IEEE Open Journal of the
Computer Society

Received 15 May 2024; revised 9 August 2024; accepted 18 August 2024. Date of publication 22 August 2024; date of current version 4 September 2024. The review of this article was arranged by Associate Editor Peng Li.

Digital Object Identifier 10.1109/OJCS.2024.3447773

Stitch-Able Split Learning Assisted Multi-UAV Systems

TINGKAI SUN[1], XIAOYAN WANG [1] (Senior Member, IEEE), XIUCAI YE [2] (Member, IEEE), AND BIAO HAN [3] (Member, IEEE)

[1]Graduate School of Science and Engineering, Ibaraki University, Ibaraki 316-8511, Japan
[2]Institute of Systems and Information Engineering, University of Tsukuba, Ibaraki 305-8577, Japan
[3]School of Computer, National University of Defense Technology, Changsha 410073, China

CORRESPONDING AUTHOR: XIAOYAN WANG (e-mail: xiaoyan.wang.shawn@vc.ibaraki.ac.jp).

This work was supported in part by JSPS (Japan Society for the Promotion of Science) Grant-in-Aid for Scientific Research(C) under Grant 23K11080 and in part by ROIS NII Open Collaborative Research 24FA01.

ABSTRACT Unmanned aerial vehicles (UAVs), commonly known as drones, have gained widespread popularity due to their ease of deployment and high agility in various applications. In scenarios such as search missions and target tracking, conducting complex and computation-intensive tasks in multi-UAV systems have become essential. Recent investigations have explored the integration of collaborative centralized learning (CL) and federated learning (FL) into multi-UAV systems. However, CL methods raise privacy concerns and may suffer from communication delays, while FL methods demand high UAV-side computation capability. To address these challenges, split learning (SL) emerges as a promising alternative, offering reduced learning iteration time and improved accuracy in resource-constrained edge clients. In this study, we leverage SL and Stitch-able Neural Network (SN-NET) to propose a novel Stitch-able Split Learning (SSL) approach for multi-UAV systems. The proposed SSL approach is capable of tackling challenges in terms of device instability and model heterogeneity that associated in multi-UAV systems. Comparative simulations are conducted, evaluating its performance against CL, FL, traditional SL and SFLV1 (SplitFed Learning V1) approaches to establish its superiority.

INDEX TERMS Split learning (SL), federated learning (FL), unmanned aerial vehicles (UAVs), privacy preservation, distributed learning, model stitching.

I. INTRODUCTION

Unmanned aerial vehicles (UAVs), also known as drones or remotely piloted aircraft, have become more and more popular for their ease of deployment and high agility in various applications in recent years. In certain application scenarios, such as executing search and rescue missions or tracking targets, the multi-UAV system is required to conduct complex and computation-intensive tasks. To this end, incorporating centralized learning (CL) and federated learning (FL) methods [1], [2] into a multi-UAV system has recently been investigated. However, the CL based methods have the privacy concern and may suffer from high communication delay, and the FL based methods require high UAV-side computation capability. To address these challenges, split learning [3], [4] has emerged as a promising alternative to execute complex tasks in resource-constrained clients, proving to be useful in domain such as medical imaging.

Nevertheless, conducting distributed learning tasks within multi-UAV systems consistently presents significant challenges, particularly in relation to device instability and model heterogeneity. These challenges create substantial obstacles to implementing conventional distributed learning methods, such as Centralized Learning (CL), Federated Learning (FL), and Split Learning (SL), in multi-UAV systems. Even in SFLV [4], a recent method that attempts to combine the strengths of FL and SL, these issues remain unresolved.

To maintain the computational efficiency provided by SL while also enabling the independent learning capability of FL, there is a need to facilitate information transfer between different models. Model distillation [5] is frequently employed to transfer knowledge between dissimilar models, but when used on devices with limited computing resources, it can inadvertently exacerbate latency issues. In response, model stitching [25] has emerged as a technique that links neural networks with varying architectures or those trained using different strategies. Unlike knowledge distillation, stitching simply creates a mapping connection without iterating the model parameters themselves.

However, the strategic integration of model stitching within the SL framework remains an unresolved research question. More research is needed to explore its potential for enhancing distributed learning systems in multi-UAV environments.

In this study, to executing computation-intensive tasks on resource-constraint UAVs, we initially propose to employ the SL approach in multi-UAV systems. Subsequently, to address challenges related to device instability and model heterogeneity, we introduce an improved Stitch-able Split Learning (SSL) approach tailored for multi-UAV systems. This enhanced approach integrates SL methodology with Stitch-able Neural Networks (SN-NET). The contributions of this study are summarized below.

- We introduce a SL assisted multi-UAV system executing with computation-intensive tasks such as image classification. The system comprises multiple UAVs seamlessly orchestrated by a central base station (BS), each equipped with specialized computing, caching, and communication gear, in addition to an on-board camera for comprehensive area surveying. By employing SL method, we effectively alleviate the computational load on the UAV side and demonstrate superiority in total computation and communication time consumption.
- To address the challenges encountered by SL assisted multi-UAV systems, we propose a novel SSL approach which integrates SL and SN-NET. By leveraging model stitching technology, our approach could transfer respective knowledge between UAVs and BS effectively. The proposed SSL approach persists in training tasks even when the network is totally offline, and it can complete training and inference tasks with other UAVs that have heterogeneous models.
- We evaluate the effectiveness of our proposed multi-UAV system assisted by stitch-able split learning using an aerial perspective geographic dataset. The simulation results demonstrate that the SL-based system achieves high accuracy in image classification with low computation costs in an ideal network environment. Particularly noteworthy is SL's ability to handle data imbalances among multi-UAV systems, significantly outperforming FL-based methods. Furthermore, the proposed SSL approach demonstrates resilience in managing device instability and client model heterogeneity, leading to improved classification accuracy. These thorough evaluation outcomes emphasize the versatility and efficacy of our proposed system across various scenarios.

II. RELATED WORK

Federated learning (FL) has emerged as a preeminent distributed learning method in recent years, garnering widespread attention and extensive research efforts aimed at optimizing its performance. A plethora of works has explored diverse avenues for enhancing FL, encompassing the design of multi-tier FL frameworks to accommodate numerous devices [6], [7], and the development of model aggregation and compression techniques to mitigate communication overhead [8], [9]. Moreover, to support FL in dynamic wireless networks, pioneering research efforts have focused on tailoring resource allocation algorithms, accounting for communication link unreliability [10], [11] and emphasizing energy efficiency [12], [13]. Recent surveys on federated learning [14], [15], [16] have been conducted for further insight.

In contrast, researches on **split learning (SL)** are still in their early stages. The foundational concept of SL was initially introduced in [14]. SL has been applied across various distributed learning domains, including medical imaging [3], [17], IoT [18], and large-model distributed computing [19]. It holds the promise of facilitating more privacy-preserving machine learning by enabling organizations to train models without sharing raw data with external entities [20], [21]. As the improvement over SL, Cluster-based Parallel SL (CPSL) [23] has been presented, which could reduce training latency in a first parallel and then sequential training manner. AdaSplit [24] enables efficiently scaling SL to low resource scenarios by reducing bandwidth consumption and improving performance across heterogeneous clients. In contrast to FL, SL emerges as a promising method for building machine learning models across vast distributed networks.

The concept of **model stitching** was initially introduced by Lenc et al. [25] to explore the equivalence of representations. They demonstrated that the early segment of one trained network could be seamlessly connected with the final segment of another trained network using a 1×1 convolution stitching layer, resulting in minimal performance degradation. In a more recent study, Yamini et al. [26] extended this idea, revealing that neural networks, even with distinct architectures or trained using different strategies, could also be stitched together with negligible impact on performance. Concurrently, Adrian et al. [27] investigated model stitching as an experimental tool for aligning neural network representations. Their work demonstrated that common similarity indices (e.g., CKA [28], CCA [29], SVCCA [30]) were not correlated with the performance of the stitched model. Building on this, Pan and Cai et al. introduced SN-NET [31], a novel approach that unleashes the potential of model stitching as a general strategy for leveraging pretrained model families in the expansive model zoo. This enables the creation of a single scalable neural network at a low cost, capable of instantly adapting to diverse deployment scenarios.

III. MAIN CHALLENGES OF DISTRIBUTED LEARNING IN MULTI-UAV SYSTEMS

In this section, we elaborate the main challenges and the potential negative effects when employing traditional distributed learning methods in a multi-UAV system.

A. DEVICE INSTABILITY

In IoT applications, the presence of a myriad of devices characterized by variations in hardware specifications (CPU, memory), network conditions (4G, 5G, WiFi), and power resources (battery level) introduces substantial heterogeneity. This diversity extends to computing, storage, and communication capacities, thereby posing challenges in distributed learning, such as elevated communication costs, the emergence of stragglers, and the imperative need for fault tolerance.

The context of UAV settings exacerbates these challenges, particularly concerning wireless communication quality over increasing distances between the BS and the UAV. It hampers the distributed training of models by not only constraining the computing power of individual devices but also impeding progress through network latency. In FL, where synchronous updates are required, devices with limited computing capacities may become stragglers, prolonging the reporting of their model updates compared to other devices in the same iteration. Similarly, in the synchronous updates of SL, network delays impede the transmission of concatenated data at the cut layer, leading to queues of subsequent UAVs participating in learning, resulting in inefficient use of computing power.

Compounding these issues, participating devices may drop out of the learning process due to connectivity issues and energy constraints, thereby detrimentally affecting distributed learning outcomes. Given the prevalence of stragglers and faults arising from the inherent device instability in complex IoT environments, addressing these practical challenges are of paramount significance.

B. MODEL HETEROGENEITY

Within the original framework of distributed learning, a consensus among participating devices regarding a specific training model architecture is imperative to effectively derive the global model. This is achieved through the aggregation of model weights collected from local models or by transmitting the concatenated data and gradients.

However, in practical IoT applications, individual devices aspire to tailor their own models to adapt to their unique application environments and resource constraints, such as computing capacity. This need for model customization is particularly prevalent in multi-UAV systems comprising different types of UAVs. Moreover, concerns related to privacy may result in reluctance to share intricate model details. Consequently, the model architectures derived from distinct local models exhibit diverse configurations, rendering traditional FL methods impractical for straightforward aggregation. The inherent model heterogeneity prevalent in IoT environments has garnered significant research attention owing to its practical importance in the context of intelligent IoT applications.

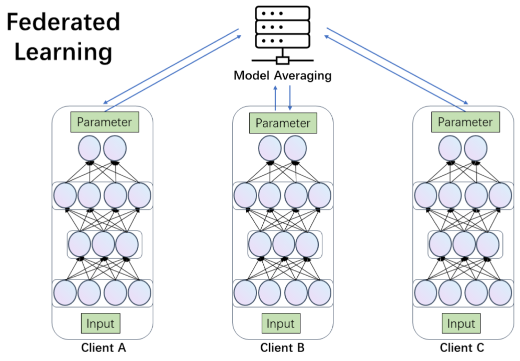


FIGURE 1. The model of federated learning.

Algorithm 1: Federated Learning.

```
Input: Initial model parameters $\theta$, number of iterations $T$
1: Initialize $\theta$ on a central server
2: for $t = 1, 2, \ldots, T$ do
3:     Distribute $\theta$ to participating parties
4:     for each party $i$ do
5:         Train model on local data and update model
           parameters $\theta_i$
6:         Send updated $\theta_i$ to central server
7:     end for
8:     Average updated $\theta_i$ to produce new global model $\theta$
9: end for
Output: Federated model parameters $\theta$
```

IV. PRELIMINARIES

A. FEDERATED LEARNING

FL is a distributed learning technique that allows models to be trained on multiple decentralized devices or edge nodes, without the need to share raw data with a central server. The model of FL and its basic algorithm are given in Fig. 1 and Algorithm 1, respectively. By training the model on decentralized devices, FL allows for privacy-preserving machine learning and can improve the accuracy of the model by leveraging a larger, more diverse dataset. Specifically, the global model are updated by averaging the local model updates from each device by (1), and the global model weights are then updated by adding the global updates to the current weights by (2). This process is repeated until the model has converged.

$$\Delta\theta_{global} = \frac{1}{n}\sum_{i=1}^{n}\Delta\theta_{local}^{i} \tag{1}$$

$$\theta_{global} \leftarrow \theta_{global} + \Delta\theta_{global} \tag{2}$$

B. SPLIT LEARNING

Unlike FL, SL divides a deep learning model into two parts via a cut layer: a client model and a server model. The client model is trained on data from a user's device, while the server model is trained on a centralized server. Through communication with each other, the client and server models can perform learning tasks together. This approach allows

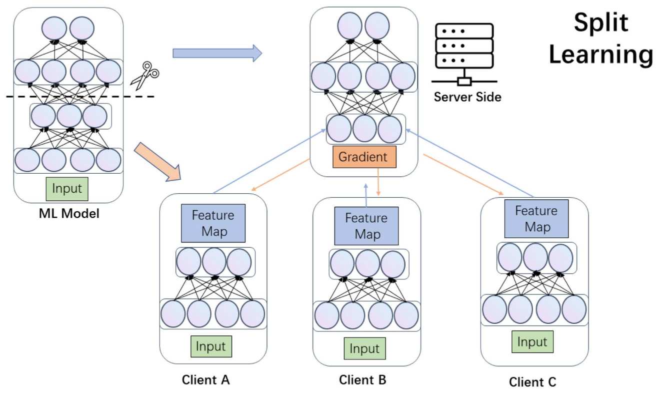


FIGURE 2. The model of split learning.

for privacy-preserving machine learning, and can also enable efficient model training and inference on resource-constrained devices.

The model of SL and its basic algorithm are given in Fig. 2 and Algorithm 2, respectively. Let D be the smashed data after passing through the client model, f be the model, and θ be the model's parameters, the process of SL can be outlined as follows. The SL divides the model into two parts, f_{client} and f_{server}, such that $f = f_{client} \circ f_{server}$. During the training process, the client performs forward propagation based on its local data and sends the smashed data D to the server. The server uses this data and the label to perform forward propagation and backward propagation, and sends the gradients of the cut layer to the client, which is used for client's backward propagation. This process continues until the model has converged, at which point the trained model, f_θ, is returned.

Split learning allows for privacy-preserving machine learning, as the users' data never leaves their devices. Meanwhile, the computation burden at the user side could be largely alleviated, since parts of the model is trained at server side.

C. STITCH-ABLE NEURAL NETWORKS

Stitch-able Neural Networks (SN-NET) [31] present a groundbreaking and versatile framework for elastic deep learning by harnessing pretrained model families from public model repositories through a process known as model stitching.

Inspired by the increasing availability of pretrained models in public repositories, where individually trained models often lack direct adaptability to dynamic resource constraints, as depicted in Fig. 3, SN-NET introduces stitching layers to seamlessly interconnect an array of pretrained models. This results in diverse stitched networks that permit real-time network selection.

A key attribute of SN-NET is its remarkable flexibility, allowing the interchangeable replacement of the two-party models involved in the stitching process. This flexibility empowers the choice of a server-side model finely tuned for specific tasks within a distributed learning system, thereby enhancing performance potential. Once an appropriate larger

Algorithm 2: Split Learning.

```
 Input: smashed data$D$, parameter $\theta$, model $f$
 1: Initialize $f_\theta$ with random weights
 2: while not converged do
 3:     Forward propagation:
 4:     The client performs forward propagation based on
        its local data and sends the smashed data $D$ to the
        server.
 5:     The server obtains loss $l$ through forward
        propagation.
 6:     Backward propagation:
 7:     The server executes backward propagation and
        sends the gradients $\Delta\theta_{server}$ of the cut layer to the
        client.
 8:     The client updates $\theta_{client}$ based on the gradients
        $\Delta\theta_{server}$
 9: end while
10: Return $f_\theta$
 Output: Trained model $f_\theta$
```

model is established, the selection of smaller models becomes equally unrestrained.

Following the SN-NET algorithm, a small client model could be stitched with a large server model. Here, multiple anchors are strategically established to ensure seamless model performance. From this array of anchor points, a suitable anchor is selected for each client, serving as a fixed reference point throughout the model stitching process. This connection will sew the two different models together like a sliding zipper and selected anchor point plays a crucial role in subsequent stitching calculations, contributing to the overall stability and efficacy of the model stitching procedure.

V. STITCH-ABLE SPLIT LEARNING ASSISTED MULT-UAV SYSTEMS

A. SYSTEM MODEL

Fig. 4 illustrates a multi-UAV system for image classification tasks in the exploration of geographic areas. The BSs are connected to UAVs within their respective operational ranges, and acted as coordinators for the UAVs. Each UAV is equipped with a transceiver, caching device, and processor, facilitating seamless communication with the BS through wireless connections. The network topology adheres to a star configuration, ensuring direct connectivity between all UAVs and the BSs. The exploration area is subdivided into smaller subareas, each assigned to UAVs equipped with onboard cameras. The primary mission of the UAVs is to identify specific objects like airports, parking lots, and facilities. The size of each sub-area is determined by the optical camera's resolution requirements for precise image capture. To optimize coverage and efficiently capture images during mission execution, we assume that multiple UAVs are strategically deployed across diverse locations to cover a larger expanse of sub-areas.

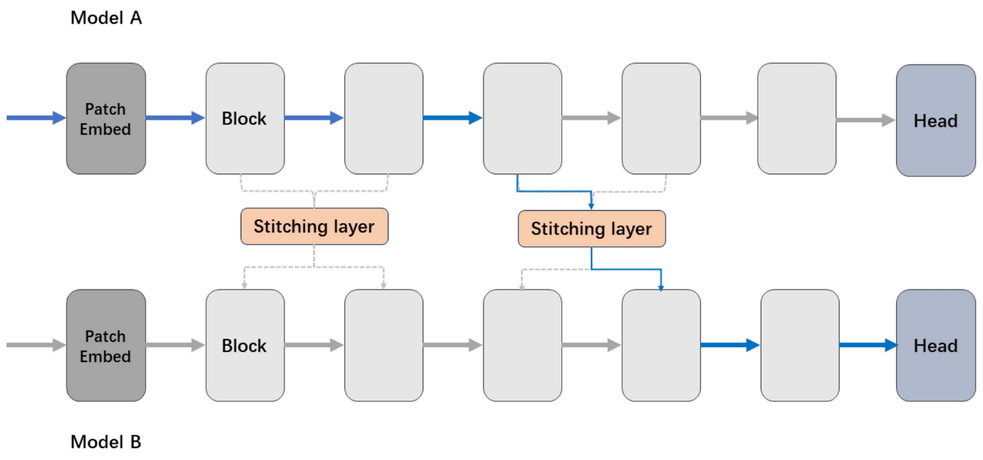


FIGURE 3. **llustration of the stitchable neural network, where 2 pretrained models are connected with simple stitching layers (1 × 1 convolutions).**

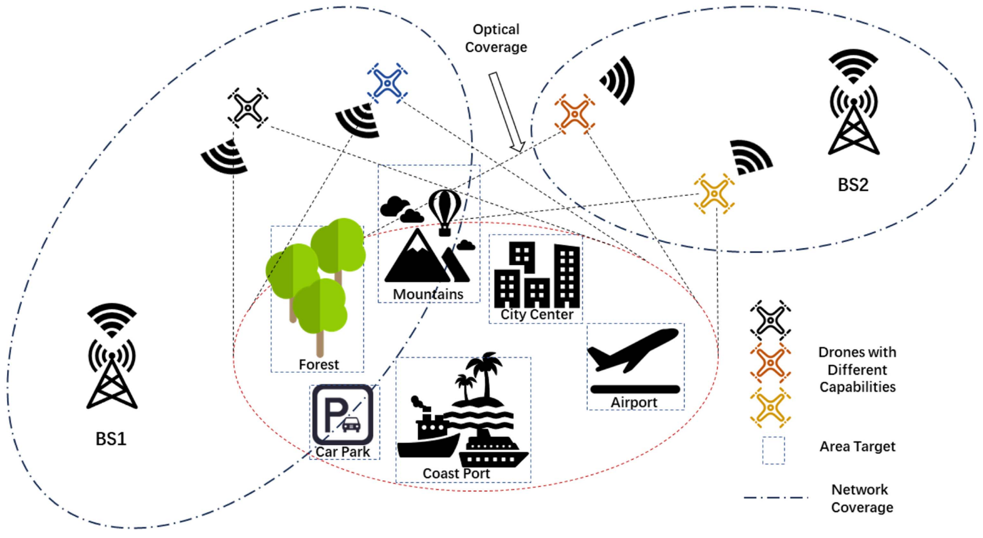


FIGURE 4. **Illustration of a multi-UAV system for image classification tasks in a geographic area exploration scenario.**

B. SPLIT LEARNING ASSISTED MULTI-UAV SYSTEMS

SL emerges as a viable solution to tackle the challenge of training deep learning models within resource-constrained UAVs, under the coordination of the BS. Fig. 5 visually represents the SL assisted multi-UAV system's operational framework, where a Convolutional Neural Network (CNN) dedicated to image classification undergoes division into client-side and server-side models, executed at the UAVs and BS, respectively.

The UAV's local training dataset comprises images stored in its cache, serving as inputs for forward propagation within the client model. Subsequently, the generated smashed data at the cut layer, coupled with corresponding target labels, is transmitted to the BS. The BS leverages this data for both forward and backward propagation of the server-side model. Consequently, the gradients of the cut layer are transmitted back to the UAV for its backward propagation. Through this distributed architecture, SL facilitates the UAVs in efficiently training models with the aid of the BS. The division of the CNN model into client-side and server-side models ensures that the resource constraints of the UAVs do not impede their participation in training deep learning models. The smashed data at the cut layer transmitted to the BS, and the gradients transmitted back to the UAV signify a bidirectional exchange of information, fostering mutual learning between the client and server models. This iterative knowledge exchange, orchestrated through SL, underscores its efficacy in enabling UAVs to partake in complex deep learning tasks despite their inherent resource limitations.

C. STITCH-ABLE SPLIT LEARNING ASSISTED MULTI-UAV SYSTEMS

SL is capable of training deep learning models on resource-constrained UAVs, with coordination from the BS. Nevertheless, it is crucial to underscore that SL assumes that the machine learning model is unique and the connection between client and server is stable. This assumption implies that SL cannot deal with heterogeneous models and lacks the autonomous learning capabilities exhibited by FL when the connection between client and server is poor. In response to this limitation, we propose an enhanced SSL assisted multi-UAV systems in this subsection.

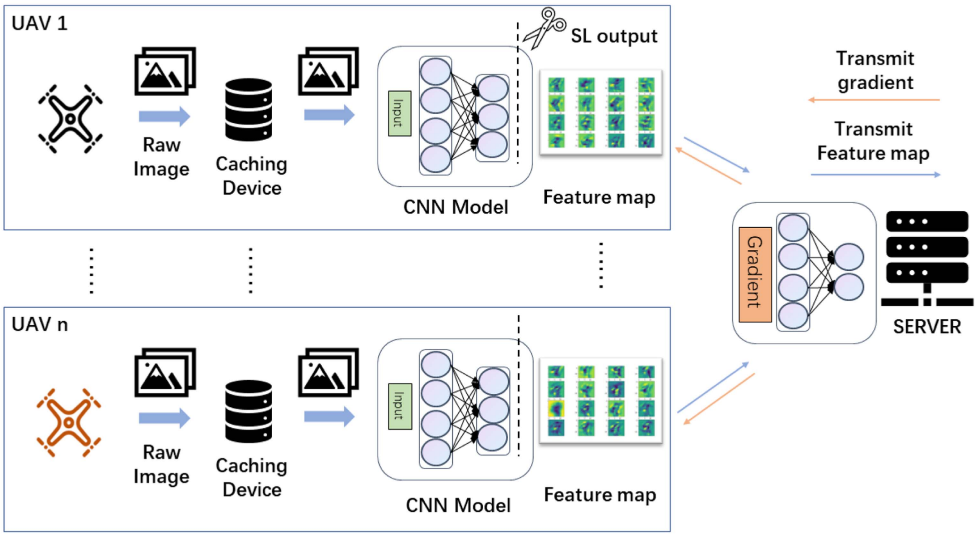


FIGURE 5. Illustration of SL assisted multi-UAV system for image classification tasks.

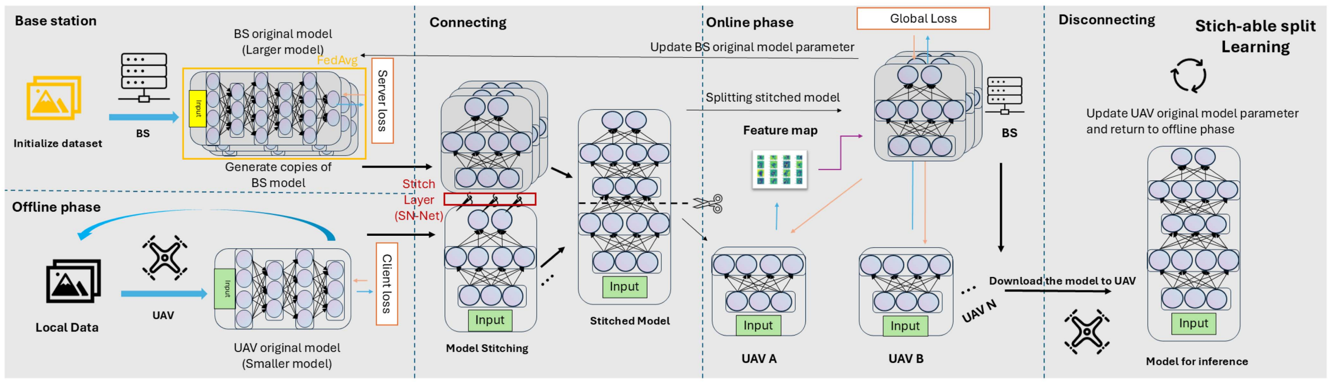


FIGURE 6. Process of the stitch-able split learning assisted multi-UAV systems.

We observed that in SL, a unified server-side model is trained through the aggregated data uploaded by each client. This segment of the model is then amalgamated with the client-side model retained by each client, forming a complete model. Conversely, FL employs a full-size model on the client and delivers an entirely new model through server-side averaging. We find that treating the models trained by these methods as pre-trained models could allow the system to dynamically switch training strategies through the utilization of model stitching technology.

To address these challenges posed by device instability and model heterogeneity, we incorporate the model stitching method SN-NET with SL in this study, and propose a novel Stitch-able Split Learning (SSL) approach for multi-UAV systems. To put it simply, our basic idea is to stitch a better-performing model deployed on the BS with a local trained model on the UAV, and generate a model of in-between sizes with higher accuracy which is trained through SL to reduce the computation load on the UAV. Specifically, the proposed SSL approach is divided into four phases, which is depicted in Fig. 6. The algorithm of the proposed SSL approach is given in Algorithm 3.

1) OFFLINE PHASE

Initially, the SSL algorithm is initialized with all participating devices, i.e., UAVs and the BS independently generate new models of varying sizes. During this phase, the BS strategically opts for a larger model deemed more suitable for the specific task, while the UAV's model exhibits flexibility, necessitating effective iteration capacity on client devices. To align with real-world application scenarios, the server-side model is initialized using a constrained set of existing datasets, serving as a pre-trained model.

After the initialization, similar to the procedures in FL, the UAV trains the local model based on locally collected data. Throughout this training process, the UAV flexibly selects the range of image data collection based on area size and endurance.

2) CONNECTION PHASE

Upon multiple UAVs establishing connections with the BS, copies of the existing BS-side models are instantiated on the BS. These copies operate in parallel with different UAVs' models and are updated at fixed intervals by employing FedAvg algorithm. Following the creation of BS model replicas,

Algorithm 3: Stitch-Able Split Learning.

```
Input:data$D$, smashed data$D_s$, parameter $\theta$, client model
  $f_c$, server model $f_s$
 1: Initialize $f_c$ and $f_s$ with random weights
 2: Train several times $f_s$ on small dataset similar to data
    $D$
 3: Server waiting for connection
 4: if Client is offline then
 5:     Train $f_c$ locally until connected to server
 6: else if Client is online then
 7:     Server check number of clients participating $N$
 8:     Create $N + 1$ th copies of the model $f_s$ $f_1 \sim f_{N+1}$
 9:     Client update $f_c$ to server
10:     Execute SN-NET algorithm to stitch $f_c$ and $f_s$
11:     if First connection then
12:         Select a anchor as an SNNET fixed anchor $ANC$
13:     else
14:         Use fixed anchor $ANC$ to stitch model $f_c$ and $f_s$
            and generate a stitched model $f$
15:     end if
16:     Select cut layer according to client situation and
        split the model $f$ and send the first half of the model
        to client
17:     while Connection do
18:         Forward propagation:
19:         Client performs forward propagation based on its
            local data $D$ and sends the smashed data $D_s$ to
            server.
20:         Server use $D_s$ to obtains loss $l$ through forward
            propagation.
21:         Backward propagation:
22:         The server executes backward propagation and
            sends the gradients $\Delta\theta_{server}$ of the cut layer to the
            client.
23:         The client updates $\theta_{client}$ based on $\Delta\theta_{server}$
24:         Iterate model $f$
25:         Update model $f_{N+1}$ by $f$ parameters.
26:         Average model $f_1$ to $f_{N+1}$ of each client to update
            participation part of server model $f_s$
27:     end while
28: end if
29: Return $f$ for for inference
Output: Trained model $f$
```

the UAV initiates a connection to the BS, simultaneously uploading its local model. This local model undergoes transmission to the BS and stitch with a server-side model's replica using the SN-NET algorithm. For each client and server replica, an SN-NET anchor is strategically selected based on the UAV's computation time consumption, this anchor point determines the relative position of the model during stitch and serves as a fixed reference point for all subsequent stitching calculations. This stitching process does not actually produce a brand new model, but changes the connection relationship through the SN-NET anchor to connect the two parties together, so all part of the original model involved in this process will be updated during the training process.

As depicted in Fig. 6, by leveraging the SN-NET algorithm we combine server-side and client-side models to generate a new model with a size between them. Notably, in contrast to FL, which mandates uniformity among client models, this step enables the system to smoothly accommodate the coexistence of heterogeneous models.

3) ONLINE PHASE

During the online phase, the previously stitched model is split according to the principles of the Split Learning (SL) method, reducing the computational load on the client and enabling drones to train new global models more efficiently. Unlike the automatic generation of anchor points by SN-NET, this splitting process requires manual specification. While model stitching provides flexibility, it is essential to have a solid understanding of the model's architecture before selecting the client model, in order to correctly determine the cut layer. After the data is processed at the cut layer, it is transmitted to the base station (BS) along with the corresponding target labels. These inputs are then used in both the forward and backward propagation processes of the server-side model. Ideally, all UAVs would participate in this phase, engaging in continuous training until a superior-performing global model is achieved.

It is crucial to note that, particularly when using a CNN in this system, the model should be split after the pooling layer or any layer responsible for reducing data dimensions. This approach is recommended to minimize the network traffic generated during data transmission.

4) DISCONNECTION PHASE

Finally, the UAV downloads the stitched model from the BS and updates the UAV's original model based on it, and returns to offline phase. This model is the final model and is only used for inference while the UAV is performing its task.

If we incorporate traditional methods into these four stages for comparison, we obtain Table 1. This table provides a comparison of the operations performed by CL, FL, SL, and SSL at different connection stages.

VI. EVALUATION RESULTS

This section presents the evaluation results for the proposed SL and SSL assisted multi-UAV system for image classification task.

A. DATASET

We use the aerial image dataset (AID) [22] to evaluate the performance, which is a large collection of aerial images that have been sourced from Google Earth. These images have undergone post-processing using RGB renderings of the original optical aerial images, but have been found to have no significant difference from the original one when it comes to pixel-level land use mapping. Noted that the AID dataset is widely used for evaluating scene classification and object identification algorithms.

TABLE 1. Comparison of CL, FL, SL, and SSL in Different Phases

	CL	FL	SL	SSL(Our proposed)
Offline	collect data	collect data and training	collect data	collect data and training
Connect	upload data	upload model	upload smashed data	upload smashed data and model
Online	N/A	N/A	training model	stitching model and training
Disconnect	download model	download model	download model	download model

FIGURE 7. Aerial image dataset: some examples.

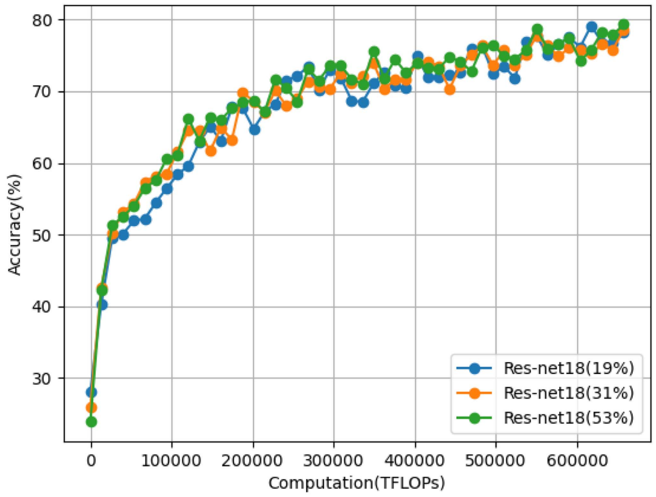


FIGURE 8. The impacts of different cut layers.

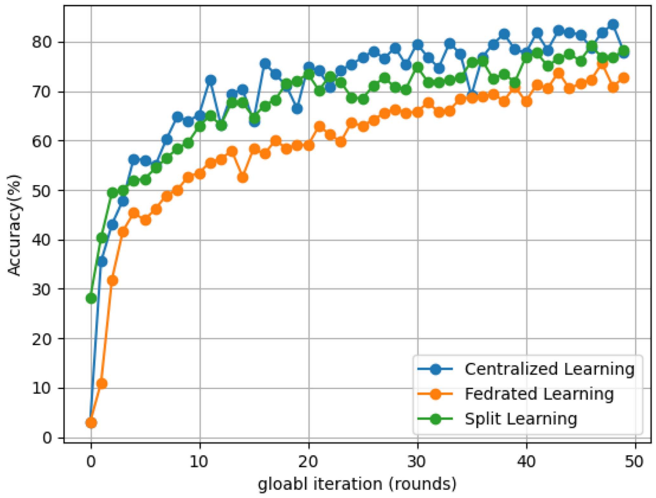


FIGURE 9. Classification accuracy on IID dataset in a system with 5 UAVs.

The dataset consists of 30 different scene classes, each of which includes a range of 200 to 400 images that are 600×600 in size. These images have been carefully labeled by experts in the field of remote sensing image interpretation, ensuring that they are accurately classified and easy to work with. Some examples of the different scene classes are shown in Fig. 7, to provide a glimpse into the diverse range of aerial images included in the dataset.

B. PERFORMANCE OF SL ASSISTED MULTI-UAV SYSTEMS

1) THE IMPACT OF SL CUT LAYER

In our simulations, A ResNet18 CNN model is applied, and the AID dataset [22] with 1×10^5 samples are used. By taking into consideration the limited computation capability of each UAV, we first evaluate the impact of SL's cut layer on the classification accuracy. We designate the 3rd, 5th and 9th convolutional layers of ResNet18 as cut layers to make client's model size approximate to 19%, 31% and 53% of the whole model.These layers are located after the pooling layer and can effectively reduce the size of the data. As shown in Fig. 8, we observed that the choice of split size does not significantly impact the convergence speed of the model, and all models achieves a high level of accuracy finally. To keep the model retained by the UAV as compact as possible, we choose to cut the first block of ResNet18, i.e., 19%, as the client-side model in subsequent simulations.

2) PERFORMANCE ON IID DATASET

Next, we evaluate its performance on IID (Independent and Identically Distributed) dataset by comparing with FL and CL based methods. Specifically, we assume that the number of UAV is 5, and thus 1×10^5 samples are partitioned into 5 subsets as IID dataset, i.e., each UAV holds 2×10^4 local training samples. For the CL, the locally captured images at all UAVs are directly transmitted to the BS, where a unique CNN model is trained for image classification. In FL, an equal-weight parameter setting for fedAVG is adopted with $\lambda_1=\lambda_2=\lambda_3=\lambda_4=\lambda_5=1$ for the sake of simplicity. The image classification accuracy of SL, FL and CL varying with number of iterations is illustrated in Fig. 9. We can observe that as the number of iterations increases, the classification accuracy improves for all methods. Our SL based approach shows faster convergence in the initial stages with an overall accuracy difference of less than 5% compared with CL. Meanwhile, FL

TABLE 2. **Computation and Communication Delay Comparison**

Evaluation Standard	**Method**			
	SL(19%)	***SL(31%)***	***FL***	***Baseline(CL)***
Bandwidth	5.8 GB	2.9GB	584.5MB	2.7GB
Transmission Delay	16.95sec	8.56sec	1.67sec	7.89sec
Client MACs	353.64 M	586.26M	1.82 G	≈0
Computation Delay	49.97sec	82.70sec	263.05sec	≈0
Server MACs	1.47G	1.24G	≈0	1.82 G
Computation Delay	88.9sec	75.48sec	≈0	108.7sec
Total Duration	155.82sec	166.74sec	264.72sec	116.59sec

achieves an approximately 73% accuracy, which is slightly inferior to the SL and CL methods throughout the same number of iterations.

Next, Table 2 compares the communication and computation delay of SL, FL and CL based methods in one local iteration. In our simulation, we calculate the bandwidth consumption of each method by assuming a WiFi5 protocol for wireless connection. We use a workstation with an A6000 GPU as the server, and a notebook with an RTX 3070laptop GPU as the client, to simulate the computation delays. Specifically, the size of the CNN model is measured in MACs (Multiply-Accumulate Operations), and all computation delays are actually measured using our machines.

As shown in Table 2, the bandwidth of the SL method is significantly higher than that of the FL method. For instance, the SL with 19% model on UAV consumes a bandwidth of 5.8GB which takes 16.95 seconds to complete the transmission in a WiFi5 network setting, which is even longer than transmitting the original image dataset to the BS. Notice that, this result is based on original image size with 600×600 . The bandwidth consumption for CL will increase if the original image size becomes larger. On the other hand, we could confirm that SL can significantly reduce computation delays at UAVs compared to FL. Notice that in the experiment, we used a high-performance GPU notebook as the client, while in reality UAVs may not have such computing resources, which will result in much longer computation delay.

Overall, due to the reduction in client's computation time, the total delay of SL is substantially lower than that of FL. This gap is likely to be substantially larger in lower-performance UAVs. Although the CL method has the shortest duration, we should notice that it could not preserve the privacy of the UAVs, and its delay will largely increase when the high-resolution image is transferred. In summary, thanks to the advances in wireless communications, we notice that the communication delay is only a small portion of the total delay compared to the computation time at UAV. In summary, SL can effectively reduce the local computation time and thus achieve a lower total learning time.

3) PERFORMANCE ON NON-IID DATASET

We evaluate the performance of the proposed SL-based approach on a non-IID dataset, i.e., each UAV could only gather specific classes of data samples. In the non-IID setting, each UAV only has 6 classes of data among total 30 classes. We use 20% of the samples as test data to characterize the classification accuracy. The accuracy of the image classification on non-IID dataset is illustrated in Fig. 10. We compare SL and FL' accuracy by setting the number of local epoch at 1 and 5. It is obvious that the performance of FL method degrades drastically on a non-IID dataset, and the label distribution skew also causes client's model to over-fit. Compared with FL, SL is only slightly affected by the non-IID dataset, and it can attain a higher accuracy rate by increasing the number of local epoch. We can conclude that SL performs well when handling imbalanced data, and it requires less data in the early stages of training than FL.

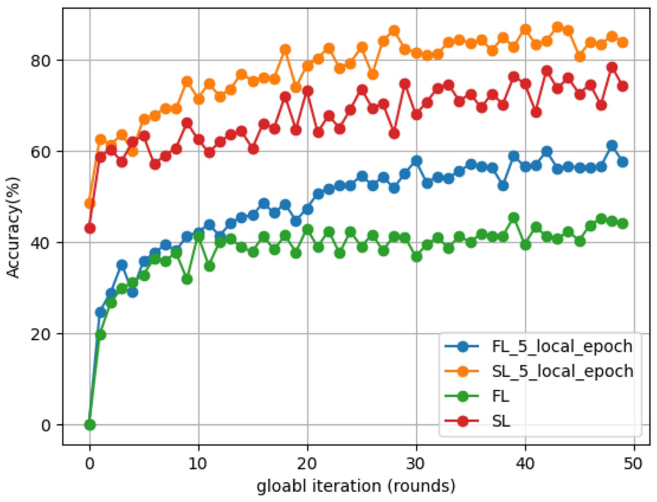


FIGURE 10. **Classification accuracy on non-IID dataset.**

C. PERFORMANCE OF SSL ASSISITED MULTI-UAV SYSTEMS

1) THE IMPACT OF STITCHING PROCESS

In our simulations, we employ the ResNet18 model as the small model on the UAV side, which is stitched with a ResNet50 model on the BS side initialized with a small sample dataset. Initially, we conducted a preliminary experiment in which both models were fully trained on the AID dataset. Subsequently, utilizing the SN-NET algorithm, we seamlessly stitch these two models and evaluate the accuracy across various SN-NET anchors on the test set. Fig. 11 illustrates the accuracy achieved with different stitching anchors. These anchors function as flexible connection points, allowing the network to be effectively connected at various positions. Notably, as the stitched model size increases, there is a corresponding improvement in accuracy. This observation highlights the minimal loss in accuracy during the stitching process and provides a strong foundation for our subsequent experiments.

2) PERFORMANCE ON IID DATASET

In this experiment, for split the stitched model in online phase, we still use the pooling layer(first block of ResNet18) as the boundary to split the model. We compare the image classification accuracy of SSL, SFLV1 (SplitFed Learning V1) [4] and traditional SL, and depict the results in Fig. 12. SFLV1, as the primary comparison subject, represents an early enhancement approach for split learning and has gained considerable

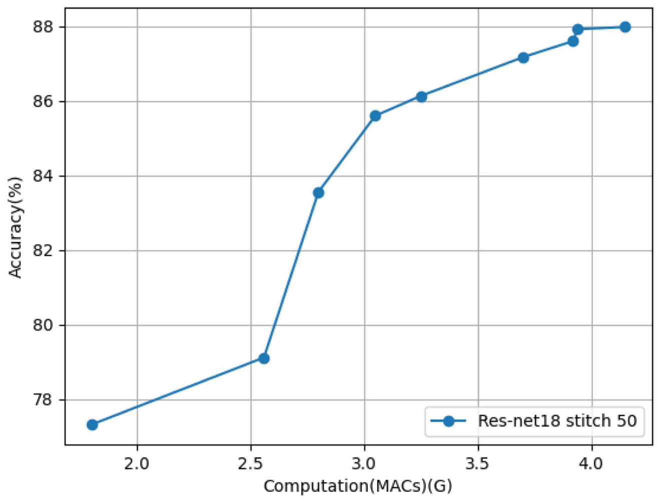


FIGURE 11. Effect of stitching model, demonstrating the accuracy of stitching with different anchors.

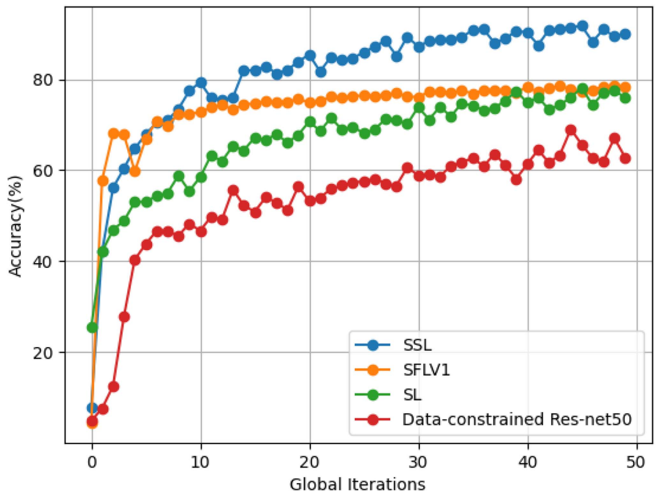


FIGURE 12. Classification accuracy versus the number of global iterations.

traction in the distributed learning domain. It integrates the federated learning algorithm, transmitting all models upon connection establishment, and averaging the models at designated intervals. SFLV1 expects to demonstrate a commendable convergence speed and accuracy in this evaluation. While the convergence speed of our SSL during the initial training phase may not match that of SFLV1, it demonstrates superior accuracy than SL and SFLV1 after convergence.

In order to ascertain the effect of the sub-dataset used to initialize the BS-side model, which was derived from a small portion of the AID dataset, we conducted an additional evaluation. Specifically, we assessed the performance of the ResNet50 model trained solely on this subset of the AID by testing it on the AID dataset. This evaluation aimed to provide insights into the influence of the sub-dataset on the accuracy of our proposed SSL method. The noticeable difference in

TABLE 3. Communication Comparison of Single UAV

Communication Bandwidth	Connection States			
	Online	*Offline*	*Connection*	*Disconnection*
SSL	138MB~1.08GB	0	58.4MB	75.9MB~153.7MB
SL(31%)	552 MB	N/A	0	40.3MB
FL	0	0	58.4MB	58.4MB
SFLV1(31%)	552 MB	N/A	58.4MB	58.4MB

TABLE 4. Computation Comparison of UAV and BS

Evaluation Standard	Method			
	SSL	*SL(31%)*	*FL*	*SFLV(31%)*
UAV-side MACs	353.64M~ 931.84M	586.26M	1.82G	586.26M
BS-side MACs	1.66G~3.61G	1.24G	≈0	1.24G

accuracy between this data-constrained model and SSL underscores the effectiveness of our innovative approach. This efficacy is evident in the proficient iteration of the server-side model, enabling the assimilation of knowledge from the data contributed by the client.

Tables 3 and 4 present the communication and computation overhead incurred by a UAV in our experimental setup, where the client and server utilize ResNet18 and ResNet50 models, respectively. As delineated in Table 3, the communication overhead of SSL during both the online and the disconnection phases varies depending on the configuration of SSL during stitching and splitting. This variability is also evident in the computation overhead depicted in Table 4. Notably, Table 4 indicates that SSL imposes a heavier computational burden on the server side compared to other methods, while still preserving the characteristic of SL whereby the computational load on the client side remains minimal. This attribute contributes to the enhanced accuracy observed in SSL.

Overall, the communication overhead of SSL is comparable to that of SL and SFLV1. Furthermore, given that UAVs in the SSL system can autonomously learn in offline states similar to FL, the bandwidth requirements of SSL fall between FL and SL when certain UAVs are offline.

3) IMPACT OF DEVICE INSTABILITY

In this section, we conduct an evaluation of the impact of device connectivity status on SSL with 5 UAVs. Through simulations, we analyze how the performance of SSL is affected by the presence of different numbers of online UAVs. Furthermore, we show the performance of FL utilizing ResNet18, equipped with offline independent learning capabilities, when all 5 clients are online, to provide a comparative analysis.

Specifically, each client remains online for an equal duration, with a corresponding number of clients randomly selected to establish connections during the simulation. As depicted in Fig. 13, our results reveal that as the number of online devices deceases, the accuracy of the system also experiences a decline. Notably, when only one device is online, the system's accuracy exhibits considerable fluctuations. Nevertheless, as the iterations progress and the full connectivity is achieved with each device, the accuracy rate exceeds 80%. Overall, SSL outperforms FL, showing superior performance even when most of the clients are offline.

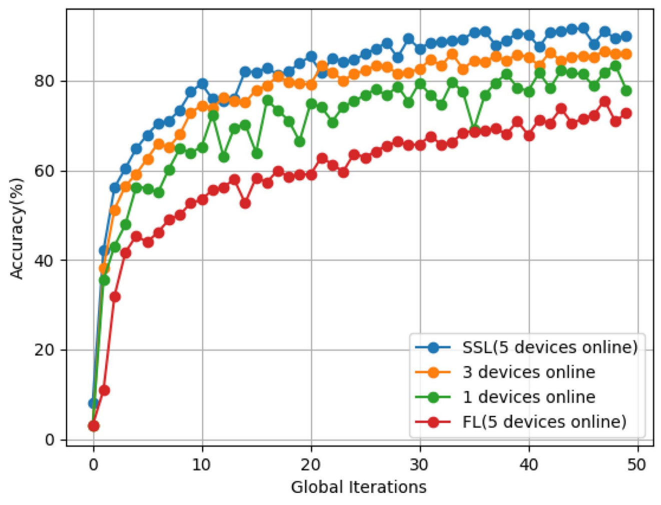


FIGURE 13. The impact of device instability on classification accuracy.

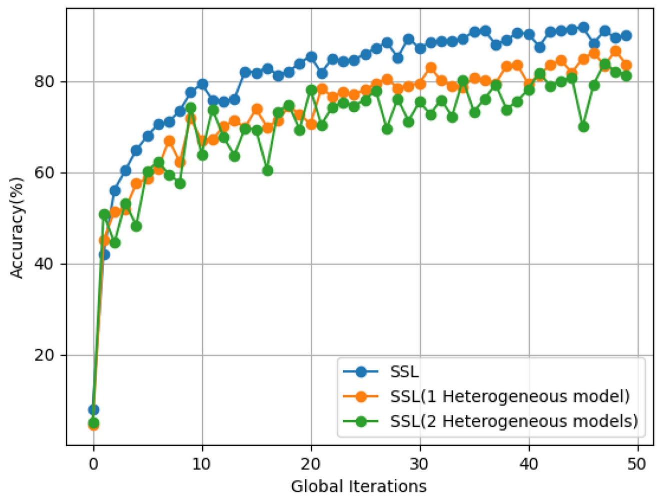


FIGURE 14. The impact of model heterogeneity on classification accuracy.

4) IMPACT OF MODEL HETEROGENEITY

In the proposed SSL approach, the selection of client models is not strictly constrained, allowing for practical deployment scenarios that may involve heterogeneous client models. To evaluate the system's accuracy under these conditions, we replaced some of the ResNet18 models with GoogleNet, which has a similar model size. The results, as shown in Fig. 14, indicate that accuracy fluctuates and declines notably in the presence of model heterogeneity, especially as the number of heterogeneous models increases. Nevertheless, the system still achieves an accuracy of over 80%. This outcome highlights the effectiveness of SSL in iterating and integrating knowledge from clients, even in heterogeneous model environments. We believe this capability can be extended to more model combinations, and we plan to further investigate the impact of different model types on this method in future work.

VII. CONCLUSION

In this article, we propose to employ SL in multi-UAV systems and present a new stitch-able split learning approach to overcome the challenges encountered in multi-UAV scenarios. To evaluate the viability of the proposed system, we use an aerial image dataset to validate the performance. Comprehensive evaluation results reveal that SL-based approach could greatly diminish local computation demands compared to FL, resulting in expedited overall learning time. Moreover, our enhanced SSL approach surpasses SFLV1 and SL in classification accuracy and exhibits robustness in the presence of poor device connectivity and heterogeneous client model conditions. SSL enables a solitary UAV to persist in the training tasks even when the network is offline. Additionally, it empowers us to modify the server-side model, adapting it to varied tasks to attain heightened accuracy.

REFERENCES

[1] B. McMahan et al., "Communication-efficient learning of deep networks from decentralized data," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[2] H. Zhang and L. Hanzo, "Federated learning assisted multi-UAV networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 14104–14109, Nov. 2020.

[3] P. Vepakomma et al., "Split learning for health: Distributed deep learning without sharing raw patient data," 2018, *arXiv:1812.00564*.

[4] C. Thapa et al., "Splitfed: When federated learning meets split learning," in *Proc. AAAI Conf. Artif. Intell.*, 2022, vol. 36, no. 8, pp. 8485–8493.

[5] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[6] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–6.

[7] W. Zhang et al., "Optimizing federated learning in distributed industrial IoT: A multi-agent approach," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3688–3703, Dec. 2021.

[8] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2022–2035, Mar. 2020.

[9] P. Han, S. Wang, and K. K. Leung, "Adaptive gradient sparsification for efficient federated learning: An online learning approach," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, 2020, pp. 300–310.

[10] J. Zhang, N. Li, and M. Dedeoglu, "Federated learning over wireless networks: A band-limited coordinated descent approach," in *Proc. IEEE Conf. Comput. Commun.*, 2021, pp. 1–10.

[11] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 1, pp. 269–283, Jan. 2021.

[12] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 3, pp. 1935–1949, Mar. 2020.

[13] X. Mo and J. Xu, "Energy-efficient federated edge learning with joint communication and computation design," *J. Commun. Inf. Netw.*, vol. 6, no. 2, pp. 110–124, Jun. 2021.

[14] M. Chen et al., "Distributed learning in wireless networks: Recent progress and future challenges," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3579–3605, Dec. 2021.

[15] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 72–80, Apr. 2020.

[16] X. Shen, J. Gao, W. Wu, M. Li, C. Zhou, and W. Zhuang, "Holistic network virtualization and pervasive network intelligence for 6G," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 1–30, Firstquarter 2022.

[17] M. G. Poirot et al., "Split learning for collaborative deep learning in healthcare," 2019, *arXiv:1912.12115*.

[18] Y. Gao et al., "End-to-end evaluation of federated learning and split learning for Internet of Things," in *Proc. Int. Symp. Reliable Distributed Syst. (SRDS)*, Shanghai, China, 2020, pp. 91–100, 10.1109/SRDS51746.2020.00017.

[19] S. Baek et al., "Visual transformer meets cutmix for improved accuracy, communication efficiency, and data privacy in split learning," 2022, *arXiv:2207.00234*.

[20] O. Li et al., "Label leakage and protection in two-party split learning," 2021, *arXiv:2102.08504*.

[21] J. Jeon and J. Kim, "Privacy-sensitive parallel split learning," in *Proc. IEEE Int. Conf. Inf. Netw.*, 2020, pp. 7–9.

[22] G. S. Xia et al., "AID: A benchmark data set for performance evaluation of aerial scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 7, pp. 3965–3981, Jul. 2017.

[23] W. Wu et al., "Split learning over wireless networks: Parallel design and resource management," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 4, pp. 1051–1066, Apr. 2023.

[24] A. Chopra et al., "AdaSplit: Adaptive trade-offs for resource-constrained distributed deep learning," 2021, *arXiv:2112.01637*.

[25] K. Lenc and A. Vedaldi, "Understanding image representations by measuring their equivariance and equivalence," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 991–999.

[26] A. Bansal, P. Nakkiran, and B. Barak, "Revisiting model stitching to compare neural representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 225–236.

[27] A. Csiszárik et al., "Similarity and matching of neural network representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 5656–5668.

[28] S. Kornblith, M. Norouzi, H. Lee, and G. E. Hinton, "Similarity of neural network representations revisited," in *Proc. Int. Conf. Mach. Learn.*, 2019, vol. 97, pp. 3519–3529.

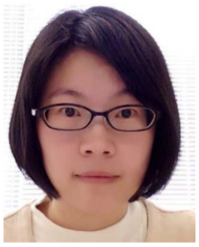
[29] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Computation*, vol. 16, no. 12, pp. 2639–2664, Dec. 2004.

[30] M. Raghu et al., "SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, vol. 30, pp. 1–17.

[31] Z. Pan, J. Cai, and B. Zhuang, "Stitchable neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 16102–16112.

TINGKAI SUN received the M.E. degree in electrical and electronic systems engineering from the Graduate School of Science and Engineering, Ibaraki University, Mito, Japan, in 2024. He is currently working toward the Ph.D. degree in computer science with the University of Tsukuba, Tsukuba, Japan. His research interests include communication security, privacy in data analysis and machine learning, as well as distributed learning.

XIAOYAN WANG (Senior Member, IEEE) received the B.E. degree from Beihang University, Beijing, China, and the M.E. and Ph.D. degrees from the University of Tsukuba, Tsukuba, Japan. From 2013 to 2016, he was an Assistant Professor (by special appointment) with the National Institute of Informatics, Tokyo, Japan. He is currently an Associate Professor with the Graduate School of Science and Engineering, Ibaraki University, Mito, Japan. His research interests include intelligent networking, wireless communications and sensing, cloud computing, Big Data systems, security and privacy.



XIUCAI YE (Member, IEEE) received the Ph.D. degree in computer science from the University of Tsukuba, Tsukuba Science City, Japan, in 2014. She is currently an Associate Professor with the Department of Computer Science, and Center for Artificial Intelligence Research, University of Tsukuba. Her research interests include machine learning algorithms and their applications, as well as bioinformatics.

BIAO HAN (Member, IEEE) received the B.S. and M.S. degrees from the National University of Defense Technology (NUDT), Changsha, China, in 2007 and 2009, respectively, and the Ph.D. degree in computer science from the University of Tsukuba, Tsukuba, Japan, in 2013. He is currently an Associate Professor with the School of Computer, NUDT. He has also been a Visiting Scholar with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA. He has authored or coauthored more than 50 research papers in peer-reviewed journals, such as IEEE Journal on Selected Areas in Communications, IEEE Transactions on Parallel and Distributed Systems, and *Computer Networks,* and conferences, such as INFOCOM and IWQoS. His research interests include UAV swarm networking, multi-path transmission and physical layer security. He was the recipient of the Best Paper Award at IEEE LANMAN'2014. He is the Poster Chair of APNet 2022, an Academic Editor of *Security and Communication Networks*, and Guest Editor of *International Journal of Distributed Sensor Networks*.