electronics

Article

Cloud-Edge Collaborative Optimization Based on Distributed UAV Network

Jian Yang [1,*], Jinyu Tao [2], Cheng Wang [2] and Qinghai Yang [2]

[1] The 54th Research Institute of China Electronics Technology Group Corporation, Shijiazhuang 050081, China
[2] The School of Telecommunications Engineering, Guangzhou Institute of Technology, Xidian University, Xi'an 710126, China; jytao@stu.xidian.edu.cn (J.T.); chengw@stu.xidian.edu.cn (C.W.); qhyang@xidian.edu.cn (Q.Y.)
* Correspondence: jianyang202408@163.com

Abstract: With the continuous development of mobile communication technology, edge intelligence has received widespread attention from academia. However, when enabling edge intelligence in Unmanned Aerial Vehicle (UAV) networks where drones serve as edge devices, the problem of insufficient computing power often arises due to limited storage and computing resources. In order to solve the problem of insufficient UAV computing power, this paper proposes a distributed cloud-edge collaborative optimization algorithm (DCECOA). The core idea of the DCECOA is to make full use of the local data of edge devices (i.e., UAVs) to optimize the neural network model more efficiently and achieve model volume compression. Compared with the traditional Taylor evaluation criterion, this algorithm consumes less resources on the communication uplink. The neural network model compressed by the proposed optimization algorithm can achieve higher performance under the same compression rate.

Keywords: unmanned aerial vehicle network; edge intelligence; neural network model optimization; model compression

check for updates

Citation: Yang, J.; Tao, J.; Wang, C.; Yang, Q. Cloud-Edge Collaborative Optimization Based on Distributed UAV Network. *Electronics* **2024**, *13*, 3763. https://doi.org/10.3390/electronics13183763

Academic Editor: Subir Halder

Received: 30 July 2024
Revised: 13 September 2024
Accepted: 20 September 2024
Published: 22 September 2024

1. Introduction

Cloud-edge collaboration in UAV networks represents an advanced intelligent system architecture designed to enable seamless interaction and coordination between UAVs, cloud servers, and edge computing resources during their operations [1,2]. In this system, cloud servers provide powerful data storage and computing capabilities for processing large-scale data analysis, complex mission planning, and global decision-making, enabling the UAV system to obtain global perception and intelligent decision-making capabilities. At the same time, edge computing resources are located close to the UAV operation area and can provide low-latency computing and communication services, enabling UAVs to obtain real-time data, perform perception processing, and make local decisions faster [3–6].

The core of cloud-edge collaboration of drone networks is to distribute computing and decision-making tasks between the cloud and the edge, thereby optimizing resource utilization [7]. In this framework, cloud servers can use their powerful processing capabilities to perform complex task planning, path optimization, and global coordination, while edge computing can process real-time perception data, obstacle avoidance calculations, and local decision-making, thereby reducing communication delays and enhancing the system's response speed and stability.

In cloud-edge collaborative learning, the training of artificial intelligence (AI) neural network models by drone edge terminals requires a large amount of computing resources [8]. In particular, when the model has a high number of parameters, the power requirements of drone edge terminals are high as well. At the same time, the communication overhead generated when the network transmits model gradient data increases. In order to solve the problem of insufficient computing and communication resources in drone networks during cloud-edge collaborative learning, model compression methods are often used to obtain lightweight and low-computation models. Model compression mainly includes the compression directions of model pruning [9], parameter quantization [10], and lightweight convolution kernel design. Existing solutions mostly simplify models by pruning redundant parameters, such as the centralized pruning method used in drone networks. The decision to prune the model is made in a central location or central server. This means that the structure and weight information of the entire model is sent to a central location, where decisions are made to determine which weights should be pruned or trimmed. However, this usually requires considerable computing resources and communication overhead, and does not fully consider data isolation and privacy issues [11].

In this paper, we propose a distributed cloud-edge collaborative optimization algorithm to optimize the neural network model. Compared with the traditional centralized neural network compression method [12], the proposed method makes full use of decentralized data and drone equipment to optimize the neural network and obtain a more streamlined model. Specifically, the UAVs are responsible for collecting data from their respective local environments, which are then used to prune the model on-site at the edge level. This localized pruning ensures that each UAV processes relevant decentralized data based on its specific operating environment, which improves the model's adaptability to dynamic conditions. The cloud server aggregates the pruning results from multiple UAVs, allowing for further fine-tuning and synchronization. By distributing the computational tasks between the drones (edge devices) and the cloud, the proposed method not only reduces the computational load on the cloud server but also decreases the overall communication overhead. This collaborative optimization between UAVs and the cloud enables the construction of a more efficient and streamlined model that retains high performance with minimal redundancy. The proposed distributed cloud-edge collaborative optimization algorithm uses NVIDIA's Taylor evaluation criterion [13] and evaluates the adjustable convolution kernel through the Taylor evaluation criterion to reduce model redundancy and achieve light model weight with minimal performance loss. This optimization method effectively simplifies the neural network model in the edge intelligence environment, thereby improving the flexible adaptability of drones in actual application scenarios.

The rest of this paper is organized as follows: Section 2 delves into the related research on compression of cloud-edge collaborative models in UAV networks; Section 3 introduces the system model discussed in this paper; Section 4 details the distributed cloud-edge collaborative optimization algorithm; and experimental results and evaluations are presented in Section 5. Finally, Section 6 concludes the paper.

2. Related Work

In recent years, the size and complexity of deep neural network (DNN) models have experienced significant growth, especially in the field of natural language processing. Research conducted by OpenAI [14] demonstrates that larger models with a higher number of parameters and richer training data generally exhibit superior performance. This trend towards increasingly large-scale models is driven by their enhanced ability to handle complex tasks and deliver high accuracy. Despite the advantages, deploying such models on edge devices such as drones presents notable challenges due to limited computing resources.

Deep neural networks are selected for deployment on edge devices despite their computational demands due to their remarkable capabilities in performing sophisticated tasks [15]. These tasks include real-time image recognition, target tracking, and autonomous navigation, all of which are critical for the effective operation of drones. The high performance of DNNs often outweighs the difficulties associated with their deployment in constrained environments. For example, DNNs enable advanced functionalities that are crucial for mission success in drone operations, such as precise object detection and intelligent decision-making during flight. However, in order to leverage these capabilities it is essential to address the challenges posed by the resource limitations of edge devices.

To facilitate the deployment of complex models on edge devices, various model compression and optimization techniques have been developed; these methods aim to reduce the computational load and memory requirements of neural networks, making them more suitable for resource-constrained environments. Key techniques include lightweight model design, model pruning, low-rank approximation, knowledge distillation, and weight quantization [16]. These techniques adjust the network structure or the number of parameters to achieve a more efficient model.

For instance, lightweight models are designed with fewer parameters and reduced complexity in order to minimize resource usage. Model pruning involves the systematic removal of less important parameters and network components, which helps to decrease computational demands while preserving model performance. Low-rank approximation simplifies network layers by approximating them with lower-rank structures. Knowledge distillation transfers knowledge from a larger and more complex model to a smaller and more efficient one, whole weight quantization reduces the precision of model weights to save memory.

In the context of drone applications, the challenges of deploying deep neural networks are particularly pronounced. Drones are constrained by their limited battery life, processing power, and memory, yet must perform real-time data processing tasks; to address these constraints, pruning optimization has become a crucial technique. Pruning reduces the computational burden and energy consumption by removing unnecessary parameters from the network. For example, MobileNet [17] employs depthwise separable convolution to reduce the size of convolution kernels, thereby decreasing the number of parameters and overall network complexity. EfficientNet [18] utilizes neural architecture search to explore the impact of input resolution, layer width, and network depth, resulting in a lightweight model with approximately 1.3 million parameters.

Further optimization techniques involve pruning existing neural network models. This approach evaluates the importance of model parameters, ranks them, and removes less significant parameters while aiming to minimize performance degradation. For instance, NVIDIA's method based on Taylor expansion [19] ranks convolution kernels by importance and removes those deemed less critical, achieving a 12-fold reduction in computational load for AlexNet with only a minimal accuracy loss.

Additional research by Han Song's team at Stanford University [20] has highlighted advanced optimization techniques involving compression, quantization, and encoding. Their approach includes pruning to create a sparse network, sharing similar weights, and using Huffman coding for efficient weight encoding. This process resulted in a dramatic reduction of parameters in AlexNet and VGG-16 without accuracy loss, demonstrating the effectiveness of these methods in maintaining performance while reducing model size.

The Lottery Ticket Hypothesis (LTH) has also influenced recent pruning methods. Proposed by Frankle and Carbin [21], the LTH suggests that there exists a subnetwork (a "winning ticket") within a larger network that can achieve comparable performance to the original network when trained from scratch. This hypothesis has spurred new approaches to identify and preserve such subnetworks for efficient model deployment.

Similarly, federated distributed pruning [22] addresses challenges in decentralized training environments such as UAV networks. This approach involves pruning models across multiple distributed devices while maintaining overall accuracy and efficiency. It contrasts with centralized methods by focusing on collaborative optimization and data sharing among devices, which is particularly relevant for resource-constrained settings.

In edge computing environments such as those involving drones, optimizing neural network models is crucial due to data privacy concerns and the exponential growth of edge devices. Although this paper does not delve into co-training technologies, understanding the background of cloud-edge co-training underscores the necessity of model optimization on edge devices. This paper focuses on the application of pruning techniques to optimize neural network models, enabling the deployment of complex deep learning models in resource-constrained environments and effectively reducing data transmission requirements.

3. System Model

Before performing cloud-edge collaborative neural network model compression on UAV networks, it is necessary to model the system.

The UAV network cloud-edge learning architecture is composed of edge devices, edge servers, and central servers that form an edge-cloud network system. It aims to meet diverse business needs efficiently and in real time. It builds edge intelligence by sinking AI capabilities to the edge of the network [23] and conducts joint training and reasoning collaboration between network edge drones, edge clouds, and central clouds. Unlike traditional federated learning, drone network cloud-edge collaborative learning combines the distributed learning characteristics and privacy protection advantages of federated learning [24] while also possessing cloud-edge collaborative reasoning capabilities, allowing the cloud center and drone edge to jointly perform intelligent task reasoning. This multilevel architecture has obvious advantages over traditional federated learning architectures in terms of distributed training.

The cloud-edge learning architecture of the drone network includes not only a central cloud server but also an edge cloud server deployed close to the drone edge device. The responsibilities of the edge cloud server include distributing tasks to the drone edge device. At the same time, the drone edge device uploads the model to the edge cloud server after updating it locally. Then, the model parameters of the drone edge terminal are weighted averaged to update the temporary model of the edge cloud. In this architecture, we assume that the UAV terminals possess uniform computing capabilities, including processing power and available memory, although we recognize that in practical scenarios UAVs may vary in terms of hardware resources. In future studies, we will explore how the system can handle heterogeneous UAV hardware environments, including different levels of computing power, memory, and network bandwidth. Additionally, in the current setup each UAV is responsible for performing local model updates, and data are transmitted in fixed intervals to synchronize model parameters across the UAVs and the edge–cloud. Communication is performed synchronously to ensure that all UAVs update their local models based on the same version of the model at the edge, reducing the chance of inconsistencies. The updated temporary model is sent to the drone edge terminal to update the local model. After executing it again, all edge clouds upload the temporary model to the central cloud in order to aggregate the model and update the global model. This process is shown in Figure 1.

In the scenario of collaborative reasoning in a cloud-edge drone network, multiple drone devices can collaborate with each other to perform AI task reasoning [25], allowing them to complete tasks that are difficult or impossible for a single drone edge device. When computing resources such as drone computing power, storage, and communication bandwidth are unevenly distributed in a drone network, drone devices with more computing resources can help devices with less computing resources to process related tasks.

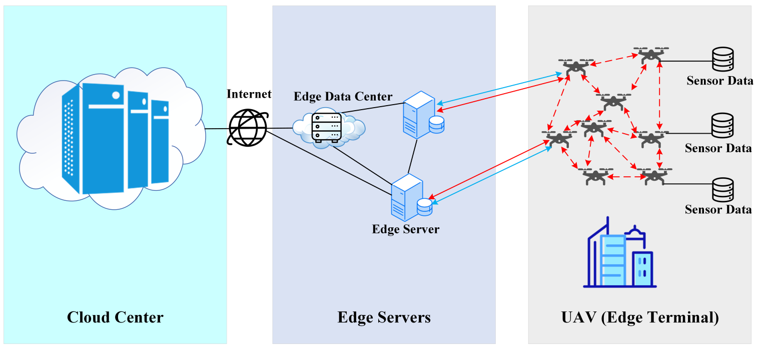

Figure 1. Cloud-edge learning of UAV networks.

In a cloud-edge collaborative training UAV network, due to the limitations of energy and computing resources, a single UAV terminal can only perform data mining on the data it receives [2] and cannot form an analysis of more data that meet the same distribution. In this case, multiple UAV terminals are relied on to exchange information in order to form a knowledge consensus on the data faced by all UAV edge terminals. For example, multiple UAVs can form a local view of the ground view, then obtain a common view of the global view by exchanging reasoning information with each other [26].

4. Distributed Cloud-Edge Collaborative Optimization Algorithm (DCECOA)

In order to effectively compress the UAV network model, this paper proposes a distributed cloud-edge collaborative optimization algorithm. The proposed algorithm uses scattered data and UAV terminals to prune the model, with the Taylor evaluation criterion used to prune the neural network parameters. Through this method, the network model can be effectively compressed and unnecessary parameters can be eliminated, thereby improving the operating efficiency and performance of the neural network. The schematic diagram of cloud-edge collaborative pruning is shown in Figure 2.

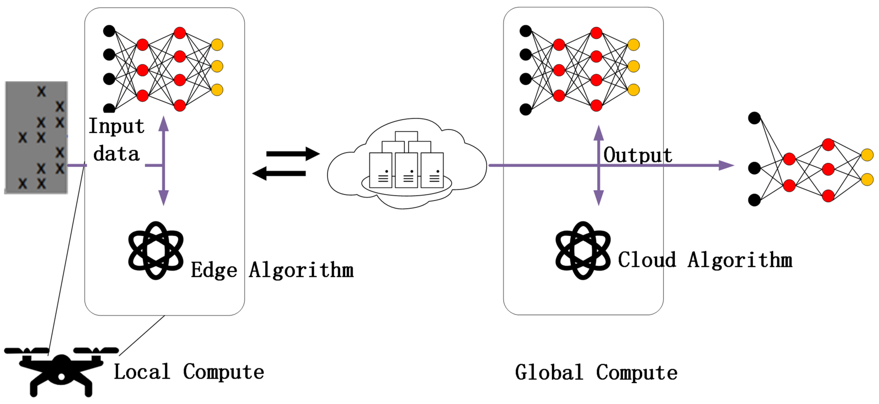

Figure 2. Illustration of cloud-edge collaborative pruning.

The architecture consists of a three-layer structure: edge terminals, edge cloud, and central cloud. The edge terminals (UAVs) locally compute and perform initial model pruning, while the edge cloud acts as an intermediate layer, aggregating the results from the edge terminals and performing further computations before forwarding the results to the central cloud for global optimization. The central cloud performs the final global computation and shares the updated model parameters with the lower layers. The diagram highlights how local, edge, and global computations interact in the DCECOA framework, allowing for efficient model pruning while minimizing communication overhead and computation requirements.

4.1. Principle of Distributed Cloud-Edge Collaborative Optimization Algorithm (DCECOA)

The distributed cloud-edge collaborative optimization algorithm decomposes the Taylor evaluation criterion into a cloud-side algorithm and an edge-side algorithm. It uses the fact that the model deployed on the edge terminal of the drone is consistent with the structure of all models on the cloud to ensure the invariance of the overall result of the Taylor evaluation criterion. The algorithm flow is shown in Algorithm 1, where M represents the number of edge cloud servers, K represents the number of terminals, T represents the number of original convolution kernels of the neural network model, F represents the upper limit of the pruning rate, P represents the number of convolution kernels to be pruned in each round of pruning, a represents the activation value of the convolution layer, g represents the gradient value of the convolution layer corresponding to a, B represents the batch size of the small-batch stochastic gradient descent algorithm of the drone terminal, f_ranks represents the calculated value of the Taylor evaluation criterion,

n_m represents the amount of data possessed by the edge cloud server, n_k represents the amount of data possessed by the drone terminal, and C represents the proportion of drone terminals participating in pruning in each round.

Algorithm 1 Distributed cloud-edge collaborative optimization algorithm

```
1:  procedure CLOUDEXECUTES($C, K, M, T, F, P$)                    ▷ Central Cloud
    Input:
        $C$: Proportion of edge terminals participating in each round
        $K$: Total number of edge terminals
        $M$: Total number of edge clouds
        $T$: Total number of convolution kernels
        $F$: Pruning ratio
        $P$: Number of convolution kernels pruned in each round
    Output:
        Updated global convolution kernel importance evaluation values
2:      $N \leftarrow \frac{T \cdot F}{P}$
3:      for iteration $t = 1 : N$ do
4:          $m \leftarrow \max(C \cdot K, 1)$
5:          $S_t \leftarrow$ random set of $m$ edge terminals
6:          for $m = 1 : M$ in parallel do
7:              $f\_ranks_t^m \leftarrow \text{EdgeUpdate}(n_k, n_m, S_t)$
8:          end for
9:          $f\_ranks_{t+1} \leftarrow \sum_{m=1}^{M} f\_ranks_t^m$    ▷ Global model convolution kernel importance
    evaluation values
10:         Search and delete the first P convolution kernels in $f\_ranks$ in the neural network
11:     end for
12: end procedure
13: procedure EDGEUPDATE($n_k, n_m, S_t$)                             ▷ Edge Cloud
    Input:
        $n_k$: Number of data samples owned by the $k$-th edge terminal
        $n_m$: Number of data samples owned by the $m$-th edge cloud
        $S_t$: Randomly selected edge terminal set for the current round
    Output:
        Convolution kernel importance evaluation values computed by the edge cloud
14:     $E_t \leftarrow$ edge terminal set of current edge
15:     $R_t = E_t \cap S_t$
16:     for each edge terminal $k \in R_t$ in parallel do
17:         $f\_ranks_k \leftarrow \text{EdgeTerminalUpdate}(B)$
18:     end for
19:     $f\_ranks = \sum_{k \in R_t} \left(\frac{n_k}{n_m}\right) \cdot f\_ranks_k$        ▷ Weighted average
20:     return $f\_ranks$ to server
21: end procedure
22: procedure EDGETERMINALUPDATE($B$)                                 ▷ Edge Terminal
    Input:
        $B$: Batch size for local training
    Output:
        Convolution kernel importance evaluation values computed by the edge terminal
23:     $\beta \leftarrow$ split local data into batches of size $B$
24:     $f\_ranks = 0$
25:     for batch $b \in \beta$ do
26:         $f\_ranks = f\_ranks + a \cdot g$          ▷ Local model convolution kernel importance
    evaluation values
27:     end for
28:     return $f\_ranks$ to edge
29: end procedure
```

First, the central cloud server selects $C \times K$ drone terminals to participate in the pruning process. Then, the central cloud server notifies the selected drone terminals to execute the side algorithm to obtain the activation value a and gradient value g of the convolution layer and calculate the evaluation value f_ranks of the convolution kernel. When the drone terminal has completed the calculation, it uploads the result of f_ranks to the edge cloud server to which it belongs. The edge cloud server performs a weighted average of the calculation results of the drone terminals within its jurisdiction and uploads the weighted average result to the central cloud server. After that, the central cloud server summarizes the results uploaded by all edge cloud servers and accumulates them into a single evaluation value of the convolution kernel of the global model.

The central cloud server then runs the cloud-side algorithm and adjusts the neural network structure based on the evaluation results of the convolution kernels. After the adjustment is completed, the neural network removes relatively unimportant P convolution kernels and returns the adjusted network to the terminals participating in the training.

The core idea of the distributed cloud-edge collaborative optimization algorithm is to entrust part of the algorithm that requires training data to the drone terminal for execution. This method can take advantage of the drone terminal's proximity to the data source and avoid large amounts of data transmission. At the same time, another part of the algorithm that relies on the calculation results of the drone terminal and does not require training data is executed by the central cloud server. In this case, only a small amount of drone terminal calculation data needs to be transmitted, and the pruning performance can reach and exceed the Taylor evaluation method based on centralized data optimization.

4.2. Distributed Cloud-Edge Collaborative Optimization Algorithm (DCECOA) Execution Process

The distributed cloud-edge collaborative pruning algorithm is divided into two main parts. First, the drone terminal executes the part called the edge-side algorithm. The edge-side algorithm uses the local data on the terminal to iteratively calculate the local model and obtain the importance evaluation value of the parameter by inputting the local data into the neural network.

The second part, called the cloud-side algorithm, is executed in the cloud center. The cloud-side algorithm summarizes the evaluation values for parameter importance uploaded from the edge terminal and derives the parameter importance evaluation value of the global model.

Finally, according to the parameter importance evaluation value of the global model, the convolution kernel that needs to be pruned is calculated to achieve the goal of streamlining the model.

5. Simulation Results

In this section, we focus on evaluating the performance of our proposed cloud-edge collaborative pruning algorithm and compare it with the common centralized pruning method. We use the previously described dataset and model architecture to verify the effect of cloud-edge collaborative pruning at different pruning rates. To better observe the overall trends, some experimental results are presented with smoothing to highlight general patterns while retaining the raw experimental data.

In terms of the experimental environment, a high-performance server equipped with an NVIDIA GeForce GTX 1080 Ti accelerator card was used as the cloud center server, three personal computers were used as the edge cloud server, ten drones equipped with low-power AI computers were used as edge terminals, and a Huawei S5720S-28P-LI-AC Gigabit Ethernet switch was used for networking. Python 3.6.13 programming was used to realize the DCECOA, socket programming was used to realize communication transmission, and the PyTorch 1.8.0 framework was used to assist in modification of the deep neural network structure. For simulations and analyses, we selected the ResNet-18 architecture due to its balance between computational complexity and suitability for resource-constrained UAV networks. Training and testing data were sourced from the CIFAR-10 dataset to ensure data quality and consistency across experiments. To avoid additional complexity, direct sensor data from the UAVs were not utilized.

The distributed cloud-edge collaborative optimization algorithm requires several key parameters to be set. First, the *prune_num* parameter is used to indicate the number of convolution kernels that need to be pruned between the terminal and the central cloud server in each round of communication, and it was set to 32. Next, the *prune_percent* parameter, which represents the percentage of the maximum number of convolution kernels that can be pruned out of the total number of convolution kernels, was set to 80%. The *iteration* parameter is used to indicate how many rounds of convolution kernel pruning the terminal and central cloud server need to perform in total, that is, the total number of communications between the two. This parameter needs to be dynamically calculated based on the initial number of convolution kernels of the model as well as *prune_num* and *prune_percent*. For a neural network model such as AlexNet, the calculated value of this parameter is 28. We set the number of central cloud servers to 1, the number of edge clouds to 3, and the number of drones to 10.

5.1. Pruning Optimization Only

In this section, we focus on evaluating the performance of our proposed cloud-edge collaborative pruning algorithm and compare it with the common centralized pruning method. We use the previously described dataset and model architecture to verify the effect of cloud-edge collaborative pruning at different pruning rates. In particular, we show the advantages of the algorithm in maintaining model accuracy and analyze the data transmission overhead incurred during the pruning process. Figure 3 shows the relationship between normal pruning and model accuracy obtained using the centralized pruning method at different pruning rates. The following trends are observed. When the pruning rate is low (less than 30%), the accuracy of the neural network fluctuates within a certain range and the accuracy decreases relatively little when the pruning rate is low. In the pruning rate range of 30% to 40%, the accuracy hardly fluctuates, and the accuracy on the test set remains at around 75%. This stable fluctuation in accuracy suggests that pruning has a positive impact on model performance. However, when the pruning rate exceeds 40%, the accuracy of the model drops sharply. It should be emphasized that these results only involve pruning of the convolution kernels; no subsequent neural network fine-tuning was performed. This shows that there are certain redundant parameters in the neural network and that the convolution kernels can be pruned according to the specific task to reduce network complexity.

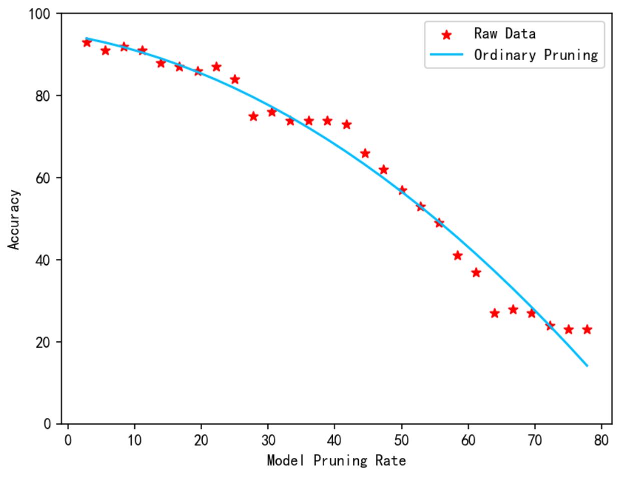

Figure 3. Ordinary centralized pruning.

Figure 4 shows the relationship between the results obtained with the cloud-edge collaborative pruning method at different pruning rates and the corresponding model accuracy. These experiments were conducted in a data-dispersed environment, focusing on the convolution kernel pruning of the neural network, and there was no fine-tuning of the model after pruning. It is worth emphasizing that the effect of the cloud-edge collaborative pruning method is almost the same as that of the centralized pruning method. When the pruning rate is lower than 20%, the accuracy decreases relatively little; in the pruning rate range of 26% to 40%, the model accuracy can be maintained at a stable level of about 80%. Within this range, the cloud-edge collaborative pruning method shows a clear performance advantage over centralized pruning. Only when the pruning rate exceeds 45% does the accuracy of the neural network model drop sharply.

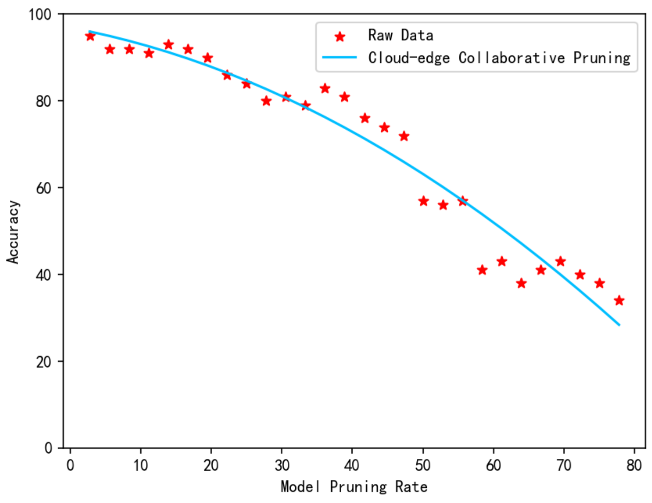

Figure 4. Cloud-edge collaborative pruning.

In comparing the performance of ordinary pruning and cloud-edge collaborative pruning, Figure 5 confirms that cloud-edge collaborative pruning consistently outperforms centralized pruning, with each curve in the figure representing the average of five trials. This result is particularly evident at higher pruning rates, where cloud-edge collaborative pruning shows better ability to maintain model accuracy. This highlights the potential and effectiveness of multilevel cloud-edge collaborative pruning algorithms in addressing the accuracy degradation issue commonly encountered during neural network pruning.

As can be seen from the data in Figure 6, a total of about 1.5 MB of data were sent during the entire pruning process, while the amount of data received reached 5 GB. This is because the multilevel cloud-edge collaborative pruning algorithm transmits the terminal's convolution kernel importance evaluation data for the local neural network model through the network. These data volumes are very small, and from a structural point of view are simple two-dimensional arrays. At the same time, the amount of data received is mainly affected by the terminal device downloading the global model from the central cloud server. Between each two communications, the terminal device needs to obtain the global model on the central cloud server to update its local model in order to support the subsequent edge-side collaborative optimization algorithm. Considering that edge devices usually have high downlink bandwidth and relatively low uplink bandwidth, the amount of data sent by the terminal device is negligible and can be ignored. Therefore, the effect of cloud-edge collaborative pruning is mainly limited by the downlink bandwidth of the terminal device. However, according to Figure 5 the actual pruning process only needs to reach a pruning rate of 40%. This is because the performance of the model becomes unfeasible as the pruning rate exceeds 40%. In this case, the terminal device needs to send about 3000 MB of data.

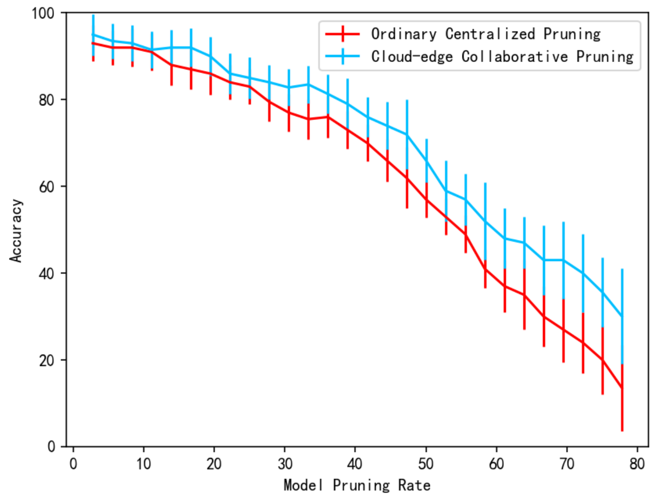

Figure 5. Ordinary pruning and cloud-edge synergistic pruning.

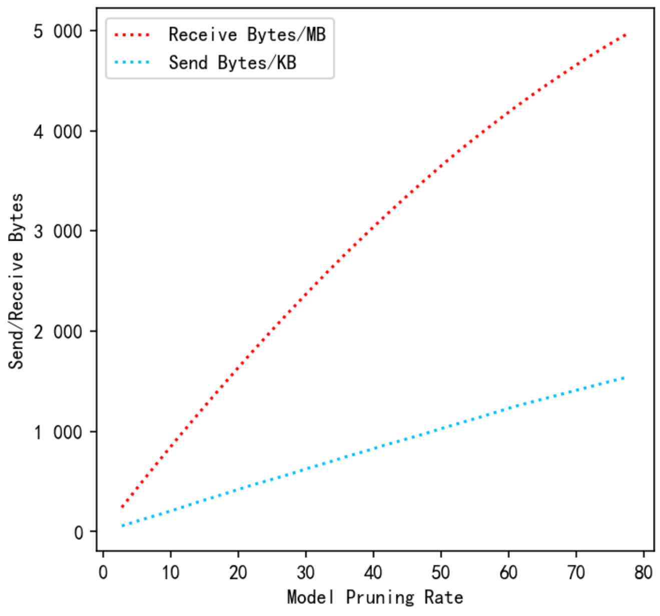

Figure 6. Cloud-edge collaborative pruning: sent and received bytes.

Figures 3 and 4 show a performance comparison between ordinary centralized pruning and cloud-edge collaborative pruning under different pruning rates. cloud-edge collaborative pruning maintains better accuracy at higher pruning rates, indicating that it has significant advantages in maintaining model performance. Figure 5 further confirms this point and emphasizes the improvement in pruning efficiency with the cloud-edge collaborative pruning method. At the same time, Figure 6 shows the overhead of cloud-edge collaborative pruning in data synchronization through a comparison of data transmission volumes. Although the communication overhead is high, the improvement in model performance makes it worthy of consideration.

5.2. Pruning and Fine-Tuning

The above experiment analyzed the impact of different pruning rates on the performance of neural network models, especially when only pruning was performed. The experimental results show that larger neural network models contain redundant information and that performance can be improved by removing some convolutional kernels. In comparing ordinary pruning and cloud-edge collaborative pruning, the performance of the cloud-edge collaborative pruning model is less affected by the pruning rate. The only cost is that the client must upload a small amount of data to the cloud center server and synchronizes the global model with the cloud center server.

To further analyze the impact of fine-tuning on the performance of neural network models after pruning, we first verified the changes in model performance after fine-tuning following ordinary pruning, then verified the changes after fine-tuning following cloud-edge collaborative pruning, and finally analyzed and compared the performance changes after fine-tuning with both ordinary pruning and cloud-edge collaborative pruning. Ordinary pruning is performed only on the server, while cloud-edge collaborative pruning requires the transmission of intermediate data needed by the cloud-side and edge-side algorithms over the network.

The experimental results for fine-tuning after ordinary pruning and cloud-edge collaborative pruning are shown in Figures 7 and 8; the horizontal axis represents the pruning rate, which is the percentage of the number of convolutional kernels pruned in the neural network relative to the initial number of convolutional kernels, while the vertical axis represents the accuracy rate of the neural network model on the test set, indicating the performance accuracy of the model.

After pruning the convolutional kernels in the neural network with ordinary pruning, a neural network training process was conducted to fine-tune the network parameters. As shown in Figure 7, the accuracy of the fine-tuned model is significantly improved and does not significantly decrease until the pruning rate approaches 50%. With a pruning rate of 50%, the accuracy on the test set can be maintained at about 80%. Without fine-tuning after pruning, the model's accuracy drops below 80% when the pruning rate exceeds 25%, indicating that fine-tuning significantly improves the performance of the pruned model.

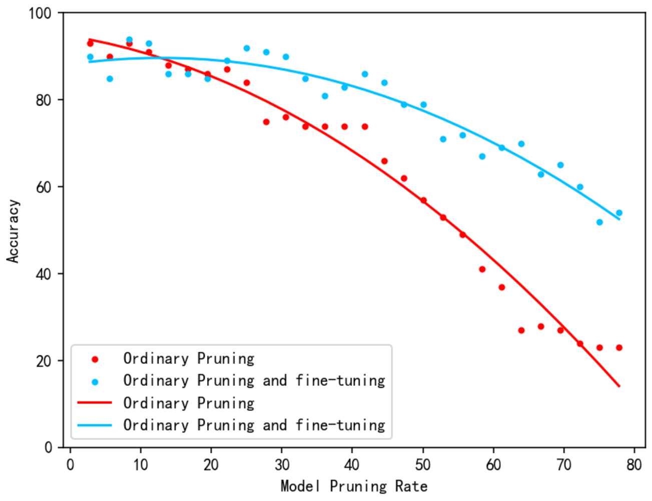

Figure 7. Ordinary pruning and fine-tuning.

After pruning the convolutional kernels with cloud-edge collaborative pruning, a neural network training process was conducted using the multilevel cloud-edge collaborative training algorithm. The impact of fine-tuning on cloud-edge collaborative pruning is essentially the same as that on ordinary pruning, with both increasing the upper limit of the pruning rate. However, the pruning rate for cloud-edge collaborative pruning can reach 60% while still maintaining an accuracy of 80% on the test set. Compared to ordinary pruning, the pruning rate is increased by 10%, as shown in Figure 8.

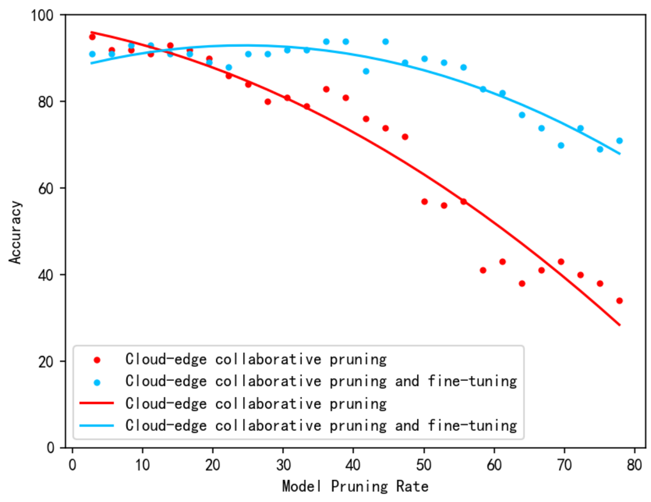

Figure 8. Cloud-edge collaborative pruning and fine-tuning.

The experimental results comparing the performance of ordinary pruning and fine-tuning with cloud-edge collaborative pruning and fine-tuning are shown in Figures 9 and 10, respectively along with the corresponding communication overhead. In these figures, the horizontal axis represents the pruning rate, which is the percentage of convolutional kernels pruned from the initial number of kernels. In the performance comparison between ordinary pruning and fine-tuning and cloud-edge collaborative pruning and fine-tuning, the vertical axis represents the accuracy of the neural network model on the test set, which is a key performance indicator of the model. In the communication overhead experiment for cloud-edge collaborative pruning and fine-tuning, the vertical axis shows the number of bytes (measured in MB) sent and received by the client.

Figure 9 shows the results of cloud-edge collaborative pruning and fine-tuning averaged over five experimental runs. The data demonstrate that cloud-edge collaborative pruning combined with fine-tuning achieves a higher pruning rate while maintaining the same level of model performance compared to centralized pruning methods. This indicates that the multilevel cloud-edge collaborative pruning algorithm not only surpasses ordinary centralized pruning algorithms in terms of pruning efficiency but also enhances pruning performance when paired with the multilevel cloud-edge collaborative training algorithms. Averaging of the results over multiple experiments ensures the robustness and reliability of the findings, providing a more accurate assessment of the algorithm's performance and its effectiveness in maintaining model accuracy despite higher pruning rates.

Judging from the results, the performance improvement appears modest. One of the primary tradeoffs of DCECOA is its focus on balancing model pruning with minimizing communication overhead, as opposed to solely maximizing the pruning rate. Simpler approaches such as standard centralized pruning may achieve higher pruning rates in static environments where computational resources and bandwidth are abundant; however, these methods often result in increased communication overhead due to the need for frequent model updates and centralized coordination. In contrast, DCECOA is specifically designed to function in UAV networks, where low-latency decision-making and efficient communication are critical, making it more suitable for deployment in dynamic and resource-constrained environments.

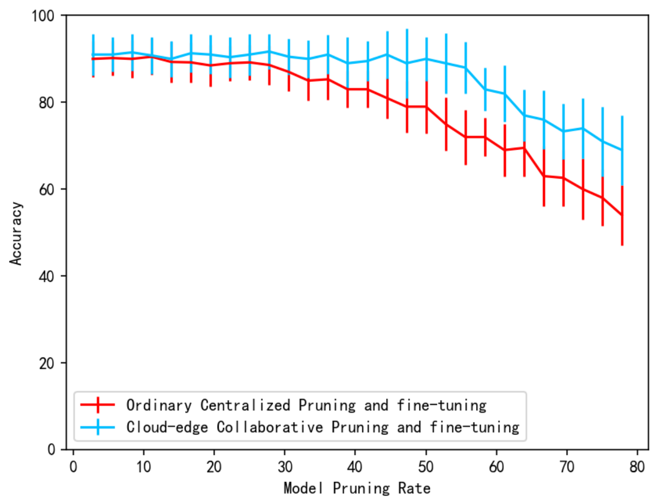

Figure 9. Performance comparison after fine-tuning

Figure 10 demonstrates that the number of bytes sent and received by the client increases significantly with both cloud-edge collaborative pruning and fine-tuning. This is because the fine-tuning process requires cloud-edge collaborative training, which involves synchronizing a large number of parameters and models, leading to increased data transmission. The client typically uploads only the gradient updates of the local model while downloading the entire model from the cloud center, resulting in a much larger amount of data being received than sent. Additionally, as shown in Figure 9, the model's accuracy decreases significantly when the pruning rate exceeds 50%. Therefore, the pruning rate should be kept within 50% to maintain high model performance. Under this condition, the client sends approximately 3000 MB of data and receives about 7000 MB of data.

The communication overhead results shown in Figure 10 show that the amount of data sent and received by the client increases significantly as the pruning rate increases. In particular, when the pruning rate exceeds 50%, the data transmission demand is significantly increased. For practical deployment in UAV networks, this data transmission requirement may pose a series of challenges.

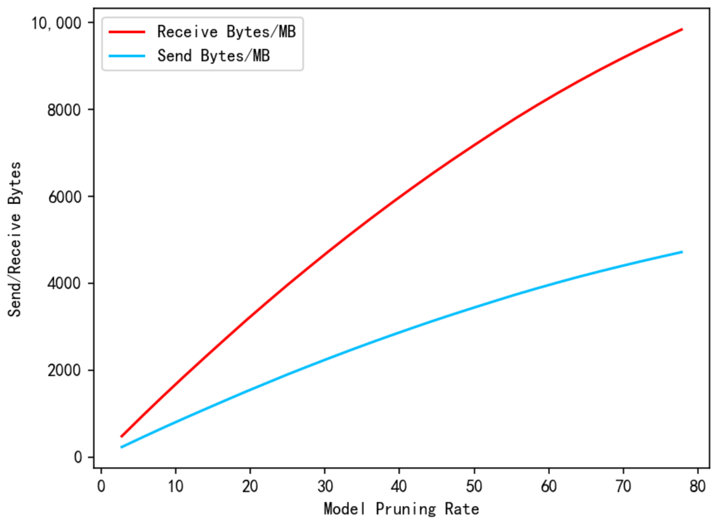

Figure 10. Cloud-edge collaborative pruning and fine-tuning.

UAV networks often operate in environments with limited bandwidth and unstable connections, especially over long distances or under adverse weather conditions. This means that the transmission of large amounts of data may lead to communication delays, data loss, and even system instability, potentially affecting the efficiency and performance of the entire model optimization process. Although cloud-edge collaborative pruning technology has certain advantages in model optimization, in the context of bandwidth-limited UAV networks it is necessary to weigh the pruning rate and communication overhead and to optimize the transmission strategy in order to ensure the feasibility and effectiveness of the model in actual deployment.

In this section, we assess the impact of fine-tuning on neural network models after pruning, comparing the ordinary and cloud-edge collaborative pruning methods. Figures 7–9 reveal that fine-tuning improves the accuracy of both methods, with cloud-edge collaborative pruning supporting up to 60% pruning while maintaining accuracy. However, Figure 10 shows that while cloud-edge collaborative pruning offers higher efficiency, it also leads to significant communication overhead, especially beyond a 50% pruning rate, posing challenges for deployment in bandwidth-constrained UAV networks.

6. Conclusions

The Distributed cloud-edge Collaborative Optimization Algorithm (DCECOA) proposed in this paper has demonstrated significant contributions in optimizing trained models in UAV networks. By pruning the model locally on the drone and introducing edge servers as relay nodes for federated learning, this method not only improves the adaptability of the model but also significantly reduces computational complexity and model volume. Experimental results show that DCECOA successfully reduces the transmission burden and computing requirements while transmitting only a small amount of communication data. This makes it especially suitable for edge smart devices with low power consumption and limited computing resources. In terms of innovation, the DCECOA method demonstrates strong innovation by performing local model pruning and optimization on UAVs. This multilevel cloud-edge collaboration architecture enhances model optimization efficiency, decreases communication frequency, and reduces communication overhead while also improving privacy protection to some extent.

Nonetheless, this study also has a number of limitations. First, the CIFAR-10 dataset is small; future research should verify the effectiveness of DCECOA on larger-scale real datasets. Second, the current research assumes a relatively uniform hardware configuration; however, in actual applications the hardware conditions may be inconsistent. In future research, attention should be paid to the performance of the algorithm in heterogeneous hardware environments. Additionally, while DCECOA shows progress in terms of the model compression rate, there is still a gap compared to some methods with high compression rate. Future research could incorporate the latest compression techniques to explore the balance between compression ratio and computational effort. Furthermore, although it was not the primary focus of this study, energy consumption is a critical factor in UAV operations. Future work should consider the energy efficiency of DCECOA and its impact on the overall operational efficiency of UAV networks. Expanding the size of the dataset, exploring privacy protection technologies, and investigating application scenarios in heterogeneous drone networks are other promising research directions.

Author Contributions: Conceptualization, J.Y. and J.T.; methodology, Q.Y.; software, C.W. and J.T.; validation, J.Y.; writing—original draft preparation, J.Y.; writing—review and editing, Q.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Guangzhou Key Research and Development Program (202206030003) and the National Key R&D Program of China (2020YFB1807700).

Data Availability Statement: All data are contained within the article.

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

Abbreviations

The following abbreviations are used in this manuscript:

UAV	Unmanned Aerial Vehicle
DCECOA	Distributed cloud-edge Collaborative Optimization Algorithm
AI	Artificial Intelligence
DNN	Deep Neural Network

References

1. Qu, Y.; Sun, H.; Dong, C.; Kang, J.; Dai, H.; Wu, Q.; Guo, S. Elastic collaborative edge intelligence for UAV swarm: Architecture, challenges, and opportunities. *IEEE Commun. Mag.* **2024**, *62*, 62–68. [CrossRef]
2. Gu, H.; Zhao, L.; Han, Z.; Zheng, G.; Song, S. AI-Enhanced Cloud-Edge-Terminal Collaborative Network: Survey, Applications, and Future Directions. *IEEE Commun. Surv. Tutor.* **2024**, *26*, 1322–1385. [CrossRef]
3. Gu, J.; Su, T.; Wang, Q.; Du, X.; Guizani, M. Multiple Moving Targets Surveillance Based on a Cooperative Network for Multi-UAV. *IEEE Commun. Mag.* **2018**, *56*, 82–89. [CrossRef]
4. Zhang, K.; Gui, X.; Ren, D.; Li, D. Energy–Latency Tradeoff for Computation Offloading in UAV-Assisted Multiaccess Edge Computing System. *IEEE Internet Things J.* **2021**, *8*, 6709–6719. [CrossRef]
5. Zhang, L.; Chakareski, J. UAV-Assisted Edge Computing and Streaming for Wireless Virtual Reality: Analysis, Algorithm Design, and Performance Guarantees. *IEEE Trans. Veh. Technol.* **2022**, *71*, 3267–3275. [CrossRef]
6. Zhang, T.; Xu, Y.; Loo, J.; Yang, D.; Xiao, L. Joint Computation and Communication Design for UAV-Assisted Mobile Edge Computing in IoT. *IEEE Trans. Ind. Inform.* **2020**, *16*, 5505–5516. [CrossRef]
7. Yang, Z.; Pan, C.; Wang, K.; Shikh-Bahaei, M. Energy Efficient Resource Allocation in UAV-Enabled Mobile Edge Computing Networks. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 4576–4589. [CrossRef]
8. Li, K.; Liu, C. Edge intelligence: State-of-the-art and expectations. *Big Data Res.* **2019**, *3*, 1–10.
9. Li, Z.; Li, H.; Meng, L. Model compression for deep neural networks: A survey. *Computers* **2023**, *12*, 60. [CrossRef]
10. Wei, L.; Ma, Z.; Yang, C.; Yao, Q. Advances in the Neural Network Quantization: A Comprehensive Review. *Appl. Sci.* **2024**, *14*, 7445. [CrossRef]
11. Liu, T.; Yang, C.; Suo, S.; Huang, Y. Edge Intelligence for Wireless Communication. *J. Signal Process.* **2020**, *36*, 1789–1803.
12. Cao, W.; Rui, J.; Li, M. A Survey of Neural Network Model Compression Methods. *J. Univ. Chin. Acad. Sci.* **2019**, *36*, 649–656.
13. Molchanov, P.; Tyree, S.; Karras, T.; Aila, T.; Kautz, J. Pruning convolutional neural networks for resource efficient inference. In Proceedings of the 5th International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
14. Kaplan, J.; McCandlish, S.; Henighan, T.; Brown, T.; Chess, B.; Child, R.; Gray, S.; Radford, A.; Wu, J.; Amodei, D. Scaling Laws for Neural Language Models. *arXiv* **2020**, arXiv:2001.08361.
15. Joshi, P.; Hasanuzzaman, M.; Thapa, C.; Afli, H.; Scully, T. Enabling all in-edge deep learning: A literature review. *IEEE Access* **2023**, *11*, 3431–3460. [CrossRef]
16. Dantas, P.V.; Sabino da Silva, W., Jr.; Cordeiro, L.C.; Carvalho, C.B. A comprehensive review of model compression techniques in machine learning. *Appl. Intell.* **2024**, *54*, 11804–11844. [CrossRef]
17. Howard, A.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
18. Tan, M.; Le, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; Proceedings of Machine Learning Research; Volume 97, pp. 6105–6114.
19. Molchanov, P.; Tyree, S.; Karras, T.; Aila, T.; Kautz, J. Pruning Convolutional Neural Networks for Resource Efficient Inference. *arXiv* **2016**, arXiv:1611.06440.
20. Han, S.; Mao, H.; Dally, W. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *arXiv* **2015**, arXiv:1510.00149.
21. Chen, T.; Frankle, J.; Chang, S.; Liu, S.; Zhang, Y.; Wang, Z.; Carbin, M. The lottery ticket hypothesis for pre-trained bert networks. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 15834–15846.
22. Wang, H.; Qin, C.; Bai, Y.; Zhang, Y.; Fu, Y. Recent advances onneural network pruning at initialization. In Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22) Survey Track, Vienna, Austria, 23–29 July 2022; pp. 5638–5645.
23. Xu, D.; Li, T.; Li, Y.; Su, X.; Tarkoma, S.; Jiang, T.; Crowcroft, J.; Hui, P. Edge Intelligence: Empowering Intelligence to the Edge of Network. *Proc. IEEE* **2021**, *109*, 1778–1837. [CrossRef]
24. Lalitha, A.; Shekhar, S.; Javidi, T.; Koushanfar, F. Fully decentralized federated learning. In Proceedings of the Third Workshop on Bayesian Deep Learning (NeurIPS), Montreal, QC, Canada, 7 December 2018; Volume 2.

25. Yanmaz, E.; Yahyanejad, S.; Rinner, B.; Hellwagner, H.; Bettstetter, C. Drone networks: Communications, coordination, and sensing. *Ad Hoc Netw.* **2018**, *68*, 1–15. [CrossRef]
26. Wang, Y.; Ge, Y.; Li, C.; Zhong, J.; Yang, Z. Multi-UAV Cooperative Autonomous Environment Exploration Strategy Based on Visual SLAM. In Proceedings of the 4th Underwater Unmanned Systems Technology Summit—Collaborative Technology for Manned/Unmanned Systems, National Harbor, MD, USA, 3–4 April 2021.