MobiFormer: Split-Federated Transfer Learning for Drone RAN Slicing With Multi-Head Attention

Hang Shen, *Member, IEEE*, Yanke Yao, Tianjing Wang, *Member, IEEE*, and Guangwei Bai

***Abstract*—This paper presents MobiFormer, a split-federated transfer learning framework with multi-head attention designed for distributed drone Radio Access Network (RAN) slicing. The objective is to optimize slice performance isolation and training costs. Based on a flexible service metric, the problem of maximizing slice performance isolation quality is formulated as a joint optimization of slice windowing and resource allocation. For single-drone autonomous operations, we construct an "unconstrained mobility and sustainable fine-tuning" paradigm, enabling drones to adapt previously trained resource slicing models to new environments with the assistance of multiple target-domain terrestrial Base Stations (BSs). This adaptation is facilitated by a Source-free Multi-target-domain Transfer Learning (SMTL) approach, where Transformer-based multi-head attention is employed on drones to integrate fine-tuned models from multiple target-domain BSs. Building on SMTL and continuing its scenario, a Clustered Split-Federated Learning (CSFL) approach is developed to support multi-drone collaborative training, where BSs serve as cluster heads to aggregate parameters from member drones. To save energy, part of the onboard models are migrated to BSs while local iterations occur through gradient exchanges. Unlike SMTL, the Transformer encoder-decoder is deployed at BSs to enhance the global model's adaptability and generalization. Extensive simulations demonstrate that MobiFormer outperforms benchmark approaches in terms of performance isolation, energy consumption, and online decision-making efficiency in distributed learning settings.**

***Index Terms*—Drone radio access network (RAN) slicing, slice windowing, source-free multi-target-domain transfer learning (SMTL), clustered split-federated learning (CSFL), transformer.**

I. Introduction

DRONE base stations (BSs), as a new type of network infrastructure, are becoming an indispensable component of beyond 5G and 6G networks. Benefiting from flexible deployment and line-of-sight communication, drone BSs can rapidly extend network coverage in remote or emergency scenarios, providing customized services for applications such as smart cities and autonomous driving, thus extending the functionality of terrestrial BSs. In 6G networks, drones are expected to handle massive device connectivity and high-speed data transmission while ensuring ultra-low latency and high reliability in communication. Using Radio Access Network (RAN) slicing technology, physical network resources are partitioned into independent logical networks to provide differentiated services for various applications [1].

The integration of drones extends RAN slicing into aerial networks, enabling agile and scalable connectivity beyond the limitations of static terrestrial infrastructure [2]. In real-world large-scale deployments, drones operate under unconstrained mobility across regions with highly diverse slicing requirements, even within similar environments. For example, one area may require high-throughput slices for streaming and navigation, while a neighboring zone may prioritize low-latency voice or Internet of Things (IoT) services. These intra-regional variations evolve with daily routines, population shifts, and contextual factors. To address such dynamic and heterogeneous demands, drone RAN slicing must support lightweight operation, timely adaptability, and efficient resource use, despite constraints on battery, storage, and onboard computation. Accordingly, slice resource allocation must consider real-time drone positions, workload dynamics, and localized constraints to ensure reliable, low-latency service delivery.

Several studies have addressed these challenges through the use of mathematical modeling and optimization techniques. The authors in [3], [4], [5], [6] explored the joint optimization of drone deployment and resource slicing, thereby improving overall efficiency in spatial and temporal dimensions. SoftFloat [7] was proposed as a software-defined framework for drone network slicing, supporting differentiated service provisioning under complex and variable conditions. Atlas [8] introduced an automated slicing configuration system designed to reduce resource usage while meeting service-level constraints. In [9], user locations and channel gains were leveraged to proactively optimize slicing decisions. However, these methods lack the flexibility to adapt rapidly after deployment.

To overcome the limitations of static optimization, recent research on drone RAN slicing has shifted to learning-based approaches. They primarily involve standalone decision-making [10] or reliance on terrestrial infrastructure [11], both of which struggle to support large-scale mobility and collaboration. Learning paradigms such as Transfer Learning (TL) [12], Federated Learning (FL) [13], and Split Learning (SL) [14] have demonstrated potential in distributed drone networks.

Received 27 October 2024; revised 16 October 2025; accepted 26 October 2025. Date of publication 31 October 2025; date of current version 6 March 2026. This work was supported in part by the National Natural Science Foundation of China under Grant 61502230 and Grant 61501224, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20201357, in part by the Market Supervision Administration Science and Technology Fund of Jiangsu Province under Grant KJ2026027, and in part by the Six Talent Peaks Project in Jiangsu Province under Grant RJFW-020. Recommended for acceptance by Z. Li. *(Corresponding author: Hang Shen.)*

The authors are with the College of Computer and Information Engineering (College of Artificial Intelligence), Nanjing Tech University, Nanjing 211816, China (e-mail: hshen@njtech.edu.cn; 202261220087@njtech.edu.cn; wangtianjing@njtech.edu.cn; bai@njtech.edu.cn).

Digital Object Identifier 10.1109/TMC.2025.3627068

TL enables drones to quickly adapt to new environments by transferring knowledge across various scenarios. FL facilitates collaborative learning of multiple drones without sharing data. SL mitigates the computational burden by splitting the model between drones and infrastructure. Some studies further incorporate Multi-Agent Reinforcement Learning (MARL) [15], [16], [17], allowing drones to make decentralized decisions while maintaining cooperative behavior.

A. Challenging Issues and Related Works

Integrating distributed learning techniques into drone RAN slicing remains challenging due to dynamic deployments, fluctuating service conditions, and the need for efficient collaboration with terrestrial infrastructure, including

1) Sustainable and flexible onboard model update: In dynamic deployments, onboard models must be retrained or fine-tuned to adapt to new scenarios, consuming storage and computational resources and hindering efficient knowledge transfer. TL enables knowledge reuse but suffers from domain discrepancies between the source and target domains, potentially leading to negative transfer [18]. Thus, there is a pressing need for flexible mechanisms that support continuous model optimization and efficient resource utilization. Yu et al. proposed a data-free single-teacher knowledge distillation solution [19] to continuously update drone knowledge. This teaching mode cannot leverage knowledge from multiple ground-based facilities. Li et al. [20] integrated TL with Mixed Noise Deep Deterministic Policy Gradient (MN-DDPG) to support continuous learning in drone tracking and planning, though it places a heavy burden on onboard computation in the absence of infrastructure. MARL frameworks with centralized training and decentralized execution have also been applied to improve multi-drone coordination [17], [21], leveraging global policies for collaborative decision-making. However, these approaches generally rely on static or periodically retrained models and lack support for fast, sustainable fine-tuning during dynamic operations. Efficiently integrating source and target knowledge in real-time remains an open challenge.

2) Cost reduction of online training and decision-making: Airborne models must adapt to new environments and tasks through in-flight fine-tuning or retraining, during which data interaction and model updates impose significant burdens, reducing the sustainability of task execution. Although model compression [22], reduced training frequency [23], or client scheduling [24] can lower FL training costs, they may weaken model performance. Liu et al. proposed a distributed learning architecture incorporating SL and FL, which improves training efficiency and balances model accuracy and resource consumption [25]. Xu et al. [21] proposed a MARL approach that improves training efficiency and decision scalability by compressing individual policies into shared, group-level strategies through agent grouping and information sharing. Another notable challenge is the efficiency of online decision-making related to slice-window partitioning. The decision generation is window-centric. Shortening the window duration can accelerate resource updates but increase decision and control overhead; extending the window reduces control overhead but may compromise performance isolation in task-intensive scenarios. Existing slice windowing strategies [26], [27] mostly adopt static modes, bringing about many unnecessary decision costs. Therefore, it is inevitable to design efficient distributed learning and window-adaptive decision-making.

3) Enhancements in adaptability and generalization: Drones performing tasks in dynamic environments require high model generalization and adaptability. Conventional FL approaches struggle to handle data heterogeneity and disparity caused by drone mobility. Due to limited coverage, the strong local characteristics of drone-collected data pose a challenge. Traditional attention mechanisms [28] struggled to optimize models by extracting locally important information. GSM-FedAVG [29] was proposed as a global, shared, model-federated averaging algorithm that improves model adaptability and generalization by sharing models across all clients. Federated Proximal (FedProx) [30] introduced regularization to address system and data heterogeneity. Ma et al. [31] utilized attention mechanisms to process local essential tasks, aiming to improve path-planning accuracy and robustness in multi-drone systems. AGUZero [32] was an attention-based MADL framework that leveraged graph-based representations to optimize drone-sensor coordination under severe communication and battery constraints. Strengthening model adaptability of local environments and integrating globally distributed contextual information remain areas for further exploration.

B. Contributions and Organization

To address the aforementioned challenges, this paper presents MobiFormer, a split-federated transfer learning framework with Transformer-based multi-head attention, designed for drone RAN slicing in both single-drone and multi-drone scenarios. The framework aims to optimize performance isolation and training cost simultaneously. The main contributions of this study are threefold:

- *Window-adaptive resource slicing strategy:* We design a drone-centric strategy that jointly adjusts slicing windows and allocates resources based on a flexible performance isolation metric. This framework strikes a balance between isolation quality and decision efficiency.
- *Source-free Multi-target-domain TL (SMTL) scheme:* This scheme is, based on the paradigm of "unconstrained mobility and sustainable fine-tuning", designed specifically for single-drone mobility scenarios, where the drone transitions across geographical regions and loses access to the original source domain data. SMTL enables fine-tuning with assistance from multiple target-domain BSs without requiring source data uploads. The drone utilizes a Transformer-based multi-head attention mechanism for fusing fine-tuned models, which improves the drone model's adaptability across heterogeneous domains.
- *Clustered Split-Federated Learning (CSFL) scheme:* Extending SMTL to multi-drone scenarios, we further develop a CSFL method where BSs serve as cluster heads. Each drone's model is split into aerial and terrestrial

TABLE I
MAIN NOTATIONS AND VARIABLES

Symbol	Definition
B_k	Total num. of RBs held by drone k
$\mathcal{I}_k/I_k$	Set/Num. of target domain BSs participating in fine-tuning the model on drone i
$\mathcal{K}/K$	Set/Num. of drones
$\mathcal{K}_i/K_i$	Set/Num. of drones in the cluster headed by BS i
$\mathcal{M}_k/M_k$	Set/Num. of slices on drone k
o_k'	Source domain model of drone k
$r_{k,m}^{(a)}$	RBs allocated to drone k's slice m in window a
$r_{k,m,t}^{(a)}$	RBs demand of slice m on drone k at $t \in \mathcal{T}_k^{(a)}$
$\tilde{r}_{k,m}^{(a)}$	Predicted RBs for slice m on drone k in window a
$\mathcal{T}_k^{(a)}/T_k^{(a)}$	Set/Num. of timeslots in window a for drone k
$w_{k,i}$	Model of drone k fine-tuned by target domain BS i
$\boldsymbol{w}_{k,i}$	Vector representation of $w_{k,i}$ encoded by TE
$\boldsymbol{w}_k$	Vector representation of drone k's model after aggregation via multi-head attention
w_k	Model of drone k fine-tuned by multiple target domain BSs and decoding by TD
$w_{i,k,b}$	Aerial model of drone k in cluster i
$\boldsymbol{w}_{i,k,b}$	Vector representation of $w_{i,k,b}$ encoded by TE
$w_{i,k,p}$	Ground model from drone k in cluster i
$\boldsymbol{W}_{i,b}$	Vector representation of the aerial model in BS i aggregated via multi-head attention
$W_{i,b}$	Aerial model aggregated at BS i and decoded by TD
$z_k^{(a)}$	Shared RBs reserved by drone k in window a

components, allowing for energy-efficient local updates via gradient exchange. Unlike SMTL, the Transformer is integrated at BSs to optimize onboard model aggregation, enhancing global generalization.

We build a distributed simulation environment that integrates real-world multi-service communication data to emulate realistic drone RAN scenarios. Simulation results demonstrate that MobiFormer outperforms typical slicing strategies based on baseline distributed learning frameworks, achieving better performance isolation and lower training costs, while significantly reducing decision-making overhead.

The rest of this paper is organized as follows. Section II introduces the resource slicing framework, performance isolation metrics, and problem modeling. In Section III, an SMTL framework addresses single-drone mobility. Section IV designs a CSFL method to support collaborative training in large-scale scenarios. Section V presents the experimental design and performance evaluation. Section VI concludes the paper and discusses future work. For clarity, the main notations and variables are listed in Table I.

II. System Model

Consider a large-scale drone RAN scenario, where drones serve as aerial small BSs, flexibly deployed over wide areas. They communicate with ground BSs via the 5G NR protocol [33] for handshakes and inquiries. If a drone needs to leave for other tasks, it sends a notification to release the connection.

A. Window-Adaptive Resource Slicing Framework

We design a window-adaptive resource slicing framework, where physical resources are pooled and divided into multiple slices and a shared area. Each drone adjusts the slice window length and resource allocation based on demand. As shown in Fig. 1, time is divided into windows, each containing a set of discrete timeslots, where $\mathcal{T}_k^{(a)}$ represents the set of timeslots in window a for drone k.

Suppose the total number of Resource Blocks (RBs) held by drone k is B_k. They are virtualized into M_k service slices, and the set of slices is denoted by $\mathcal{M}_k$. At the start of each window, the RBs allocated to each service slice and the shared area are redistributed and held until the window ends. At the end of each window, drone k computes the resource demand of each slice for the next window and adjusts the size of the shared area accordingly. Tasks are scheduled to different slices based on their attributes. Each slice allocates RBs to tasks in a timeslot-based manner. Let $\mathcal{K}$ denote the set of drones, with K being its cardinality. At the start of window a, the number of RBs allocated to slice m and the shared area on drone k are denoted as $r_{k,m}^{(a)}$ and $z_k^{(a)}$, which satisfy

$$r_{k,m}^{(a)} + z_k^{(a)} \leq B_k \tag{1}$$

and

$$z_k^{(a)} \leq \omega B_k, \qquad \forall k \in \mathcal{K}. \tag{2}$$

Constraint (1) demonstrates the requirements on RB allocation for slices and the shared area at drone k, while (2) limits the shared RB proportion to ω. If a slice's resources are insufficient, shared RBs can be temporarily borrowed.

B. Adaptive Windowing Strategy

The shared RBs help to buffer sharp fluctuations in resource demand, preventing slice performance degradation. When the resource pressure is high (low) in slice window a, the number of shared RBs, $z_k^{(a)}$, should be increased (decreased). We regard $z_k^{(a-1)}$ as a hint to determine window a's length, $T_k^{(a)}$. Let T_k be the default window duration for drone k, with an adjustment step size of ΔT_k. Given $z_k^{(a-1)}$ and $z_k^{(a)}$, the window duration (i.e., the cardinality of $\mathcal{T}_k^{(a)}$) is determined at the start of window a as

$$T_k^{(a)} = \begin{cases} T_k + \Delta T_k, & \text{if } z_k^{(a-1)} > z_k^{(a)} \\ T_k, & \text{if } z_k^{(a-1)} = z_k^{(a)} \\ T_k - \Delta T_k, & \text{if } z_k^{(a-1)} < z_k^{(a)}. \end{cases} \tag{3}$$

This adaptive windowing policy ensures that resource allocation can flexibly respond to each drone's demand variations. When the shared area expands, the window shortens to speed up resource updates; when it shrinks, the window lengthens to lower decision costs.

C. Performance Isolation Metric

Performance isolation ensures that resource occupation in one slice does not interfere with the services provided by other slices, which is critical for multi-slice coexistence [34]. The resource allocation at the beginning of each slice window determines the performance isolation quality within that window.

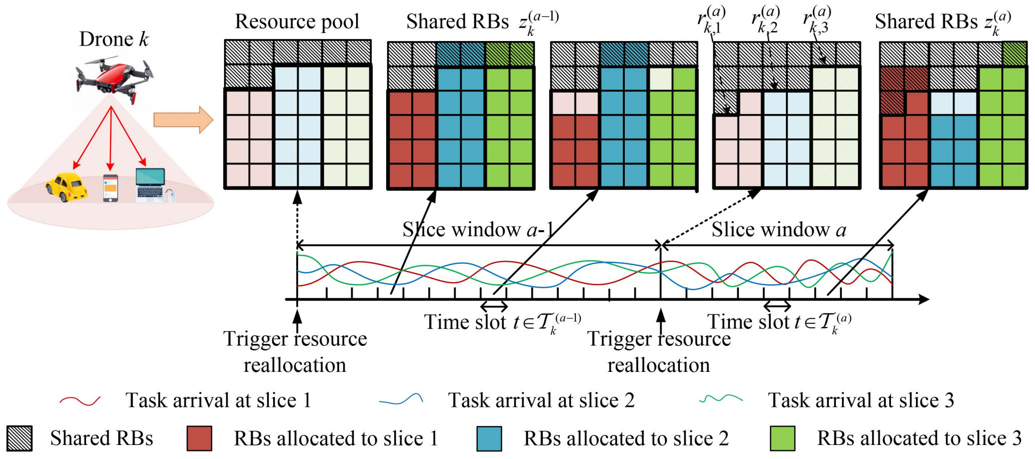


Fig. 1. Window-adaptive dynamic resource slicing framework.

Let $r_{k,m,t}^{(a)}$ be the number of RBs required by slice m on drone k at timeslot t. The number of RBs occupied in the shared area is $\sum_{m\in\mathcal{M}_k}\max\{|r_{k,m,t}^{(a)}-\tilde{r}_{k,m}^{(a)}|\}\leq z_k^{(a)}$. The performance isolation quality at timeslot t is defined as

$$q_{t,k}^{(a)} \triangleq 1-\frac{\sum_{m\in\mathcal{M}_k}\max\left\{|r_{t,k,m}^{(a)}-\tilde{r}_{k,m}^{(a)}|\right\}}{M_k B_k}. \tag{4}$$

From (4), the performance isolation is inversely correlated with $|r_{k,m,t}^{(a)}-\tilde{r}_{k,m}^{(a)}|$. If $r_{k,m,t}^{(a)}\leq r_{k,m}^{(a)}$, slice m can meet its current demand without occupying the shared area, and the isolation quality reaches its highest level (with $q_{k,t}^{(a)}=1$). If the number of shared RBs, $z_k^{(a)}$, are insufficient to meet the slice's service demand, the isolation is broken (with $q_{k,t}^{(a)}=0$). If $r_{k,m,t}^{(a)}>r_{k,m}^{(a)}$, the shared RBs are temporarily occupied, at which the performance isolation is compromised (with $0<q_{k,t}^{(a)}<1$).

D. Problem Formulation and Solution Framework

The resource allocation vector for M_k slices on drone k for window a is denoted as $\boldsymbol{R}_k^{(a)}=[r_{k,1}^{(a)},r_{k,2}^{(a)},\ldots,r_{k,M_k}^{(a)}]$. Based on (4), the average performance isolation quality over window a is computed as $q_k^{(a)}=\frac{1}{T_k^{(a)}}\sum_{t\in\mathcal{T}_k^{(a)}}q_{k,t}^{(a)}$. By applying it to $\mathcal{A}_k$, the long-term performance isolation optimization problem for drone k is formulated as:

$$\mathcal{P}1:\max_{\{T_k^{(a)},\boldsymbol{R}_k^{(a)},z_k^{(a)}\}_{a\in\mathcal{A}_k}}\lim_{A\to\infty}\sum_{a\in\mathcal{A}_k}\frac{1}{T_k^{(a)}}\sum_{t\in\mathcal{T}_k^{(a)}}q_{t,k}^{(a)}$$

s.t. (1), (2), and (3)

The essence of Problem $\mathcal{P}1$ is to allocate resources to each slice, $\boldsymbol{R}_k^{(a)}$, and the shared area, $z_k^{(a)}$, and to determine the slice window duration, $T_k^{(a)}$, based on (3), maximizing the long-term performance isolation quality.

We now introduce the solution framework for this problem. Assume that the predicted number of RBs required by each slice on drone k in window a is $\tilde{\boldsymbol{R}}_k^{(a)}$. The RBs allocated to slice m on drone k are calculated as

$$r_{k,m}^{(a)}=\frac{(\tilde{r}_{k,m}^{(a)}-\min\{\tilde{\boldsymbol{R}}_k^{(a)}\})\cdot(\max\{\boldsymbol{R}_k^{(a)}\}-\min\{\boldsymbol{R}_k^{(a)}\})}{\max\{\tilde{\boldsymbol{R}}_k^{(a)}\}-\min\{\tilde{\boldsymbol{R}}_k^{(a)}\}}+\min\left\{\boldsymbol{R}_k^{(a)}\right\} \tag{5}$$

By substituting (5) into (1), we compute $z_k^{(a)}$, and using it in (3), we derive $T_k^{(a)}$. Clearly, $\tilde{\boldsymbol{R}}_k^{(a)}$ serves as the starting point, determining both $z_k^{(a)}$ and $T_k^{(a)}$. Sections III and IV introduce the SMTL and CSFL methods for solving $\tilde{\boldsymbol{R}}_k^{(a)}$ in single-drone and multi-drone scenarios, respectively.

III. SMTL Scheme

When a drone transitions into a new operational area, its onboard model often becomes outdated due to domain shifts, necessitating fine-tuning to adapt to the new environment. Traditional retraining approaches incur significant communication and computational overhead and typically fail to reuse prior knowledge. To address this issue, we propose an SMTL scheme that supports sustainable in-flight model adaptation without requiring uploads of source-domain data.

In the SMTL workflow, a Mobile Edge Computing (MEC)-enabled controller coordinates model adaptation between the drone and nearby BSs. As illustrated in Fig. 2, when the drone moves from the source domain into a new region jointly served by multiple BSs (e.g., BSs 1 and 2), it shares its current model for collaborative fine-tuning using BSs' local data under MEC supervision. This process avoids exposing any source-domain information while accelerating adaptation. The SMTL process consists of the following steps:

① *Multi-target-domain Data Selection:* The drone transmits latent features representing the source-domain distribution (not raw data) to the MEC controller, which compares them with BS-side data distributions to select suitable target BSs. Each chosen BS filters representative local samples for fine-tuning.

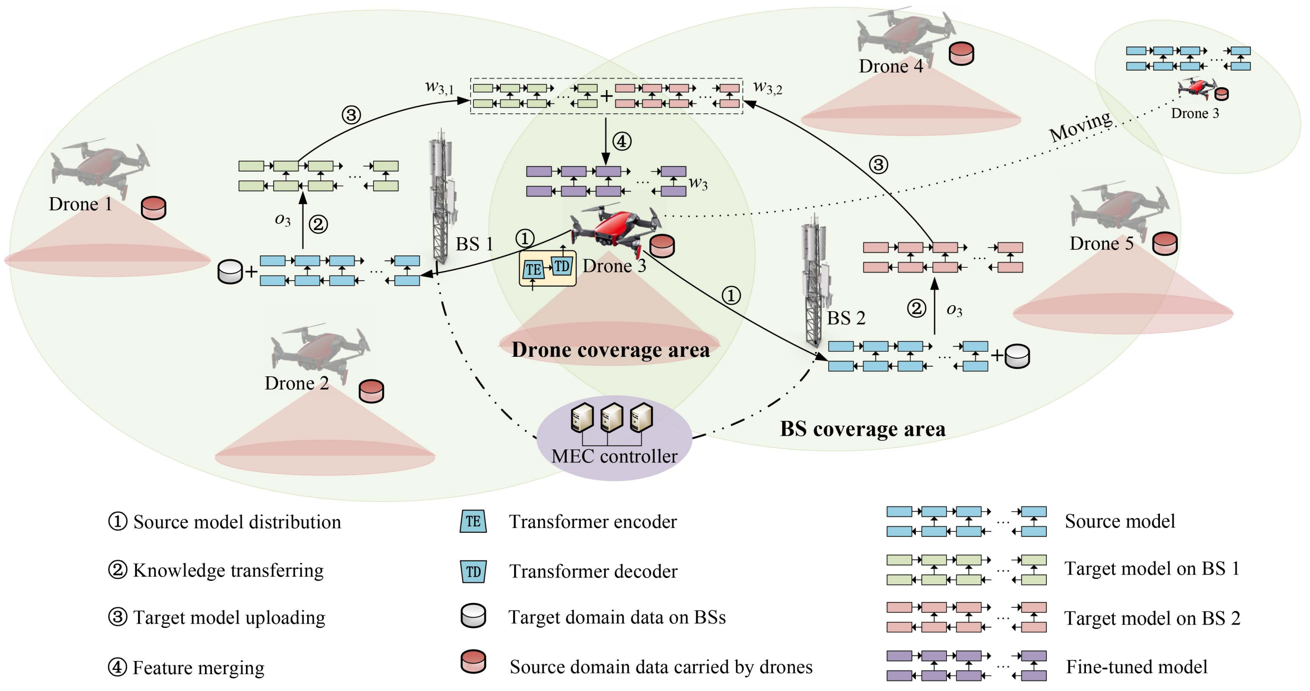


Fig. 2. SMTL for the single-drone autonomous scenario.

② *Multi-target-domain Knowledge Transfer:* The selected BSs 1 and 2 collaboratively fine-tune the drone's model using their local data subsets, adapting it to region-specific traffic or service patterns.

③ *Multi-target-domain Feature Fusion:* Drone 3 utilizes a Transformer-based Encoder (TE)-Decoder (TD) [35] architecture to perform adaptive feature fusion, yielding a refined model suited to the new environment.

A. Multi-Target Domain Data Selection

A multi-target domain data selection strategy is designed to mitigate the adverse effects of differences in the data of BSs as target domains on fine-tuning.

In the SMTL framework, the source and a target domain correspond to drone k and BS i. Let $x_{i,j}$ represent the j-th input sequence data from target domain BS i, $y_{i,j}$ represent the j-th label, and N_i represent the number of samples in the target domain. Let $o_k{}'$ denote the source domain model carried by drone k, and $f_i(x_{i,j}; o_k{}')$ represent the model for target domain BS i of the j-th input sequence data. The loss function for them is expressed as

$$\mathcal{L}_i(o_k{}') = \frac{1}{N_i}\sum_{j=1}^{N_i} \ell(y_{i,j}, f_i(x_{i,j}; o_k{}')) \tag{6}$$

where $\ell(\cdot)$ is the loss function for a single sample.

The Kullback-Leibler (KL) divergence [36] is used as the criterion for selecting target domain BSs for model fine-tuning. A smaller (larger) KL divergence value indicates greater (less) similarity between the drone's historical data distribution and the BS's distribution. Similar data is used as much as possible to fine-tune the model, reducing overfitting risk. Suppose the distributions of drone k and BS i are P_k and $P_i \in \mathbb{R}^{d\times 1}$.

$$\begin{cases} P_k = \frac{P_k{}'}{\sum P_k{}'}, & P_k{}' = \frac{1}{N_k}\sum_{j=1}^{N_k} h_{k,j} \\ P_i = \frac{P_i{}'}{\sum P_i{}'}, & P_i{}' = \frac{1}{N_i}\sum_{j=1}^{N_i} h_{i,j}. \end{cases} \tag{7}$$

In (7), $h_{k,j}$ and $h_{i,j}$ represent the hidden representations of the drone and BS models. The KL divergence between them is calculated as

$$\begin{aligned} D_{KL}^{(k,i)} &= D_{KL}(P_k || P_i) = H(P_i, P_k) - H(P_k) \\ &= \sum_{x\in\mathcal{X}} P_k(x) \log \frac{P_k(x)}{P_i(x)} \end{aligned} \tag{8}$$

where $H(P_k)$ is the entropy of P_k, and $H(P_i, P_k)$ is the cross-entropy between P_i and P_k. The set of BSs covering drone k is represented as $\mathcal{I}_k'$ with I_k' being its cardinality. The set of target domain BSs selected for fine-tuning is determined as

$$\mathcal{I}_k = \left\{ i \in \mathcal{I}_k{}' | D_{KL}^{(k,i)} < \theta \right\} \tag{9}$$

where θ is a KL divergence threshold. BS i selects a portion of historical data similar to P_k to fine-tune o_k. The proportion of this data is inversely proportional to the KL divergence, expressed as

$$\varepsilon_i = \frac{\min_{i\in\mathcal{I}_k} \{D_{KL}^{(k,i)}\}}{D_{KL}^{(k,i)}}. \tag{10}$$

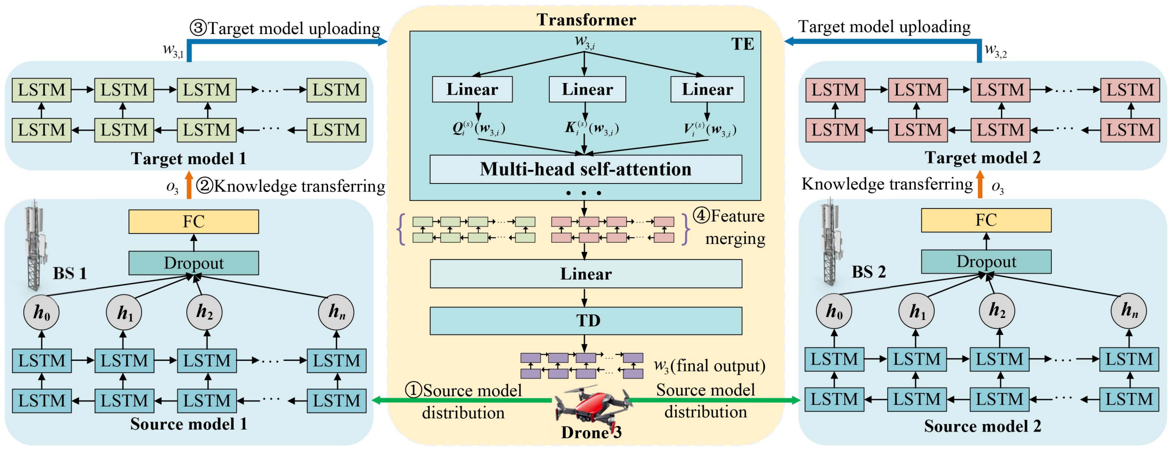


Fig. 3. SMTL workflow using Drone 3 as an example.

B. Multi-Target Domain Knowledge Transfer

A Bidirectional Long Short-Term Memory (Bi-LSTM) network [37], a lightweight model, is deployed on each drone to predict resource demand. Let $\overrightarrow{\tilde{r}}_{t,k,m}^{(a)}$ and $\overleftarrow{\tilde{r}}_{t,k,m}^{(a)}$ represent the forward and backward predictions of the LSTM at timeslot $t \in \mathcal{T}_k^{(a)}$. The prediction by Bi-LSTM is then given by

$$\tilde{r}_{t,k,m}^{(a)} = \overrightarrow{\tilde{r}}_{t,k,m}^{(a)} \oplus \overleftarrow{\tilde{r}}_{t,k,m}^{(a)}, \tag{11}$$

where $\oplus$ represents the combination of the two hidden states.

The number of RBs required by a slice on the drone during window a is predicted as

$$\tilde{r}_{k,m}^{(a)} = f\left(r_{k,m}^{(a-1)}; o_k{}'\right). \tag{12}$$

Loss function $\ell(f(r_{k,m}^{(a-1)}; o_k{}'), \tilde{r}_{k,m}^{(a)})$ is used to measure the prediction error of the required RBs for slice m on drone k, for which the aggregated loss function is expressed as

$$\mathcal{L}_k(o_k{}') = \frac{1}{T_k^{(a)}} \frac{1}{M_k} \sum_{m \in \mathcal{M}_k} \ell\left(f(r_{k,m}^{(a-1)}; o_k{}'), \tilde{r}_{k,m}^{(a)}\right). \tag{13}$$

From Fig. 3, the SMTL workflow begins with the onboard model of drone k being updated using gradient descent as

$$o_k = o_k{}' - \alpha_k \nabla_{o_k{}'} \mathcal{L}_k(o_k{}'), \tag{14}$$

where α_k is the learning rate of drone k and $\nabla_{o_k{}'}$ represents the gradient of the loss function for $o_k{}'$. The objective of fine-tuning BS i is to minimize the following loss function

$$\mathcal{L}(o_k) = \sum_{i \in \mathcal{I}_k} e^{-\beta D_{KL}^{(k,i)}} \mathcal{L}_i(o_k). \tag{15}$$

In (15), β is a regularization factor that controls the impact of KL divergence on the weighting. A dynamic weighting mechanism is employed to adaptively adjust β based on the domain shift between the source and target distributions. When the target domain exhibits a large distributional divergence (high $D_{KL}^{(k,i)}$), lowering the value of β suppresses the influence of noisy or misaligned target data, thereby preventing negative transfer. Conversely, when the distribution shift is moderate, increasing β enhances alignment and encourages effective knowledge transfer. This adaptive adjustment mechanism helps strike a balance between knowledge reuse and robustness, improving the fine-tuning performance in multi-target scenarios with varying domain characteristics. o_k is fine-tuned in the target domain BS i as

$$w_{k,i} = o_k - \eta_i \nabla \mathcal{L}(o_k), \tag{16}$$

where η_i is BS i's learning rate and ∇ represents the gradient of the loss function. A portion of BS i's local data, denoted by η_i, is used to fine-tune the model. Throughout the iterations, $w_{k,i}$ is adapted to BS i's target tasks, improving the adaptability of o_k in target domain i. Afterward, the fine-tuned model from the BSs in $\mathcal{I}_k$ is delivered to drone k.

C. Multi-Target Domain Feature Fusion

After receiving multiple fine-tuned models from the selected BSs, the drone performs an onboard feature fusion procedure using its TE and TD. This process dynamically integrates multi-target knowledge while maintaining inter-domain consistency and resource-aware adaptability.

The fusion begins with each fine-tuned model ($\boldsymbol{w}_{k,i}$) being encoded by the TE into a continuous latent sequence that preserves resource-aware dependencies and model-specific characteristics. The TE maps each sequence into three vector spaces (queries $\boldsymbol{Q}_i^{(s)}$, keys $\boldsymbol{K}_i^{(s)}$, and values $\boldsymbol{V}_i^{(s)}$) through multiple independent fully connected layers, following the multi-head attention formulation [38], where s denotes the head index. $\boldsymbol{Q}_i^{(s)}$ captures cross-domain resource alignment dependencies, $\boldsymbol{K}_i^{(s)}$ encodes contextual correlations among BS fine-tuned parameters, and $\boldsymbol{V}_i^{(s)}$ retains the performance-relevant features of each model. Each attention head analyzes a feature subspace to learn inter-domain relations. For head h, the attention output for BS i

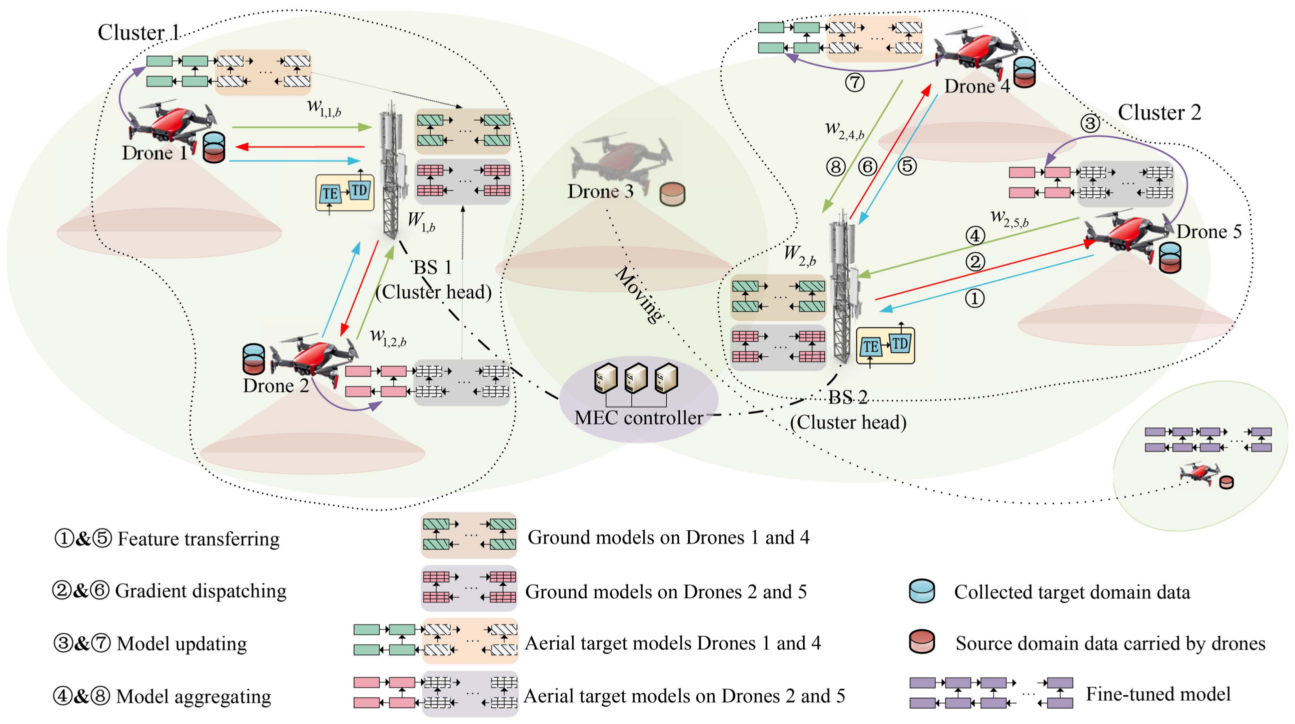


Fig. 4. CSFL framework for multi-drone collaboration scenario (Continuation of the scenario in Fig. 2).

is denoted as

$$\delta_i^{(s)} = \text{softmax}\left(\frac{\boldsymbol{Q}_i^{(s)}\boldsymbol{K}_i^{(h)\top}}{\sqrt{d}}\right)\boldsymbol{V}_i^{(s)}, \tag{17}$$

representing the weight assigned to BS i by the s-th attention head. Each head focuses on distinct statistical or resource distribution patterns across target domains, enabling the drone to jointly exploit multi-view resource cues during adaptation. Through (17), the attention weight vector for all BSs in $\mathcal{I}_k$ is obtained as $[\delta_1, \delta_2, \ldots, \delta_{I_k}]$, representing each BS's relative contribution to the fusion.

Based on (16) and (17), the drone aggregates all locally fine-tuned models into the following unified representation

$$\boldsymbol{w}_k = \sum_{i\in\mathcal{I}_k} \delta_i \boldsymbol{w}_{k,i}. \tag{18}$$

This fusion allocates adaptation effort across the drone's sliced computing resources under BS guidance, mitigating negative transfer from less relevant domains. The TD reconstructs $\boldsymbol{w}_k$ into the final output w_k to enforce feature consistency, while backpropagated gradients refine attention and shared parameters within the TE, balancing specialization and generalization. The fused parameters guide onboard slice adjustment based on service demands, enabling in-flight adaptation with low overhead and without exposing source-domain information.

IV. CSFL Scheme

After completing model adaptation via SMTL, a drone can choose to remain in the current area or proceed to a neighboring region. For drones that stay within adjacent areas, we extend the SMTL framework and scenario by introducing a CSFL scheme that enables opportunistic multi-drone collaboration. As illustrated in Fig. 4, CSFL integrates federated clustering, local training of split models, and Transformer-enhanced federated aggregation, enabling participating drones to share and refine knowledge in a communication-efficient, scalable manner.

A. Federated Clustering

When multiple drones are found hovering in the managed area, the MEC-enabled controller opportunistically initiates clustering using weighted k-means, broadcasting invitations to drones via ground BSs. These BSs serve as cluster heads and drones as members. This design ensures that even in sparse or dense deployments, only eligible drones with stable connectivity and overlapping BS coverage participate in aerial collaborative training.

The weighted coefficients of drone positions are included in the Euclidean distance calculation of the k-means algorithm. The positions of drone k and BS i are represented as g_k and g_i, with the weighting coefficient determined by the following standard deviation of position dispersion.

$$\rho = \sqrt{\frac{1}{K}\sum\nolimits_{i\in\mathcal{I}}\sum\nolimits_{k\in\mathcal{K}_i}(g_k - \bar{g}_k)^2} \tag{19}$$

In (19), $\bar{g}_k$ is the aggregated position, $\mathcal{I}$ is the set of all BSs, and $\mathcal{K}_i$ the number of drones in the cluster headed by BS i. The weighted Euclidean distance between drone k and BS i

Algorithm 1: Federated Clustering Algorithm.

```
input : $\cup_{k\in\mathcal{K}} g_k$ and $\cup_{i\in\mathcal{I}_k} g_i$
output: $\mathcal{K}_i$
1 Randomly select $I_k$ BSs as initial centroids;
2 Initialize clusters: $\rho \leftarrow \sqrt{\frac{1}{K}\sum_{k\in\mathcal{K}}(g_k-\bar{g}_k)^2}$;
3 foreach $k \in \mathcal{K}$ do
4     foreach $i \in \mathcal{I}_k$ do
5         $d_{k,i} \leftarrow \sqrt{\rho(g_k-g_i)^2}$;
6         $i^* = \arg\min \sum_{i\in\mathcal{I}_k}\sum_{k\in\mathcal{K}_i} d_{k,i}$;
7         $\mathcal{K}_i \leftarrow \mathcal{K}_i \cup \{k\}$;
```

(centroid) is calculated as

$$d_{k,i} = \sqrt{\rho(g_k-g_i)^2}. \tag{20}$$

The MEC controller determines the drone-to-BS assignments by minimizing $\sum_{i\in\mathcal{I}}\sum_{k\in\mathcal{K}_i} d_{k,i}$, which is the summation of all weighted distances.

While the designed clustering policy mainly considers node positions, the preceding SMTL stage indirectly aligns the data distribution among drones within each cluster. This is largely due to overlapping learning targets among neighboring drones during the SMTL stage, which causes interactions among their data distributions. The clustering procedure is detailed in Algorithm 1, where the randomly selected set of BSs, $\mathcal{I}_k$, serves as the initial centroids, and I_k is its cardinality. Based on (19) and (20), each drone is assigned to the cluster with the nearest centroid according to the weighted distance metric. In line 7, $\mathcal{K}_i$, the set of drones assigned to cluster i, is updated. This method reduces communication costs in joint training by aligning data in each cluster.

B. Model Splitting and Local Iteration

An SL scheme is designed to reduce collaborative training costs. The Bi-LSTM model carried by drone k, denoted as $w_{i,k}$, is split into an airborne model and a ground model [39], denoted as $w_{i,k,b}'$ and $w_{i,k,p}'$, corresponding to the shallow and deep layers of the original model. The former remains local, while the latter is transferred to BS i as a cluster head. The two parts exchange backpropagated gradients wirelessly.

The MSE loss, denoted as $\frac{1}{K_i}\sum_{k\in\mathcal{K}_i}(r_{k,m}^{(a)}-\tilde{r}_{k,m}^{(a)})^2$, is used to measure the loss of model output [40]. Substituting $w_{i,k}$ into (12), the number of RBs required by a slice on drone k in window a is predicted as

$$\tilde{r}_{k,m}^{(a)} = f\left(r_{k,m}^{(a-1)}; w_{i,k}\right). \tag{21}$$

A loss function, $l(f(r_{k,m}^{(a-1)}; w_{i,k}), \tilde{r}_{k,m}^{(a)})$, is used to measure the prediction error of the required RBs for slice m on drone k. The loss function with respect to $w_{i,k}$ is represented as

$$\mathcal{L}_k(w_{i,k}) = \frac{1}{T}\frac{1}{M_k}\sum_{m\in\mathcal{M}_k} l\left(f(r_{k,m}^{(a-1)}; w_{i,k}), \tilde{r}_{k,m}^{(a)}\right). \tag{22}$$

Let $\mathcal{L}_b(w_{i,k,b}')$ and $\mathcal{L}_p(w_{i,k,p}')$ denote the loss functions for $w_{i,k,b}'$ and $w_{i,k,p}'$, for which the cumulative loss is given by

$$\mathcal{L}_k(w_{i,k}) = \mathcal{L}_p(w_{i,k,p}') + \mathcal{L}_b(w_{i,k,b}'). \tag{23}$$

Fig. 5 illustrates the local iteration process after the model is split on drone k. Unlike in SMTL, where drone k only carries source domain data, both the source domain data and newly collected data are used for forward propagation for $w_{i,k,b}'$. The resulting intermediate features, or smashed data, are passed to BS i (e.g., step ①/⑤ in Fig. 5). The BS performs forward and backward propagation to update $w_{i,k,p}'$, and the gradients are sent back (e.g., step ②/⑥ in Fig. 5). Upon receiving the gradients, the airborne model on drone k is updated through backpropagation (e.g., step ③/⑦ in Fig. 5) as

$$w_{i,k,b} = w_{i,k,b}' - \mu_k \nabla\mathcal{L}_b\left(w_{i,k,b}'\right), \tag{24}$$

where μ_k is the learning rate of drone k. The complete forward and backward propagation process is called one local iteration.

C. Federated Aggregation of Split Models

After receiving the shallow aerial model parameters $w_{i,k,b}$, BS i initiates federated aggregation to update the global model (e.g., step ④/⑧ in Fig. 5). In the CSFL stage, unlike SMTL in Subsection III-C where the TE-TD resides on drones, the TE-TD is deployed on BSs, serving as the federated aggregator to align features, coordinate gradient interactions, and improve generalization across multiple drones.

The TE-TD serves a dual role in feature alignment and gradient interaction during model aggregation. When a drone transmits intermediate features to BS i, the TE encodes them into query-key-value representations, capturing cross-layer and cross-drone dependencies in the resource-aware feature streams. The TD then fuses these encoded sequences into globally contextualized representations and generates feedback signals that guide gradient propagation. In this way, the Transformer functions as a feature-and-gradient alignment bridge between the drone's shallow Bi-LSTM layers and the BS's deeper layers, ensuring that exchanged gradients are adaptively weighted to support efficient resource-slicing predictions.

Specifically, the partitioned aerial model, $w_{i,k,b}$, is first encoded by TE into an input sequence, which is then processed through a multi-layer stack to produce, for each attention head s, the corresponding query, key, and value vectors $\boldsymbol{Q}_k^{(s)}$, $\boldsymbol{K}_k^{(s)}$, $\boldsymbol{V}_k^{(s)}$. Here, $\boldsymbol{Q}_k^{(s)}$ captures global dependencies among parameters, $\boldsymbol{K}_k^{(s)}$ models contextual relationships within the sequence, and $\boldsymbol{V}_k^{(s)}$ preserves the inherent feature representations of the sliced resources. Within the federated aggregation framework, $W_{i,b}$ and $w_{i,k,b}$ are respectively used as $\boldsymbol{Q}_k^{(s)}$ and $\boldsymbol{K}_k^{(s)}$ inputs to the Transformer module.

The built-in multi-head attention mechanism computes the importance of each position based on the dot-product similarity between $\boldsymbol{Q}_k^{(s)}$ and $\boldsymbol{K}_k^{(s)}$. These attention weights, denoted as $[\gamma_1,\ldots,\gamma_{K_i}]$ for drone $k\in\mathcal{K}_i$ and $[\sigma_1,\ldots,\sigma_{I_k}]$ for BS $i\in\mathcal{I}_i$, are used to aggregate the encoded value vectors $\boldsymbol{V}_k^{(s)}$

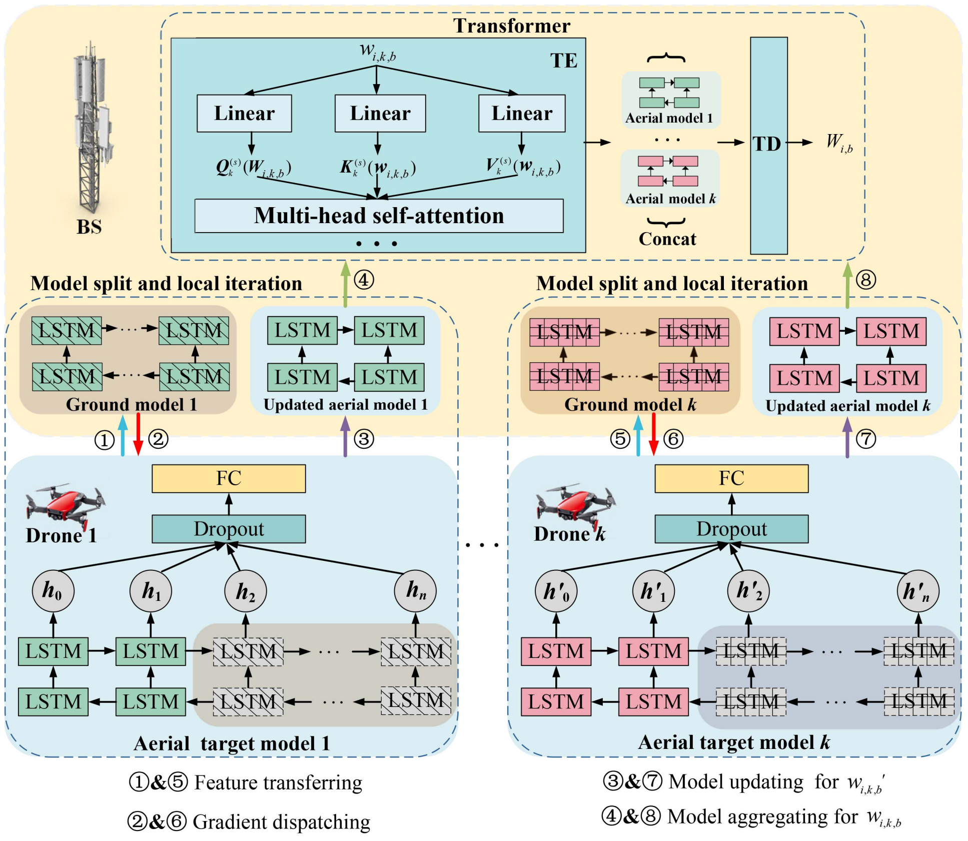


Fig. 5. Model splitting, local iteration, and Transformer-enhanced aggregation in CSFL.

and produce globally contextualized model representations

$$\gamma_k = Att(\boldsymbol{W}_{i,b}, \boldsymbol{w}_{i,k,b}, \boldsymbol{V}_k^{(s)}) = \text{softmax}\left(\frac{\boldsymbol{Q}_k^{(s)} \cdot \boldsymbol{K}_k^{(s)\top}}{\sqrt{d}}\right) \boldsymbol{V}_k^{(s)}. \tag{25}$$

This TE-TD interaction enables adaptive weighting of gradient updates, giving higher influence to more informative or stable drone models, thereby improving both convergence speed and robustness. The aggregated model parameters from drones in $\mathcal{K}_i$ are combined at BS i as

$$\boldsymbol{W}_b = \frac{1}{I_k} \sum_{i \in \mathcal{I}_k} \sigma_i \boldsymbol{W}_{i,b}, \tag{26}$$

after which the TD on BS i decodes and redistributes the updated global parameters W_b to all drones in $\mathcal{K}_i$ for the next training round.

The parameter aggregation procedure is summarized in Algorithm 2. During aggregation, BS i sequentially collects the split model parameters $w_{i,k,b}$ from the drones in $\mathcal{K}_i$. These parameters are first encoded into sequence representations $\boldsymbol{w}_{i,k,b}$ via TE (line 3), then aggregated into $\boldsymbol{W}_{i,b}$ through the multi-head attention mechanism (lines 4–6) across attention heads, and finally decoded into the updated model $W_{i,b}$ by TD (line 7). The resulting global aerial model W_b is subsequently distributed to all member drones for the next local training round. This aggregation process bridges sequence-aware local updates and global model synchronization, balancing sequence learning on drones with high-level feature fusion and generalization at the BS.

Algorithm 2. Transformer-Enhanced Split Aggregation.

```
input : $w_{i,k,b}$
output: $W_{i,b}$
1 foreach $i \in \mathcal{I}_k$ do
2     foreach $k \in \mathcal{K}_i$ do
3         $w_{i,k,b}$ is encoded as $\boldsymbol{w}_{i,k,b}$ via TD;
4         $\boldsymbol{w}_{k,i}$ is processed through multiple layers to
          generate $\boldsymbol{Q}_k^{(s)}, \boldsymbol{K}_k^{(s)}, \boldsymbol{V}_k^{(s)}$ for each head $s$;
5         $[\gamma_1, \gamma_2, \ldots, \gamma_{K_i}] \leftarrow Att(\boldsymbol{w}_{i,b}, \boldsymbol{w}_{i,k,b}, \boldsymbol{V}_k^{(s)})$;
6         $\boldsymbol{W}_{i,b} \leftarrow \frac{1}{K_i} \sum_{k \in \mathcal{K}_i} \gamma_k \boldsymbol{w}_{i,k,b}$;
7         $\boldsymbol{w}_{i,b}$ is decoded by TD as $w_{i,b}$;
```

D. Training Cost Analysis

This subsection provides a theoretical analysis of the impact of model splitting on communication and computation costs. The former refers to the data exchanged between drone k and BS i during training. Let $|w_{i,k}|$ be the number of model parameters carried by drone k. In traditional FL (e.g., FedAvg [41]), the entire parameter set $|w_{i,k}|$ must be uploaded to BS i, resulting in bi-directional transmission, which doubles the data to $2|w_{i,k}|$.

TABLE II
COST ANALYSIS PER EPOCH OF TRAINING APPROACHES

Model	Communication Cost		Computation Cost	
	drone k	BS i	drone k	BS i
Proposed	$2\lvert w_{i,k}\rvert$	$K_i 2\lvert w_{i,k}\rvert$	$\mathcal{T}(w_{i,k})$	$K_i\mathcal{T}(w_{i,k})$
FedAvg	L	$4K_iL$	$\mathcal{T}(w_{i,k,b})$	$K_i\mathcal{T}(w_{i,k,p})$

With model splitting, only the backpropagated gradients are transmitted. Assuming the size of each gradient transmission is L, and considering both forward and backward propagation, the total data size for the split model becomes $4L$. As the cluster head, BS i interacts with K_i drones within its cluster, leading to a total data volume of $4K_iL$, where K_i is the number of drones in $\mathcal{K}_i$. As long as $4L < 2\lvert w_{i,k}\rvert$, the communication cost of the proposed scheme is lower than that of FedAvg. Computation cost represents the time required for training. Only the sub-model parameters $w_{i,k,b}$ need to be updated with model splitting. Denoted by $\mathcal{T}(\cdot)$ the computation cost function. As long as $\mathcal{T}(w_{i,k,b}) \ll \mathcal{T}(w_{i,k,p}) < \mathcal{T}(w_{i,k})$, where $w_{i,k,p}$ represents the remaining part of the split model, the computation cost of our approach remains lower than that of FedAvg. Table II summarizes the costs for drone k and BS i in a training round.

V. PERFORMANCE EVALUATION

To validate the effectiveness of the proposed MobiFormer framework in realistic aerial RAN scenarios, we constructed a distributed simulation environment on a high-performance server equipped with an Intel Core i9-14900 K CPU, 64 GB DDR5 RAM, 4 TB PCIe 4.0 SSD, and two NVIDIA RTX 4090 GPUs. Each drone and BS was instantiated as an independent Docker container with localized models and private data partitions. These containers communicated over a virtual network using binary messaging protocols. To emulate real-world aerial communication behavior, the platform incorporated automatic reconnection and checkpoint-based resumption mechanisms, enabling tolerance to intermittent connectivity and node failures. The system also supported flexible scaling of drone-BS configurations to simulate deployments of varying sizes.

We leveraged a multi-source mobile communication dataset from the Trentino province in Italy [42], containing regionally distributed voice and data service records. This dataset enabled the emulation of fine-grained, temporally varying slice demands across diverse geographic zones, reflecting heterogeneous user behaviors such as video streaming, voice communication, and IoT data transmission. Drones were dynamically assigned to subregions with evolving slice demands and mobility patterns, inducing natural domain shifts. Real-time updates to drone movement, task arrivals, and service heterogeneity allowed comprehensive evaluation of the framework's adaptability to environmental variation, multi-domain TL, and collaborative model refinement.

Each drone utilizes a Bi-LSTM model with two bidirectional LSTM layers, each containing 128 hidden neurons, followed by a linear layer for prediction. Both drones and BSs are equipped with Transformers for feature fusion and parameter aggregation.

TABLE III
CATEGORIZATION OF PROPOSED METHODS FOR ABLATION

Name	Model fine-tuning & training			Slice windowing	
	SMTL	CFL	CSFL	Static	Dynamic
Proposed-1	✓			✓	
Proposed-2	✓				✓
Proposed-3		✓		✓	
Proposed-4			✓	✓	
Proposed-5			✓		✓
Proposed-6	✓	✓		✓	
Proposed-7	✓		✓	✓	
Proposed-8	✓		✓		✓

TABLE IV
CATEGORIZATION OF BASELINE METHODS

Name	TL scheme		FL scheme		
	Single domain [43]	SMTL	FedAvg [41]	Weight aggregate [44]	Self-attention [45]
Baseline-1	✓				
Baseline-2		✓			
Baseline-3				✓	
Baseline-4			✓		
Baseline-5					✓
Baseline-6		✓			✓

TABLE V
DEFAULT PARAMETER SETTINGS

Parameter	Value
Num. of drones (K)	24
Num. of terrestrial BSs (I_k)	8
Number of slices on drone k (M_k)	3
Default window size of drone k (T_k)	600sec
Window adjustment step size (ΔT_k)	10sec
Total num. of RBs on each drone (B_k)	250
Num. of shared RBs (z_k)	10, 20, 30
Local batch size for each drone	20
Learning rates (α_k, η_i, μ_k)	10^{-4}, 10^{-3}, 10^{-2}
Proportion of shared RBs (ω)	0.15

For the ablation study, the proposed MobiFormer is divided into eight categories, as shown in Table III. Proposed-1 and -2 include only SMTL; Proposed-3 belongs to a Clustered FL (CFL) approach, equivalent to CSFL without SL; Proposed-4 and -5 correspond to CSFL; Proposed-6 is equivalent to SMTL plus Proposed-3; Proposed-7 and -8 cover both SMTL and CSFL schemes, differing in slice windowing strategies. Six baseline methods in Table IV exclude SL and dynamic windowing: Baseline-1 relies solely on single-target-domain TL; Baseline-1 employs SMTL but excludes subsequent federation; Baseline-3, -4, and -5 are FL-based; Baseline-6 is a Federated Transfer Learning (FTL) method that combines SMTL and self-attention-based FL.

The method proposed by Thammawichai et al. [46] calculates communication costs, including energy consumption for sending and receiving parameters, while the method described by Tian et al. [47] quantifies computational cost, considering

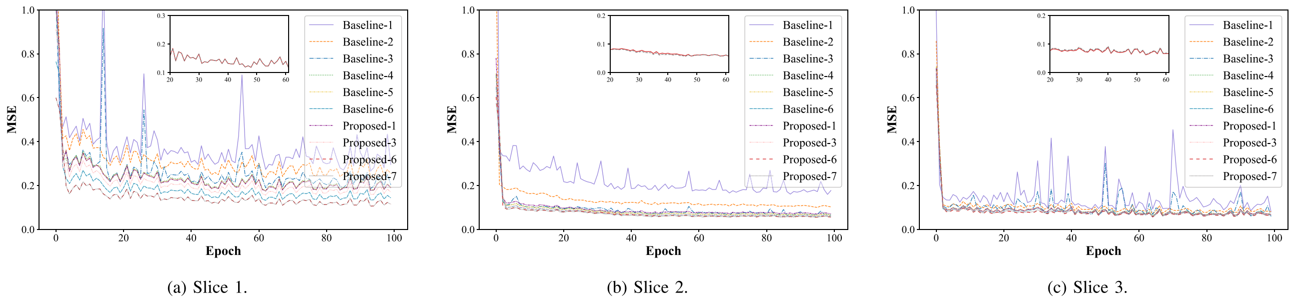


Fig. 6. Prediction error variation across communication rounds for different slices.

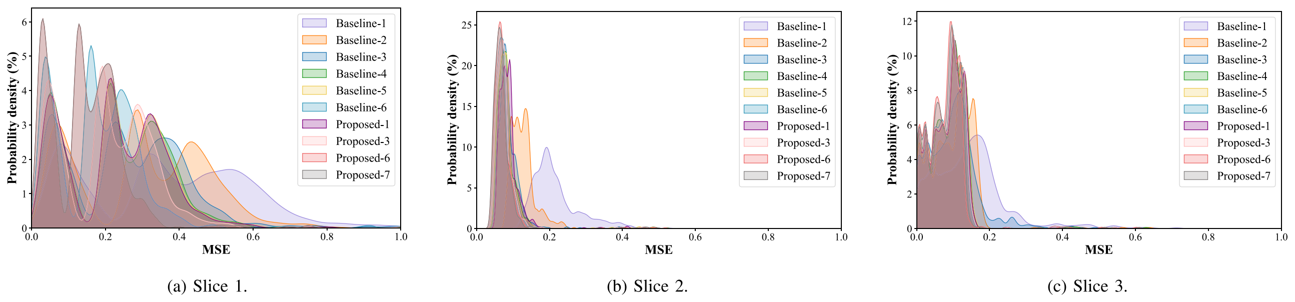


Fig. 7. PDF for prediction errors for different slices.

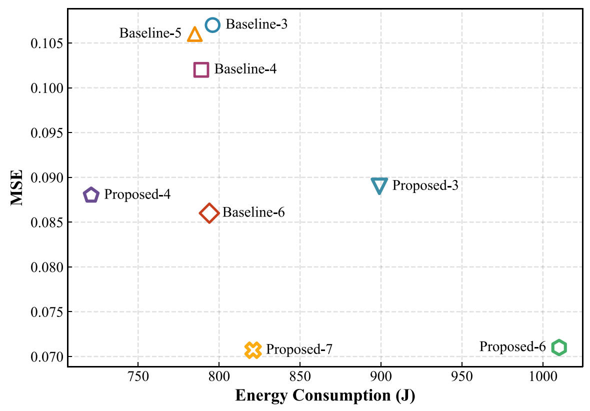


Fig. 8. MSE vs. energy.

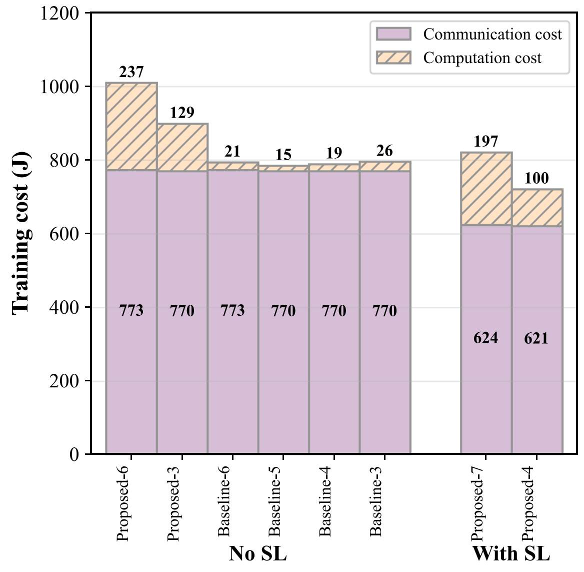
CPU usage and memory occupancy. Stochastic Gradient Descent (SGD) is chosen as the optimizer. The detailed parameter settings are shown in Table V.

A. Resource Demand Prediction Analysis

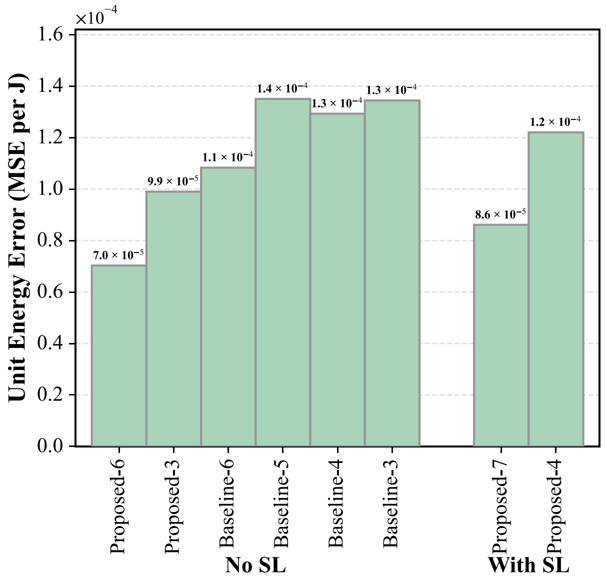
In this simulation, Mean Squared Error (MSE) measures the difference between predicted and actual resource demand. Among all eight proposed schemes, we selected four representative variants (Proposed-1, 3, 6, and 7) for detailed analysis, as they capture the main combinations of SMTL, CSFL, and windowing strategies without visual redundancy in the figures. Fig. 6 shows the prediction errors of different methods over communication rounds, illustrating changes in prediction error for each slice's resource demand. As rounds increase, the prediction errors for all slices decrease and converge. The proposed methods consistently achieve lower prediction errors than the baseline methods. Additionally, it leverages the Transformer's multi-head attention mechanism to focus on nodes that significantly impact global model updates, thereby accelerating convergence and reducing computation costs. Compared to TL-based Baseline-1 and -2, Proposed-1 integrates multi-target domain data more effectively, improving accuracy and generalization, with performance gains of 10.31%-25.54%, 29.32%-55.18%, and 7.51%-27.46% across the three slices, respectively. Proposed-3, which combines Transformer and FL, outperforms FL-based Baseline-3, -4, and -5 in handling heterogeneous data, achieving gains of 8.67%-23.78%, 7.19%-14.46%, and 2.6%-8.67% across the three slices, respectively. Proposed-6 captures inter-correlations more accurately than FTL-based Baseline-6, with performance gains of 18.10%, 7.04%, and 8.09% across the three slices.

We selected Proposed-7 and -6 (with split and unsplit models) to investigate how model splitting affects prediction accuracy. All models were configured with the same splitting point. To facilitate the observation of subtle differences, we cut the local features of Proposed-6 and -7 into each sub-graph of Fig. 6 as internal graphs. The trends of the former are almost identical to those of the latter as the number of communication rounds increases, indicating that the model splitting does not compromise prediction accuracy.

Fig. 7 illustrates the prediction errors through the Probability Density Function (PDF). For Slice 1, Proposed-6 has a 73% probability of errors being below 0.2, compared to 58% for Baseline-6. For Slice 2, the probabilities of errors being less than 0.1 are approximately 92% for Proposed-6 and 90% for Baseline-6. For Slice 3, Proposed-6 achieves a 70% probability



(a) Training energy consumption.



(b) Unit Energy Error.

Fig. 9. Training energy efficiency comparison.

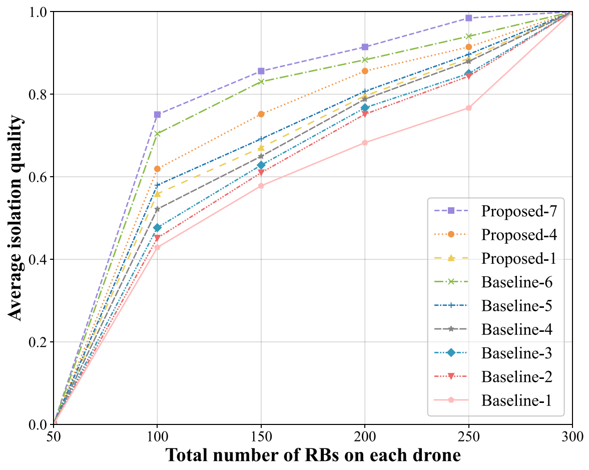


Fig. 10. Impact of resources on isolation quality.

of errors below 0.1, while Baseline-6 reaches 62%. These results demonstrate that Proposed-6 more accurately captures variations in resource demand.

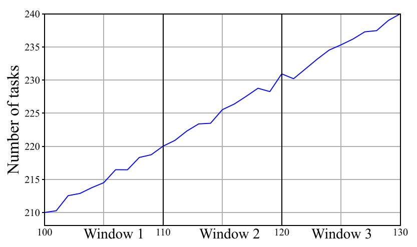
B. Training Energy Efficiency Analysis

During the SMTL phase, model fine-tuning is offloaded to multiple target-domain BSs, while the drone is responsible only for onboard model fusion, thereby minimizing its energy consumption. Hence, our energy analysis primarily focuses on the CSFL phase. Since the training energy consumption is not affected by the windowing strategy, Proposed-3, -4, -6, and -7 are selected for evaluation. For fair comparison, representative baseline methods incorporating FL, Baseline-3 through -6, are included. Based on this setup, we conduct a comprehensive analysis in terms of accuracy–energy trade-off, energy distribution, and unit energy efficiency. Furthermore, we examine the effect of introducing SL on energy consumption.

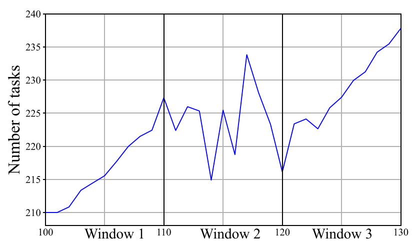
Fig. 8 illustrates the trade-off between prediction accuracy (MSE) and energy consumption (Joules). The proposed methods consistently outperform the baseline approaches, achieving lower MSE values under comparable or lower energy budgets. Notably, Proposed-7 delivers the lowest MSE while maintaining moderate energy usage, demonstrating its superior accuracy-to-energy ratio. Proposed-4 also stands out for its efficiency, achieving competitive accuracy while consuming the least energy of all methods, making it particularly suitable for deployment on energy-constrained drone platforms. In contrast, even the best-performing baseline (Baseline-6) fails to match the accuracy–energy profiles of the proposed variants. These results collectively validate MobiFormer's effectiveness in jointly optimizing accuracy and energy efficiency, reinforcing its practical viability for aerial RAN slicing.

Fig. 9(a) presents the breakdown of training energy consumption, distinguishing between computation and communication costs. The benefits of model splitting are clearly reflected: by offloading parts of the model to BSs, the framework significantly reduces both data transmission and onboard training overhead. Proposed-4 and -7 demonstrate substantial communication energy savings compared to all baseline methods, primarily due to the CSFL scheme's effective clustering and aggregation mechanisms. Proposed-4 consumes the lowest energy, attributed to efficient model offloading and BS-side Transformer deployment without invoking SMTL.

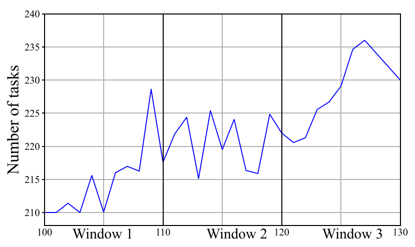
While Proposed-7 incurs higher computational energy during the SMTL phase, this overhead arises from its enhanced feature representation capability. Although SMTL is decoupled from CSFL, it alters the model initialization, leading to more complex gradient updates and requiring additional local iterations for convergence. Nevertheless, the accuracy gain offsets the extra energy cost. When considered alongside the accuracy results in Figs. 6 and 7, Fig. 9(a)



(a) Drone 1.

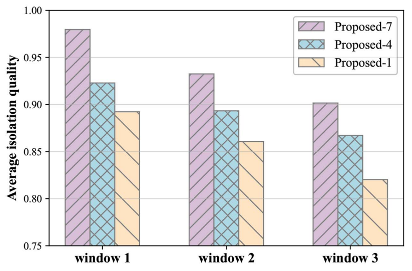


(b) Drone 2.

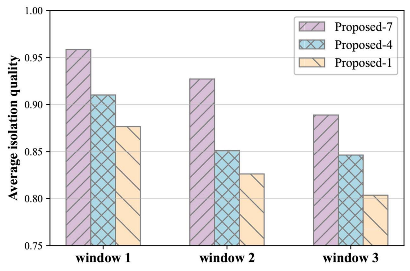


(c) Drone 3.

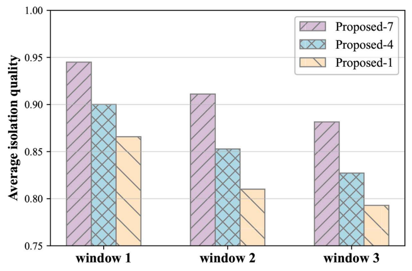
Fig. 11. Task traffic fluctuations of drones in the same cluster.



(a) Drone 1.

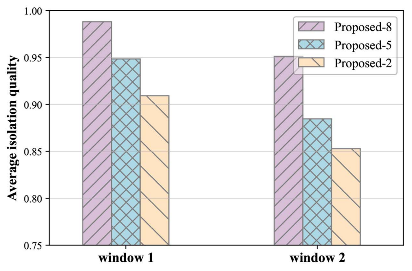


(b) Drone 2.

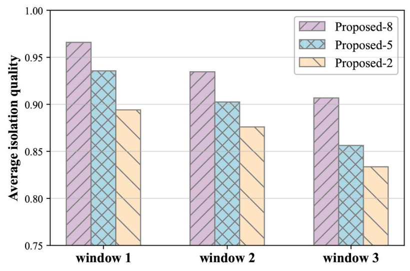


(c) Drone 3.

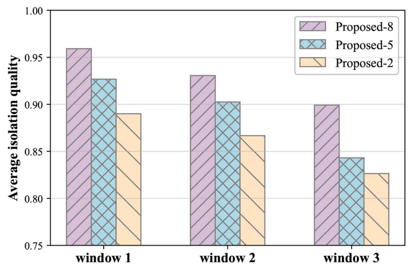
Fig. 12. Performance isolation under static windowing.



(a) Drone 1.



(b) Drone 2.



(c) Drone 3.

Fig. 13. Performance isolation under dynamic windowing.

demonstrates that Proposed-4 and -7 achieve the best trade-off between energy consumption and prediction accuracy, demonstrating superior cost-effectiveness in aerial collaborative learning.

Fig. 9(b) further analyzes unit energy efficiency, defined as prediction error per joule consumed. The proposed methods consistently outperform baseline approaches across this metric. Among them, one variant achieves the lowest per-energy-unit prediction error, indicating high energy efficiency. Another variant, while not the most energy-frugal, achieves the best absolute accuracy with still-competitive energy use, making it ideal for accuracy-sensitive RAN slicing. Even the most efficient baseline lags behind all proposed models, highlighting the inefficiencies of conventional FL-based designs. These findings reinforce that MobiFormer not only enhances predictive accuracy but also significantly improves energy utilization, making it highly suitable for real-world, resource-constrained drone-based RAN deployments.

C. Impact of Resource Pressure on Performance Isolation

We further examined how varying levels of available RBs affect performance isolation. To ensure fairness, all methods adopted the same windowing strategy. As illustrated in Fig. 10, all methods exhibit a noticeable improvement in isolation quality as resource pressure decreases in the early phase. Among them, Proposed-7 consistently achieves the highest performance isolation, underscoring the effectiveness of the Transformer-based multi-head attention mechanism in prioritizing critical features during model aggregation and optimizing resource allocation. With the increase in resource availability, the performance isolation of all methods approaches an ideal state. Specifically,

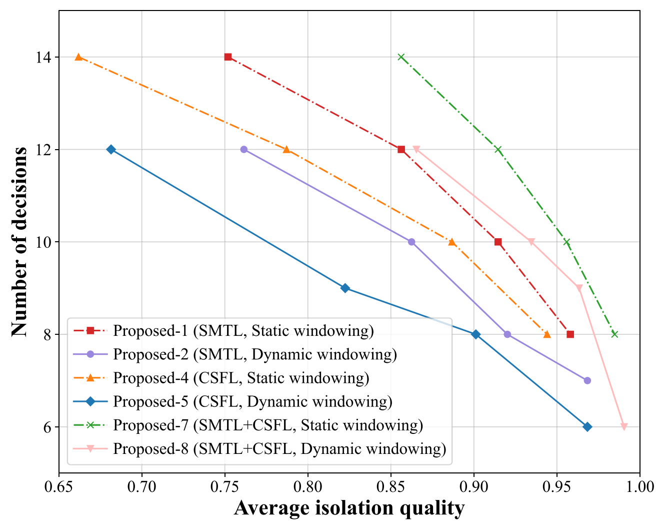


Fig. 14. Impact of slice windowing on decision efficiency.

Proposed-1 delivers improvements of 18.55% and 9.62% in slice performance isolation quality over TL-based Baseline-1 and -2, respectively. Compared to FL-based Baseline-3, -4, and -5, Proposed-4 achieves gains of 15.39%, 10.66%, and 5.58%, respectively. The result of Proposed-7 also demonstrates a 4.38% improvement over FTL-based Baseline-6. These gains are attributed to the accurate prediction of slice resource demand.

When combined with the running cost results presented in Fig. 9, the results confirm that the proposed models deliver superior performance isolation at comparable training costs. These results demonstrate that MobiFormer not only optimizes training cost and prediction accuracy but also sustains high-quality service provisioning under varying system loads.

D. Impact of Windowing on Performance Isolation

This subsection explores the impact of window partitioning strategies on performance isolation. In the experimental scenario, task arrivals over a selected period for a cluster of three drones were captured and visualized, as shown in Fig. 11. Fig. 12 illustrates the static windowing strategy, which divides the timeline into evenly spaced intervals, while Fig. 13 presents the dynamic windowing strategy that generates unevenly spaced windows based on observed task dynamics. As seen in Figs. 12(b), (c) and Figs. 13(b), (c), the average isolation quality of Proposed-7 and -8 consistently exceeds that of Proposed-4 and -5 (based on CSFL only) and Proposed-1 and -2 (based solely on SMTL). This observation confirms that the window-adaptive resource slicing strategy enables more effective performance isolation, particularly under time-varying task loads. Notably, by adjusting the window size in response to task arrival patterns, the framework can reduce decision frequency and overhead without sacrificing performance isolation. This balance demonstrates MobiFormer's flexibility and adaptability in handling fluctuating service demands across distributed drone networks.

E. Online Decision Efficiency Analysis

The real-time prediction and slice demand estimation are performed by a lightweight Bi-LSTM model with fewer than 150 K parameters, ensuring the online inference module remains highly efficient and suitable for drone deployment. Importantly, the resource slicing decision module does not require continuous or high-frequency inference. Instead, under our window-adaptive slicing framework, resource reallocation decisions are triggered periodically rather than constantly. This significantly reduces inference frequency and ensures that the latency incurred per decision is negligible in practice.

Lastly, we evaluated the impact of windowing strategies on long-term decision efficiency, referring to resource reallocations triggered (equivalent to the number of time windows divided) under given performance isolation levels. From Fig. 14(a)–(c), the proposed dynamic windowing strategy has a lower (higher) long-term decision cost (performance isolation) compared to the static window under the same performance isolation (decision cost). While dynamic windowing may incur high online decision costs during significant task fluctuations, it offers lower decision costs over more extended periods (e.g., a day). Proposed-8, with dynamic windows, reduces decision costs by 8.89% to 15.83% compared to Proposed-7, which uses static windowing. Proposed-5 outperforms Proposed-4 with static windowing and CSFL with a 19.51%–27.78% reduction in online decision costs, while Proposed-2 offers a 9.33%–15.83% improvement over Proposed-1 with static windowing and SMTL. Together with the observations from Figs. 12 and 13, these findings demonstrate that dynamic windowing, as a built-in optimization strategy, offers a favorable balance between decision efficiency and performance isolation.

VI. Conclusion

This study presents MobiFormer, a Transformer-enhanced distributed learning framework for drone RAN slicing, designed to achieve the dual-optimization of performance isolation and energy efficiency. For single-drone autonomous scenarios, we introduce an SMTL method based on an "unconstrained mobility and sustainable fine-tuning" paradigm, enabling drones to adapt their slicing models efficiently in new work environments. To support scalable multi-drone collaborative training, we develop a CSFL scheme in which terrestrial BSs serve as aggregation points to reduce communication and computation costs. Extensive simulations confirm the effectiveness of MobiFormer. Compared to baseline methods, our adaptive slice windowing strategy significantly reduces the long-term decision-making cost under comparable levels of performance isolation. Conversely, under the same decision cost, MobiFormer achieves superior performance isolation, validating its robustness in resource-constrained and dynamic aerial networks. The MobiFormer framework can be extended to other drone applications such as target detection, environmental sensing, and coordinated data collection.

References

[1] M. Dai, G. Sun, H. Yu, and D. Niyato, "Maximize the long-term average revenue of network slice provider via admission control among heterogeneous slices," *IEEE/ACM Trans. Netw.*, vol. 32, no. 1, pp. 745–760, 2024.

[2] H. Shen, Y. Tian, T. Wang, and G. Bai, "Slicing-based task offloading in space-air-ground integrated vehicular networks," *IEEE Trans. Mob. Comput.*, vol. 23, no. 5, pp. 4009–4024, May 2024.

[3] F. Wei, G. Feng, S. Qin, Y. Peng, and Y. Liu, "Hierarchical network slicing for UAV-assisted wireless networks with deployment optimization," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 12, pp. 3705–3718, Dec. 2024.

[4] H. Shen, Z. Tong, T. Wang, and G. Bai, "UAV-relay-assisted live layered video multicast for cell-edge users in NOMA networks," *IEEE Trans. Broadcast.*, vol. 70, no. 1, pp. 135–147, Mar. 2024.

[5] H. Shen, Y. Heng, N. Shi, T. Wang, and G. Bai, "Drone-small-cell-assisted spectrum management for 5G and beyond vehicular networks," in *Proc. IEEE Int. Symp. Comput. Commun.*, 2022, pp. 1–8.

[6] H. Shen, Q. Ye, W. Zhuang, W. Shi, G. Bai, and G. Yang, "Drone-small-cell-assisted resource slicing for 5G uplink radio access networks," *IEEE Trans. Veh. Technol.*, vol. 70, no. 7, pp. 7071–7086, Jul. 2021.

[7] D. Basu, U. Ghosh, and R. Datta, "SoftFloat: Softwarized 5G-driven network slicing framework for QoS aware UAV communications," in *Proc. ACM MobiCom Workshop Drone Assist. Wireless Commun. 5G Beyond*, 2022, pp. 13–18.

[8] Q. Liu, N. Choi, and T. Han, "Atlas: Automate online service configuration in network slicing," in *Proc. ACM Int. Conf. Emerg. Netw. Exp. Technol.*, 2022, pp. 140–155.

[9] P. Yang, X. Xi, K. Guo, T. Q. Quek, J. Chen, and X. Cao, "Proactive UAV network slicing for URLLC and mobile broadband service multiplexing," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 10, pp. 3225–3244, Oct. 2021.

[10] L. Le, T. N. Nguyen, K. Suo, and J. He, "5G network slicing and drone-assisted applications: A deep reinforcement learning approach," in *Proc. ACM Workshop Drone-Assist. Wireless Commun. 5G Beyond*, 2022, pp. 109–114.

[11] H. Hao, C. Xu, W. Zhang, S. Yang, and G.-M. Muntean, "Joint task offloading, resource allocation, and trajectory design for multi-UAV cooperative edge computing with task priority," *IEEE Trans. Mob. Comput.*, vol. 23, no. 9, pp. 8649–8663, Sep. 2024.

[12] M. Wang, Y. Lin, Q. Tian, and G. Si, "Transfer learning promotes 6G wireless communications: Recent advances and future challenges," *IEEE Trans. Reliab.*, vol. 70, no. 2, pp. 790–807, Jun. 2021.

[13] F. Shi, W. Lin, X. Wang, K. Li, and A. Y. Zomaya, "The analysis and optimization of volatile clients in over-the-air federated learning," *IEEE Trans. Mob. Comput.*, vol. 23, no. 12, pp. 13144–13157, Dec. 2024.

[14] C. Thapa, P. C. M. Arachchige, S. Camtepe, and L. Sun, "SplitFed: When federated learning meets split learning," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 8485–8493.

[15] H. Fu, J. Wang, J. Chen, P. Ren, Z. Zhang, and G. Zhao, "Dense multiagent reinforcement learning aided multi-UAV information coverage for vehicular networks," *IEEE Internet Things J.*, vol. 11, no. 12, pp. 21274–21286, Jun. 2024.

[16] H. Bai, H. Wang, J. Du, R. He, G. Li, and Y. Xu, "Multi-hop UAV relay covert communication: A multi-agent reinforcement learning approach," in *Proc. Int. Conf. Ubiquitous Commun.*, 2024, pp. 356–360.

[17] Z. Sun, Z. Yu, B. Guo, B. Yang, Y. Zhang, and D. W. K. Ng, "Integrated sensing and communication for effective multi-agent cooperation systems," *IEEE Commun. Mag.*, vol. 62, no. 9, pp. 68–73, Sep. 2024.

[18] C. T. Nguyen et al., "Transfer learning for wireless networks: A comprehensive survey," in *Proc. IEEE*, vol. 110, no. 8, pp. 1073–1115, Aug. 2022.

[19] G. Yu, "Data-free knowledge distillation for privacy-preserving efficient UAV networks," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2022, pp. 52–56.

[20] B. Li, Z.-P. Yang, D.-Q. Chen, S.-Y. Liang, and H. Ma, "Maneuvering target tracking of UAV based on MN-DDPG and transfer learning," *Def. Technol.*, vol. 17, no. 2, pp. 457–466, 2021.

[21] Y. Xu, J. Zha, J. Ren, X. Jiang, H. Zhang, and X. Chen, "Scalable multi-agent reinforcement learning for effective UAV scheduling in multi-hop emergency networks," in *Proc. Annu. Int. Conf. Mob. Comput. Netw.*, 2024, pp. 2028–2033.

[22] Y. Lu, Z. Liu, and Y. Huang, "Parameters compressed mechanism in federated learning for edge computing," in *Proc. IEEE Int. Conf. Edge Comput. Scalable Cloud*, 2021, pp. 161–166.

[23] H. Wang, H. Zhao, and B. Li, "Bridging multi-task learning and meta-learning: Towards efficient training and effective adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 10991–11002.

[24] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4298–4311, Apr., 2020.

[25] X. Liu, Y. Deng, and T. Mahmoodi, "Wireless distributed learning: A new hybrid split and federated learning approach," *IEEE Trans. Wirel. Commun.*, vol. 22, no. 4, pp. 2650–2665, Apr. 2023.

[26] J. Li et al., "A hierarchical soft RAN slicing framework for differentiated service provisioning," *IEEE Wirel. Commun.*, vol. 27, no. 6, pp. 90–97, Dec. 2020.

[27] Q. Ye, W. Zhuang, S. Zhang, A.-L. Jin, X. Shen, and X. Li, "Dynamic radio resource slicing for a two-tier heterogeneous wireless network," *IEEE Trans. Veh. Technol.*, vol. 67, no. 10, pp. 9896–9910, Oct. 2018.

[28] G. Brauwers and F. Frasincar, "A general survey on attention mechanisms in deep learning," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 3279–3298, Apr. 2023.

[29] O. B. Tanmoy, M. Al Mamun, S. Hasan, and A. Anwar, "Enhancing federated learning with globally shared model: A modified FedAVG approach (GSM-FedAVG)," in *Proc. Int. Conf. Electr. Inf. Commun. Technol.*, 2023, pp. 1–6.

[30] Y. Zhao, "Comparison of federated learning algorithms for image classification," in *Proc. Int. Conf. Data Analytics Comput. Artif. Intell.*, 2023, pp. 613–615.

[31] Z. Ma, H. Gong, and X. Wang, "Cooperative optimal control of UAVs based on brain-attention mechanism neural network," in *Proc. Int. Conf. Automat. Robot. Appl.*, 2023, pp. 263–267.

[32] J. Ren, Y. Xu, Z. Li, C. Hong, X.-P. Zhang, and X. Chen, "Scheduling UAV swarm with attention-based graph reinforcement learning for ground-to-air heterogeneous data communication," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput. ACM Int. Symp. Wearable Comput.*, 2023, pp. 670–675.

[33] X. Lin et al., "5G new radio: Unveiling the essentials of the next generation wireless access technology," *IEEE Commun. Mag.*, vol. 3, no. 3, pp. 30–37, Sep. 2019.

[34] M. He, H. Wu, C. Zhou, and X. Shen, "Resource slicing with cross-cell coordination in satellite-terrestrial integrated networks," in *Proc. IEEE Int. Conf. Commun.*, 2024, pp. 2501–2506.

[35] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process.*, 2017, pp. 5998–6008.

[36] Z. Cao, Y. Zhou, A. Yang, and S. Peng, "Deep transfer learning mechanism for fine-grained cross-domain sentiment classification," *Connect. Sci.*, vol. 33, no. 4, pp. 911–928, 2021.

[37] P. Xu, C. Wang, J. Ye, and T. Ouyang, "State-of-charge estimation and health prognosis for lithium-ion batteries based on temperature-compensated Bi-LSTM network and integrated attention mechanism," *IEEE Trans. Ind. Electron.*, vol. 71, no. 6, pp. 5586–5596, Jun. 2024.

[38] L. Hebert, L. Golab, P. Poupart, and R. Cohen, "FedFormer: Contextual federation with attention in reinforcement learning," in *Proc. Int. Jt Conf. Auton. Agents Multiagent Syst.*, 2023, pp. 810–818.

[39] Y. Liao, Y. Xu, H. Xu, L. Wang, Z. Yao, and C. Qiao, "MergeSFL: Split federated learning with feature merging and batch size regulation," in *Proc. IEEE Int. Conf. Data Eng.*, 2024, pp. 2054–2067.

[40] T. Kim, J. Oh, N. Y. Kim, S. Cho, and S.-Y. Yun, "Comparing kullback-leibler divergence and mean squared error loss in knowledge distillation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. 2628–2635.

[41] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "FedAvg with fine tuning: Local updates lead to representation learning," in *Proc. Adv. Neural. Inf. Process. Syst.*, 2022, pp. 10572–10586.

[42] G. Barlacchi et al., "A multi-source dataset of urban life in the city of Milan and the province of trentino," *Sci. Data*, vol. 2, no. 1, pp. 1–15, 2015.

[43] M. Woźniak, M. Wieczorek, and J. Siłka, "Deep neural network with transfer learning in remote object detection from drone," in *Proc. ACM Workshop Drone-Assist. Wireless Commun. 5G Beyond*, 2022, pp. 121–126.

[44] Z. Li, T. Lin, X. Shang, and C. Wu, "Revisiting weighted aggregation in federated learning with neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 19767–19788.

[45] N. K. Singh et al., “Self-attention mechanism based federated learning model for cross context recommendation system,” *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 2687–2695, Feb. 2024.
[46] M. Thammawichai, S. P. Baliyarasimhuni, E. C. Kerrigan, and J. B. Sousa, “Optimizing communication and computation for multi-UAV information gathering applications,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 54, no. 2, pp. 601–615, Apr. 2018.
[47] Y. Tian, Y. Wan, L. Lyu, D. Yao, H. Jin, and L. Sun, “FedBERT: When federated learning meets pre-training,” *ACM Trans. Intell. Syst.*, vol. 13, no. 4, pp. 1–26, 2022.

Hang Shen (Member, IEEE) received the PhD degree with honors in computer science from the Nanjing University of Science and Technology, Nanjing, China, in 2015. He worked as a full-time postdoctoral fellow with the Broadband Communications Research (BBCR) Lab, Electrical and Computer Engineering Department, University of Waterloo, Waterloo, ON, Canada, from 2018 to 2019. He is an associate professor with the Department of Computer Science and Technology, Nanjing Tech University, Nanjing, China. His research interests include involve space-air-ground integrated networks, drone/vehicle vision-language navigation, and cybersecurity. He serves as an associate editor for *Journal of Information Processing Systems*, *Frontiers in Blockchain*, and *IEEE Access*, and was a guest editor for *Peer-to-Peer Networking and Applications*. He was a Program Committee Member of the IEEE International Conference on High Performance Computing and Communications (HPCC), the Annual International Conference on Privacy, Security and Trust (PST), the International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP), and the International Conference on Artificial Intelligence Computing and Systems (AICompS). He is a CCF Senior Member and an Executive Committee Member of the ACM Nanjing Chapter.

Yanke Yao received the BEng degree in network engineering from the North China University of Water Resources and Electric Power, Zhengzhou, China, in 2022, and the MEng degree in computer science from Nanjing Tech University, Nanjing, China, in 2025. Her research interests include distributed learning for drone communications and networking, as well as radio access network slicing.

Tianjing Wang (Member, IEEE) received the BSc degree in mathematics from Nanjing Normal University, in 2000, the MSc degree in mathematics from Nanjing University, in 2002, and the PhD degree in signal and information system from the Nanjing University of Posts and Telecommunications (NUPT), in 2009, Nanjing, China. From 2011 to 2013, she was a full-time postdoctoral fellow with the School of Electronic Science and Engineering, NUPT. From 2013 to 2014, she served as a visiting scholar with the Electrical and Computer Engineering Department, State University of New York at Stony Brook, NY, USA. She is an associate professor with the Communication Engineering Department, Nanjing Tech University, Nanjing, China. Her research interest includes the Internet of Vehicles. She is a CCF member.

Guangwei Bai received the BEng and MEng degrees in computer engineering from Xi’an Jiaotong University, Xi’an, China, in 1983 and 1986, respectively, and the PhD degree in computer science from the University of Hamburg, Hamburg, Germany, in 1999. From 1999 to 2001, he worked as a research scientist with the German National Research Center for Information Technology, Bonn, Germany. In 2001, he joined the University of Calgary, Calgary, AB, Canada, as a research associate. Since 2005, he has been working with Nanjing Tech University, Nanjing, China, as a professor in computer science. From October to December 2010, he served as a visiting professor with the Electrical and Computer Engineering Department, University of Waterloo, Waterloo, ON, Canada. His research interests include wireless networking and cybersecurity. He has authored and co-authored more than 80 peer-reviewed papers in international journals and conferences. He is a CCF Distinguished Member.