algorithms

MDPI

Review

Overview of Distributed Machine Learning Techniques for 6G Networks

Eugenio Muscinelli †, Swapnil Sadashiv Shinde *,† and Daniele Tarchi †

Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi", University of Bologna, 40126 Bologna, Italy; eugenio.muscinelli@studio.unibo.it (E.M.); daniele.tarchi@unibo.it (D.T.)

* Correspondence: swapnil.shinde2@unibo.it

† These authors contributed equally to this work.

Abstract: The main goal of this paper is to survey the influential research of distributed learning technologies playing a key role in the 6G world. Upcoming 6G technology is expected to create an intelligent, highly scalable, dynamic, and programable wireless communication network able to serve many heterogeneous wireless devices. Various machine learning (ML) techniques are expected to be deployed over the intelligent 6G wireless network that provide solutions to highly complex networking problems. In order to do this, various 6G nodes and devices are expected to generate tons of data through external sensors, and data analysis will be needed. With such massive and distributed data, and various innovations in computing hardware, distributed ML techniques are expected to play an important role in 6G. Though they have several advantages over the centralized ML techniques, implementing the distributed ML algorithms over resource-constrained wireless environments can be challenging. Therefore, it is important to select a proper ML algorithm based upon the characteristics of the wireless environment and the resource requirements of the learning process. In this work, we survey the recently introduced distributed ML techniques with their characteristics and possible benefits by focusing our attention on the most influential papers in the area. We finally give our perspective on the main challenges and advantages for telecommunication networks, along with the main scenarios that could eventuate.

Keywords: machine learning; distributed learning; telecommunications; 6G

Citation: Muscinelli, E.; Shinde, S.S.; Tarchi, D. Distributed Machine Learning Techniques for 6G Networks. *Algorithms* **2022**, *15*, 210. https://doi.org/10.3390/a15060210

Academic Editors: Andras Farago, Ionut Brandusoiu and Héctor Migallón

Received: 16 May 2022
Accepted: 14 June 2022
Published: 15 June 2022

1. Introduction

The rapid growth of data and information availability in recent years produced a paradigm change in wireless networks: "big data" cannot be anymore managed as a whole. New parceled "small data" should be envisaged where large amounts of data are distributed among several nodes for their processing [1]. The 6G standard will provide worldwide connection, reduced latency, and allow the growth of applications based on extremely dense and diverse wireless networks, such as Internet of Things (IoT) systems [1,2]. Intelligent interactions between devices across the network, which is the goal of developing distributed machine learning (ML) methods, will be essential to manage and exploit this continuously increasing amount of data and communications (often from machine to machine) within networks that are not in optimal conditions. In these kinds of applications, there is no need to send large amounts of data to a central system that produces the learning model; instead, data processing and local observations are handled by the devices themselves at the network's edge. There are various advantages to such solutions: a significant reduction in transmission load and the possibility to optimize how resources are utilized, in addition to cost saving [3].

In contrast to older generations, the 5G wireless network brings in a truly digital society by achieving substantial advancements in latency, data rates, mobility, and the number of connected devices. The 5G network with integrated technologies, such as IoT,

mobile edge computing (MEC), fog computing, and ML, spans different domains of society, including robotics [4], vehicular networks [5], healthcare [6], etc. 5G is predicted to reach its limit in a decade; hence, the academia and the standardization bodies are already working toward a new generation of wireless communications, named 6G [7]. The key technical objectives for 6G networks will be:

- Ultra-high data rate (up to 1 terabit per second (Tbps)) and ultra-low latency communications.
- High energy efficiency for resource-constrained devices.
- Ubiquitous global network coverage.
- Trusted and intelligent connectivity across the whole network.

The *space* of needs for 6G applications will encompass a wide range of throughput, latency, dependability, scalability, service availability, service continuity, and security dimensions [8]. Simultaneously, obtaining a significantly higher resource efficiency than in 5G is a critical step, not only toward expanded spectrum capacity, but also toward meeting the ambitious energy reduction goals for future networks. As a result, the design of 6G radio access must be flexible and resource efficient, having the ability to be adjusted in real-time. As an example, multi-antenna communication is a crucial component of tomorrow's wireless networks. Cellular multiple-input multiple-output (MIMO) and decentralized extensions conform to the old cellular paradigm, which is inefficient in dense networks due to overly frequent handovers and pilot sequence re-assignments. This encourages a new user-centric cell-free MIMO strategy, in which the mobile device fluidly navigates through a *sea of access points* that are dynamically associated. Another aspect to be considered in 6G is to achieve communications that use terahertz-scale frequencies (THz). THz communication allows one to cover the spectrum well beyond 100 GHz and has been the focus of significant research for the past two decades [8]. While propagation studies and channel characterization were important at the start of this research, in recent years, various new approaches have been developed, and the feasibility of THz communications has been demonstrated. IEEE in 2017 published the world's first wireless standard (IEEE Std. 802.15.3d-2017 [9]), which operates at a frequency of roughly 300 GHz and has a bandwidth of up to 69 GHz [10]. This big block of the spectrum has the potential to support 1 Tbps data rates in the future without requiring overly sophisticated baseband processing. Nonetheless, there are significant obstacles to overcome. To offset the high-path loss in this frequency band, high-gain antennas are required. This complicates THz communications for mobile applications, as device detection, beam steering, and beam tracking are more difficult in THz systems than in millimeter wave systems.

Supporting the requirements of the large range of 6G applications and use cases is a major challenge. In addition, in order to support such new technological advancements, much higher intelligence levels and processing efforts should be encompassed. This is the reason why artificial intelligence (AI) and ML are considered as fundamental elements of the forthcoming 6G, and not just an optimization tool for the performance evaluation. However, the majority of today's AI/ML solutions use centralized learning, in which data are collected throughout the system but training is performed in a single spot. Centralized learning is not ideal and often too expensive on a distributed platform, such as in carrier networks. At the centralized data center, there are significant energy supply requirements to be covered by the responsible tenant/owner, and privacy concerns.

As a result, as we move closer to 6G, it will be critical to design new procedures that will allow the system to function and perform well, and coordinate learning and execution across a pool of—possibly—computation, networking, storage, and energy resources. Engineering-wise, the ability to manage exceedingly varied resources, system dynamism, and efficient real-time learning are the primary obstacles to achieving this kind of technological development. In the algorithmic field, on the other hand, future objective solutions will include the ability to perform training using highly fragmented data, increase efficiency in dynamic environments, and manage collaborative learning transfer agents in order to optimize anomaly handling.

The goal of this paper is to survey on the most influential papers on distributed learning approaches for 6G and wireless communications. We have surveyed the articles related to machine learning, wireless communication, applications of machine learning in wireless communication, upcoming 6G technology vision, distributed learning for 6G, privacy and security related issues in distributed learning, distributed learning over joint terrestrial and non-terrestrial networks, etc. Most of the articles are from well-known IEEE journals and magazine papers. We have considered the most recent papers that are from the last few years to avoid outdated information.

In Section 2, we have described the various generations of wireless communication technology and presented the upcoming 6G network challenges. After recalling the fundamentals of ML in Section 3, we describe the main distributed learning approaches introduced so far (Section 4). In Section 5, we survey the most influential papers considering the applications of DL algorithms for 6G. In Section 6 we drive future directions for the usage of DL in 6G systems. In Section 7, we discuss the main points of the paper. Finally, in Section 8, we conclude the paper.

2. Wireless Communication Technology

Wireless communication technology has evolved through several generations. In the beginning, the original mobile network, also known as a 1G, supported voice transmission based on the frequency modulation technique in the 1980s. The next generation of wireless communication, known as 2G, supported digital signal communication over analog. It was supported by Global System Mobile Communications (GSM) and was able to send text and picture messages, along with voice-related services. Later in 1995, a 2.5 G standard supporting voice and data communication was introduced. In 2001, a 3G standard was introduced with increased data transmission capabilities and various new video-related services, including video streaming, video conferences, and live video chat. In 2009, a 4G revolution occurred, supporting the high-quality video communication, HD video streaming, and online gaming-related services. 4G technology was succeeded by 5G in the last decade, supporting a plethora of new services and applications. 5G services can be classified into three groups, known as eMBB (enhanced mobile broadband services), URLLC (ultra-reliable low latency communications), and mMTC (massive machine type communication services) [11]. This service often has a stringent data rate, latency, reliability, and connection requirements. 5G technology has advanced with various new technology revolutions, such as the Internet of Things (IoT), MEC, mmWave, and MIMO technique. Additionally, various ML new ML-related services have also emerged for supporting more intelligent networking platforms. With evolving wireless technologies, various new transmission technologies have emerged and are used effectively. For example, code-division multiple access (CDMA) in 3G, MIMO, and orthogonal frequency-division multiplexing (OFDM) in 4G; the mmWave technique; and massive MIMO in 5G have improved data rates by several fold in every generation. The upcoming 6G network is expected to have a more flexible approach while utilizing the time-frequency and space resources to fulfill more goals.

The upcoming 6G technology is expected to serve the 2030s' data-driven society with instant unlimited connectivity requirements [12]. The current version of 5G is insufficient to satisfy the requirements of newly developed forthcoming applications and services, and thus requires a major revolution. Therefore, 6G research already begun at the end of the last decade with the goal of providing improved data rates, trillion-bit-level connection capacity, and undetectable or nonexistent latency [2]. However, new challenges are emerging in the domain of 6G research and need to be handled carefully. Here we present the main 6G challenges, as shown in Figure 1:

- **Device energy consumption:** With the emerging trends of IoT technology, it is expected that the 6G network will provide communication services for a large set of IoT devices with limited energy supply. Therefore, the 6G network will be required to support various energy-saving mechanisms for higher energy efficiency. Various

energy harvesting techniques with energy-efficient communication and computation paradigms can help 6G to overcome the devices' energy scarceness.

- **ML-related challenges:** Several new artificial intelligence applications will be integrated into the 6G network for creating an intelligent networking platform. However, this can also present several challenges, including higher resource requirements, additional energy costs, security, privacy aspects, etc. The edge-based distributed learning and the split learning technologies are expected to play key roles.
- **Terahertz (THz) communication and corresponding challenges:** The upcoming 6G technology is expected to use the THz frequency band to fulfill the users' data requirements. However, higher propagation loss and limited communication ranges are some of the main challenges that need to be addressed.
- **Joint Terrestrial and Non-terrestrial network and possible challenges:** Various new non-terrestrial networking platforms are expected to be integrated into the traditional terrestrial communication network for creating a more reliable 6G communication network. However, proper channel models, mobility management, savior channel loss due to natural causes such as rain fading, and proper resource allocation are some of the main challenges that need to be handled.
- **Mobile edge computing (MEC) and corresponding challenges:** MEC has emerged as a promising technology that brings cloud computing resources to the proximity of end-users. However, size and coverage restrictions often limit the MEC servers' computation and communication capabilities. With upcoming 6G technology, various new latency-critical and data-intensive applications are expected to be enabled. Allocating the proper networking services to the edge servers, proper user-server assignments, proper resource allocation, and user mobility issues are some of the main challenges that require proper attention.

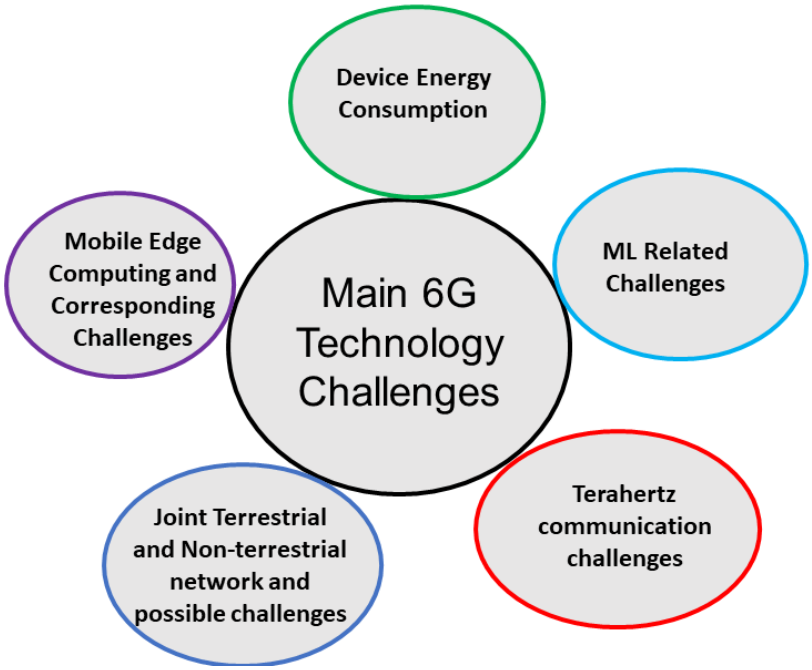

Figure 1. Main applications of distributed learning.

3. Machine Learning in Wireless Communication

Wireless communication and ML are two major revolutionary technologies shaping the world to serve human needs. In the last decade, two technologies were merged and have shown lots of promising results. The complex wireless communication problems are being solved through innovative ML techniques. On the other hand, wireless communication with technologies such as edge intelligence [13] is helping ML algorithms to optimize

their performances. In this section, we introduce two of the main ML techniques used in the recent past for solving the various challenging problems in wireless communication research. We highlight their fundamental concepts, properties, and main applications in the field of wireless communications.

3.1. Deep Learning with Artificial Neural Networks (DL-ANN)

Deep learning is a sub-branch of the vast ML field containing several tools for extracting useful information from various types of datasets. With the recent big data revolution and innovations in hardware technologies, the deep learning field reemerged with the availability of two main ingredients: a large amount of data and greater processing power. Deep learning techniques can be used to achieve various ML tasks, including predictions [14], classifications [15], sequential decision making in uncertain environments [16], etc. The traditional ML methods heavily depend on the feature definitions. The deep learning algorithms can extract the important features from the raw data through the heretical processing of the sequence of layers constituted by the nonlinear processing units called neurons [17]. This extracted information can be used to achieve some predefined objective, i.e., prediction, classification, etc.

Deep learning, in association with neural network implementations, has its origins in the previous century [18]. Based on the number of hidden layers present between input and output layers, neural network models can be classified into two groups. One group consists of a shallow neural network with limited hidden layers (mostly one or two). On the other hand, deep neural network models can have more number of hidden layers, which can also determine their complexity and performance. An artificial neural network, used for tasks such as logistic regression, can have an input layer where the inputs are stored, and output can be a single point towards which the inputs are directed. A neuron in an artificial neural network is a connected processing element that takes one or more weighted inputs and creates real-valued activations by passing them through the activation function [19]. The logistic regression can be modeled as a one-neuron neural network [18]. In general, simple neural networks can have several neurons, forming layers. Each neuron in a layer is connected to all neurons in the following layer, and their outputs get multiplied by a weight associated with particular connection. In general, weight specifies how much information from the previous layer needs to be passed on to the next. The weight is determined by the neuron–target pair, not by the neuron itself. This is to say that the weight $w_{i,j}$ between neuron i and j can be different from $w_{j,i}$, indicating the weight between neuron j and i—obviously from different layers. These weights almost always have distinct values. The information flows through the neural network from the first input layer to the output layer, passing through the hidden layer(s).

As an example, we refer to a simple three-layered artificial neural (ANN) network [18]. The input layer consists of three neurons, each of which can accept an input value. The three variables used to represent them are x_1, x_2, and x_3. The first layer performs only one operation: accepting input values. A single output is available for each of the first layer neurons. The neural network may have fewer input values than the first layer's neurons, but not the other way around. As stated previously, each neuron in the input layer is connected to all neurons in the hidden layer; however, neurons in the same layer are not interconnected.

Each connection between neuron j in layer k and neuron m in layer n has a weight that we will denote as w_{jm}. The weight determines how much of the starting value is sent to each neuron. The weight's value is not fixed; it can both decrease and enhance the input's value. When a neuron in the second layer receives many inputs, it receives the sum of each of them multiplied by the associated weight as a quantity.

In the second layer's third neuron, the inputs are x_3, x_2, and x_3; and the weights are w_{13}, w_{23}, and w_{33}. Each neuron contains a modifiable value called *bias*, which is denoted as b_3 in this case and is applied to the prior sum. The end result is known as *logit*, and it is commonly represented by the letter z. Some basic models simply return the logit value as

an output, and most of them apply a non-linear function to it. The output is traditionally indicated by the letter y. The most common non-linear function is the logistic function, which receives logit z and returns as an output $\sigma(z) = \frac{1}{1+e^{-z}}$. This function "narrows" the field and returns a value between 0 and 1, and it can be given the intuitive probability calculation interpretation of the output given a certain input.

Different layers may have different non-linearities, but all neurons in the same layer apply the same non-linearity to their logits. Weights and biases are essential components: the goal of a neural network is to discover the best set of weights and biases possible. This is accomplished through training, which is done with backpropagation. Its idea is to calculate the network error and then adjust the weight to minimize it as much as possible. Intuitively, the error backpropagation is nothing more than a method of descending the gradient; that is, mathematically speaking:

$$w_{updated} = w_{old} - \eta \nabla E$$

where w denotes the weight, η is the learning rate, and E denotes the cost function measuring the performance of a prediction operation in terms of a mean square error between the perdition results and the target values [18].

In the hidden layer, the weights and biases are initialized at random, multiplied by the relevant inputs, summed together, and then reduced to values between 0 and 1 using logistic regression; and in the end these steps are repeated.

3.2. Reinforcement Learning

Reinforcement learning (RL) seeks the correlations between situations and actions in order to maximize a numerical result [20,21]. The three characteristics mentioned below are the three most significant characteristics of RL implementations. It is essentially a kind of closed loop, as the actions of the learning system influence its future inputs. Furthermore, unlike many ML domains, the learning system is not taught about the action to be taken, but is instead asked to figure out which action is the most convenient for it to execute. In the most interesting circumstances, the actions have an impact on not only the immediate but also all subsequent consequences. The trade-off between exploration and exploitation (see [20]) is a key difficulty in RL that does not exist in most other types of learning frameworks. To optimize earnings, a learning agent must prioritize behaviors that have been tested and proven to be productive in the past. To uncover and recognize these activities, though, it will have to try with ones that have never been done before.

A learning agent must therefore exploit what it knows in order to gain rewards, while also exploring in order to find any margins for optimization. Neither option can be carried out effectively on its own. In addition to the agent and the environment in which it is located, there are four fundamental elements that make up a RL system: a *policy*, a *reward signal*, a *value function*, and optionally, a *model of the surrounding environment*.

A policy specifies how an agent should act at a specific point in time. A policy, in general, is the mapping of perceptions of environmental conditions into actions to be carried out when you are in them. It corresponds to what is known as the collection of rules or stimulus–response associations in psychology.

Within RL, the goal is defined by a reward signal. The environment provides the agent a number and a reward, at regular intervals. In the long run, the agent's ultimate goal is to maximize total rewards. As a result, the reward signal tells the agent which events are positive and which are negative. They are the immediate characteristics that define the problem faced by the agent. The reward received by the agent always depends on the current action and state of the environment. Given that, the reward signal indicates the goodness of an action in the immediate future; the value function specifies which is the best choice in the long run. Using the appropriate simplifications, the value of a state can be understood as the total amount of reward that an agent expects to accumulate in the future, starting from that state. The rewards determine the immediate intrinsic desirability of the state in which the agent operates, whereas the values determine the long-term desirability

of the state in which the agent operates, taking into account the most likely sequence of states.

Rewards are in a sense a primary magnitude, whereas values, as reward predictions, are a secondary magnitude. However, it is the values that should be taken into account most when evaluating a decision. The choice of an action is made based on values, as what you want is a choice that leads to a high value of rewards in the long run. The values are much more complicated to estimate, whereas the rewards are provided by the environment itself. It is no coincidence that the most important component of almost all RL algorithms is the value estimation method.

The model of the environment is the fourth and last element. It is something that tries to mimic the behavior of the environment, or more accurately, something that helps to predict how the environment will behave. Models are used in programming as decision-making tools for a set of activities that take into account potential future events before they occur.

The fact that RL uses training information to evaluate the actions taken, rather than instructing by offering the proper actions to take, is the most fundamental element that distinguishes it from other methods of learning. The purely evaluative input tells us how good the action recently taken is, but it does not suggest whether it is the best or worst option available. Figure 2 shows the RL framework, highlighting the important elements.

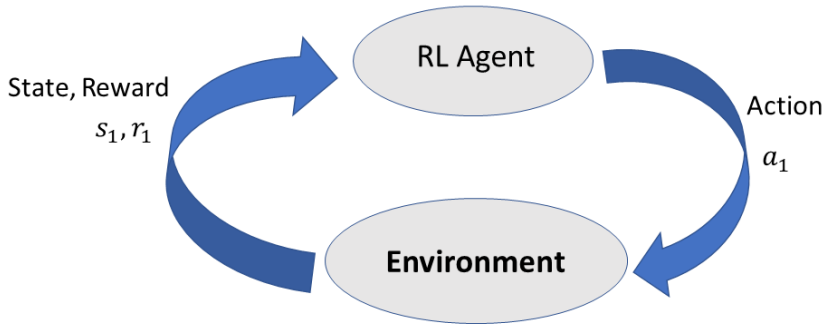

Figure 2. Reinforcement learning framework.

4. Distributed Machine Learning Algorithms

With the advances of the 5G wireless communication standard, a large set of heterogeneous devices with sensors, storage, and processing capabilities are being integrated into the networking infrastructures. These devices can collect tons of data that can be utilized to analyze and solve various networking challenges. Various ML algorithms can be useful for these operations [22]. In the traditional centralized ML approaches, the data collected by the devices are required to be transmitted towards a centralized node equipped with more powerful processing capabilities. However, wireless devices can have limited resources, and such an approach can be challenging, especially in the case of latency-critical applications and services. Additionally, network users are becoming more aware of privacy and are reluctant to share their sensitive data with the outside world to avoid potential data theft. With recent progress in hardware and software technologies, devices can process a limited set of data onboard, allowing them to create the local ML models based on their data. However, such models can have limited accuracies, since most wireless environment users can impact each other's decisions. By integrating the device's local computing capabilities and centralized processor coverage capabilities, various distributed and federated learning (FL) techniques can be implemented. Additionally, the cloud computing-based network settings are increasingly being replaced by processing at the network's edge, i.e., edge computing, which allows for the implementation of ML algorithms to make the best use of the data these devices produce. The concept of centralized ML is no longer dominant [23]. This trend reversal has as its cornerstone the introduction of distributed learning algo-

rithms [3]. In the following we will focus on the most important DL paradigms, i.e., FL and multi-agent RL (MARL).

4.1. Federated Learning

Google proposed the concept of FL in 2016 [24,25]. Given the vast amount of data now available on the network, the goal is to construct ML models based on datasets dispersed across a large number of devices while avoiding dispersion of data. The goal of this strategy is to provide users with the benefits of having a huge volume of data available without having to store it centrally [26]. While defining a number N of data owners as $F_1, F_2, ..., F_N$ and having the goal of training a ML algorithm on their respective data $D_1, D_2, ..., D_N$, a conventional method is to group all the data together and use a set $D = D_1 \cup D_2 \cup ... \cup D_N$ in order to train a M_{SUM} model. In a FL system, data owners collaborate to train an M_{FED} model, in which each owner of F_i data does not need to expose their D_i data to other users [27]. The main characteristics of ideal scenarios for FL applications are:

- Real-world data training on mobile devices has an advantage over training data obtained via proxy and storage in data centers.
- As these data are massive and sensitive to privacy, it is better to avoid storing them in a data center for the sole purpose of training on a certain model.
- In supervised tasks, data labeling can be done directly by the user.

Many applications involving intelligent implementations in the mobile environment, such as picture classification, language models, and speech recognition, reflect the characteristics described above. In fact, even when communicating *anonymized* information to the data center, the user is still at risk, whereas in FL, the information communicated is strictly necessary to enhance a particular model. The goal of FL is typically to minimize an objective function such as:

$$\min_{\omega \epsilon \mathbb{R}^d} F(\omega) \quad \text{where} \quad f(\omega) \triangleq \sum_{i=1}^{m} p_k F_i(\omega)$$

where m is the number of devices ω is the model parameter vector, $p_k > 0$, and $\sum_{i=1}^{m} p_k F_i(\omega) = 1$. F_k is the objective function of the k-th device. The objective function is often defined as the empirical risk to local data; e.g.,

$$F(\omega) = \frac{1}{n_k} \sum_{jk=1}^{n_k} f_{jk}(\omega; x_{jk}; y_{jk})$$

where n_k is the number of examples locally available. Instead, p_k specifies the relative impact that each device has, and is defined as

$$p_k = \frac{1}{n} \quad \text{or} \quad p_k = \frac{n_k}{n}$$

where $n = \sum_k n_k$ is the total amount of examples [28]. FL optimization approaches must deal with the following issues: since the data in use for training are typically dependent on the user's use of the mobile device, the variables usually are not independent and identically distributed (non-IID). As a result, any given information set of a user will not be representative of the population distribution.

Some users will use services or applications much more than others, so there will be extremely variable amounts of data available per user. The number of devices (client) involved in the optimization process will far exceed the number of examples available for each of them, and as mobile devices typically result in slow or expensive connections, communication is limited. Communication costs are modest in data center optimization, whereas computing costs are the most significant. In federate optimization, on the other hand, the key expenses to consider are those related to communication. In fact, since each

dataset on a device is relatively small in comparison to the overall dataset, and because of the high-performance processors found in today's smartphones, computing becomes almost irrelevant. As a result, the goal is to increase computational capacity while decreasing the number of communication steps required to train the model. There are two basic approaches to accomplishing this: the first is to improve parallelism, which requires one to use more devices that operate independently, and the second is to boost the computation of each device, or to have each perform more complex operations.

The heterogeneity of the network is another significant challenge in FL. Due to the variability of the hardware, connectivity, and available power in a federated system, the computing, communication, and data storage capabilities may change. Furthermore, the limits imposed by the network's and system's scale often result in just a small percentage of all available devices being active at any given moment. As a result, the proposed FL systems must account for low participation, be able to accept high heterogeneity, and be sufficiently robust in the case that active devices are turned off during an iteration owing to energy or connectivity constraints.

Federated Averaging (FedAvg) Algorithm

In terms of optimization, a large number of recent successful deep learning applications rely almost entirely on multiple variants of the stochastic gradient descent (SGD) method. As a result, the SGD method is considered as the starting point for federate optimization as well. At first, in might be used intuitively by calculating the gradient for the amount of samples to be processed and then updating the model in a single step for each communication cycle. This method is fast in terms of computing, but it requires a lot of training cycles to have a good model.

In a federate environment, the cost of involving multiple devices is not particularly high; therefore, it is convenient to use a synchronous SGD method as the basis for optimization of a large set of samples (batch) before going to update the model.

For each cycle, a section C of devices is chosen, and the gradient is calculated using all of the data available to them. As a result, C governs the amount of data that are processed, and C = 1 denotes the total amount of data handled. This basic algorithm is called FederatedSDG [26]. With $C = 1$ and a fixed learning rate η, a common implementation of this algorithm has each device compute $g_k = \nabla F_k(\omega_t)$ and save the average gradient in the local data for the current model ω_t; and the central server aggregates these gradients and applies the update:

$$\omega_{t+1} \leftarrow \omega_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} g_k \quad since \quad \sum_{k=1}^{K} \frac{n_k}{n} g_k = \nabla f(\omega_t)$$

Each device, using its own local data, conducts a gradient descent step in the current model, and the server takes the average value of the resulting models. Increasing each device's computational load can be done by iterating its local update before doing the averaging calculation, having implemented the algorithm this way.

This approach is called FederatedAveraging, or FedAvg [26,28]. Three parameters determine the amount of computing: C, the fraction of devices that conduct the computation in each cycle; E, the number of times the training is completed locally every cycle; and B, the size of the data partition to be processed (batch) locally before executing the update. Figure 3 shows the FL framework with FedAvg process. The main FL process elements, i.e., local FL device processing, FL communication, and the FedAvg process at a centralized FL server, are shown.

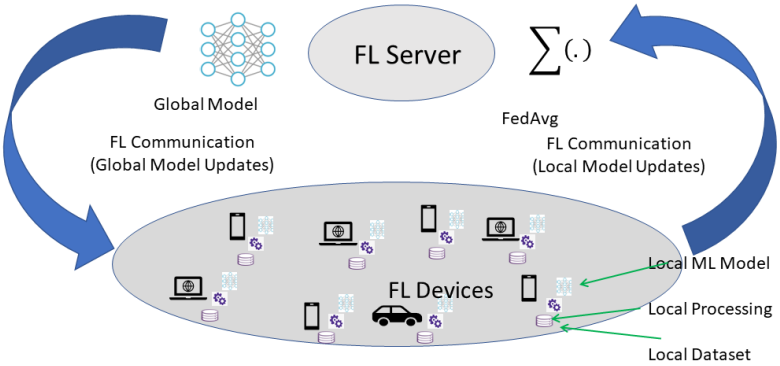

Figure 3. Federated learning framework.

4.2. Multi-Agent Reinforcement Learning (MARL)

Various wireless communication problems can be modeled as sequential decision-making problems and can be effectively solved through the RL-based approaches. Most wireless environments require multiple agent settings where agents can interact with each other and can impact each other's decision-making processes. Therefore, naturally, the multi-agent reinforcement earning (MARL) is a potential candidate for solving various sequential decision-making problems in diverse wireless environments. In particular, mainly due to the developments in the deep learning-based techniques for functional approximations, operations research, and multi-agent systems, MARL has gained lots of attention in the last decade [16,29]. The MARL settings can be classified into three major groups based on the agents' interactions with each other [29]. In the first case of fully cooperative settings, all agents can work together toward optimizing the common goal or reward signals. In the case of such collaborative settings, usually agents can have similar rewards. In another case of fully competitive MARL settings, agents can compete with each other; each agent selfishly tries to maximize its reward. Therefore, the total sum of reward values for all the agents involved in the training process can be zero. There are also hybrid MARL systems where both cooperative and competitive agents can be present.

In Figure 4, we show an example of a MARL system involving the N agents. At a given time step, from a given state $s \in \mathcal{S}$, with $\mathcal{S}$ being a possible state space of the environment, each nth agent with $n \in N$ can take action $a \in \mathcal{A}_n$. Here $\mathcal{A}_n$ is the available action space of the nth agent. Effectively, the total set of actions $A = A_1 \cup A_2 \cdots \cup A_n$ can grow exponentially and become a bottleneck for the implementation of the MARL process. Based upon the combined action from all agents $a = a_1 \cup a_2 \cdots \cup a_n$, each agent can receive the new observation set including the possible new state and rewards. Implementing such a centralized training process can be challenging, especially in dynamic wireless environments where each agent can have limited observability, making the problem non-stationary.

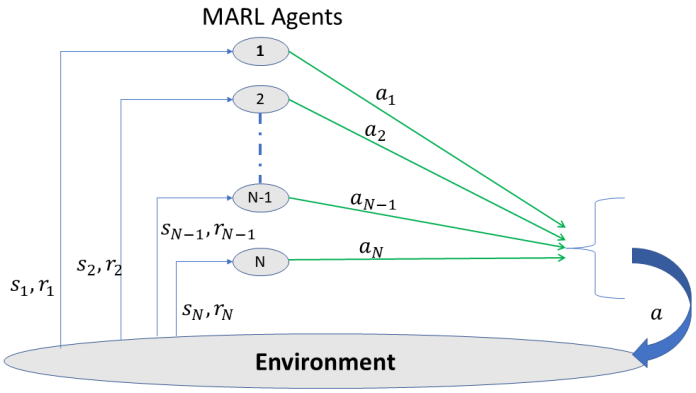

Figure 4. Multi-agent reinforcement learning framework.

4.3. Main Applications

The main applications of distributed learning methods related to 6G scenarios can be individuated in the following main categories (Figure 5):

- **Non-terrestrial Networks:** Recently, various new non-terrestrial networking platforms have been integrated into the terrestrial networking systems to increase the available resource pool, limit the security-related challenges, and add flexibility and more sustainability to plans for natural disasters. Various distributed learning techniques have found their applications in the newly added non-terrestrial networks. The main applications include the cooperative spectrum sharing [30], trajectory design [31], traffic optimization [32], security [33], and task and resource allocation [34].
- **Vehicular Networks:** Distributed learning methods are widely used to solve the challenging problems in vehicular networks. The main applications include intelligent object detection [35], network resource allocation, vehicular data sharing [36], computation offloading to edge-computing-enabled systems [37], traffic light control [38], spectrum sharing [39], and intrusion detection [40].
- **Power System:** Recently, various distributed learning methods hav been used to solve power system-related problems [41]. Voltage control [42], energy management [43], demand predictions [44], transient stability enhancement, and resilience enhancement [45] are some of the main areas of power systems where distributed learning is succinctly used.
- **E-health:** E-health systems are getting packed with various new applications with high computational complexities and resource requirements [46]. Various distributed learning techniques have found applications in e-health systems. Given the sensitive nature of medical data, privacy-preserving FL approaches have gained a lot of interest. The recent advances in FL technology, the motivations, and the requirements of using FL in smart healthcare, especially for the Internet of Medical Things, are presented in [47]. In [48], main security challenges and the mitigation techniques for using an FL for healthcare systems are discussed. Additionally, a multilayered privacy-protective FL platform is proposed for healthcare-related applications.

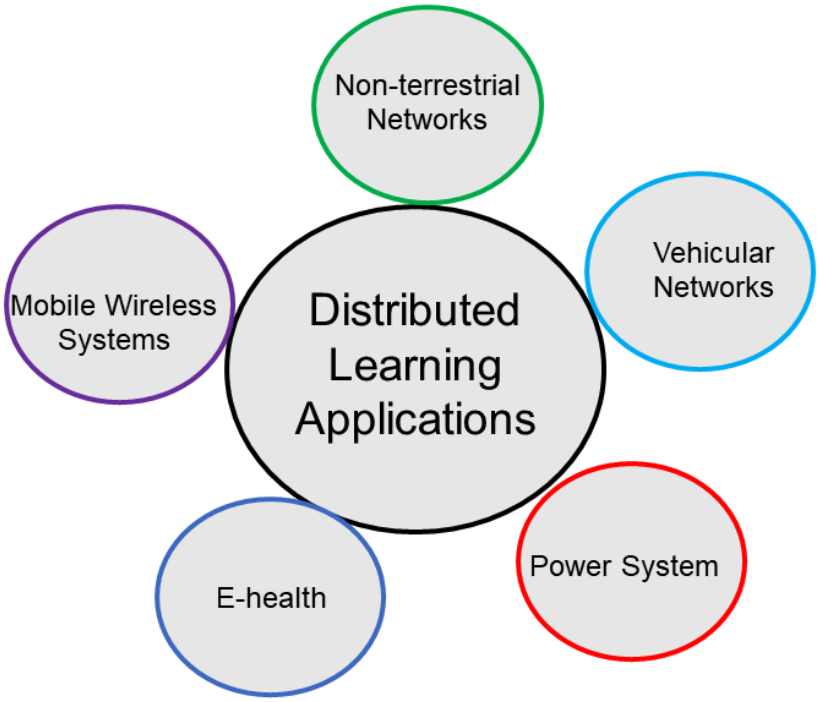

Figure 5. Main applications of distributed learning.

5. Distributed Machine Learning for 6G

Artificial intelligence has taken the central role of defining the vision for upcoming 6G technologies [49]. ML techniques, particularly, distributed ML, can play important roles in achieving the goals set by the 6G technology for society in the 2030s. In this section, we introduce and analyze some of the emerging distributed ML techniques in the field of telecommunications by surveying the most influential papers available in the literature.

5.1. Communication-Efficient Federated Learning

The traditional FedAvg technique, which was previously described, gives a global ML model that is suitable for all devices. Such a technique can be suitable if all devices have independent and identically distributed (IID) datasets. However, in practice, such IID assumptions are rarely true, especially for wireless scenarios with heterogeneous nodes. Therefore, traditional FedAvg-based FL systems can have limited performance in the wireless networking domain. By taking this into account, in [3], researchers proposed two distinct FL algorithms. In particular, they proposed federated multi-task learning (FMTL) and FL based on model agnostic meta-learning (MAML) methods. In the case of FMTL, it is possible to define the minimization problem as:

$$\min_{M,\Omega} \sum_{i=1}^{U} \sum_{k \epsilon k} f(m_i, x_{i,k}, y_{i,k}) + R(M, \Omega) \tag{1}$$

where $M = [m_1, m_2, \ldots, m_U], \Omega$ models the relationship between the different device learning operations and R is a regulation function. The problem can be divided into sub-problems in order to offer the devices the ability to operate in a distributed manner in order to achieve the given goal. It is worth noting that the devices in FedAvg all use the same convergence model. In the case of FMTL, however, the devices may have different convergences, as one model may have less training losses than another due to the non-IID distribution.

The FL technique based on MAML tries to develop a ML model that allows each device to reach its own model after a few gradient descent iterations. In this situation, the goal is stated as follows:

$$\min_{m} \sum_{i=1}^{U} \frac{P_i}{k_i} \sum_{k \epsilon K_i} f(m - \lambda \nabla f_i, x_{i,k}, y_{i,k}) \tag{2}$$

where ∇f_i is the gradient descent of the local ML model of the i-th device, and λ denotes the learning speed. The main characteristics of the FMTL and MAML approaches are given as:

- The FMTL algorithm directly optimizes the customized model of each device, while the federated MAML optimizes the initial model of all devices.
- When working with IID data, the FedAvg algorithm is essential, although FMTL and MAML are more practical for non-IID data.
- The parameter server (PS) must be aware of the data distributions in the devices to choose between FMTL and MAML.
- All FL algorithms must be trained by a distributed iterative process.

The qualities used to evaluate the performance of FL implementations for wireless networks are training loss, convergence time, energy consumption, and reliability.

Training loss is the loss experienced by the function f described in (1) and (2). The ML models of the devices are communicated across imperfect connections in wireless networks; therefore, transmission problems that severely effect training losses are possible. Furthermore, due to limited energy and processing power, only a small subset of devices can actively engage in the learning process, resulting in only a portion of the ML models being used to construct the global model, resulting in negative consequences once again. The convergence time T for FL implementations in wireless networks is expressed as:

$$T = (T_C + T_T) \times N_T$$

where T_C is the time it takes each device to update its local model at each learning step, T_T represents the maximum transmission time per ML model step, and N_T represents the number of steps required for FL to converge. Both the N_T and T_C factors are dependent. In particular, increasing the number of SGD steps to update the local model at each learning step (for example by increasing T_C) can decrease the number of steps required for the FL to converge. The energy consumption E of each device participating in the FL training is expressed as:

$$E = (E_C + E_T) \times N_T$$

where E_C denotes the energy consumed by each device during the training of its model, and E_T is the energy required to communicate the parameters to the parameter server during each step.

The reliability of FL is defined as the probability that it will be able to reach a certain value of loss in training. Since the devices that actually participate in each training phase are a subset of the total number of devices, the devices that communicate the parameters may change each step. This has the power to impact convergence time and loss.

These performance metrics are strongly influenced by the wireless communication factors: in particular, the spectral resources allocated in each device determine the signal to interference-plus-noise ratio (SINR), the data rate, and the probability that the parameter transmits includes errors. As a result, they have an impact on training T_T, E_T, and reliability.

The number of updates to the gradient descent method that may be made for each learning step is determined by the computational capacity. As a result, it has impacts on the amount of time and energy spent on local training. In the meantime, decreasing the number of SGD updates will increase the loss in training and the number of steps required for convergence. SINR, data rate, and the chance to include mistakes in parameter transfer are all determined by transmit power and wireless channel. As a result, increasing the transmission power of each device reduces losses, T_T, N_T, and dependability while increasing E_T. In FL, learning losses and N_T decrease as the number of participants increases, whereas T_T and reliability improve. Training losses, reliability, and the overall number of training steps may decrease as the amount of the FL parameters trained by each device grows. However, the amount of energy and time spent training the model will rise.

5.1.1. Compression and Sparsification

The "bottleneck" in the field of communication that is formed owing to the high size of the trained models is one of the most significant issues in the field of distributed learning, especially with regard to wireless channels. Transmission of all these locally trained parameters to the parameter server and repeating this for each iteration of the learning algorithm across a shared wireless channel is a difficult operation for emerging deep neural networks (DNNs), which have millions of training parameters.

The traditional approach to the problem involves first separating the compression of the DNN parameters from their transmission over the channel. This so-called "digital" approach turns all local changes into bits, which are then successfully transferred over the channel, and all decoded reconstructions are then mediated by the parameter server. In the field of ML, the efficient communication strategies proposed to reduce the amount of information (i.e., the number of bits exchanged by the device with the parameter server for each global iteration) have been many. They can be classified into two main groups: sparsification and quantization [50].

Sparsification aims to convert the updating of the device's d-dimensional model m into its **m*** representation by setting some of its elements to zero. Sparsification can also be thought of as applying a mask vector $\mathbf{M}\,\epsilon[0,1]^d$ to $\mathbf{m}$ in such a way that $\mathbf{m^*} = \mathbf{M} \otimes \mathbf{m}$,

where the mathematical operator $\otimes$ represents element-by-element multiplication. We can define the sparsification level of this vector mask as:

$$\phi \triangleq \|\mathbf{M}\|_1 / d,$$

that is, the quantity of its non-zero elements compared to its size. It is important to note that when the "approximate" (sparse) model is sent to the parameter server, not all the update values are sent; each device only sends non-null values and its location. Therefore, the lower the sparsification level, the higher the compression level and the lower the communication load. In the distributed learning world, the Top-K sparsification approach is arguably the most popular. At each iteration of this technique, each device creates its own sparsification mask $\mathbf{M}_{i,t}$ by finding the K values in its local update that have the largest absolute values. The rand-K is a simpler variant that picks a sparsification mask at random from the collection of masks available with sparsification level K. The Top-K technique has been demonstrated to be significantly more effective in terms of precision and convergence speed; nevertheless, its additional complexity can cause the learning process to slow down.

5.1.2. Federated Learning Training Method Design

In addition to using wireless transmission techniques, new training methods can be used to increase the learning parameters, allowing FL to be applied to wireless networks more effectively.

As wireless devices have limited processing capabilities and energy, the number of the model parameters that can be trained and sent by them is often limited, as is the time frame in which they can be employed for training. When creating a FL training method, all of these factors must clearly be taken into account. Furthermore, the FL training methods determine the topologies of the networks produced by the devices, which has an impact on both the training complexity and the convergence time. As a result, not only the characteristics of the wireless channel, but also the location and mobility of the devices, must be taken into account while designing FL training techniques.

The authors of [51] used decentralized averaging approaches to update each device's local ML model in order to include more devices in FL participation and lessen their reliance on PS. In particular, with these methods, each device needs to transmit its parameters only to the devices close to it to make an estimate of the global model. In this way, the communication load is considerably reduced, and provides the possibility to connect and give one's contribution to the FL even to devices limited by scarcity of resources or low connectivity.

The performance of FL with centralized averaging can be seen in the example considered in [51], where six devices interacting with a BS were involved. In a first implementation of the original FL, only 4 of the 6 devices would be able to contribute, due to insufficient requirements in terms of latency. On the other hand, with this decentralized architecture, even devices that would have been excluded in the prior situation are allowed to contribute by just connecting with the device closest to them. This type of architecture gives a significant improvement in terms of accuracy.

As already highlighted, centralized algorithms will not be able to meet the latency and privacy requirements needed for cellular network applications in post-5G technologies. For this reason, approaches that allow data to remain at the edge of the network (smartphones or IoT devices) and implement a learning algorithm without having to be transferred are rousing increasing interest, both in telecommunications and in the IoT industry.

5.2. Privacy and Security Related Studies in FL Framework

The main advantage of a distributed FL framework is improved user data security, which traditional ML training approaches are unable to provide. Instead of sending the raw user data to the centralized servers, the FL devices are only required to send the locally trained ML model updates and hence enhancing the data security. However, recently it

has been noticed that user information can still be leaked over an FL framework through new security attacks. Proper measures can be taken to boost the client and server-side privacy preservation while implementing more secure FL models [52]. The FL framework security and privacy issues can be classified as FL-device-side and FL-server-side. From the FL device point of view, adequate measures can be taken to elevate the device data-privacy-based perturbation techniques that are employed to protect the valuable user data's parameters by making them more elusive [53]. Otherwise, FL-devices can send the dummy model parameters to the server along with the true parameters to conceal the true parameters [54]. Similarly, various techniques can be added on the server-side for elevating the FL privacy. A proper aggregation technique for limiting the server's access to the individual device parameters, and using encryption techniques such as secure multi-party computation, are some of the common approaches. Techniques such as homomorphic encryption [55] and back-door defenders [56] can be used to discourage the malicious devices from stealing the FL models or from adding any backdoors in the global model. In addition to this various poisoning, attackers can elevate the security threats in the FL models [57,58]. Poison attacks can occur when a malicious user introduces fake training data leading to improper training performances. In the case of FL, devices being in charge of the learning process can easily attack the training process. They can also insert the backdoors in the intermediate training states, leading to the theft of sensitive data from the other devices.

Application of Blockchain Technology for Creating a Secure FL Platform

Recently, blockchain-based methods have been increasingly used to develop more secure and privacy-protective FL architectures. In [59], authors designed a blockchain-based collaborative data sharing architecture with enhanced security for the distributed entities. This differential privacy is then incorporated into the FL process for creating a privacy-protective FL framework. In [60], a fully decentralized, blockchain-based federated learning framework was proposed for a vehicular networking system that ensures end-to-end trustworthiness. The FLchain paradigm, which integrates FL and blockchain technologies to create decentralized, secure, and privacy-enhancing distributed systems, is being increasingly investigated. In [61], authors studied the FLchain by analyzing the fundamental technologies, main challenges, and possible future directions. The applications of blockchain-enabled FL for the Internet of Vehicles (IoVs) are surveyed in [62]. Given the importance of non-terrestrial considerations, in [63], authors studied the integration of FL, blockchain, UAVs, and beyond 5G technologies together.

5.3. Collaborative Machine Learning for Energy-Efficient Edge Networks in 6G

The authors of [64]suggested a decentralized and collaborative ML architecture for 6G edge networks, which allows agents to interact and collaborate in order to improve user request adaption tactics. Furthermore, a multi-agent DRL (Deep RL) algorithm is described that is based on the allocation of computational resources in such a way that overall energy consumption is minimized but latency requirements are always fulfilled. Finally, a federated approach based on DRL was created to reduce computational complexity and training excesses by requiring agents to communicate only the parameters of their models, rather than their full training data.

A typical hierarchical architecture of a 6G network has been proposed in [64], a three-layer architecture: cloud intelligence, edge intelligence, and user intelligence.

The user layer is made up of a large number of terminals that require computation from the edge intelligence layer or the deployment of numerous applications from the cloud intelligence layer. Smartphones, for example, rely on real-time computing services to handle more complicated tasks, such as language recognition, high-definition video, and 3D rendering. Automated robots rely on computing services in IoT industry (IIoT) networks to analyze production lines, and if necessary, change production strategy. For autonomous

driving applications, a huge set of sensing data must be processed within very tight latency limits.

Both cloud and the edge intelligence layers have computing and caching capabilities, allowing them to meet the needs of the user layer terminals. In general, the edge intelligence layer is used for services that require a modest amount of computing power, whereas the cloud intelligence layer is used for applications that demand a lot of processing power. Various network infrastructures, such as macro-base stations (MBSs), small cell base stations (SBSs), roadside units (RSUs), unmanned aerial vehicles (UAVs), and access points (APs), will be introduced in the edge network to give unlimited connectivity to a large number of terminals.

These geographically distributed infrastructures, with computational skills, providing data storage and learning functions, will be fundamental for applications that require the use of artificial intelligence. The edge intelligence layer can be thought of as a collection of smart agents. Since the edge layer is so heterogeneous, each agent will be able to evolve independently and adapt its functions to the environment and users' requests. The proposed decentralized ML architecture integrates multiagent RL and FL into the edge network. In particular, autonomous agents can make local decisions on their own in the execution phase, based on their own observations of the surrounding environment, yet in the training phase, two collaborative schemes can be exploited, local information sharing and model parameter sharing. In the first, the agents investigate the environment by cooperating, allowing each individual agent to perfect their local strategies while learning more about the others. Model parameter sharing, on the other hand, involves agents training their own models independently, each based on their own observations, and then aggregating the data. This procedure can be performed locally by any agent or within the parameter server. The agents in the first scenario load their models into the PS and then receive the global model, whereas in the second situation, the model characteristics are transferred directly amongst the agents.

5.3.1. Energy Efficient Computational Offloading Empowered By Multiagent DRL

Consider an edge network with M small cells, N user equipment, and K orthogonal channels, where each small cell is covered by an SBS and all small cells reuse the system's whole spectrum. The SBS are equipped with MEC servers and are capable of making decisions. Each EU contains a computationally intensive operation that can be partially downloaded to the MEC server and performed remotely.

Offloading performance is connected with each other due to the interference between the cells. The problem of efficiently delegating computation activities and distributing resources can be stated as a problem of minimizing total energy usage while also optimizing resource control decisions. This problem can be solved by exploiting a multi-agent DRL model, in which each SBS behaves like an agent that interacts with the environment to improve its performance. Three fundamental elements in the multi-agent DRL process are the status, the action, and the reward:

- **Status**: each SBS in a real network is unable to acquire the global state of the entire surrounding environment; instead, each SBS has its own limited knowledge. The channel gains of each agent's signals for each channel, the interference power received, and information on the actions requested by the user within its coverage area are all part of the state observed by each agent for the characterization of the environment.
- **Action**: Each instant, each of the agents carries out an action based on its decisions. In this case, the action consists of the computational download decisions, channel and resource allocation for each user served, and uplink power control.
- **Reward**: Through a reward-based learning process, each SBS agent refines its rules. To increase performance and meet the goal of lowering global energy consumption, an incentives system can be devised to encourage collaboration among SBS agents.

The multi-agent DRL process for offloading computation and resource allocation was proposed in [64]. An SBS agent m gets a local observation $s_m[t]$ on the status of the local

environment each instant of time, and executes an action $a_m[t]$ to decide on the offloading of the computation and resource allocation for the customers it serves. Following that, the SBS agent receives a reward $r_m[t]$ based on all agents combined actions, and the environment advances to the next state. As a result, the SBS agent receives the new observation $s_m[t+1]$.

5.3.2. Energy Efficient Computational Offloading Empowered By Federated DRL

To train each agent to use past global information, distributed agents need to be able to exchange local information with each other, which could lead to significant signal overheads. Different cells in small cell networks are very closely linked to each other due to interference. After receiving the interference, each cell can decide on its own computational offloading and resource allocation policy. As a result, the energy consumption reduction problem can be divided into M subproblems, each of which corresponds to a cell whose consumption should be minimized. In this situation, each SBS agent develops its own strategy based on its observations, and the other agents are treated as components of the environment.

However, training a performing DRL model for each agent might be difficult due to strict latency constraints. To solve this issue, a federated learning strategy can be used to increase training performance without requiring data to be shared centrally. The main features of each local model can be defined in the same way as the previously stated DRL algorithm, with the exception that the reward for each cell is local in this case, consisting of energy consumption and the usefulness of ensuring necessary delay in the required tasks.

The offloading system proposed in [64] is based on the presence of multiple SBS agents executing a DDPG model, in which the actor network takes its own local observations of the environment as inputs and returns the selected action; and the critic network has local observation and action as inputs the local action, and outputs its Q value. The FL process consists of a multiple number of coordinated rounds. In each round, each SBS agent independently trains its local model based on its data, without having information on the surrounding environment. Each agent loads the weights of its model to the coordinator, who aggregates them only after these several training rounds. Finally, the coordinator delivers the averaged global model to all agents, who update their local models using it.

5.3.3. Performance Evaluation

The previously discussed approaches have been evaluated in [51], considering a network of small cells with 6 SBSs, 10 UEs in each cell and 16 orthogonal channels. The sizes of computational operations were randomly distributed between 200 and 300 kB. The energy consumption coefficient for each UE was 1 J/GCycle, and for each SBS it was 100 mJ/GCycle. For the DDPG model adopted, the learning rate and the discount factor were set, respectively, at 0.001 and 0.9. To evaluate the performance of the algorithms, the independent learning algorithm (IL-DRL), the full offloading method, and the local computing one were taken as benchmarks.

In the IL-DRL algorithm, each SBS agent learns a policy based on its observations and interactions with the environment, there is no exchange or sharing of models between agents. The performances of the algorithms in terms of convergence were compared. As depicted in [64], the MA-DRL algorithm performed best, thanks to using the information regarding observations and actions of the other SBS agents, which allows a clear improvement in terms of stability.

A comparison of the energy consumption of the various algorithms was also performed in [64]. The MA-DRL and F-DRL algorithms consume less energy than the IL-DRL and full offloading methods, but they still consume more energy than local computing.

5.4. Federated Learning with Over-the-Air Computation

One of the most difficult aspects of FL in wireless networks, known as federated edge learning (FEEL), is overcoming the communication bottleneck caused by the need for multiple devices to upload high-dimension model updates to the parameter server [3]. Various

approaches were used to reduce the resulting latency, including removing slower devices, picking only those whose updates would greatly accelerate learning, and compressing the update data.

Designing new, targeted multiple access systems for FEEL is another option. The main disadvantage of traditional orthogonal access techniques is that they do not adjust to the number of devices as well as they should. The radio resources required, in particular, expand linearly with the number of transmitters; otherwise, delay would increase linearly. The scalability required for multi-access in FEEL is provided by a developing approach known as over-the-air computation, also known as AirComp [3]. In fact, the aggregate of the models is achieved by utilizing the waveform's superposition property. As a result of the simultaneous access, the latency is unaffected by the number of devices. Given the devices' simultaneous and synchronized transmission, their signals are superimposed "over-the-air," and the PS receives their weighted sum, which is described as an aggregate signal; the weights correspond to the channel coefficients.

It is preferable to have uniform weights in the case of AirComp FEEL, so that the aggregate signal is not oriented towards one device over another. Each device must modulate its own signal using analog linear modulation and invert the channel fading by regulating the transmission power in order for this to be achievable. The first procedure is required to exploit the analog waveforms' superposition property, and the second aligns the values that make up the distinct signals. The basic principle of AirComp is proposed in [3].

The use of seemingly basic analog modulations may raise questions about its utility; yet, it has been demonstrated that AirComp can be a great solution for minimizing the mean squared error (MSE) of the distortion when all channels and sources are Gaussian and independent. The time synchronization of device transmission is a critical need for the deployment of AirComp. Uplink transmissions (TDMA and SC-FDMA) for systems such as LTE and 5G also have this requirement.

A fundamental synchronization process in these systems is known as "timing advance," and it can also be used to synchronize AirComp. This process requires involving each device in the estimation of the relevant propagation delay, and then transmitting this estimate ahead of time in order to eliminate the delay. When it comes to estimating the propagation delay, the precision of a synchronization channel is related to its bandwidth. With a bandwidth of 1 MHz, for example, the estimation error is less than 0.1 microseconds. When AirComp is used in an OFDM broadband system, the error causes just a phase shift in a symbol received on a sub-channel if the error is less than the cyclic prefix. This phase shift can then be adjusted. The cyclic prefix in an LTE system is several microseconds long, which is more than enough time to correct synchronization issues. Quantization and decoding mistakes cause distortion in digital modulations. The main sources of distortion in AirComp, on the other hand, are channel noise and interference, which directly disrupt the analogically modulated signals.

In a broadband system, the spectrum is divided into sub-carriers, using OFDM modulation. The implementation of AirComp-FEEL in systems of this type involves the simultaneous and over-the-air aggregation of the model update coefficients, transmitted through the sub-carriers. AirComp FEEL was proposed in [3].

For each OFDM symbol, ideally each sub-carrier is linearly and analogically modulated, with a single element, whose power is determined by the channel inversion. However, due to power limits, reversing the sub-carriers is not feasible, so they are not transmitted. This is known as channel truncation. As all devices must have the same mapping between update coefficients and sub-carriers, channel truncation will remove the coefficients mapped to sub-carriers in deep fade because they cannot be remapped. Near–far problems can arise as a result of channel truncation, in which the fraction of deleted coefficients is substantially higher in devices close to the parameter server (PS) than in devices further away. This phenomenon causes biases and has a negative influence on learning. One option is to use channel truncation solely in circumstances of small-scale fading, which has two

major benefits: equalization of the truncation ratio between devices and the ability for the PS to use data from extremely far away devices.

SignSGD in a Broadband AirComp-FEEL System

Let us look at a real-world example of how wireless channels effect AirComp-FEEL convergence. Consider the implementation of *signSGD* in an AirComp-FEEL broadband system. It requires the replacement of the linear analog modulation with a binary/BPSK modulation. Let us look at a real-world example of how wireless channels affect AirComp-FEEL convergence.

Consider the application of *signSGD* in an AirComp-FEEL broadband system; it necessitates the substitution of a binary/BPSK modulation for the linear analog modulation. The PS choice is based on the received signal's sign, and corresponds to an *over-the-air majority vote* approach that turns the received aggregate gradient into a binary vector. The averaged/aggregated gradient norm reaching during the cycles, represented with the notation $\overline{G}$, is a standard yardstick for the model's convergence level, given a generic attenuation function.

The expected value of $\overline{G}$ for the considered system can be analyzed as a function of the given numbers of rounds and devices, which quantifies the speed of convergence. We have that:

$$E[\overline{G}] \leq \frac{a}{\sqrt{N}}(f_1 + \frac{1}{\sqrt{k}}f_2 + b)$$

in which the factors f_1 and f_2/K correspond to the descent of the gradient. The two parameters a and b concern the effects of the wireless channel. In the ideal case with perfect channels, the values a = 1 and b = 0 are taken. If the channels are AWGN, they are expressed as:

$$a_{AWGN} = \frac{1}{1 - \frac{1}{K\sqrt{SNR}}}, b_{AWGN} = \frac{f_2}{1 - \frac{1}{K\sqrt{SNR}}}$$

where SNR refers to the SNR of the device. They tend to the ideal channel when the factor $K\sqrt{SNR}$ increases, because K removes noise through aggregation and raises SNR by increasing signal power. The convergence speed analysis can be used to estimate the number of communication rounds required to train the model. In the instance of FEEL, learning delay, measured in seconds, is a more practical option in terms of performance measures.

AirComp achieves lower model accuracy than OFDMA, but it considerably reduces latency when there is a large number of devices. The experiment considered an AirComp-FEEL system with over 100 devices running on broadband channels. The purpose was to train a convolutional neural network for handwriting recognition using MNIST data. This has been also discussed in [65].

5.5. Federated Distillation

Despite the fact that FL is intrinsically efficient in terms of communications, it nevertheless necessitates the sharing of huge models. A significant number of parameters are frequently used in modern DNN architectures (e.g., the GPT-3 model [66,67] is the latest architecture for language processing operations, and has 175 billion parameters, which corresponds to about 350 GB) [68,69]. The complete exchange of parameters is a costly operation that obstructs real-world communications, especially when primary resources are limited. Federated distillation was developed in order to address this issue.

Federated distillation only swaps the outputs which are significantly smaller than the model. In a classification procedure, for example, each device executes its own local iterations for each class while saving the average value of the model output. These local average outputs are then loaded into the parameter server at regular intervals, and the local outputs for each class are then aggregated and averaged. The global result of the average of the outputs is then downloaded from each device, which then executes its own local iterations with its own loss function, and a regulator that measures the difference

between its output prediction and the global average, to pass it to its local model. *Knowledge distillation* (KD) is the name given to this approach of regulation [3].

5.6. Multi-Agent Reinforcement Learning for UAV Trajectory

In [70], a collection of drones must work together to manage blocks of land users who make uplink requests in a dynamic and unpredictable manner in this paradigm. The UAVs must move in a coordinated and cooperative manner in order to provide the maximum coverage demanded by consumers. The purpose of this trajectory's design is to maximize the number of people served by each individual drone, which is expressed as an optimization problem (successful service rate). However, the volatility of user requests makes the optimization goal for drones random. Due to this, the traditional optimization strategies, such as branch and bound, are ineffective for this task. As a result, VD-MARL, a new MARL that combines the concepts of the value decomposition network, the agnostic meta-learning model, and the gradient method to optimize the trajectories of all UAVs, has been developed. This technique allows each UAV to assess the expected values of all the other UAVs' successful service rates in order to choose the best possible route. By using the VD-MARL algorithm, each UAV is required to share its rewards with other UAVs, considerably lowering the amount of data exchanged.

5.7. Adaptive FL for the Resource-Constrained Networking Scenarios with Stringent Task Requirements

The FL process is an iterative process wherein the FL devices at each iteration perform the training operation with their own data, transmit the parameter update to centralized servers for averaging purposes, and receive back the global updated model from the FL server. Since each iteration involves both communication and computation processes, it adds an extra burden to the resource-constrained nodes, especially wireless devices with limited energy, time, computation, and networking resources, i.e., vehicles. Additionally, various new services with stringent requirements in terms of latency, data rates, reliability, etc., can pose many new challenges for enabling the FL-based applications in resource-constrained wireless networks. With this in mind, the work done in [71] has proposed a resource-efficient adaptive FL process for the vehicular scenario.

An edge-computing-enabled vehicular network is considered where individual vehicular users (VUs) $m = 1, \cdots, M$ are requesting the services with limited latency requirements. The VUs are employing the nearby edge servers for performing the partial-offloading-based task processing by offloading a certain portion of their tasks, i.e., α_m. The FL-based distributed learning approach is used to find the optimal offloading amount. The FL process is implemented over the joint air–ground network, where air-network-based high altitude platforms (HAPs) are employed as an FL server. Vehicles with their datasets with certain mobility patterns are training the local ML models for solving the computation offloading problem, i.e., finding the optimal amount of vehicular data to be offloaded at the edge computing servers. Given that the FL process is an iterative process where each FL iteration creates a more reliable FL model for optimizing the offloading process, each vehicle is tentative to participate in a maximum number of FL iterations. However, since each FL iteration adds cost and vehicles are bound to perform both FL and task processing operations within a limited time, it is important to select the optimal number of FL iterations performed by each vehicle. Performing less of a number of FL iterations can add additional costs in terms of energy and latency and can make the system unreliable. By taking this into account, a joint latency and energy minimization problem for both FL and the vehicular task processing phases can be formed in terms of a constrained nonlinear optimization problem and solved through various clustered and distributed approaches. Figure 6, provides the overview of the proposed adaptive FL process.

The simulation results show that the proposed FL-based platform over the joint air-ground network can outperform the traditional FL methods in terms of latency, energy, service failures, etc. Given that the upcoming 6G network is expected to be highly

automated, resource-constrained, densified with heterogeneous users and networking platforms, and a complex ecosystem of new services and applications, such adaptive FL strategies would be very useful.

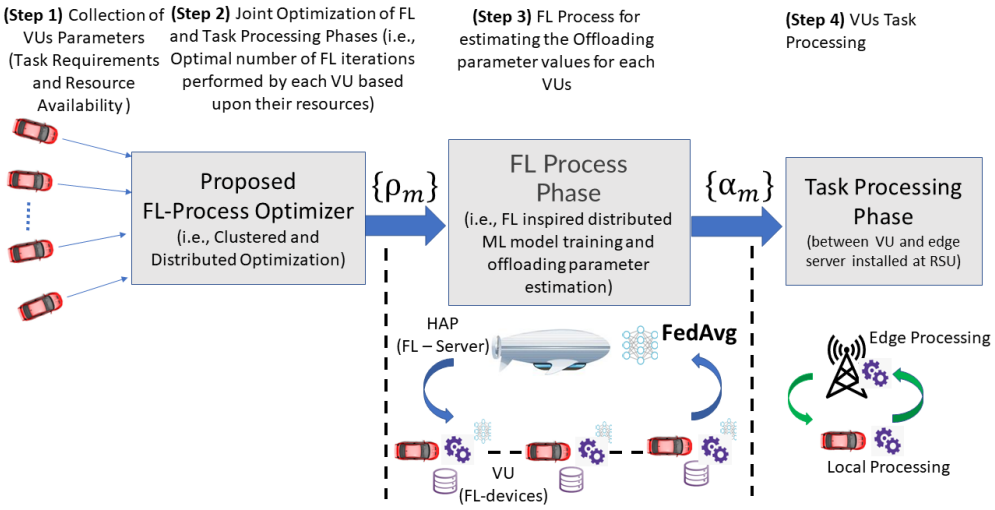

Figure 6. Proposed scheme for the joint FL and task-offloading processes optimization. Reprinted/adapted with permission from Ref. [71]. ©2022, IEEE.

5.8. Distributed Learning Over the Joint Terrestrial and Non-Terrestrial Network

The non-terrestrial networks have gained lots of attention in both 5G and 6G technologies in recent times [72–75]. Various joint terrestrial and non-terrestrial networking solutions are considered for solving challenging problems in different wireless scenarios [76,77]. Given that the ground-based fixed networking infrastructure is saturating in terms of resources, it is expected that the non-terrestrial networks, being easy to deploy, flexible, and highly secure entities, will be used more and more in the next generation of wireless technologies. Various air and space networking platforms, such as unmanned aerial vehicles (UAVs); high altitude platforms such as balloons; and very low earth orbit-based sidelights (VLEO), have already shown great promise and are expected to play important roles. With this in mind, the authors of [78] proposed the novel air–ground distributed-computing-enabled intelligent platform for the vehicular scenario. This work also highlighted the important problem to be solved for the traditional centralized FL process and proposed different FL-based distributed networking solutions for solving vehicular problems.

A hierarchical FL process for the joint air–ground network with multiple edge computing layers was proposed for solving the FL communication problem (see Figure 7). By allowing the FL devices, i.e., vehicles, to transmit their local model updates towards the intermediate edge computing layers, the FL communication latency can be reduced. Additionally, link failures and resource requirements can be limited, which can improve the FL model's training performance compared with the traditional centralized FL training process. Another solution includes the computation offloading enabled by the FL process, where FL devices, i.e., vehicles, can employ the nearby edge servers for improving the performance of the FL process (see Figure 8). With vehicular mobility and resource limitations, vehicles alone can perform the local training operations with limited success only. By allowing vehicular nodes to use the nearby reliable edge servers and their computing capabilities through the split learning-based approaches, FL process performance can be successfully improved. Such solutions will be important to achieve the main goals of 6G technology in upcoming decade.

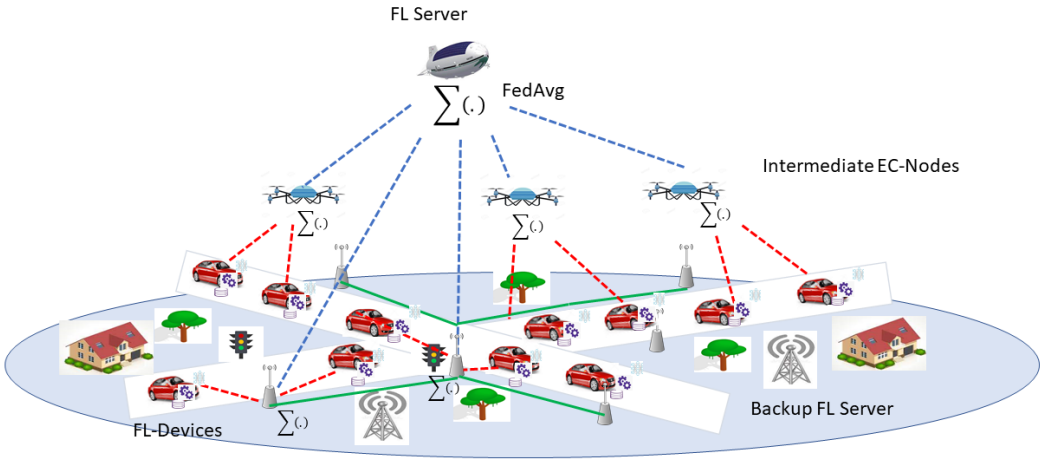

Figure 7. The hierarchical FL platform in a Multi-EC VN architecture. Reprinted/adapted with permission from Ref. [78]. 2021, Attribution 4.0 International (CC BY 4.0) .

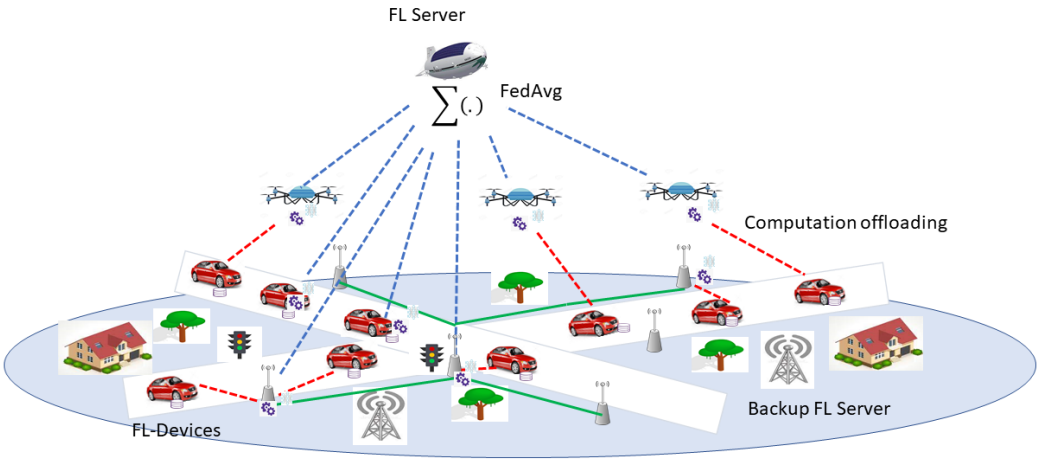

Figure 8. Computation offloading for FL. Reprinted/adapted with permission from Ref. [78]. 2021, Attribution 4.0 International (CC BY 4.0).

6. Future Directions

In this section, we will give our views of possible future directions regarding the application of distributed learning in 6G and wireless system scenarios.

6.1. Collaborative MARL with Wireless Information Sharing

In the past, various MARL techniques have been proposed for implementation in different wireless networking environments. They are mainly based upon the different natures of the training processes. In general, MARL-based solutions involve multiple agents with a common environment trying to optimize a common reward signal. The training policies adapted by the individual agents can impact the other agents' surrounding environments, and as result can dictate their policies. In most wireless environments, agents can have a limited knowledge of the surrounding environments (i.e., partial observability), which makes the MARL problem non-stationary. Additionally, in the case of fully centralized training methods, exponential growths of action and observation spaces can become unbearable in most resource-constrained wireless networking scenarios, i.e., vehicular networks. Therefore, fully centralized training-based MARL solutions have shown limited performances, and other training policies are required. In recent times, other training approaches have been proposed wherein agents can learn effectively. One such approach can be found in [79], where the authors proposed a value-decomposition network-based cooperative MARL architecture. A deep neural network-based backpropagation approach is used to

decompose a common value function into a set of agent-based value functions. However, the complexity of the considered decomposition network, limited uses of state information during training, and applicability towards a reduced class of centralized value functions are bottlenecks. Additionally, in [80], another value-function-based approach, is considered by estimating a joint action-value function from a set of local observation-based action individual agents values through neural networks. The issues of complexity, scalability, etc., can limit the uses of this MARL-based method for solving the given problem.

In the following Figure 9, we propose a collaborative MARL-based solution based upon the data-sharing capabilities of wireless nodes. A dynamic vehicular scenario is considered where a set of moving vehicles, i.e., $m = 1, 2, \cdots M$, are employing the MARL for performing the task T. Task T can be a generic task, such as computation offloading, path planning, and collision avoidance. By employing the V2V and V2I communication technologies, vehicles can share the important environmental parameters, which can be used to reduce issues such as partial observability. MARL agents can have proper command over the nearby environment through such data sharing approaches and can be able to perform the training process with proper knowledge. This way, a centralized training method can be employed by avoiding the challenges introduced by the partial observability and unintended complexity of action and/or observation spaces.

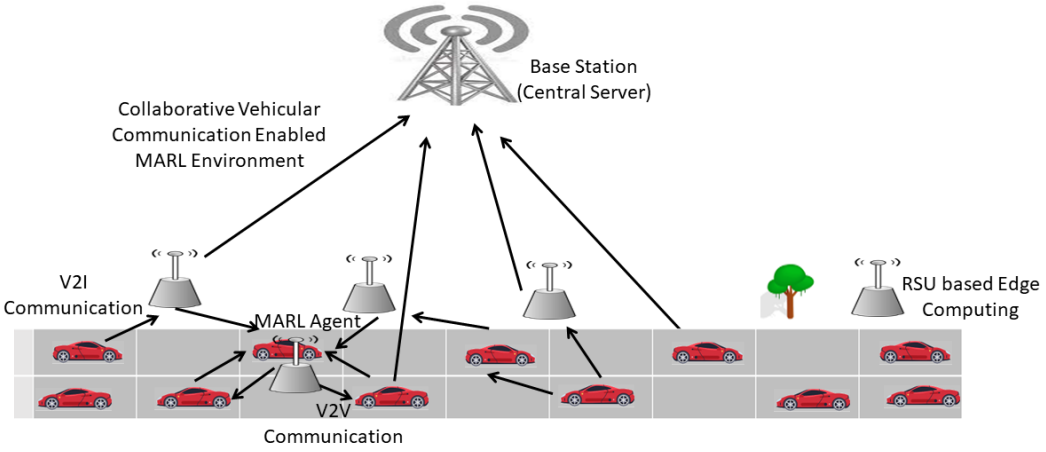

Figure 9. Vehicular-communication-based collaborative MARL process.

6.2. Distributing FL over Multi-EC Enabled Wireless Networks

Though FL has several advantages in terms of privacy, security, and resource requirements, implementing the FL over resource-constrained, dynamic wireless communication nodes can be challenging, especially in cases of latency-critical applications. Compared to the previous generations, the 6G technology has set high goals, and fulfilling them will be challenging, especially with the traditional centralized FL process. Many researchers in recent times have proposed different FL algorithms by optimizing the various aspects of the FL process. Most of the work done so far is closely related to the FL device selection, finding a suitable averaging process for FL servers, resource allocation, analyzing the tradeoff between FL process performance and resource requirements, and analyzing the tradeoff between global and local FL iterations. Given the availability of various edge computing platforms, a distributed FL process can be useful for implementing the FL over resource-constrained wireless nodes. In particular, the overall FL process can be distributed among several edge computing nodes for limiting the FL communication costs. Figure 10 shows one such case for vehicular networks where the FL process is distributed among terrestrial and non-terrestrial networking platforms. In this scenario, the traditional centralized process can require FL devices, i.e., vehicles, to send their model updates to the centralized nodes, i.e., satellites. This can add larger communication delay and noise during each iteration. Due to this, many devices may not be able to participate in the process due to various reasons, such as insufficient resources, unbearable communication

delay, and noise. By distributing the FL averaging process over various intermediate layers, these problems can be tackled. However, this could add additional computation delay over the intermediate layers, and proper node selection is required for having better results.

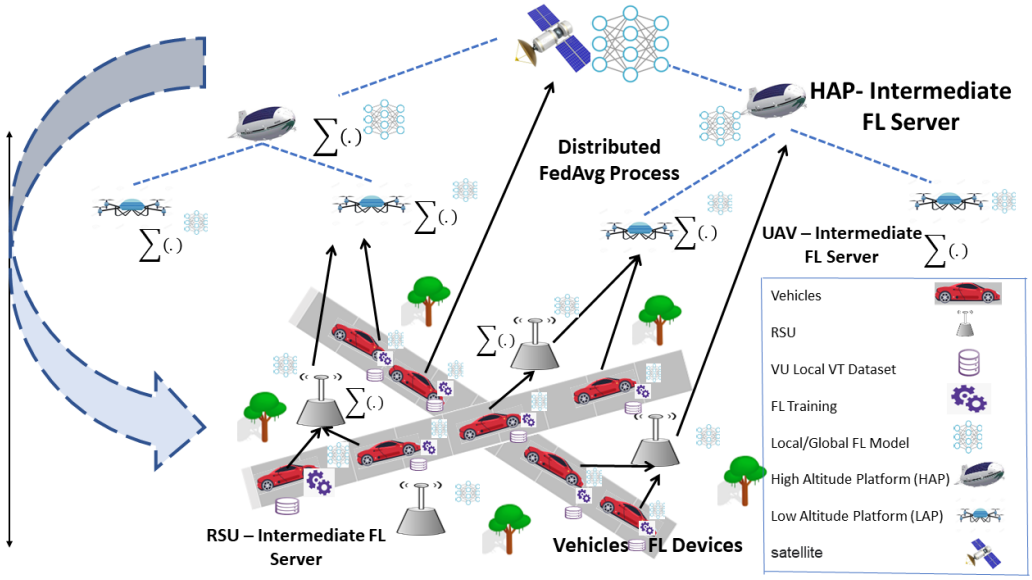

Figure 10. Distributed FL process.

6.3. Privacy and Security-Related Challenges

As discussed in Section 4.2, in recent times various studies have highlighted data security and privacy-related issues in the distributed learning frameworks. Many distributed frameworks allow wireless users to participate in the training process, which can also invite the various adversarial attacks and malicious entities. Distributed frameworks such as FL can limit the data privacy leakage by allowing the individual nodes to perform the training operations and limiting the device–server communication for model parameters. However, sensitive data still can be compromised through various data poisoning attacks. Such security and privacy-related issues pose several challenges to the upcoming 6G networks with heterogeneous users, devices, services, and applications. Technologies such as blockchain can be explored more to tackle these issues.

6.4. FL Hybridization with Heuristic and Meta-heuristic Techniques

Traditionally, various heuristic and meta-heuristic approaches are used to solve complex optimization problems in various fields. The FL-based distributed learning methods are susceptible to various security and privacy-related attacks and can also be modeled as optimization problems. Additionally, the training performance can also be impacted by various factors, such as the number of users, types of datasets, and types of learning algorithms. Recently, the trend of using the heuristic or meta-heuristic approaches to solve the ML issues has been growing. Hybridization-based solutions are being proposed that consider the heuristic/meta-heuristic approaches, along with FL-inspired distributed learning paradigms [81,82]. In [81], authors proposed a hybridization of meta-heuristic and FL frameworks for increasing detection accuracy and minimizing the vulnerability to poisoning attacks. A meta-heuristic technique to manage the FL process that includes the model aggregation, separation, and evaluation. The results from the image database showed higher accuracy and poisoning attack detection. In [82], authors modified this framework for creating a new framework for selecting image classifiers. They proposed the hybridization approach for FL-inspired distributed policy and augmentation method based upon the heuristic approach. The results show that the augmentation technique proposed can provide reliable data, which can increase the classifier's accuracy level.

Such an approach of using heuristic and meta-heuristic techniques to manage and solve the distributed learning-based approaches could be useful in upcoming 6G networks by providing better training accuracy, higher security, etc.

7. Discussion

As most of the current research points out, the upcoming 6G technology is anticipated to create a highly connected society with diverse applications and services. The distributed ML technology has acquired a central place in the 6G research and is expected to play an important role in shaping the 6G world. However, it is important to analyze the various distributed learning solutions available in the literature for using them for various 6G applications. Through surveying the current distributed learning solutions, we have individuated how they can be effectively used in the 6G technology. As analyzed, DL and RL are two of the main ML approaches, and have gained lots of attention in the last decade. However, distributed learning approaches are becoming increasingly considered; in particular, FL and MARL are two of the main distributed learning techniques used to solve the different challenging problems in wireless communication research currently. In particular, FL and MARL are very important for solving various 6G technology scenarios. As highlighted, there could be various innovative solutions associated with distributed learning-based research. These solutions are expected to play key roles in the upcoming days in 6G related solutions.

8. Conclusions

In this paper, we introduced the main challenges that today's telecommunications technology faces, examining the possible future wireless systems and the reasons why the integration of the latter with artificial intelligence will be of fundamental importance. We then provided a general overview of some of the main ML techniques, and then examined in more detail examples of some emerging applications, evaluating their fundamental concepts and major challenges, in terms of implementation and optimization. We also presented various future directions for distributed learning research in the field of 6G.

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

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

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

References

1. Saad, W.; Bennis, M.; Chen, M. A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems. *IEEE Netw.* **2020**, *34*, 134–142. https://doi.org/10.1109/MNET.001.1900287.
2. Yang, P.; Xiao, Y.; Xiao, M.; Li, S. 6G Wireless Communications: Vision and Potential Techniques. *IEEE Netw.* **2019**, *33*, 70–75. https://doi.org/10.1109/MNET.2019.1800418.
3. Chen, M.; Gündüz, D.; Huang, K.; Saad, W.; Bennis, M.; Feljan, A.V.; Poor, H.V. Distributed Learning in Wireless Networks: Recent Progress and Future Challenges. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 3579–3605. https://doi.org/10.1109/JSAC.2021.3118346.
4. Voigtländer, F.; Ramadan, A.; Eichinger, J.; Lenz, C.; Pensky, D.; Knoll, A. 5G for Robotics: Ultra-Low Latency Control of Distributed Robotic Systems. In Proceedings of the 2017 International Symposium on Computer Science and Intelligent Controls (ISCSIC), Budapest, Hungary, 20–22 October 2017; IEEE: Budapest, Hungary, 2017; pp. 69–72. https://doi.org/10.1109/ISCSIC.2017.27.
5. Shah, S.A.A.; Ahmed, E.; Imran, M.; Zeadally, S. 5G for Vehicular Communications. *IEEE Commun. Mag.* **2018**, *56*, 111–117. https://doi.org/10.1109/MCOM.2018.1700467.
6. Bishoyi, P.K.; Misra, S. Enabling Green Mobile-Edge Computing for 5G-Based Healthcare Applications. *IEEE Trans. Green Commun. Netw.* **2021**, *5*, 1623–1631. https://doi.org/10.1109/TGCN.2021.3075903.

7. Huang, T.; Yang, W.; Wu, J.; Ma, J.; Zhang, X.; Zhang, D. A Survey on Green 6G Network: Architecture and Technologies. *IEEE Access* **2019**, *7*, 175758–175768. https://doi.org/10.1109/ACCESS.2019.2957648.
8. Taking Communications to the Next Level. Position Paper, one6G. 2021. Available online: https://one6g.org/download/1350/ (accessed on 1 May 2022).
9. *IEEE Std 802.15.3d-2017 (Amendment to IEEE Std 802.15.3-2016 as amended by IEEE Std 802.15.3e-2017)*; IEEE Standard for High Data Rate Wireless Multi-Media Networks–Amendment 2: 100 Gb/s Wireless Switched Point-to-Point Physical Layer. IEEE: Piscataway, NJ, USA, 2017; pp. 1–55. https://doi.org/10.1109/IEEESTD.2017.8066476.
10. Petrov, V.; Kurner, T.; Hosako, I. IEEE 802.15. 3d: First standardization efforts for sub-terahertz band communications toward 6G. *IEEE Commun. Mag.* **2020**, *58*, 28–33.
11. Shinde, S.S.; Marabissi, D.; Tarchi, D. A network operator-biased approach for multi-service network function placement in a 5G network slicing architecture. *Comput. Netw.* **2021**, *201*, 108598.
12. David, K.; Berndt, H. 6G Vision and Requirements: Is There Any Need for Beyond 5G? *IEEE Veh. Technol. Mag.* **2018**, *13*, 72–80. https://doi.org/10.1109/MVT.2018.2848498.
13. Deng, S.; Zhao, H.; Fang, W.; Yin, J.; Dustdar, S.; Zomaya, A.Y. Edge Intelligence: The Confluence of Edge Computing and Artificial Intelligence. *IEEE Internet Things J.* **2020**, *7*, 7457–7469. https://doi.org/10.1109/JIOT.2020.2984887.
14. Emmert-Streib, F.; Yang, Z.; Feng, H.; Tripathi, S.; Dehmer, M. An introductory review of deep learning for prediction models with big data. *Front. Artif. Intell.* **2020**, *3*, 4.
15. Minaee, S.; Kalchbrenner, N.; Cambria, E.; Nikzad, N.; Chenaghlu, M.; Gao, J. Deep learning–based text classification: A comprehensive review. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–40.
16. Feriani, A.; Hossain, E. Single and multi-agent deep reinforcement learning for AI-enabled wireless networks: A tutorial. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 1226–1252.
17. Zhang, C.; Patras, P.; Haddadi, H. Deep Learning in Mobile and Wireless Networking: A Survey. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 2224–2287. https://doi.org/10.1109/COMST.2019.2904897.
18. Skansi, S. *Introduction to Deep Learning—From Logical Calculus to Artificial Intelligence*; Springer: Cham, Switzerland, 2018. https://doi.org/10.1007/978-3-319-73004-2.
19. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117.
20. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*, 2nd ed.; The MIT Press: Cambridge, MA, USA, 2018.
21. Gronauer, S.; Diepold, K. Multi-agent deep reinforcement learning: A survey. *Artif. Intell. Rev.* **2022**, *55*, 895–943.
22. Balkus, S.V.; Wang, H.; Cornet, B.D.; Mahabal, C.; Ngo, H.; Fang, H. A Survey of Collaborative Machine Learning Using 5G Vehicular Communications. *IEEE Commun. Surv. Tutor.* **2022**, *24*, 1280–1303.
23. Guo, Y.; Zhao, R.; Lai, S.; Fan, L.; Lei, X.; Karagiannidis, G.K. Distributed machine learning for multiuser mobile edge computing systems. *IEEE J. Sel. Top. Signal Process.* **2022**, *16*, 460–473.
24. Konečný, J.; McMahan, B.; Ramage, D. Federated Optimization:Distributed Optimization Beyond the Datacenter. *arXiv* **2015**, arXiv:1511.03575.
25. Konečný, J.; McMahan, H.B.; Ramage, D.; Richtárik, P. Federated Optimization: Distributed Machine Learning for On-Device Intelligence. *arXiv* **2016**, arXiv:1610.02527.
26. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; Aguera y Arcas, B. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Ft. Lauderdale, FL, USA, 20–22 April 2017; Singh, A., Zhu, J., Eds.; Proceedings of Machine Learning Research; Volume 54, pp. 1273–1282. Available online: https://proceedings.mlr.press/v54/mcmahan17a.html (accessed on 1 Apr 2022).
27. Yang, Q.; Liu, Y.; Chen, T.; Tong, Y. Federated Machine Learning: Concept and Applications. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 12:1–12:19. https://doi.org/10.1145/3298981.
28. Li, T.; Sahu, A.K.; Talwalkar, A.; Smith, V. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Process. Mag.* **2020**, *37*, 50–60. https://doi.org/10.1109/MSP.2020.2975749.
29. Zhang, K.; Yang, Z.; Basar, T. Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms. In *Handbook of Reinforcement Learning and Control*; Vamvoudakis, K.G., Wan, Y., Lewis, F.L., Cansever, D., Eds.; Springer: Cham, Switzerland, 2021; pp. 321–384. https://doi.org/10.1007/978-3-030-60990-0_12.
30. Shamsoshoara, A.; Khaledi, M.; Afghah, F.; Razi, A.; Ashdown, J. Distributed cooperative spectrum sharing in uav networks using multi-agent reinforcement learning. In Proceedings of the 2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 11–14 January 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6.
31. Wu, F.; Zhang, H.; Wu, J.; Song, L. Cellular UAV-to-device communications: Trajectory design and mode selection by multi-agent deep reinforcement learning. *IEEE Trans. Commun.* **2020**, *68*, 4175–4189.
32. Qin, Z.; Yao, H.; Mai, T. Traffic optimization in satellites communications: A multi-agent reinforcement learning approach. In Proceedings of the 2020 International Wireless Communications and Mobile Computing (IWCMC), Limassol, Cyprus, 15–19 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 269–273.
33. Zhang, Y.; Zhuang, Z.; Gao, F.; Wang, J.; Han, Z. Multi-agent deep reinforcement learning for secure UAV communications. In Proceedings of the 2020 IEEE Wireless Communications and Networking Conference (WCNC), Seoul, Korea, 25–28 May 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–5.

34. Wang, S.; Chen, M.; Yin, C.; Saad, W.; Hong, C.S.; Cui, S.; Poor, H.V. Federated Learning for Task and Resource Allocation in Wireless High-Altitude Balloon Networks. *IEEE Internet Things J.* **2021**, *8*, 17460–17475.
35. Zhou, X.; Liang, W.; She, J.; Yan, Z.; Kevin, I.; Wang, K. Two-layer federated learning with heterogeneous model aggregation for 6g supported internet of vehicles. *IEEE Trans. Veh. Technol.* **2021**, *70*, 5308–5317.
36. Li, X.; Cheng, L.; Sun, C.; Lam, K.Y.; Wang, X.; Li, F. Federated-learning-empowered collaborative data sharing for vehicular edge networks. *IEEE Netw.* **2021**, *35*, 116–124.
37. Prathiba, S.B.; Raja, G.; Anbalagan, S.; Dev, K.; Gurumoorthy, S.; Sankaran, A.P. Federated learning empowered computation offloading and resource management in 6G-V2X. *IEEE Trans. Netw. Sci. Eng.* **2021**. https://doi.org/10.1109/TNSE.2021.3103124.
38. Wu, T.; Zhou, P.; Liu, K.; Yuan, Y.; Wang, X.; Huang, H.; Wu, D.O. Multi-agent deep reinforcement learning for urban traffic light control in vehicular networks. *IEEE Trans. Veh. Technol.* **2020**, *69*, 8243–8256.
39. Liang, L.; Ye, H.; Li, G.Y. Spectrum sharing in vehicular networks based on multi-agent reinforcement learning. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 2282–2292.
40. Liu, H.; Zhang, S.; Zhang, P.; Zhou, X.; Shao, X.; Pu, G.; Zhang, Y. Blockchain and federated learning for collaborative intrusion detection in vehicular edge computing. *IEEE Trans. Veh. Technol.* **2021**, *70*, 6073–6084.
41. Gholizadeh, N.; Musilek, P. Distributed learning applications in power systems: A review of methods, gaps, and challenges. *Energies* **2021**, *14*, 3654.
42. Wang, S.; Duan, J.; Shi, D.; Xu, C.; Li, H.; Diao, R.; Wang, Z. A data-driven multi-agent autonomous voltage control framework using deep reinforcement learning. *IEEE Trans. Power Syst.* **2020**, *35*, 4644–4654.
43. Xu, X.; Jia, Y.; Xu, Y.; Xu, Z.; Chai, S.; Lai, C.S. A multi-agent reinforcement learning-based data-driven method for home energy management. *IEEE Trans. Smart Grid* **2020**, *11*, 3201–3211.
44. Saputra, Y.M.; Hoang, D.T.; Nguyen, D.N.; Dutkiewicz, E.; Mueck, M.D.; Srikanteswara, S. Energy demand prediction with federated learning for electric vehicle networks. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, Hawaii, USA, 9–13 December 2019; IEEE: Piscataway, NJ, USA, 2019, pp. 1–6.
45. Wang, P.; Govindarasu, M. Multi-agent based attack-resilient system integrity protection for smart grid. *IEEE Trans. Smart Grid* **2020**, *11*, 3447–3456.
46. Bishoyi, P.K.; Misra, S. Towards Energy-and Cost-Efficient Sustainable MEC-Assisted Healthcare Systems. *IEEE Trans. Sustain. Comput.* **2022**. https://doi.org/10.1109/TSUSC.2022.3170508.
47. Nguyen, D.C.; Pham, Q.V.; Pathirana, P.N.; Ding, M.; Seneviratne, A.; Lin, Z.; Dobre, O.; Hwang, W.J. Federated learning for smart healthcare: A survey. *ACM Comput. Surv.* **2022**, *55*, 1–37.
48. Kasyap, H.; Tripathy, S. Privacy-preserving decentralized learning framework for healthcare system. *ACM Trans. Multimed. Comput. Commun. Appl. (TOMM)* **2021**, *17*, 1–24.
49. Wikstrom, G.; Persson, P.; Parkvall, S.; Mildh, G.; Dahlman, E.; Balakrishnan, B.; Ohlrn, P.; Trojer, E.; Rune, G.; Arkko, J.; et al. 6G—Connecting a Cyber-Physical World. White Paper 28, Ericsson, 2022. Available online: https://www.ericsson.com/en/reports-and-papers/white-papers/a-research-outlook-towards-6g (accessed on 1 May 2022).
50. Khan, L.U.; Saad, W.; Han, Z.; Hossain, E.; Hong, C.S. Federated Learning for Internet of Things: Recent Advances, Taxonomy, and Open Challenges. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 1759–1799. https://doi.org/10.1109/COMST.2021.3090430.
51. Chen, M.; Poor, H.V.; Saad, W.; Cui, S. Wireless Communications for Collaborative Federated Learning. *IEEE Commun. Mag.* **2020**, *58*, 48–54. https://doi.org/10.1109/MCOM.001.2000397.
52. Ma, C.; Li, J.; Ding, M.; Yang, H.H.; Shu, F.; Quek, T.Q.S.; Poor, H.V. On Safeguarding Privacy and Security in the Framework of Federated Learning. *IEEE Netw.* **2020**, *34*, 242–248. https://doi.org/10.1109/MNET.001.1900506.
53. Geyer, R.C.; Klein, T.; Nabi, M. Differentially private federated learning: A client level perspective. *arXiv* **2017**, arXiv:1712.07557.
54. Kido, H.; Yanagisawa, Y.; Satoh, T. Protection of Location Privacy using Dummies for Location-based Services. In Proceedings of the 21st International Conference on Data Engineering Workshops (ICDEW'05), Tokyo, Japan, 3–4 April 2005; pp. 1248–1248. https://doi.org/10.1109/ICDE.2005.269.
55. Papernot, N.; Abadi, M.; Erlingsson, U.; Goodfellow, I.; Talwar, K. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv* **2016**, arXiv:1610.05755.
56. Andreina, S.; Marson, G.A.; Möllering, H.; Karame, G. Baffle: Backdoor detection via feedback-based federated learning. In Proceedings of the 2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS), Washington, DC, USA, 7–10 July 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 852–863.
57. Tolpegin, V.; Truex, S.; Gursoy, M.E.; Liu, L. Data poisoning attacks against federated learning systems. In *Lecture Notes in Computer Science*; Chen, L., Li, N., Liang, K., Schneider, S., Eds.; Computer Security—ESORICS 2020; ESORICS 2020; Springer: Cham, Switzerland, 2020; Volume 12308, pp. 480–501.
58. Cao, D.; Chang, S.; Lin, Z.; Liu, G.; Sun, D. Understanding Distributed Poisoning Attack in Federated Learning. In Proceedings of the 2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS), Tianjin, China, 4–6 December 2019; pp. 233–239. https://doi.org/10.1109/ICPADS47876.2019.00042.
59. Lu, Y.; Huang, X.; Dai, Y.; Maharjan, S.; Zhang, Y. Blockchain and Federated Learning for Privacy-Preserved Data Sharing in Industrial IoT. *IEEE Trans. Ind. Inform.* **2020**, *16*, 4177–4186. https://doi.org/10.1109/TII.2019.2942190.
60. Pokhrel, S.R.; Choi, J. Federated Learning With Blockchain for Autonomous Vehicles: Analysis and Design Challenges. *IEEE Trans. Commun.* **2020**, *68*, 4734–4746. https://doi.org/10.1109/TCOMM.2020.2990686.

61. Nguyen, D.C.; Ding, M.; Pham, Q.V.; Pathirana, P.N.; Le, L.B.; Seneviratne, A.; Li, J.; Niyato, D.; Poor, H.V. Federated Learning Meets Blockchain in Edge Computing: Opportunities and Challenges. *IEEE Internet Things J.* **2021**, *8*, 12806–12825. https://doi.org/10.1109/JIOT.2021.3072611.
62. Billah, M.; Mehedi, S.; Anwar, A.; Rahman, Z.; Islam, R. A Systematic Literature Review on Blockchain Enabled Federated Learning Framework for Internet of Vehicles. *arXiv* **2022**, arXiv:2203.05192.
63. Saraswat, D.; Verma, A.; Bhattacharya, P.; Tanwar, S.; Sharma, G.; Bokoro, P.N.; Sharma, R. Blockchain-Based Federated Learning in UAVs Beyond 5G Networks: A Solution Taxonomy and Future Directions. *IEEE Access* **2022**, *10*, 33154–33182. https://doi.org/10.1109/ACCESS.2022.3161132.
64. Huang, X.; Zhang, K.; Wu, F.; Leng, S. Collaborative Machine Learning for Energy-Efficient Edge Networks in 6G. *IEEE Netw.* **2021**, *35*, 12–19. https://doi.org/10.1109/MNET.100.2100313.
65. Zhu, G.; Wang, Y.; Huang, K. Broadband Analog Aggregation for Low-Latency Federated Edge Learning. *IEEE Trans. Wirel. Commun.* **2020**, *19*, 491–506. https://doi.org/10.1109/TWC.2019.2946245.
66. Floridi, L.; Chiriatti, M. GPT-3: Its nature, scope, limits, and consequences. *Minds Mach.* **2020**, *30*, 681–694.
67. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
68. Mishra, A.; Latorre, J.A.; Pool, J.; Stosic, D.; Stosic, D.; Venkatesh, G.; Yu, C.; Micikevicius, P. Accelerating sparse deep neural networks. *arXiv* **2021**, arXiv:2104.08378.
69. Chen, L.; Lin, S.; Lu, X.; Cao, D.; Wu, H.; Guo, C.; Liu, C.; Wang, F.Y. Deep Neural Network Based Vehicle and Pedestrian Detection for Autonomous Driving: A Survey. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 3234–3246. https://doi.org/10.1109/TITS.2020.2993926.
70. Hu, Y.; Chen, M.; Saad, W.; Poor, H.V.; Cui, S. Distributed Multi-Agent Meta Learning for Trajectory Design in Wireless Drone Networks. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 3177–3192. https://doi.org/10.1109/JSAC.2021.3088689.
71. Shinde, S.S.; Bozorgchenani, A.; Tarchi, D.; Ni, Q. On the Design of Federated Learning in Latency and Energy Constrained Computation Offloading Operations in Vehicular Edge Computing Systems. *IEEE Trans. Veh. Technol.* **2022**, *71*, 2041–2057. https://doi.org/10.1109/TVT.2021.3135332.
72. Rinaldi, F.; Maattanen, H.L.; Torsner, J.; Pizzi, S.; Andreev, S.; Iera, A.; Koucheryavy, Y.; Araniti, G. Non-Terrestrial Networks in 5G amp; Beyond: A Survey. *IEEE Access* **2020**, *8*, 165178–165200. https://doi.org/10.1109/ACCESS.2020.3022981.
73. Verdone, R.; Mignardi, S. Joint Aerial-Terrestrial Resource Management in UAV-Aided Mobile Radio Networks. *IEEE Netw.* **2018**, *32*, 70–75. https://doi.org/10.1109/MNET.2018.1800036.
74. Lin, X.; Rommer, S.; Euler, S.; Yavuz, E.A.; Karlsson, R.S. 5G from Space: An Overview of 3GPP Non-Terrestrial Networks. *IEEE Commun. Stand. Mag.* **2021**, *5*, 147–153. https://doi.org/10.1109/MCOMSTD.011.2100038.
75. Giordani, M.; Zorzi, M. Non-Terrestrial Networks in the 6G Era: Challenges and Opportunities. *IEEE Netw.* **2021**, *35*, 244–251. https://doi.org/10.1109/MNET.011.2000493.
76. Shah, H.A.; Zhao, L.; Kim, I.M. Joint Network Control and Resource Allocation for Space-Terrestrial Integrated Network Through Hierarchal Deep Actor-Critic Reinforcement Learning. *IEEE Trans. Veh. Technol.* **2021**, *70*, 4943–4954. https://doi.org/10.1109/TVT.2021.3071983.
77. Zhang, S.; Zhu, D.; Wang, Y. A survey on space-aerial-terrestrial integrated 5G networks. *Comput. Netw.* **2020**, *174*, 107212. https://doi.org/10.1016/j.comnet.2020.107212.
78. Shinde, S.S.; Tarchi, D. Towards a Novel Air-Ground Intelligent Platform for Vehicular Networks: Technologies, Scenarios, and Challenges. *Smart Cities* **2021**, *4*, 1469–1495. https://doi.org/10.3390/smartcities4040078.
79. Sunehag, P.; Lever, G.; Gruslys, A.; Czarnecki, W.M.; Zambaldi, V.; Jaderberg, M.; Lanctot, M.; Sonnerat, N.; Leibo, J.Z.; Tuyls, K.; et al. Value-Decomposition Networks For Cooperative Multi-Agent Learning. *arXiv* **2017**, arXiv:1706.05296.
80. Rashid, T.; Samvelyan, M.; de Witt, C.S.; Farquhar, G.; Foerster, J.; Whiteson, S. Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. *J. Mach. Learn. Res.* **2020**, *21*, 1–51. Available online: http://jmlr.org/papers/v21/20-081.html (accessed on 1 May 2022).
81. Połap, D.; Woźniak, M. Meta-heuristic as manager in federated learning approaches for image processing purposes. *Appl. Soft Comput.* **2021**, *113*, 107872.
82. Połap, D.; Woźniak, M. A hybridization of distributed policy and heuristic augmentation for improving federated learning approach. *Neural Netw.* **2022**, *146*, 130–140.