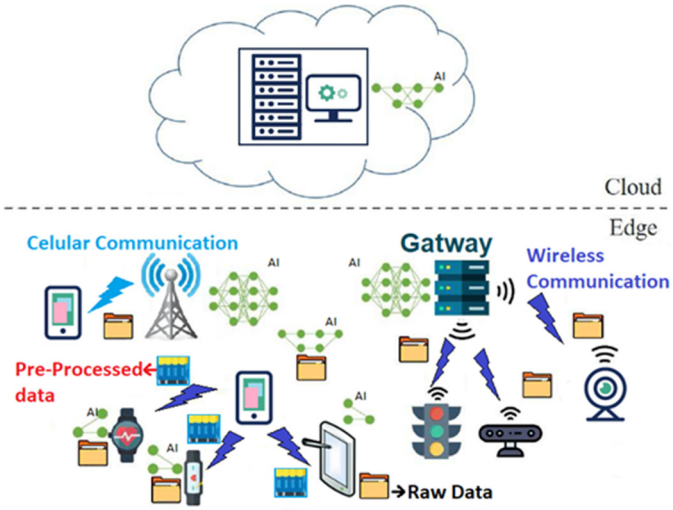

Figure 5. Intelligence network edge.

Note that transferring huge amounts of data gathered from IoT devices to the cloud and then returning the data after processing is costly and time consuming. Owing to the interesting features of AI, employing AI at the network edge can help to solving problems such as increased power consumption, latency, and security issues. AI uses machine learning algorithms. These algorithms assist AI in optimizing the network edge. In this section, we introduced some techniques that use these algorithms. Furthermore, we made a comparison between several research articles that used the AI techniques for optimizing the network edge.

Federated learning is a distributed machine learning technique. Federated learning helps to train the model in devices that collect data, through computational devices connected to a decentralized system. Fast edge learning is difficult due to latency in communications. In [27], to address this issue, the authors designed a low-latency multi-access method. This method is named federated edge learning (FEEL). A federated neural-network-based AI model is used at the edge-server by averaging local models trained at the edge devices. Broadband analog aggregation (BAA) is also used in this work to reduce communications latency. The reduced latency is between 10× to 1000×. Then, a comparison between BAA and OFDMA is done and it is shown that the scale and the device crowd have a linear relation. As mentioned previously, most current optimization-based methods need a self-adaptive ability in a dynamic environment. To address these challenges, the authors used learning-based algorithms in a centralized way. In this paper, as we see, the latency reduction ratio of the BAA algorithm is proportional to the number of devices.

In [28], another federated learning-based network edge is proposed called federated delayed averaging (FedDelAvg). FL-based optimization techniques consist of three basic steps:

- In each device, using the local data set, the local model training is repeated several times.
- As a global model, all local models are integrated into one server.
- All models in the devices are synchronized with this universal model.

This paper investigated the communication delays between the edge devices and the aggregators. The authors studied the effect of federated learning-based delay optimization in various models. They also examined the effect of delay on the convergence of FL with simulations using the TensorFlow federated (TFF) framework. Furthermore, the result of the experiments shows on comparing the FedAvg algorithm and the FedDelAvg algorithm, FedDelAvg achieved better improvement, attaining an accuracy of 80%. If FedDelAvg is tuned, the convergence rate is the same under delay or without delay.

Edge caching is a technology for addressing huge content access in mobile networks. In [29], two methods are proposed on cooperative caching to predict users' content demand in a mobile edge caching network using deep learning (DL) technology. In the first method, a content server collects data from mobile edge nodes in the network. Then, the DL algorithm is made to predict the network content demand. The second method is a distributed deep learning (DDL)-based algorithm. The mobile edge node uses the DDL to decrease the error in the content demand prediction. In simulation results, the authors showed that through this method, compared with other algorithms in machine learning, the accuracy increased by decreasing the root-mean-squared error (RMSE) by up to 33.7% and the service delay decreased to 47.4%. In the DDL-based algorithm that performs at the edge, only the gradient information of the users is used in the DL for safe user demand prediction. A performance comparison between MFA, SVD, NMF, SLL, DL, and DDL methods showed that the learning-based methods have a better cache hit rate and less delay.

In [30], to improve edge caching performance, big data analytics is investigated. The authors discussed learning-based methods for network edge caching with big data. Then they categorized machine learning schemes for edge caching as Classification and Regression Analysis, Clustering, Reinforcement Learning, Transfer Learning, Deep Learning Approach, Similarity Learning Approach. They made two experimental studies. In one of these studies, unsupervised learning, deep learning, and optimization algorithms incorporated edge caching and the result was mitigation of the computation time. In the second study, using the similarity, learning for design D2D caching was analyzed to evaluate user satisfaction. The result was the enhancement of satisfaction performance. Caching at the edge of the wireless network is important to increase network power, improve energy efficiency (EE), decrease service latency, and reduce the traffic load of the cellular backhaul. In conclusion, the authors showed that it is possible to analyze big data at the edge of the network to improve the edge storage capability.

Mobile edge caching (MEC) is used to mitigate the traffic in the network and increase the quality of experience (QoE) of mobile users. However, MEC has challenges described in [31] that are solved with the use of the deep Q-network algorithm for mobile edge caching. The authors classified the learning-based mobile edge caching method into the popularity-prediction-based approach and the reinforcement-learning-based approach. They used MovieLens data to give proof of the performance of the A3C-based mobile edge caching policy. They used least frequently used (LFU), least recently used (LRU), and First in First Out (FIFO) for evaluating the common baseline caching policies for assessing the efficiency of the policy on cache hit ratio and offloaded traffic. The results showed that the introduced policy performs better than others do while the training of the neural network is effectively guaranteed. Finally, they proved that caching using DRL can provide a dynamic environment. The dynamic environment adaptation problem, the multiple balance problem, and the integrity of deep models into the DRL-based mobile edge caching problem were solved, as shown in the paper.

Mobile edge computing can improve the computational capacity at the edge of the network by leveraging the computation tasks to the MEC server. In [32], the authors adjusted the sum cost of delay and energy consumptions. They optimized the offloading decision and computational resource allocation. They used the Q-learning and deep reinforcement learning (DRL) algorithms in a multi-layer user. For simulation, they used four algorithms and compared the results. These four algorithms are Q learning, deep Q learning (DQN), full local, and full offloading. Full local means that all user equipment (UE) executes its tasks by local computing, and full offloading means that all UE offloads its tasks to the MEC server. With a high number of UE, the sum cost of all algorithms increases and DQN achieves the best result. We can conclude that between the DQN and Q-learning algorithms for a small number of UE, there were small differences in performance.

In [33], the authors introduced a multiple algorithm service model (MASM). This model uses a heterogeneous algorithm for optimizing the energy consumed and the delay in computation offloading. The authors suggested a tide ebb algorithm (TEA) for the MASM model. They optimized the workload assignment weights (WAWs) joint computing talent of virtual machines (VMs). Moreover, their approach had an effect on the quality of the results (QoRs). A comparison of the MASM and single algorithm service model (SASM) algorithms showed that the MASM significantly reduces the energy consumption and delay.

There are many technologies created with a combination of AI algorithms for optimization development. In [34], some optimization methods were used to face unsure input, dynamic status, and temporal isolation. The authors studied the mobile edge computing optimization, the caching, and the communication by combining the deep reinforcement learning and federated learning algorithms. By mobile content caching techniques, the important information is cached in the middleboxes, gateways, routers, or intermediate servers, which causes the omission of duplicate transmissions from the cloud and, hence, reduces traffic. They denoted F = {1, . . . , F} as a library of popular content files and (pf)×1 as the content popularity. The DRL at the edge is used for smart cache decision. They used the Markov Decision Process (MDP) to model the cache substitution problem. As shown in experiment results, the authors concluded that the edge caching efficiency and the computation offloading performance is greatly improved compared to the conventional networks. It can be seen that in the performance comparison of all methods, the FL algorithm at the edge has the best performance and with the integrity of FL-based double deep Q learning (DDQN), the efficiency improves drastically.

In [35], the authors proposed a federated deep-reinforcement-learning-based cooperative edge caching (FADE) framework. The main problem in most optimization methods is incompatibility with the dynamic environment, and to tackle this problem, the DQN is proposed. The FADE algorithm enables the base station to learn the predictive model in the first round of training and in the next round of global training, the FADE uploads the local parameters to the BS. Simulation results displayed that the FADE algorithm reduces loss in performance by 92% and leads to 60% improvement in the system payment over the DRL and compared to LRU, LFU, and FIFO, achieves 7%, 11%, and 9% improvement, respectively. The results of the practical experiments show that this algorithm solves the problem of offloading duplicated traffic and mitigates the delay. Furthermore, by using a decentralized scheme, the FADE algorithm also solves the compatibility problem, which is a big problem in other edge caching methods.

In [36], DRL and federated learning is combined in the IoT environment. Federated learning is used to train DRL operators in a distributed model to decrease transmission costs between node and devices. In addition, FL is used to coordinate the training process among multiple IoT devices and also for privacy preservation. The authors used two models of architecture: the static system model and the dynamic system model. They used DDQN. As the results showed, the standard deviation of centralized training is smaller than that of the FL-based DRL training. The IoT devices selected for the experiment use the same environment as the ones used with centralized DRL training. As we can see in the

results, there are no big differences between the centralized training DRL and IoT devices by increasing the training period for computation offloading performance.

In [37], the authors proposed a new scheme for computation and communication heterogeneity determination. This scheme minimizes the total cost (energy consumption, overhead, and delay). Additionally, it is used in resource allocation for FL systems such as transmission power or CPU frequency. The authors illustrated a time-sharing scheduling method that analyzed the method of choosing the number of participants (K) and the number of local iterations (E), which are substantial control parameters in FL for minimizing the total cost. Their algorithm is a control algorithm based on sampling. Ultimately, this method reduces the learning time and the energy consumption.

In [38], the authors proposed FL algorithms for optimizing the edge network. They aimed to optimize the system-wide cost efficiency of FL using Lyapunov optimization theory. They designed a cost-efficient optimization framework (CEFL) to make online control decisions. This technique reduces the cost and the queue congestion. Moreover, it encourages on-demand privacy preservation.

In [39], users' computation offloading problem with mobile edge computing for optimizing the computation offloading decision making policy is addressed. Since wireless networks and computing have probabilistic properties with dynamic environment, the authors used the model-free reinforcement learning (RL) framework to tackle the computation offloading problem. The authors used the combination of RL method Q-learning with the deep neural network (DNN). For the network model, they used one macrocell and N small cells with LTE standards. They used the Markov decision process (MDP) to solve the consecutive decision problem. Face recognition was chosen for the computation task. The result of the experiment shows that the efficiency of the proposed decision-making algorithm in terms of the total overhead of mobile users is better than that of other algorithms, with a reduction in latency and energy consumption and computation performance improvement.

In [40], the authors proposed an alternative direction method of multipliers (ADMM) optimizer in order to reduce the dimensions of the Kernel matrix using the Nystrom technique. This ADMM algorithm reduces the dimensions by 32 times with a 2% decrease in accuracy. This proposed algorithm, compared with the conventional sequential minimal optimization (SMO) algorithm, achieves 9.8×10^{-7} shorter latency, 62% reduction in computational complexity, 60% reduction in memory size, and 153.310 times higher energy efficiency. As a result, this algorithm results in low energy consumption and latency for edge.

One of the famous SVM uses is image classification at the network edge as the SVM is combined with the convolutional neural network (CNN) for achieving this goal. The CNN is similar to the traditional NN in its initial structure, but it has one or more convolution layers that help the network by reducing parameters, which leads to less overfitting, making the model less complex. The CNN with the SVM is used in image classification to achieve a test accuracy of more than 99.04% [41,42]. In [42], the authors proposed AI at the edge, which is used for object detection and tracking. They used the convolutional neural network (CNN) for surveillance systems. Their algorithm is known as the lightweight CNN (L-CNN) algorithm. In this paper are deployed two models of human object detection, the Haar cascade and HOG, for feature extraction and the support vector machine (SVM) for classification. The SVM is a supervised ML algorithm that classifies the data. The SVM in a high dimensional space separates clearly two or more groups of data with a hyperplane. We can see in the results that the Haar cascade is the best algorithm in terms of speed and resource efficiency and the proposed L-CNN is the second after the Haar cascade. However, from the average false positive rate (FPR), the L-CCN algorithm has the best performance (6.6%) and the false negative rate (FNR) is 18.1%, which is better than that of the Haar cascade (26.3%). Moreover, the accuracy of the L-CNN is (5.3%), which is comparable with that of SSD GoogleNet (15.6%). The L-CNN is 64% faster than MobileNet, and it uses less memory.

5. Discussion

Increase in the use of the network edge has resulted in some new open issues, such as control and management of data, distributed computing, and security. The development of optimization approaches can help to solve these issues. According to what we have stated in Section 4, there are two categories of AI techniques that are used in the edge network, as follows: techniques based on FL and techniques based on the DNN. The edge network based on FL is trained with private data optimization. The main disadvantage of FL techniques are restrictions that arise due to the heterogeneity of IoT devices, with different energy requirements and computation capabilities. The edge networks based on the DNN are capable of adapting to varying service demands. However, DNN-based systems have shown to be sensitive to the quality of training data, which can result in poor performance.

Table 1 summarizes ANN-based optimization technologies reviewed in Section 4. The table contains 15 algorithms along with their four important components: the network structure, metrics, the dataset, and some details. The network structures applied for these AI algorithms are edge network, MEC, IoT, and edge device. The metrics used are energy efficiency, delay, cache hit rate, accuracy, overhead, QoS, QoE, convergence, and so on.

Table 1. Summary AI-Enabled Edge Network Optimization Technology.

Algorithm	Network Structure	Metrics	Dataset	Details	Ref.
FL	Edge network	Delay	MNIST	The accuracy of BAA and OFDMA schemes is the same.	[27]
FL	Edge network	Delay	NIST	If $\alpha \varepsilon (0,1]$, then the convergence rate is the same in FedDelAvg and FedAvg.	[28]
DNN	Mobile edge network	Delay, cache hit rate	MovieLens 1M-dataset	Using DL raises concerns on privacy.	[29]
DNN	Mobile edge network	User satisfaction, storage capability, energy efficiency, delay	Alexanderplatz	With increasing training set size from 500 to 5000, the prediction accuracy increases by 40%+.	[30]
DQN	Mobile edge caching	QoE, cache hit rate	MovieLens	Improvement percentage on the offloaded traffic in the A3C method~2 × baseline method improvement percentage on the offloaded traffic.	[31]
QL and DQN	Mobile edge computing	Energy efficiency, delay	N/A	With more than 8 GHz capacity of the MEC server, the sum cost of all algorithms except the full offload is a fixed number.	[32]
Tide ebb	Edge network	Energy efficiency, overhead, QoS, delay	N/A	This algorithm optimizes the workload assignment weights (WAWs) in addition to the joint computing talent of virtual machines.	[33]
DQN and DDQN	Mobile edge computing, caching	Cache hit rate, utility	N/A	FedAvg is used to solve the Non-IDD and unbalancing problem in FL.	[34]
DDQN	IoT	Cache hit rate, average delay, convergence	MSN dataset from Xender	FADE has low efficiency in reducing delay.	[35]
DDQN	IoT	Training complexity	N/A	-	[36]

Table 1. *Cont.*

Algorithm	Network Structure	Metrics	Dataset	Details	Ref.
FL	Mobile edge network	Energy efficiency, overhead, accuracy, delay	MNIST, EMNIST	The algorithm is a control algorithm based on sampling.	[37]
DNN	IoT	Energy efficiency, overhead, queue congestion, accuracy, delay	Cifar-10, MFLOPS	Lyapunov optimization is used.	[38]
QL and DNN	Mobile edge computing	Energy efficiency, overhead, delay	N/A	λ m(t) is important to a user sensitive to delay; λ m(e) is important to a user in a low-battery state.	[39]
SVM	Edge devices	Accuracy, energy efficiency, delay	CHB-MIT Scalp Epilepsy, Wisconsin Breast Cancer	The dimension of the kernel matrix is reduced by the Nystrom method.	[40]
SVM and CNN	Edge network	Energy efficiency, overhead, accuracy, delay	VOC12, VOC07	The proposed L-CNN algorithm enforces security and privacy policies.	[42]

In Table 1, we can observe that the purpose of most technologies is to increase the accuracy, reduce the delay, improve the energy consumption, improve the cache hit rate, and improve the QoS/QoE. Since the nature of the wireless networks is dynamic, they cannot respond perfectly to these complex network settings. The traditional NN cannot respond to such an environment. With the advent of the DNN algorithms, this problem is being actively solved. Therefore, as shown in Table 1, most of the recent AI techniques have been based on the deep learning algorithm. However, the AI-based edge network faces many challenges, such as distributed computation load problem and security problem.

On top of that, the usage of the network edge in IoT systems raises the importance of security at the network edge because of increasing vulnerable access points. Moreover, implementing AI and ML at the network edge inherently creates complex connections between the network edge and the Internet to empower updates to AI models. IoT developers should invest in protecting the intelligence modules in the edge network against internal and external attacks. Owing to the lack of centralized storage, an AI-enabled edge preserves data privacy. However, there are many cases where attackers can disable each edge node individually.

Depending on the hardware types, we can choose appropriate techniques of protecting the AI module at the network edge. According to this, we should consider the following: the memory for encrypting the model, the level of model contribution with other applications, and the hardware that the application uses for execution [43]. For instance, some of the approaches are as follows: The first approach is encrypting a version of the model in nonvolatile memory. In this approach, when we need the model, it is authenticated. The second approach is isolating some of the sensitive parts of the model in memory. This approach does not allow the model to be exposed to a non-secure application environment. The application is authenticated and then decrypted and run in a private environment using virtualization, resulting in the AI model being safe.

6. Conclusions

In this paper, we studied the use of AI at the edge network with the network concept's introduction and a survey of most recent research works on AI-based edge. In addition, we presented the AI techniques studied in the surveyed papers to optimize the edge network in detail. Compared to conventional edge-based IoT, security issues in AI-based edge IoT are more critical. With an increase in the use of AI at the network edge, the security problem requires more attention. This is because each device that interacts with the network becomes an attack gate. AI is developed with massive amounts of data, and the security of these data inherently affects the performance of the AI. AI is based on

two stages: training and inference. The data used in the training stage need to be secure due to personal identification information at the network edge. The attackers can steal these data. In addition, AI may be faced with data injection with malicious intent [44]. As data are exchanged between AI inside the network edge and IoT devices, the security and authentication of data are more important at the AI-enabled network edge. According to Section 4, FL optimization approaches aid in privacy preservation by developing a shared model across users without direct access to the data. Unfortunately, there is almost no research in defense of AI at the network edge, but one promising method against attacks targeting neural networks is using complex network theory for the optimization of the neural network topology [45]. Another important work on the security of the AI-based network edge is [46]. In this, the edge network defense strategy with attack model estimation capability was studied to solve the attack challenge in a heterogeneous dynamic environment. The authors proposed the use of the DQN-based offloading and hotbooting technique for solving this problem. It is essential that we understand the potential security issues surrounding edge-based IoT devices and ensure that the system is secure.

Author Contributions: Conceptualization, M.P. and I.S.; investigation, M.P. and I.S.; writing—original draft preparation, M.P.; writing—review and editing, M.P. and I.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRFK) funded by the Ministry of Education (2018R1D1A1B07041981).

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

References

1. Debauche, O.; Mahmoudi, S.; Mahmoudi, S.A.; Manneback, P.; Lebeau, F. A new edge architecture for ai-iot services deployment. *Procedia Comput. Sci.* **2020**, *175*, 10–19. [CrossRef]
2. Murshed, M.G.; Murphy, C.; Hou, D.; Khan, N.; Ananthanarayanan, G.; Hussain, F. Machine learning at the network edge: A survey. *arXiv* **2019**, arXiv:1908.00080. [CrossRef]
3. Yu, W.; Liang, F.; He, X.; Hatcher, W.G.; Lu, C.; Lin, J.; Yang, X. A survey on the edge computing for the Internet of Things. *IEEE Access* **2017**, *6*, 6900–6919. [CrossRef]
4. Chang, Z.; Liu, S.; Xiong, X.; Cai, Z.; Tu, G. A Survey of Recent Advances in Edge-Computing-Powered Artificial Intelligence of Things. *IEEE Internet Things J.* **2021**, *8*, 13849–13875. [CrossRef]
5. Ling, L.; Xiaozhen, M.; Yulan, H. CDN cloud: A novel scheme for combining CDN and cloud computing. In Proceedings of the 2nd International Conference on Measurement, Information and Control, Harbin, China, 16–18 August 2013; pp. 16–18.
6. Lin, C.F.; Leu, M.C.; Chang, C.W.; Yuan, S.M. The study and methods for cloud based CDN. In Proceedings of the International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery, Beijing, China, 10–12 October 2011; pp. 469–475.
7. Rahman, M.; Iqbal, S.; Gao, J. Load balancer as a service in cloud computing. In Proceedings of the IEEE 8th International Symposium on Service Oriented System Engineering, Oxford, UK, 7–11 April 2014.
8. Feng, T.; Bi, J.; Hu, H.; Cao, H. Networking as a service: A cloud-based network architecture. *J. Netw.* **2011**, *6*, 1084. [CrossRef]
9. Wu, J.; Ping, L.; Ge, X.; Wang, Y.; Fu, J. Cloud storage as the infrastructure of cloud computing. In Proceedings of the 2010 International Conference on Intelligent Computing and Cognitive Informatics, Kuala Lumpur, Malaysia, 22–23 June 2010; pp. 380–383.
10. Lu, G.; Zeng, W.H. Cloud computing survey. In *Applied Mechanics and Materials*; Trans Tech Publications Ltd.: Bäch, Switzerland, 2014; Volume 530, pp. 650–661.
11. Moghe, U.; Lakkadwala, P.; Mishra, D.K. Cloud computing: Survey of different utilization techniques. In Proceedings of the 2012 CSI Sixth International Conference on Software Engineering (CONSEG), Madhay Pradesh, India, 5–7 September 2012; pp. 1–4.
12. Zhao, Y.; Wang, W.; Li, Y.; Meixner, C.C.; Tornatore, M.; Zhang, J. Edge computing and networking: A survey on infrastructures and applications. *IEEE Access* **2019**, *7*, 101213–101230. [CrossRef]
13. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge computing: Vision and challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. [CrossRef]
14. Sonmez, C.; Ozgovde, A.; Ersoy, C. Edgecloudsim: An environment for performance evaluation of edge computing systems. *Trans. Emerg. Telecommun. Technol.* **2018**, *29*, e3493. [CrossRef]
15. Rajavel, R.; Ravichandran, S.K.; Harimoorthy, K.; Nagappan, P.; Gobichettipalayam, K.R. IoT-based smart healthcare video surveillance system using edge computing. *J. Ambient. Intell. Humaniz. Comput.* **2021**, 1–3. Available online: https://link.springer.com/article/10.1007/s12652-021-03157-1 (accessed on 16 November 2021). [CrossRef]
16. Dillon, T.; Wu, C.; Chang, E. Cloud computing: Issues and challenges. In Proceedings of the 2010 24th IEEE International Conference on Advanced Information Networking and Applications, Perth, Australia, 20–23 April 2010; pp. 27–33.

17. Avram, M.G. Advantages and challenges of adopting cloud computing from an enterprise perspective. *Procedia Technol.* **2014**, *12*, 529–534. [CrossRef]
18. Čolaković, M.; Hadžialić, M. Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues. *Comput. Netw.* **2018**, *144*, 17–39. [CrossRef]
19. Stuurman, K.; Kamara, I. IoT Standardization-The Approach in the Field of Data Protection as a Model for Ensuring Compliance of IoT Applications? In Proceedings of the IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW), Vienna, Austria, 20–24 August 2016; pp. 22–24.
20. Sha, K.; Yang, T.A.; Wei, W.; Davari, S. A survey of edge computing-based designs for iot security. *Digit. Commun. Netw.* **2020**, *6*, 195–202. [CrossRef]
21. El Naqa, I.; Murphy, M.J. What is machine learning? In *Machine Learning in Radiation Oncology*; Springer: Cham, Switzerland, 2015; pp. 3–11.
22. Amutha, J.; Sharma, S.; Sharma, S.K. Strategies based on various aspects of clustering in wireless sensor networks using classical, optimization and machine learning techniques: Review, taxonomy, research findings, challenges and future directions. *Comput. Sci. Rev.* **2021**, *40*, 100376. [CrossRef]
23. Yao, X. Evolving artificial neural networks. *Proc. IEEE* **1999**, *87*, 1423–1447.
24. Yang, Q.; Liu, Y.; Cheng, Y.; Kang, Y.; Chen, T.; Yu, H. Federated learning. *Synth. Lect. Artif. Intell. Mach. Learn.* **2019**, *13*, 1–207. [CrossRef]
25. Hu, K.; Li, Y.; Xia, M.; Wu, J.; Lu, M.; Zhang, S.; Weng, L. Federated Learning: A Distributed Shared Machine Learning Method. *Complexity* **2021**, 8261663. [CrossRef]
26. Zhu, G.; Wang, Y.; Huang, K. Broadband analog aggregation for low-latency federated edge learning. *IEEE Trans. Wirel. Commun.* **2019**, *19*, 491–506. [CrossRef]
27. Lin, F.P.C.; Brinton, C.G.; Michelusi, N. Federated Learning with Communication Delay in Edge Networks. *arXiv* **2020**, arXiv:2008.09323.
28. Saputra, Y.M.; Hoang, D.T.; Nguyen, D.N.; Dutkiewicz, E.; Niyato, D.; Kim, D.I. Distributed deep learning at the edge: A novel proactive and cooperative caching framework for mobile edge networks. *IEEE Wirel. Commun. Lett.* **2019**, *8*, 1220–1223. [CrossRef]
29. Chang, Z.; Lei, L.; Zhou, Z.; Mao, S.; Ristaniemi, T. Learn to cache: Machine learning for network edge caching in the big data era. *IEEE Wirel. Commun.* **2018**, *25*, 28–35. [CrossRef]
30. Zhu, H.; Cao, Y.; Wang, W.; Jiang, T.; Jin, S. Deep reinforcement learning for mobile edge caching: Review, new features, and open issues. *IEEE Netw.* **2018**, *32*, 50–57. [CrossRef]
31. Sutton, R.S.; Barto, A.G. Reinforcement Learning: An Introduction. *Robotica* **1999**, *17*, 229–235. [CrossRef]
32. Li, J.; Gao, H.; Lv, T.; Lu, Y. Deep reinforcement learning based computation offloading and resource allocation for MEC. In Proceedings of the 2018 IEEE Wireless Communications and Networking Conference (WCNC), Barcelona, Spain, 15–18 April 2018; pp. 1–6.
33. Zhang, W.; Zhang, Z.; Zeadally, S.; Chao, H.C.; Leung, V.C. MASM: A multiple-algorithm service model for energy-delay optimization in edge artificial intelligence. *IEEE Trans. Ind. Inform.* **2019**, *15*, 4216–4224. [CrossRef]
34. Wang, X.; Han, Y.; Wang, C.; Zhao, Q.; Chen, X.; Chen, M. In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning. *IEEE Netw.* **2019**, *33*, 156–165. [CrossRef]
35. Wang, X.; Wang, C.; Li, X.; Leung, V.C.M.; Taleb, T. Federated deep reinforcement learning for internet of things with decentralized cooperative edge caching. *IEEE Internet Things J.* **2020**, *7*, 9441–9455. [CrossRef]
36. Ren, J.; Wang, H.; Hou, T.; Zheng, S.; Tang, C. Federated learning-based computation offloading optimization in edge computing-supported internet of things. *IEEE Access* **2019**, *7*, 69194–69201. [CrossRef]
37. Luo, B.; Li, X.; Wang, S.; Huang, J.; Tassiulas, L. Cost-Effective Federated Learning in Mobile Edge Networks. *arXiv* **2021**, arXiv:2109.05411. [CrossRef]
38. Zhou, Z.; Yang, S.; Pu, L.; Yu, S. CEFL: Online admission control, data scheduling, and accuracy tuning for cost-efficient federated learning across edge nodes. *IEEE Internet Things J.* **2020**, *7*, 9341–9356. [CrossRef]
39. Wei, Y.; Wang, Z.; Guo, D.; Yu, F.R. Deep Q-learning based computation offloading strategy for mobile edge computing. *Comput. Mater. Contin.* **2019**, *59*, 89–104. [CrossRef]
40. Huang, S.A.; Yang, C.H. A hardware-efficient ADMM-based SVM training algorithm for edge computing. *arXiv* **2019**, arXiv:1907.09916.
41. Agarap, A.F. An architecture combining convolutional neural network (CNN) and support vector machine (SVM) for image classification. *arXiv* **2017**, arXiv:1712.03541.
42. Nikouei, S.Y.; Chen, Y.; Song, S.; Xu, R.; Choi, B.Y.; Faughnan, T. Smart surveillance as an edge network service: From harr-cascade, svm to a lightweight cnn. In Proceedings of the 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC), Philadelphia, PA, USA, 18–20 October 2018; pp. 256–265.
43. Gong, C.; Lin, F.; Gong, X.; Lu, Y. Intelligent cooperative edge computing in internet of things. *IEEE Internet Things J.* **2020**, *7*, 9372–9382. [CrossRef]
44. Zhou, C.; Liu, Q.; Zeng, R. Novel defense schemes for artificial intelligence deployed in edge computing environment. *Wirel. Commun. Mob. Comput.* **2020**, 8832697. [CrossRef]

45. Xiao, L.; Wan, X.; Dai, C.; Du, X.; Chen, X.; Guizani, M. Security in mobile edge caching with reinforcement learning. *IEEE Wirel. Commun.* **2018**, *25*, 116–122. [CrossRef]
46. Kaviani, S.; Sohn, I. Influence of random topology in artificial neural networks: A Survey. *ICT Express* **2020**, *6*, 145–150. [CrossRef]