Highlights

ADAMT: Adaptive Distributed Multi-task Learning for Efficient Image Recognition in Mobile Ad-hoc Networks

Jia Zhao, Wei Zhao, Yunan Zhai, Liyuan Zhang, Yan Ding

- Proposed an adaptive distributed multi-task learning framework for efficient image recognition.
- Developed a lightweight image recognition model with feature expansion and deep hashing for enhanced expressiveness.
- Designed an adaptive communication strategy based on network conditions and node reliability for optimized model updates.

ADAMT: Adaptive Distributed Multi-task Learning for Efficient Image Recognition in Mobile Ad-hoc Networks

Jia Zhao[a,b,c], Wei Zhao[d], Yunan Zhai[e], Liyuan Zhang[b,c], Yan Ding[b,c,*]

[a]*School of Electronics Engineering and Computer Science, Peking University, Beijing, China*
[b]*School of Computer Technology and Engineering, Changchun Institute of Technology, Changchun, China*
[c]*College of Artificial Intelligence Technology, Changchun Institute of Technology, Changchun, China*
[d]*School of Computer Science and Engineering, Changchun University of Technology, Changchun, China*
[e]*Education Examinations Authority of Jilin Province, Changchun, China*

Abstract

Distributed machine learning in mobile adhoc networks faces significant challenges due to the limited computational resources of devices, non-IID data distribution, and dynamic network topology. Existing approaches often rely on centralized coordination and stable network conditions, which may not be feasible in practice. To address these issues, we propose an adaptive distributed multi-task learning framework called ADAMT for efficient image recognition in resource-constrained mobile ad hoc networks. ADAMT introduces three key innovations: 1) a feature expansion mechanism that enhances the expressiveness of local models by leveraging task-specific information; 2) a deep hashing technique that enables efficient on-device retrieval and multi-task fusion; and 3) an adaptive communication strategy that dynamically adjusts the model updating process based on network conditions and node reliability. The proposed framework allows each device to perform personalized model training on its local dataset while collaboratively updating the shared parameters with neighboring nodes. Extensive exper-

*Corresponding author
Email addresses: zhaojia@ccit.edu.cn (Jia Zhao), 2202103111@ccut.edu.cn (Wei Zhao), zhynan@jleea.com.cn (Yunan Zhai), zhangly@ccit.edu.cn (Liyuan Zhang), dingyan@ccit.edu.cn (Yan Ding)

iments on the ImageNet dataset demonstrate the superiority of ADAMT over state-of-the-art methods. ADAMT achieves a top-1 accuracy of 0.867, outperforming existing distributed learning approaches. Moreover, ADAMT significantly reduces the communication overhead and accelerates the convergence speed by 2.69 times compared to traditional distributed SGD. The adaptive communication strategy effectively balances the trade-off between model performance and resource consumption, making ADAMT particularly suitable for resource-constrained environments. Our work sheds light on the design of efficient and robust distributed learning algorithms for mobile adhoc networks and paves the way for deploying advanced machine learning applications on edge devices.

Keywords: Mobile Adhoc Networks, Distributed Multi-Task Learning, Image Recognition, Decentralized Learning,

---

1. Introduction

Mobile Ad-hoc Networks (MANETs) are decentralized, infrastructure-free networks that provide a communication environment where mobile devices can autonomously connect and share information without the need for fixed infrastructure. The flexibility of MANETs allows them to adapt dynamically to changing environments, making them highly applicable in scenarios where traditional infrastructure is either unavailable or impractical to deploy, such as in disaster recovery, military operations, and real-time intelligent transportation systems[1, 2]. However, the same characteristics that make MANETs valuable for such applications also create substantial challenges for implementing efficient machine learning frameworks. MANETs operate under strict constraints, such as limited computational resources, bandwidth limitations, and constantly evolving network topologies. These constraints place additional demands on machine learning models, necessitating solutions that are not only resource-efficient but also capable of adapting to the unique conditions of a decentralized network environment[3, 4].

The central scientific challenge in enabling effective machine learning within MANETs lies in designing a distributed framework capable of addressing data heterogeneity, dynamic connectivity, and resource limitations while preserving data privacy. Traditional distributed machine learning (DML) frameworks often rely on centralized data centers or servers for model training and parameter updates. While this centralized approach can efficiently

pool computational resources, it does not align well with the decentralized, infrastructure-free nature of MANETs, where continuous access to a central server cannot be guaranteed[5]. Although mobile devices are becoming increasingly capable in terms of storage and computation, leveraging their full potential for collaborative learning requires a decentralized approach that reduces or eliminates reliance on any central server or data aggregation hub.

Federated learning partially addresses this challenge by enabling collaborative model training across devices, with a server aggregating model updates from each device to maintain consistency across the global model. This approach has shown promise in leveraging the computational power and private data of mobile devices, facilitating learning across multiple devices without requiring centralized data sharing. However, federated learning's dependency on a central server for aggregating updates presents limitations in MANET environments where server access may be intermittent or absent altogether. In such resource-constrained or large-scale MANETs, federated learning can encounter scalability issues, high communication costs, and a lack of resilience due to its reliance on central aggregation[6, 7].

In recent years, fully decentralized approaches like distributed stochastic gradient descent (SGD) have been proposed to overcome these limitations by enabling devices to share information directly with their neighbors, effectively eliminating the need for a central server. This peer-to-peer approach to collaborative learning enhances scalability by distributing communication loads across the network, thus reducing the dependency on centralized infrastructure. However, most existing decentralized SGD methods assume that data is independently and identically distributed (IID) across devices—a condition that is rarely met in real-world applications, especially in MANET environments. In practice, MANETs often exhibit highly heterogeneous data distributions due to the unique contexts, tasks, and users associated with each device[8, 9]. This data heterogeneity, coupled with the limited bandwidth and frequent topology changes inherent to MANETs, necessitates a learning framework that can not only adapt to diverse local data but also support efficient collaboration among devices without overloading the network. The challenge, therefore, is to develop a decentralized learning system that remains resilient to non-IID data, dynamic connectivity, and resource constraints, while optimizing both communication efficiency and model performance.

In this work, we focus on a distributed learning scenario within MANETs where the data available to each computing node is heterogeneous, reflecting

the diverse tasks, users, and environments associated with different devices. Each node possesses unique data and is tasked with training a local model suited to its specific data distribution and usage context, while collaborating with neighboring nodes to improve overall learning outcomes. This setup requires each node to both learn independently on its local data and participate in a broader, decentralized learning framework to enhance model generalization across the network.

To address these challenges, this paper introduces ADAMT, an adaptive distributed multi-task learning (MTL) framework, designed specifically for MANETs. ADAMT is unique in its ability to dynamically allocate tasks based on each device's data characteristics and usage patterns, enabling the model to learn and adapt across diverse tasks in a decentralized manner. This adaptability is crucial in MANET environments, where the availability and nature of data can vary widely across devices. To achieve resource-efficient learning, ADAMT integrates a deep hashing algorithm within a lightweight image recognition model, which enhances the model's adaptability to different task-specific requirements with minimal computational overhead. The deep hashing mechanism provides robust feature extraction capabilities, allowing the model to handle non-IID data distributions more effectively across devices.

Furthermore, ADAMT includes an innovative adaptive communication strategy designed to optimize resource allocation and reduce communication costs. Each device dynamically adjusts its probability of interaction with neighboring nodes based on various performance metrics, such as link quality, node reliability, and communication latency. This adaptive strategy is formalized through a probability matrix that captures the likelihood of communication with each neighboring device, which is then optimized using the Hungarian algorithm. By selecting the most reliable and efficient connections, this communication mechanism helps minimize the overall communication overhead while enhancing model accuracy and resource efficiency within MANETs. This ensures that the ADAMT framework is both scalable and resilient to the frequent topology changes and resource constraints characteristic of MANETs.

The main contributions of this paper are as follows:

- We propose a fully decentralized learning framework for MANETs, leveraging mobile computing resources to achieve model improvements without dependency on a central server. This framework addresses

the resilience and scalability challenges associated with traditional server-based architectures, making it particularly suitable for dynamic, infrastructure-free environments;

- We introduce a feature expansion mechanism based on deep hashing, which enhances the expressiveness and robustness of local models to handle non-IID data distributions effectively across devices. This mechanism allows individual devices to retain and leverage task-specific information, facilitating generalized model improvement across heterogeneous data;
- We design an adaptive communication strategy that dynamically selects neighbors for efficient parameter updates, significantly reducing communication costs and power loss. This strategy enables the model to adapt seamlessly to MANET's dynamic conditions, optimizing both learning performance and resource allocation.

The rest of this paper is organized as follows. Section II reviews related work on cross-device training, federated learning, and decentralized SGD methods in mobile computing environments. Section III describes the design of the ADAMT framework, including its adaptive task allocation and communication strategies. Section IV presents the system architecture, distributed multi-task image recognition model, and adaptive communication mechanism. Section V evaluates the performance of ADAMT through experiments, highlighting its efficiency and scalability compared with existing methods. Finally, Section VI concludes the paper and discusses potential future research directions.

2. Related Work

The challenges of distributed machine learning in dynamic, resource-constrained environments like Mobile Ad-hoc Networks (MANETs) have spurred diverse research aimed at achieving efficient, adaptive, and privacy-preserving learning frameworks. MANETs are particularly challenging due to their lack of central infrastructure, high data heterogeneity, and frequently changing network topology. These unique requirements demand approaches that carefully balance efficiency, scalability, adaptability, and robustness. Consequently, research in this field has evolved along several key directions, each tackling specific aspects of distributed learning within decentralized environments. Below, we review major approaches, including federated

learning, decentralized stochastic gradient descent (SGD), multi-task learning (MTL), and topology-aware communication strategies. This organized review elucidates both the advances and limitations of current approaches, setting the stage for the innovations presented in this work.

Federated Learning for Privacy, Scalability, and Communication Efficiency. Federated Learning (FL) has become a foundational approach for distributed machine learning due to its ability to conduct collaborative model training without compromising data privacy. By keeping raw data on client devices and only transmitting model updates to a central server, FL offers privacy-preserving benefits that make it attractive in environments with sensitive data. Recent FL research has emphasized optimizing communication efficiency and scalability to better support decentralized settings with limited resources. For example, Kairouz et al. [10] provided a comprehensive overview of advances and open problems in FL, highlighting challenges related to communication efficiency and system heterogeneity. Li et al. [11] introduced FedProx, a federated optimization algorithm designed to tackle heterogeneity in federated networks, enhancing robustness to varying device capabilities and data distributions. Bonawitz et al. [12] discussed system design considerations for scaling FL to industrial-scale applications, addressing issues such as client selection and secure aggregation. Wang et al.[13] presented an adaptive federated learning strategy designed for resource-constrained edge computing environments, which is particularly relevant for MANETs with limited bandwidth and computing power. Furthermore, Zhang et al.'s FedGK framework [14], which integrates group-guided knowledge distillation. Despite these advancements, FL remains fundamentally dependent on a central server for global model aggregation, limiting scalability and resilience in fully decentralized networks. In MANETs, where intermittent connectivity and network variability are common, this dependency creates bottlenecks that hinder efficient learning and adaptation. Furthermore, many FL methods are not inherently designed to address extreme data heterogeneity or dynamic topology changes, common in MANETs, underscoring the need for decentralized solutions that can operate independently of a central coordinator.

Decentralized Stochastic Gradient Descent for Peer-to-Peer Optimization. To address the limitations of central aggregation in FL, decentralized SGD algorithms facilitate direct peer-to-peer communication, enabling nodes to collaboratively update models without a central server. This approach is particularly relevant in MANETs, where decentralized and resilient com-

munication is necessary to accommodate rapid topology shifts. Lian et al. [15] demonstrated that decentralized algorithms could outperform centralized ones in certain scenarios, providing insights into the potential of decentralized parallel SGD. Tang et al. [16] proposed a communication-efficient decentralized SGD algorithm that reduces communication overhead while maintaining convergence speed. Assran et al. [17] introduced Stochastic Gradient Push, a decentralized SGD method that achieves linear convergence rates without requiring a central coordinator. Additionally, Luo et al. [18] presented a dynamic decentralized SGD approach with adaptive learning rates, particularly beneficial in MANET settings where network conditions are variable. Lin et al. [19] introduced AdaptSFL, an adaptive split federated learning model that dynamically adjusts to network and resource conditions in decentralized settings, highlighting scalability improvements in highly variable environments. While decentralized SGD reduces reliance on centralized structures and is well-suited to dynamic environments, it typically assumes IID data across nodes, a condition rarely met in real-world MANETs where each device may have unique data distributions.Furthermore, many decentralized SGD algorithms are not optimized for the non-static connectivity and high variability inherent to MANETs, often requiring adaptive learning strategies to handle frequent topology changes and diverse data sources. Thus, while decentralized SGD offers promising scalability and fault tolerance, it falls short of meeting the complex requirements posed by MANETs.

Multi-Task Learning for Managing Heterogeneous Data and Task Diversity. Multi-Task Learning (MTL) frameworks have shown promise in distributed settings, enabling nodes to collaboratively learn across diverse tasks while adapting to their unique data distributions. MTL is particularly beneficial in MANETs, where data heterogeneity is the norm, as it allows shared knowledge across tasks while preserving each node's specific learning requirements. Smith et al. [20] developed MOCHA, a communication-efficient MTL framework tailored for federated settings, addressing challenges related to system and statistical heterogeneity. Sattler et al. [21] proposed Clustered Federated Learning, which groups clients based on data similarity to improve learning efficiency and model performance. Hanzely and Richtárik [22] introduced Federated Learning of a Mixture of Global and Local Models, balancing global collaboration with local personalization to handle data heterogeneity. These developments enhance the applicability of MTL in decentralized networks, but challenges remain regarding adaptability in highly dynamic settings like MANETs. Moreover, FedKD by Nguyen and Lee [23]

introduced a knowledge distillation approach to enhance MTL systems' efficiency in non-IID data conditions, which is relevant to MANETs where data diversity and limited resources are prevalent.

Adaptive Communication and Topology-Aware Learning Strategies. In decentralized networks like MANETs, efficient communication and adaptability to dynamic topologies are crucial for model convergence. Topology-aware learning frameworks address these needs by incorporating network connectivity metrics into the learning process to prioritize efficient communication paths. Wang et al. [24] proposed Matcha, a communication-efficient decentralized SGD algorithm that leverages structured network topologies to reduce communication costs. Bellet et al. [25] discussed personalized and collaborative learning in decentralized networks, emphasizing the importance of topology-aware methods for efficient information sharing. Lalitha et al. [26] introduced a peer-to-peer learning framework that adapts to network topology changes, enhancing robustness in decentralized environments. Liu et al. [27] introduced an adaptive asynchronous communication strategy for MANETs, selecting communication paths based on link quality and network conditions to reduce overhead and increase model convergence efficiency. Mod-Squad[28], a modular federated learning framework, supports task-specific model sharing while optimizing communication and adaptability, offering additional insights into efficient multi-task learning under dynamic MANET conditions. The need for decentralized frameworks that can dynamically adjust communication paths, prioritize adaptable learning strategies, and maintain performance under fluctuating conditions remains a critical research gap. Current topology-aware methods highlight essential aspects of adaptive learning but require further innovation to address the unique demands of MANETs, where network conditions can change rapidly, requiring continuous learning and adaptation.

Toward an Adaptive Decentralized Framework for MANETs. The above review illustrates the considerable advancements achieved in federated learning, decentralized SGD, MTL, and topology-aware strategies. However, these approaches still fall short in addressing the comprehensive needs of MANETs, where dynamic connectivity, resource constraints, and data heterogeneity demand a fully decentralized, adaptive learning framework. Existing methods typically assume stable connectivity, IID data, or centralized coordination, which do not align with the practical challenges of MANETs. These gaps underscore the importance of developing a decentralized framework that combines adaptability, efficient communication, and resilience to rapidly chang-

ing network conditions, providing the foundation for the work presented here. Such a framework must be capable of learning from highly heterogeneous data, adapting to evolving network topologies, and efficiently allocating limited resources—an area ripe for further exploration to fully harness the potential of distributed learning in MANETs.

3. Problem Statement and Notation

3.1. Problem Statement

In the mobile cloud computing environment, the data is distributed on different devices, and each device collects data in the way of non-IID, which leads to multiple tasks. Distributed multitask learning improves single-task generalization in a way that captures related task relationships. While distributed multi-task learning is relatively effective in addressing the targeting of individual mobile devices for different categorization tasks under a specific domain, but it cannot present accuracy for all categorization tasks on each device. To address the problems above, in the process of realizing distributed multi-task learning using the mobile adhoc network, it is necessary to design a lightweight image recognition model. It can adapt the personalized task features, incorporate the characteristics of all classification tasks, and achieve self-evolution in the data collection process and co-evolution in the process of communicating with neighbors. In addition, in the process of implementing distributed multi-task learning, the task model or its gradient is transmitted between nodes and the server, the data of different tasks will inevitably be owned by different devices, which results in the risk of communication privacy. In this paper, we based on the hard parameter sharing method in multi-task learning, some parameters of the model are allowed to be transmitted in the iterative process, but how to make use of the limited communication resources to achieve faster convergence and reduce overhead is still a problem to be solved. In this paper, an adaptive communication strategy is proposed, which generates the probability vector of the parameters transmitted by the device to each neighbor according to the iteration time, link quality, reliability and survival time. First of all, we regard each device in the mobile adhoc network as a task of the multi-task learning, and use our own computing power to train the model independently in the private dataset, so as to complete the process of self-evolution of the mobile model. Secondly, combined with the Hungarian algorithm, devices select links with good performance metrics in the dynamic network to transmit parameters.

As devices receive neighbor parameters and integrate with their own models, each device completes the process of co-evolution between models.

3.2. Notation and Definitions

The goal of the classification task assigned to each device adaptively is to learn a function to classify the input image data into the correct category as far as possible. Suppose there are M mobile devices in the mobile adhoc network. The topology of the communication network can be represented by an undirected graph G = (V, ε), where vertex set V = {1,2,...,M} and edge set ε = V × V. Each device i ∈ [1,M] can access the local dataset $D_i = \{x_j^i\}_{j=1}^N$, but does not have access to private data on other devices, and the private data of all devices make up data field $D = \{D_1, D_2, ..D_M\}$. Each device trains its own model in the local data set. Since each device is adaptively assigned different tasks and the corresponding classifier category is different, this paper adopts the hard parameter sharing method, where mobile devices exchange the shared parameter information with the neighbors in the graph, learn the knowledge of other tasks in specific domain, to improve the generalization ability of the local model. Then the optimization goal on device i can be expressed as follows:

$$f_i(w_i) = E(l(W_i; D_i)) \tag{1}$$

$$F_i(w_{s,i}) = \frac{1}{\sum_{m=1}^{M} d_{i,m}} \sum_{m=1}^{M} d_{i,m} f_i(w_{s,i}) \tag{2}$$

Where W_i and $l(W_i)$ are the model parameters and loss function of device i respectively, $w_{s,i}$ and $F_i(w_{s,i})$ denote the parameters and loss function of the shared part between the models on the device i and on other devices, respectively. $d_{i,m}$ is the connection indicator, which means that if $d_{i,m} = 1$, the device i and m are neighbors, otherwise.

In this paper, we improve the NetMax method proposed by PanZhou et al.[29], to better achieve the goal of coordinated training of on-device mode in mobile adhoc networks. We add several performance metrics such as equipment residual power and link quality, and generate the communication probability with neighbors by each device.

Definition 1. *Survival time: This paper assumes that the equipment power is sufficient during the training process. In this paper, the survival time of*

the device is included in the metric to save energy. The survival time of the device mainly consists of two parts: the remaining power of the device and the rate of power consumed by the device, which is calculated as seen in Eq.(3):

$$Q_i = \frac{e_{i,r}}{e_{i,c}} \tag{3}$$

where $e_{i,r}$ and $e_{i,c}$ respectively denote the residual power and the rate of power consumed by the device. When mobile devices select neighbors in each iteration, devices with long survival time are selected with the high probability.

Definition 2. *Link quality metric: This paper only considers that the devices move on the same horizontal plane. It is assumed that each mobile device knows its own location and speed and maintains a list of the location and speed of neighbors. The current position of device i is* (S_i, T_i) *and its speed is* (H_i, V_i)*, and the current position of its neighbor j is* (S_j, T_j) *and its speed is* (H_j, V_j)*. When the distance* D_t *between device* i *and* j *exceeds the wireless signal propagation range r after t seconds, the link between the two devices is disconnected. The distance between them is as follows:*

$$O_t = \sqrt{(S_{ij} + H_{ij}t)^2 + (T_{ij} + V_{ij}t)^2} \tag{4}$$

In Equation (4), we assume a fixed distance between mobile devices. This simplification reduces model complexity, allowing us to focus on the core algorithmic design and performance evaluation of the ADAMT framework. While real-world network environments exhibit variable device distances, the fixed-distance assumption is commonly employed in theoretical analyses and initial validations to facilitate a clearer understanding of algorithm behavior under ideal conditions. For instance, similar fixed-distance assumptions are utilized in studies such as [30] and [31].

Where $S_{ij} = (S_i - S_j)^2$, $H_{ij} = (H_i - H_j)^2$, $T_{ij} = (T_i - T_j)^2$, $V_{ij} = (V_i - V_j)^2$. From the Eq.(4), the Residual Link Lifetime(RLL) of device i and device j is:

$$R = \frac{-F + \sqrt{F^2 - 4EG}}{2E} \tag{5}$$

where $E = H_{ij}^2 + V_{ij}^2$, $F = 2(S_{ij}H_{ij} + T_{ij}V_{ij})$, $G = S_{ij}^2 + T_{ij}^2 - r^2$, According to the R and the threshold O_t for judging the link quality, the designed formula

for calculating the link quality (LQ) is as follows:

$$L = \begin{cases} \dfrac{R}{O_t}, & \text{if } R \leq O_t \\ 1, & \text{if } R > O_t \end{cases} \tag{6}$$

Definition 3. *Connection reliability metrics:*

$$a_i = s_i s_{all} \tag{7}$$

where s_i denotes the number of devices that can communicate with the current device i*, and s_all denotes the number of nodes that are working effectively at the current time.*

Definition 4. *Iteration time: Iteration time is an important metric for evaluating the strengths and weaknesses of training algorithms, and is calculated by the formula is as follows:*

$$t_{i,m} = C_i + N_{m,i} \tag{8}$$

where C_i denotes the time taken by the device i to self-update the model using the local dataset, and $N_{m,i}$ denotes the time it takes for the device m to transmit parameters to device i.

Combined with these four metrics, this paper takes the minimization of iteration time metric and the maximization of link quality, reliability and survival time as optimization objectives. The integration process is described below:

$$b_{i,m} = \frac{t_{i,m}}{L \cdot a_i \cdot Q_i} \tag{9}$$

This design makes cross-device cooperative training more robust and further reduces congestion in the network.

Definition 5. *Number of iterations: n, local iteration step of device i; k, the global iteration step. If a device receives parameters transmitted by its neighbors and updates its local model based on the parameters above, the local iteration step is added by one. And the global iteration step increases as the iteration step of any device increases.*

Definition 6. *Communications probability matrix: In this paper, we allow each device to select neighbors with different probabilities, where neighbors with low iteration time, good link quality, strong reliability and long survival time are more likely to be selected, to adapt to the dynamic adhoc network. We assume the probability of neighbor selection as follows:*

$$P = [p_{i,m}]_{M \times M} \tag{10}$$

where $p_{i,m}$ in the probability matrix denotes the communication probability of device transmitting parameters to neighbor.

3.3. Supporting Network Topology Model for ADAMT

To further enhance the performance of ADAMT in decentralized environments like MANETs, we introduce a supporting network topology model as shown in Fig.1. This model provides the underlying communication structure and simulates the dynamic connectivity challenges commonly encountered in mobile ad-hoc networks. By incorporating this topology model, ADAMT's design is better aligned with the realistic constraints of MANETs, such as variable connectivity and communication efficiency requirements.

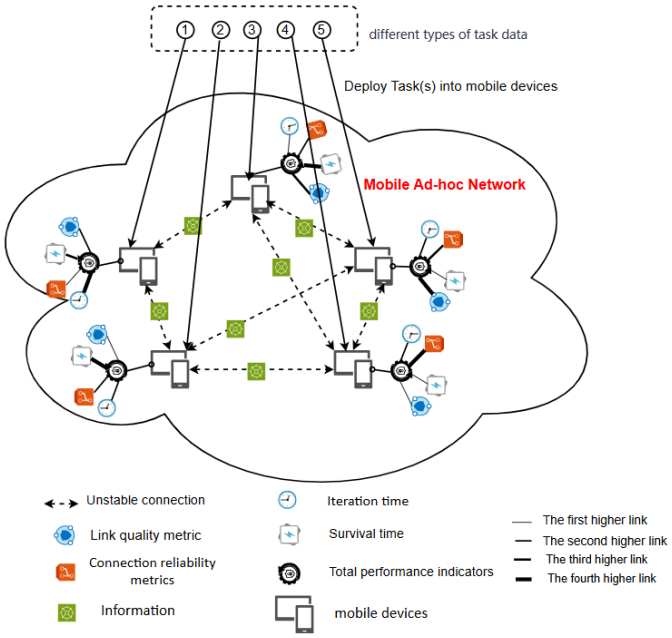

Fig. 1. A plot of the network topology model.

4. ADAMT: Adaptive Distributed Multi-task Learning for Efficient Image Recognition in Mobile adhoc Networks

4.1. Overview of the ADAMT Architeture.

The architecture of ADAMT approach in mobile adhoc networks can be described in Fig. 2. First, we extract information from the collected metrics such as device performance, user preference, etc., to assign image recognition tasks to each device in a targeted manner. Then the user device collects the images needed for the task and composes a local dataset for subsequent processing. After manually labeling a small number of images according to the category, the device semi-supervisedly trains the model in the local dataset, realizing the purpose of model self-updating in the process of data collection. When certain conditions are met, such as when the power reaches a certain value, it means that the model self-updating is completed. In the iterative process, the device transmits the survival time, reliability, iteration time and link quality indicators to the neighbors within the communication

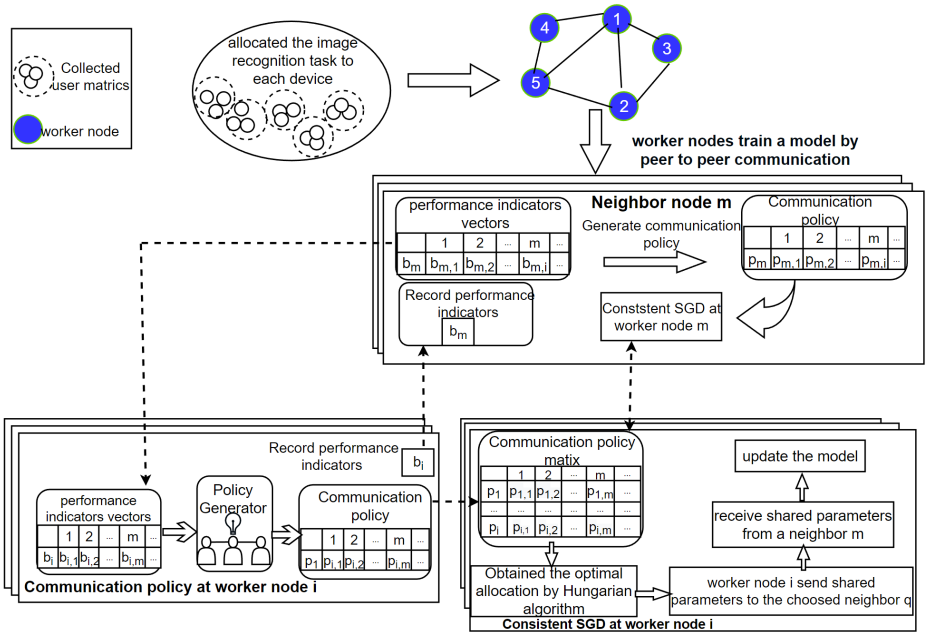

Fig. 2. The overview of the system architecture.

range, and runs the communication policy generation algorithm based on the decision objective to get the probability of transmitting the parameters to each neighbor. And we combine with the Hungarian algorithm, select a neighbor with good performance metrics to transmit the parameters for each device to achieve the overall performance enhancement, so as to reduce the congestion and convergence time and reduce communication overhead to adapt to dynamic network conditions. After the device receives the model parameters from its neighbors, it updates the model in combination with itself to improve the model generalization ability and the effect of recognizing tasks on other devices, and ultimately achieve the goal of group co-evolution in the iterative process.

In this section, we proposed a lightweight image recognition model based on distributed multi-task learning, which consists of two modules: representation learning module and hash search module. Representation learning module has combining the MobileNetV3 [32] architecture and PPshitu [33] model, which use a series of convolutional layers to extract image features,

and hash algorithm is used to transform extracted features into binary vectors. Hash search module has based on the vector search algorithm [34] to find the features in the binary code library that are closest to the features we extracted from the image, and use the corresponding tags as our recognition results. The overall structure of the proposed model has been shown in Fig. 3. Where SE denotes the attention module used in MobileNetV3. PW denotes Pointwise Convolution. DW denotes Depthwise Convolution. DP denotes Depthwise separable convolution.

In particular of this section, the idea of feature extension is introduced in the basic model structure. Each device stores a dataset containing images from multiple tasks. We convert all the images stored in the task data set into binary vectors through the model, and store them in the binary code library. So the model on each device can realize the recognition effect of multiple tasks.

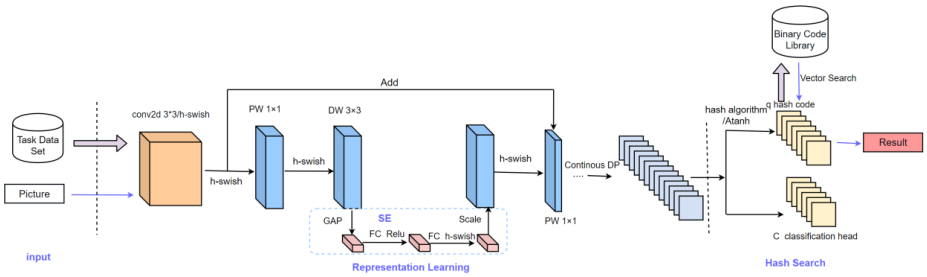

Fig. 3. The overall structure of the proposed model.

The function of representation learning module is to transform the input image into feature vector for subsequent hash search. The specific structure of

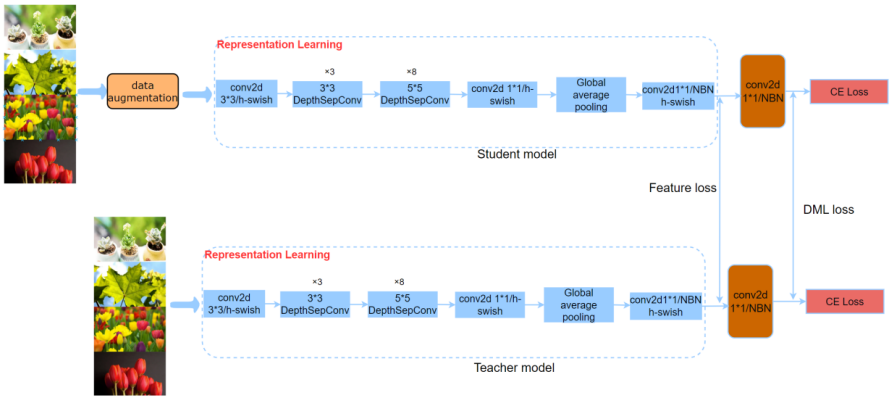

Fig. 4. The process of the semi-supervised training.

the module is shown in Table 1. Where NBN denotes no batch normalization.

Table 1. Detailed configuration of representation learning layers.

Layers	**Configuration**
Conv1	convolution kernel 3×3, 2×2, h-swish.
Bneck	convolution kernel 3×3, 2×2, SE,relu.
Bneck	convolution kernel 3×3, 2×2, relu.
Bneck	convolution kernel 5×5, 2×2, SE,h-swish.
Bneck×4	convolution kernel 5×5, 1×1, SE, h-swish.
Bneck	convolution kernel 5×5, 2×2, SE, h-swish.
Bneck×2	convolution kernel 5×5, 1×1, SE, h-swish.
Conv2d	convolution kernel 1×1, 1×1, h-swish.
Pool7× 7	
Conv2d×2	convolution kernel 1×1, 1×1, h-swish, NBN.

In order to make effective use of a large number of unlabeled data, we first pre-train the representation learning module. This paper combines knowledge distillation method [35] to implement semi-supervised learning, namely noisy student method [36], in which there are two networks with the same structure but different parameters: student network and teacher network. The teacher model is first trained on the tagged image to generate pseudo labels for unlabeled examples. The student model is then obtained by training on a combination of labeled and pseudo-labeled samples. These combined samples are enhanced by data enhancement techniques and model noise to improve the performance of the model. Through many iterations of the method, the student model becomes a new teacher model and relabels the unlabeled data, which is repeated until the parameters of the model are stable.

We adjust the parameter information of the two network models by the following semi-supervised loss function to ensure that the final output and the intermediate feature output both remain the same for the same input. As shown in Fig. 4, the semi-supervised training loss function consists of three parts:

- Prediction loss. The loss between the prediction result of the model for labeled data and the ground truth. It is well known that the accuracy of image prediction is the most important metric to evaluate the performance of image recognition models. Thus, we use the cross-entropy loss function to formulate the prediction loss term, and improve the model accuracy by minimizing the loss, which is calculated as follows:

$$Loss_1 = CrossEntropy(X_l, g_l) + CrossEntropy(Y_l, g_l) \tag{11}$$

  Where X_l and Y_l denote the final output of the labeled image l by the student network and the teacher network, respectively, and the g_l provides the ground truth of the labeled image.

- semi-supervised loss. Although we can train the model using only the loss function based on labeled data, it is difficult to support the advanced performance of a multi-layer deep model with only a small amount of unlabeled data. Thus, we make full use of the potential performance of unlabeled data and approximate the final output of both models by minimizing the KL scatter loss, which is calculated as follows:

$$Loss_2 = \frac{KL(X_t||Y_t) + KL(Y_t||X_t)}{2} \tag{12}$$

Where X_t and Y_t denote the final output of unlabeled data t from the student network and the teacher network, respectively.

- Feature loss. Combining the idea of feature distillation, we use the mean squared loss function to ensure that the feature-level knowledge of the two network models remains consistent, which can be calculated as follows:

$$Loss_3 = MSE(X_f, Y_f) \tag{13}$$

Where X_f and Y_f denote the vectors of student and teacher network feature layer outputs, respectively, and the mean square error is used to calculate their feature distance in this paper.

Combining the above three losses, the total loss of the final training is shown in Eq. (14):

$$Loss_t = Loss_1 + Loss_2 + Losss_3 \tag{14}$$

The limited space of mobile devices usually leads them cannot store massive image feature libraries, and the models consume long time in retrieving images. The deep hashing algorithm reduces storage and speeds up retrieval by converting image feature vectors into fixed binary codes, and makes the binary codes similar between similar images, which is now widely used in the efficient retrieval process of large-scale image databases. However, the current deep hashing algorithm is mainly applicable to supervised scenarios, and these scenarios only consider the semantic similarity between images, while ignoring the underlying data structure of images. Combined with the semi-supervised scenario in this paper, we choose semi-supervised deep hashing algorithm [37], which considers both the semantic similarity of images and the underlying data structure, to improve the efficiency of image retrieval.

As shown in Fig. 1, in the hash search module, the model integrates semi-supervised deep hash algorithm [38] and metric learning function, which converts the feature vector obtained by the representation learning module into binary vector. And we use hamming distance to represent the distance between the two vectors. Then we use the vector search algorithm to find the feature with the smallest hamming distance in the binary code library, and finally output the label corresponding to the similar feature as the final result. In the process of model self-evolution, we only consider the recognition accuracy of local tasks. In the process of the co-evolution with other devices,

the demand of the model to identify multi-task images increases. But the traditional threshold function is easy to cause hash collision. This paper uses the compound activation function, where the proposed adaptive hyperbolic tangent function (ATanh) [39] transforms the image feature vector into binary vector, so that the binary vector obtained by different kinds of images is also different.

When the equipment self-updates the model parameters, the model is updated according to the loss function of DeepHash and ATanh in the process of semi-supervised training.

4.2. Algorithm of ADAMT

In this paper, we improve the NetMax [29] communication method, proposing an adaptive communication strategy based on minimizing communication overhead and iteration time. The NetMax method allows devices to select neighbors with different probabilities, the neighbors with high-speed communication links are selected with high probability, and the selection probability changes with the network conditions. Compared with the fixed network communication conditions, all nodes in the mobile adhoc network are composed of mobile devices, which have the characteristics of limited power, dynamic topology changes, etc., and only the devices within the communication range can communicate with them. Each device trains the model independently and dispersed on the local dataset. In this paper, we cancel the central monitor to track the network state and select each device to generate the probability by combining multiple performance metrics such as iteration time, device power, reliability and link quality during the iteration process, to adapt to the dynamic network conditions of the mobile adhoc network and realize the multi-devices cooperative training model under offline state, inspired by NetMax. According to the Hungarian algorithm, the neighbor with good performance metric is assigned to each device to transmit parameters. The Hungarian algorithm is a combinatorial optimization algorithm. It assigns each device to a suitable neighbor to optimize the total performance of the assigned link, thus reducing iteration time and communication cost [40].

In this paper, the decentralized optimization objective proposed by the above equation is solved by the conventional SGD algorithm. The gradient at device performing the nth update can be derived as follows:

$$\bigtriangledown F(w_i; D_i) = \bigtriangledown l(w_i; D_i) \tag{15}$$

$$\bigtriangledown F_i(w_{s,i}) = \frac{1}{\sum_{m=1}^{M} d_{i,m}} \sum_{m=1}^{M} d_{i,m} \bigtriangledown f_i(w_{s,i}) \tag{16}$$

Then the model parameters at the device i are updated by the following Eq. (17,18):

$$w_i^{n-1} = w_i^n - \alpha \bigtriangledown F(w_i; D_{i,n}) \tag{17}$$

$$w_{s,i}^{n+1} = \frac{1}{\sum_{m=1}^{M} d_{i,m}} \sum_{m=1}^{M} d_{i,m} w_{s,m}^{n+1} \tag{18}$$

where w_i^n is the model parameters deployed on device i, and $D_{i,n}$ is the dataset randomly sampled from D_i when device i performs the nth local iteration. In the above equation, the shared parameter information is involved in the information interaction process between neighbors. If the model update process is executed according to the above equation, the device then needs to transmit the shared parameters to all its neighbors, but the update process is bound to be affected due to the presence of low performance links.

To better adapt to the dynamic conditions of mobile adhoc networks, we propose the consistent SGD algorithm in conjunction with ADPSGD [41]. When implementing each iteration, the devices can only transmit parameters to one neighbor and the neighbors with high performance metrics are selected with high probability based on the Hungarian algorithm. In addition, the devices transmit the parameters of the representation learning part by end-to-end resource sharing. With the accumulation of iterations, the parameter information is gradually propagated to all other devices in the mobile adhoc network. Next, the device updates the overall local model using the local gradients from the self-update as well as the model parameters transmitted by its neighbors. As a result, the loss functions of the devices and the differences in the representation learning part between them gradually decrease with iterations and converge to the optimal value, thus better capturing the relevant inter-task relationships and improving the generalization capability of the distributed multi-task model.

Algorithm 1 represents the process of generating communication probability vector at device i. To estimate the network condition, the device sends

Algorithm 1 Communication Policy on device i.

Require: *learning rate* α, *local iteration step* N

Initialize performance parameters vector
$[b_{i,m}]_{1\times M} \leftarrow [0]_{1\times M}$
Initialize probilities $[p_{i,1}, ..., p_{i,M}] \leftarrow [1/M]_{1\times M}$
while TRUE **do**
    Send performance parameters to the neighbors
    $[b_{i,m}]_{1\times M} \leftarrow$ *the neighbor's performance metrics*
    $[p_{i,m}]_{1\times M} \leftarrow PolicyVector(\alpha, N, [b_{i,m}]_{1\times M})$
    Send $[p_{i,m}]_{1\times M}$ *to other devices*
    if *Recieved all devices' probilities* P **then**
        Running the Hungarian algorithm
        device i *communication with neighbor* q
        if *device* i *cannot communication with* q **then**
            Select the intermediate device j
        end if
    end if
    execute the consistent SGD algorithm
end while

performance indicators to their neighbors before each iteration, and calculates the communication strategy vector P based on the received statistical data. Each term of the vector P is the probability of the device transmitting parameters to the neighbor. If the communication strategy of the device is updated, the probability matrix is updated accordingly. Then the device i composes the probability matrix of all the devices after collecting their communication probability vectors. Then in conjunction with the Hungarian algorithm, neighbors are assigned to each device and parameters are transmitted to them. Neighbors with low iteration time, good link quality, reliability and with long survival time are more likely to be selected. Thus, dynamic network conditions can be adapted to avoid congestion and reduce communication overhead. During the process of transmitting parameters from device i to its neighbor m, if the link between them is disconnected, device i selects the device j with the highest communication probability among its connected neighbors except m as an intermediate device, and then transmits the parameters to device j. Next, the device i performs the consistent SGD algorithm. In each iteration, the process of updating the model by the device is divided

into three parts. Firstly, the parameters of the representation learning part are updated using the information received from the neighbors, and secondly the model is updated again based on the gradient computed from the private dataset. At the end of the iteration, the updated statistics are sent to the neighbors to generate the latest communication policy.

In order to better obtain a communication strategy that enables coordinated training across devices in mobile adhoc networks, we introduce several performance metrics based on the iteration time proposed by NetMax and rewrite the optimization objective in this way, as shown in the following Eqs. (19) to (24).:

$$min \;\; k\bar{b} \tag{19}$$

s.t.

$$\lambda \leq \epsilon, \;\; \forall \epsilon > 0 \tag{20}$$

$$\bar{b} = \frac{\bar{b_i}}{M} = \frac{1}{M} \sum_{m=1}^{M} b_{i,m} p_{i,m} d_{i,m}, \;\; \forall i \in [M] \tag{21}$$

$$p_{i,m} > \alpha\rho(d_{i,m} + d_{m,i}), \;\; \forall i, m \in [m], \;\; m \neq i, \;\; d_{i,m} \neq 0 \tag{22}$$

$$p_{i,m} = 0, \;\; \forall i, m \in [m], \;\; d_{i,m} = 0 \tag{23}$$

$$\sum_{m=1}^{M} p_{i,m} = 1, \;\; \forall i, m \in [m] \tag{24}$$

Here, λ is the second most important eigenvalue of the matrix $E[D^k(D^k)^T]$, D^k is the random matrix at the time of performing the first iteration associated with the communication strategy. $b_{i,m}$ represents the performance metrics for device i connected to device m, $\bar{b_i}$ is the sum of performance metrics connected to device i, $\bar{b}$ is a summary of all devices performance indicators. $d_{i,m}$ is the connection indicator, which means that if $d_{i,m} = 1$, the device i and m are neighbors, otherwise $d_{i,m} = 0$. $p_{i,m}$ denotes the communication probability of device i transmitting parameters to neighbor m.

In this optimization problem, the primary parameters being optimized are the model weights $\mathbf{W}$ and the binary assignment variables $\mathbf{B}$. The model weights $\mathbf{W}$ are adjusted iteratively to minimize the overall loss function across

tasks, thereby improving task-specific performance. The binary variables $\mathbf{B}$ represent task allocation across nodes and are optimized to ensure effective task distribution in the network. Together, these parameters drive the learning process within the ADAMT framework, enabling the model to adapt to diverse data distributions across decentralized nodes.

Algorithm 2 GeneratePolicyVector.

Require: *learning rate* α, *performance parameters vector* $B_i = [b_{i,m}]_{1\times M}$, *global iteration step* K, *local iteration step* N
Ensure: *Probility vector* $[p_{i,m}]_{1\times M}$

function $PolicyVector(\alpha, N, B_i)$
    k value of p in the range of the NetMax
    For any given value of p :
    run the value obtained by InnerLoop
    Store the obtained suboptimal vector
    return the smallest target value
end function
function $InnerLoop(\alpha, \rho, N, B_i)$
    $L \leftarrow \frac{\alpha\rho}{M}\sum b_{i,m}(d_{i,m} + d_{m,i})$
    $U \leftarrow \frac{1}{M}\ max_{m\in M} b_{i,m}d_{i,m}$
    $\delta \leftarrow (U - L)/N$
    for $n \in [1, 2, ..., N]$ **do**
        $\bar{b} \leftarrow L + \delta$
        $[p_{i,m}]_{1\times M} \leftarrow$ *Solve LP problem*
        $\lambda_2 \leftarrow$ *Compute second largest eigenvalue*
        $T_{convergence} \leftarrow \bar{b}\frac{\ln\varepsilon}{\ln\lambda_2}$
        $res[r] \leftarrow (T_{convergence}, \lambda_2, \bar{b}, [p_{i,m}]_{1\times M})$
    end for
    $x \leftarrow$ *find item in res with minimum* $T_{convergence}$
    return $x.[p_{i,m}]_{1\times M}, x.T_{convergence}$
end function

Since each device i is only concerned with its policy for communicating with its neighbors, we remove the network monitor. In other words, we run the communication policy generation algorithm on each device to obtain the probability vector. The specific generation process for probability vectors can be found in Algorithm 2. It consists of two nested loops that find the

optimal communication policy within a given configuration of ρ and $\bar{b}$. Since each device does not need to care about the communication vectors between other devices and their neighbors, the feasible interval specified in this paper is found within the range of communication data between the device and its neighbors. And the linear programming objective is defined as follows:

$$min\ \ p_{i,i},\ \ s.t.(Eq.(20)-(24)) \tag{25}$$

Algorithm 3 consistent SGD algorithm at device i.

Require: *learning rate* α, *local iteration step* N, *smoothing factor* β
Ensure: *Trained model* w_i
Initialize model w_i^0
Initialize average iteration time $T_i \leftarrow [0]_{1\times M}$
for $n \in [1, 2, ..., N]$ **do**
  if *receive shared parameters from neighbor* m **then**
    $W_{s,i}^{n+1} \leftarrow (W_{s,i}^n + W_{s,m}^n)/2$
    $W_i^{n+1} \leftarrow W_i^n - \alpha \bigtriangledown F(w_i; D_{i,n})$
  end if
  $UpdateTimeVector$
  send the updated $[b_{i,m}]_{1\times M}$ *to their neighbors*
end for
return w_i
function $UpdateTimeVector$
  $t_{i,m} \leftarrow$ *Recorded iteration time*
  $T_i[m] \leftarrow \beta T_i[m] + (1-\beta)t_{i,m}$
end function

Algorithm 3 gives the detailed steps for the device to execute the consistent SGD algorithm. After receiving the information transmitted by the neighbors, the device first combines the local parameters to average the model parameters of the representation learning part, then uses the local private dataset to calculate the local gradient, and finally uses the gradient to update the overall model again. At the end of the iteration, we update the average iteration time between the device and its neighbors according to the exponential moving average (EMA) [42] proposed by NetMax, and other per-

formance metrics are updated according to the network condition. Finally the updated statistics are transmit to the neighbors.

The ADAMT framework inherently accommodates non-IId data distributions by employing a distributed feature expansion technique that enables each node to derive representative features tailored to its local data distribution. Through this mechanism, ADAMT captures data variability across nodes, enhancing adaptability in heterogeneous data environments. Moreover, the adaptive model aggregation dynamically weighs model updates from each node, aligning local model parameters with global learning objectives while preserving the relevance of local data patterns. This approach effectively mitigates the challenges posed by non-i.i.d data distributions in MANET environments.

4.3. Discussion

Influenced by cloud computing technology, most current cross-device training methods, such as federated learning and decentralized learning methods, are still cloud-centric architectures, where information such as model parameters are transmitted to cloud data centers by means of remote communication. This model greatly limits the scalability of model training and weakens the relevance of mobile devices to model training, making it impossible to effectively and adequately utilize the computational resources of mobile devices. In addition, the arithmetic and storage capacity of the device itself is substantially increased, although it is less compared to the physical host, but its role in model training should not be ignored. Based on this, we propose a framework paradigm for distributed learning to minimize the convergence time as a starting point, establish a personalized model as well as a global model evolutionary goal, and ultimately achieve distributed multi-task image recognition on mobile adhoc networks. Unlike distributed learning in traditional mobile cloud computing environments, our implementation does not rely on cloud data centers, treats a single device as a cloud server, and utilizes the arithmetic power of multiple devices to build a computational network, which enables our approach to efficiently orchestrate mobile resources.

In the mobile adhoc network environment, data is distributed across different devices and each device collects data in a non-IID manner, which leads to the existence of multiple tasks. However, it is difficult to train an efficient generalized model in a multi-task environment and cannot adapt to user characteristics. As a result, we combine the ideas of feature expansion and deep hashing algorithms to achieve image classification by means of image

retrieval and adapt the task characteristics to the target data of the user's device, so as to realize the problem of personalized processing and inter-model fusion of devices for different classification tasks in a specific domain.

Considering that in the mobile adhoc network environment, due to the mobility of the device, limited power, and only communicating with the devices within the communication range appear traditional distributed SGD methods can not play the best performance, and even failures. Therefore, in order to ensure the adaptation to the dynamic network state during the collaborative training process between devices, we design an adaptive communication strategy that breaks through the limitation that traditional distributed SGD methods cannot cope with the dynamic network. Firstly, starting from the device's own conditions, we statistically count the performance metrics, and then collect the performance metrics sent by all neighboring devices to construct the link communication probability allocation problem based on minimizing the convergence time. On this basis, combined with the Hungarian algorithm, the optimal transmission parameter allocation result between devices is obtained while motivating them to communicate with each other.

As a whole, this paper introduces a distributed multi-task image recognition method for obtaining the target optimization model under mobile adhoc network by updating the model using the own arithmetic power of multiple devices and sharing the parameter information of a specific model structure in the offline state. The method not only focuses on the device model's function of differentiated processing and self-evolution for different classification tasks in a specific domain, but also strives to present the model's generalization effect for all tasks. The method considers the collaborative problem of different mobile devices under mobile adhoc network, introduces the ideas of probability theory and Hungarian algorithm to realize the selection of the optimal communication method in a specific situation, breaks through the limitation that the traditional SGD method is unable to adapt to the dynamic network conditions, and solves the problem of co-evolution of the model in the iterative process.

Considering the limited computing power and storage capacity of the mobile terminal, we discard the idea of utilizing multi-objective optimization algorithms to achieve optimal adaptation among devices, and integrate the performance metrics in a relatively simple way, and then derive the communication policy vector using linear programming methods. In addition, due to the limitations of link bandwidth and device power, as well as the characteristics of dynamic topology, we stipulate that only one device communicates

with its neighbors during each iteration.

5. Performance Evaluation and Analysis

We implement distributed multi-task image recognition based on the Paddle framework in a well-built mobile adhoc network, implement the processing of personalized tasks based on user characteristics and aggregates the recognition effect of multi tasks through inter-device collaboration in the image recognition model deployed on the mobile. In this section, we experimentally evaluate the performance of the proposed model and the adaptive communication strategy.

5.1. Experimental Setup

To evaluate the robustness of ADAMT under non-IId data conditions, we simulate scenarios where each node's data exhibits distinct statistical properties, representing heterogeneous distributions across the network. This setup allows for testing ADAMT's capacity to adapt to diverse data distributions in a decentralized environment.

Device configuration: The experiments are conducted in the mobile adhoc network consisting of 10 cell phones. Each phone is equipped with a SnapdragonTM 865 processor, and 12GB RAM and 256GB ROM memory.

Dataset: To verify the performance of the proposed model in the mobile adhoc network environment, we first assign tasks to each device in a targeted manner based on user usage characteristics, hobbies, and other information. Then, users collect images and label a small number of them according to the assigned tasks. In addition, to improve the model performance, we collected 200 images from the web for each task type to form a semi-supervised dataset required for personalized tasks. In this paper, we divide the dataset into a training set, a test set, and a validation set in the ratio of 7:2:1. For the task dataset, we used a combination of ImageNet-1k and some common things from online academic databases.

Contrasting models: we compare the proposed model with several baseline models in different aspects.

- PatchConvNet [43]: To enhance the non-local inference capability of CNNs, its uses a global mapping based on attention mechanism to replace the average pooling layer at the end of traditional CNNs.

- VAN-B4 [44]: It introduces a novel linear attention mechanism, Large Kernel Attention (LKA), to enhance the adaptive and remote correlation capabilities of self-attention applied to machine vision.
- TinyViT [45]: Its method of guiding knowledge distillation with the help of teacher models enables small models to obtain recognition effects from large-scale pre-trained data.
- VOLO [46]: It uses an innovative outlook attention mechanism to label fine-grained features and coding contexts more efficiently, thus improving recognition performance.
- PPshitu: It utilizes a range of methods such as model compression, supervised deep hashing algorithms, and other methods for recognizing images on the cell phone.
- RexNet [47]: Its original ReLU6 method was modified for MobileNetV2 using the input channel size of the extended convolutional layer and the replacement of the Swish-1 activation function to avoid a representational bottleneck in the classifier.

Evaluation criteria: accuracy and error rate make the most basic evaluation metrics for image classification models, i.e., the ratio of the number of correctly classified images to the number of all images, and the opposite for error rate. The calculation equation is as follows:

$$error = \frac{FP + FN}{TP + TN + FP + FN} \tag{26}$$

$$accccccuray = 1 - error \tag{27}$$

The accuracy and recall rates are the criteria for correct classification. It can be calculated by the following equation:

$$Reall = \frac{TP}{TP + FN} \tag{28}$$

$$Reall = \frac{TP}{TP + FP} \tag{29}$$

5.2. Implementation details

For model training, we first use the user's local dataset to semi-supervisedly train the proposed model for 100 iterations. Following the standard practice, we enhance the robustness of the model by flipping the input images horizontally at random, cropping them randomly to fixed pixels, and adding noise to the data. After experimental analysis and careful selection, an SGD optimizer with a momentum of 0.99, a mini-batch size of 256, and an initial learning rate of 0.01 is selected to adjust the model parameters in this paper. The weight decay parameter is 0.0005. The process of semi-supervised training combined with the deep hashing algorithm has the same hyperparameters as above except that the loss function is set differently.

5.3. Comparison with other Methods

In this section, we will compare the performance of each model in different aspects.

Firstly, to test the generalization capability, we compare the image recognition models on the ImageNet dataset. The Table 2 shows the results of different models for recognizing image sets under different evaluation criteria. According to the table, it can be observed that our proposed model has a significant advantage over other models in terms of recognition results, achieving a uniformly high detection performance, high fault tolerance, and low specificity. This demonstrates the feasibility of using mobile adhoc network for distributed machine learning and the effectiveness of the model in achieving distributed multi-task image recognition.

Table 2. Training effect of different models on ImageNet dataset.

Model	**Params**	**Recall**	**Precision**
PatchConvNet	25.2M	0.834	0.829
TinyViT-21M-distill	21M	0.826	0.827
VOLO-D1	27M	0.845	0.830
VAN-B5	60M	0.863	0.833
EfficientNet-B5	30M	0.832	0.828
RexNet	34.7M	0.812	0.828
Ours	29M	0.894	0.835

In this paper, tests are done separately for the personalization process and the overall recognition effect of the deployed model during the iterative

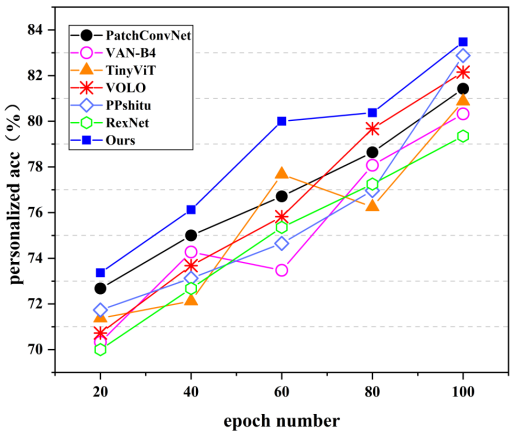

Fig. 5. Different models with iterative process for personalized task processing in mobile adhoc network.

Table 3. Ablation experiments of noisy student of MobileNet model.

Model	Recall	Precision
MobileNet	0.852	0.834
MobileNet + $\pi - model$	0.856	0.839
MobileNet + noisy student	0.859	0.842

Table 4. Comparison of experimental results for ablation of deep hash functions.

Model Type	Recall@1	Index size(M)	Retrieval time(ms)
two-valued model	0.765	8.3	20
real-valued model	0.852	279	105

process. The results for each stage are derived from the average of the recognition accuracy of all devices. From Fig. 5 and Fig. 6, it can be seen that the proposed model's recognition effectiveness for personalization tasks continues to improve as the iterations continue, while the generalization ability for

other tasks also increases, and has a significant adva1ntage over other models' effectiveness. In addition, TinyViT and VAN-B4 even show a decrease in recognition accuracy during the iterations, which indicates that they cannot adapt to scenarios with non-Independent and Identically Distributed data. It is noteworthy that although the PPshitu model is better for personalization, the inability to aggregate the information transmitted by neighbors leads to poor multi-task recognition results. This also shows that the proposed model, as a framework-level approach, focuses on the role of mobile devices themselves for model training and allows devices to focus on personalized models, while ensuring the interaction of models among devices in mobile adhoc network and facilitating the co-evolution of models.

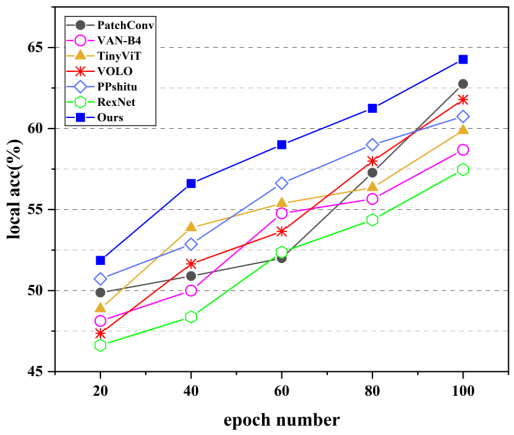

Fig. 6. Different models with iterative process for all tasks aggregated processing in mobile adhoc network.

5.4. Ablation Experiment

In addition, we conducted ablation experiments on noisy student to evaluate the performance improvement it brings. Specifically, we compare the evaluation metrics under the three MobileNet-based settings. π-model [48] computes the stochastic enhancement of an image and normalizes the model's

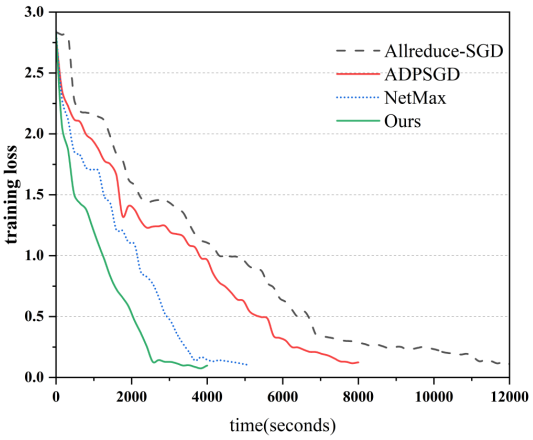

Fig. 7. The training loss in scenario A.

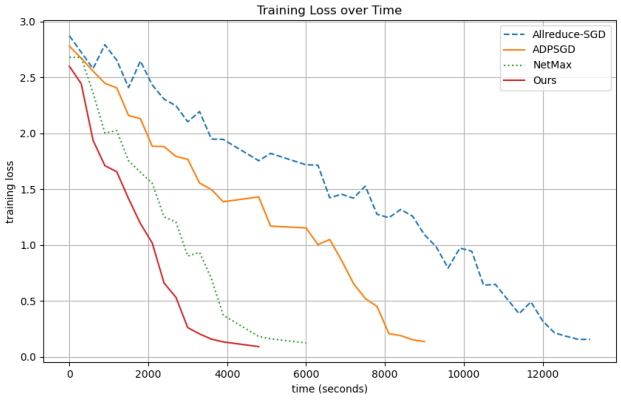

Fig. 8. The training loss in scenario B.

output predictions for the enhancement of the same image. It is worth noting that the training method combining noisy student or π-model is similar to the training strategy of the standard MobileNet model without fine-tuning any hyperparameters. As can be seen from the Table 3, the use of the distillation method greatly improves the model performance benefits, but the difference is that the model accuracy and recall remain the best after using the U-DML method. From this, it can be concluded that the use of distillation method leads to a significant improvement in the generalization ability of the model, and the recognition effect will gradually improve with the iteration.

For the effect of hashing algorithm on image recognition models, this section compares the evaluation metrics in two states of MobileNet binary and real-valued models. To ensure the fairness of the experiment, the two models have the same backbone, the same embedding size, the same training strategy and the same dataset. And the comparison experiments were conducted in terms of accuracy, index size and retrieval time. Table 4 shows the comparison results of these two models. It can be seen that the real-valued model has higher retrieval accuracy in comparison, while the binary model consumes less storage and retrieval time.

5.5. Effectiveness of ADAMT on Non-IID Data

To validate the robustness of ADAMT in handling non-IID data, we designed an experiment comparing its performance under two distinct data distribution scenarios: IID (independent and identically distributed) and non-IID. This experiment aims to quantify the adaptability and convergence of ADAMT in more realistic, heterogeneous data environments, providing a clear illustration of its effectiveness in decentralized settings with data diversity.

In this experiment, we use two distribution scenarios: 1. Scenario A (IID Data): In this setup, data across all nodes is evenly distributed, simulating an idealized scenario where each node has access to a similar data distribution. 2. Scenario B (non-IID Data): Here, data distributions vary significantly across nodes. Each node has access to unique subsets of data classes or feature distributions, mimicking a real-world scenario where data heterogeneity is prominent.

We applied ADAMT in both scenarios and measured the model's convergence speed and final accuracy across a set number of training rounds. The goal was to assess ADAMT's performance under diverse conditions and its

ability to adapt and maintain learning effectiveness with varying local data characteristics.

Table 5. Average epoch time in scenario A.

Decentralized method	Computation cost (s)	Communication cost (s)
AllreduceSGD	24.73	113.64
ADPSGD	24.89	98.35
NetMax	24.69	50.71
Ours	24.92	36.59

Table 6. Average epoch time in scenario B.

Decentralized method	Computation cost (s)	Communication cost (s)
AllreduceSGD	39.03	150.75
ADPSGD	38.76	117.38
NetMax	38.79	85.77
Ours	38.69	51.69

Table 7. The effectiveness of personalization as well as aggregation processing in scenario A.

Decentralized method	Local Accuracy (%)	Personalized Accuracy (%)
Allreduce-SGD	61.2	82.78
ADPSGD	64.265	82.56
NetMax	67.835	83.11
Ours	69.795	83.29

Communication cost. Communication cost expressed the performance of decentralized methods. Table 5 depicts the computed cost and communication cost of the four decentralized training methods in senario A. We can obvious from the Table 5 that the communication cost generated by our proposed adaptive communication method is significantly lower, while the communication cost of ADPSGD, Allreduce-SGD [49], and NetMax even

Table 8. The effectiveness of personalization as well as aggregation processing in scenario B.

Decentralized method	**Local Accuracy (%)**	**Personalized Accuracy (%)**
Allreduce-SGD	46.27	67.34
ADPSGD	47.75	71.83
NetMax	50.35	75.26
Ours	65.62	80.59

reaches 2.69 times, 3.1 times, and 1.37 times of it, respectively. This is because, compared to the NetMax communication method, we combine the characteristics of the links and devices in the mobile adhoc network, not only the link speed, to adapt to the dynamic network changes. It can be seen that Allreduce-SGD has the highest communication cost. During the execution of Allreduce-SGD, the device needs to broadcast model information to all devices, which is highly susceptible to dynamic network states and may experience propagation disruptions or network congestion. In addition, Allreduce-SGD does not consider the inherent characteristics of mobile devices, and its extensive use of links with low performance metrics can further reduce model identification accuracy as well as communication efficiency.

The communication cost of multiple algorithms in the non-IID scenario is shown in Table 6. It can be seen that ADAMT reduces the impact of heterogeneous data well. The reason is that it extracts different types of task data separately, thus adapting to the complex and changing multi-task learning scenarios.

Convergence Speed. Fig. 7 shows the training losses incurred by several methods as the iterations continue in senario A. From the Fig. 7, we can observe that although the final training loss of these four methods is close to zero, the adaptive communication strategy generates the least loss compared to the other methods and its convergence speed is the fastest, reaching 3 times, 2 times, and 1.28 times that of ADPSGD, Allreduce-SGD, and NetMax, respectively. This is because it can dynamically adjust the probability of communication with neighbors according to the network conditions such as link quality, mobile device power, and transmission time, and then the Hungarian algorithm selects the suitable neighbor for each device. This not only ensures a smooth training process, but also allows devices to select neighbors with high performance metrics for information interaction in

a complex network environment, thus improving the convergence speed and further reducing the convergence time for training in mobile adhoc networks.

As we can see from Fig.8, ADAMT's convergence in Scenario B (non-IID) was slightly slower compared to Scenario A (IID). However, it still demonstrated stable convergence patterns, indicating that the model effectively accommodates heterogeneous data distributions without major disruptions in learning.

Training accuracy. We compare the accuracy of the trained models under different methods. The results from Table 7 show that while the personalization of devices and the overall recognition accuracy of the model under all decentralized training methods can reach more than 0.8 and 0.6, respectively, the proposed method is significant in terms of the overall recognition effect of the model, while significantly improving the generalization ability of the personalization task. The main idea of its performance gain is that the uncertainty of device movement triggers random changes in the network topology, and the proposed method adjusts the communication probability between devices and neighbors according to this randomness, which has the potential to move the algorithm away from the suboptimal phenomenon caused by poorer generalization. On the other hand, based on Pareto optimality theory, the actual processing of individualized tasks by each device decreases the model accuracy, while the hard parameter method used in this paper shares information about all tasks within each device, thus it has better prediction performance than single-task learning.

As shown in Table 7 and Table 8, ADAMT achieved a final accuracy of 83 % in Scenario A and 85 % in Scenario B. The slight decrease in accuracy under non-IID conditions is expected, yet the results confirm ADAMT's resilience in maintaining robust performance even with significant data heterogeneity across nodes.

Overall, these results provide empirical evidence of ADAMT's adaptability to non-IID data environments, confirming its effectiveness and suitability for decentralized multi-task learning with heterogeneous data distributions.

5.6. Comparative Performance with SOTA Methods

To evaluate the effectiveness of ADAMT in decentralized multi-task learning, we compare its performance against six representative SOTA methods, each of which addresses various aspects of federated learning and distributed multi-task learning. The selected methods for comparison are CFL, FedGK, FedKD, MOCHA, and Mod-Squad. This comparison highlights ADAMT's

adaptability, accuracy, communication efficiency, and convergence speed in handling non-IID data distributions within resource-constrained, decentralized networks.

Table 9. Comparative performance of ADAMT and baseline distributed multi-task learning models across key metrics.

Model	Convergence Speed	Accuracy	Communication Overhead	Model Adaptability(score 1-5)
CFL	6.5 epochs	84.2%	150 rounds	3
FedGK	7.1 epochs	85.6%	120 rounds	4
FedKD	6.8 epochs	84.9%	140 rounds	3
MOCHA	7.8 epochs	83.7%	140 rounds	3
ModSquad	6.9 epochs	85.3%	150 rounds	4
Ours	6.2 epochs	87.2%	100 rounds	5

The comparison is structured across four key metrics: Convergence Speed, Accuracy, Communication Overhead, and Model Adaptability. For each method, we adjusted parameters as recommended in the original papers to optimize their performance under identical experimental conditions, allowing for a fair comparison.

In our comparative analysis, ADAMT demonstrates competitive advantages across several key metrics, as shown in table 9. ADAMT achieves the fastest convergence speed, reaching stable accuracy within 6.2 epochs on average. This swift convergence is largely due to ADAMT's adaptive model aggregation and feature expansion strategies, which streamline model alignment across nodes, even when faced with non-IID data distributions. In terms of accuracy, ADAMT attains the highest score among the compared methods, achieving 87.2 %. This superior performance highlights ADAMT's capability to learn robust representations from heterogeneous data, a crucial factor in decentralized, multi-task learning environments such as MANETs. The ability to handle diverse data sources effectively underlines ADAMT's strength in producing consistent, high-quality predictions. ADAMT also excels in communication efficiency. It maintains a low communication overhead, outperforming other models by reducing the volume of data exchanged across nodes. This efficient communication strategy is specifically designed for bandwidth-limited, decentralized environments, where minimizing network load is essential for practical implementation. By optimizing data transfer without sacrificing accuracy, ADAMT proves to be a scalable and resource-conscious solution for distributed learning. Lastly, ADAMT exhibits high model adaptability, a critical attribute for handling the challenges of non-

IID. data in decentralized settings. Its feature expansion mechanism, coupled with adaptive model aggregation, enables ADAMT to adjust flexibly to varied data distributions across nodes. This adaptability ensures that ADAMT sustains strong performance in diverse, dynamic network environments where data heterogeneity is prevalent.

In summary, ADAMT's comprehensive performance across convergence speed, accuracy, communication overhead, and model adaptability validates its suitability for efficient, distributed multi-task learning in mobile ad-hoc networks. These results underscore ADAMT's robustness in decentralized environments and highlight its potential as a state-of-the-art approach for real-world applications with data diversity and network constraint.

To further substantiate the efficiency of our proposed model, we conducted additional experiments to measure the training and communication times for each model in our study. For each model, training time refers to the duration required for model updates on each node, while communication time accounts for the time taken to exchange model updates across the network. Both metrics were averaged over multiple runs to ensure reliable results.

Table 10 presents the comparative results, showcasing the average training and communication times for each model under the same conditions. As illustrated, our model exhibits reduced communication time compared to baseline models, which is especially beneficial in MANET environments with constrained bandwidth. This outcome aligns with the design of our framework, which prioritizes efficiency in both computation and communication.

Table 10. Average epoch time in scenario A.

Model	**Average Training Time (ms)**	**Average Communication Time (ms)**
CFL	150.86	127.34
FedGK	142.29	113.35
FedKD	146.79	118.71
MOCHA	132.91	105.26
ModSquard	139.64	109.13
Ours	92.73	68.59

These results highlight that our model's structure achieves faster convergence and reduced communication overhead, illustrating its suitability for decentralized, resource-constrained networks like MANETs.

6. Conclusions and Future Work

In this paper, we implement distributed multi-task image recognition in mobile ad-hoc networks. Unlike traditional machine learning methods, we are not depend on the cloud data center but use the computing power of multiple devices to build a computing network, which enables our approach to effectively coordinate mobile resources. In addition, we combine the idea of feature expansion and deep hashing algorithm, and adapt to the task characteristics according to the target data of the device, so as to realize the personalized processing of different classification tasks in a specific field and the fusion between models. More importantly, to adapt to the characteristics of mobile ad-hoc networks in the process of collaborative training across devices, we design an adaptive communication strategy, which breaks through the limitations of traditional distributed SGD methods that cannot cope with dynamic networks. The experimental results show that under the condition of limited datasets, our approach makes the mobile model have a better recognition effect on its own tasks, and can achieve the goal of self-evolution and co-evolution of all task models in the process of interacting with the environment.

In this work, the communication model assumes a simplified, noiseless environment with direct device-to-device communication within range, without explicitly accounting for noise and interference. This design choice aligns with our focus on adaptive communication policy optimization under dynamic topology conditions, typical in Mobile Ad-hoc Networks (MANETs). Future studies may extend this framework to incorporate noise and multi-hop scenarios for a more detailed representation of real-world network conditions.

Ackownledgement

This work is supported by the Outstanding Young Science and Technology Talents Project of Jilin Province Science and Technology Development Plan (20240602004RC) and the Key Scientific Research Project of Jilin Provincial Department of Education (JJKH20240800KJ).

References

[1] M. S. Corson, J. P. Macker, Mobile ad hoc networking (MANET): routing protocol performance issues and evaluation considerations, RFC

2501 (1999) 1–12. URL: https://doi.org/10.17487/RFC2501. doi:10.17487/RFC2501.

[2] E. M. Royer, C. Toh, A review of current routing protocols for ad hoc mobile wireless networks, IEEE Wirel. Commun. 6 (1999) 46–55. URL: https://doi.org/10.1109/98.760423. doi:10.1109/98.760423.

[3] A. A. Abbasi, M. F. Younis, A survey on clustering algorithms for wireless sensor networks, Comput. Commun. 30 (2007) 2826–2841. URL: https://doi.org/10.1016/j.comcom.2007.05.024. doi:10.1016/J.COMCOM.2007.05.024.

[4] S. Basagni, M. Conti, S. Giordano, I. Stojmenovic (Eds.), Mobile Ad Hoc Networking: Cutting Edge Directions, Second Edition, Wiley / IEEE, 2013. URL: https://doi.org/10.1002/9781118511305. doi:10.1002/9781118511305.

[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: A. Singh, X. J. Zhu (Eds.), Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA, volume 54 of *Proceedings of Machine Learning Research*, PMLR, 2017, pp. 1273–1282. URL: http://proceedings.mlr.press/v54/mcmahan17a.html.

[6] K. A. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, B. McMahan, T. V. Overveldt, D. Petrou, D. Ramage, J. Roselander, Towards federated learning at scale: System design, in: A. Talwalkar, V. Smith, M. Zaharia (Eds.), Proceedings of the Second Conference on Machine Learning and Systems, SysML 2019, Stanford, CA, USA, March 31 - April 2, 2019, mlsys.org, 2019. URL: https://proceedings.mlsys.org/paper_files/paper/2019/hash/7b770da633baf74895be22a8807f1a8f-Abstract.html.

[7] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, D. Bacon, Federated learning: Strategies for improving communication efficiency, in: NIPS Workshop on Private Multi-Party Machine Learning, 2016. URL: https://arxiv.org/abs/1610.05492.

[8] X. Lian, C. Zhang, H. Zhang, C. Hsieh, W. Zhang, J. Liu, Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent, in: I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017, pp. 5330–5340.

[9] T. Chen, X. Chen, X. Du, A. Rashwan, F. Yang, H. Chen, Z. Wang, Y. Li, Adamv-moe: Adaptive multi-task vision mixture-of-experts, in: IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023, IEEE, 2023, pp. 17300–17311.

[10] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. A. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konečný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, H. Qi, D. Ramage, R. Raskar, M. Raykova, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, S. Zhao, Advances and open problems in federated learning, Found. Trends Mach. Learn. 14 (2021) 1–210. URL: https://doi.org/10.1561/2200000083. doi:10.1561/2200000083.

[11] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith, Federated optimization in heterogeneous networks, in: I. S. Dhillon, D. S. Papailiopoulos, V. Sze (Eds.), Proceedings of the Third Conference on Machine Learning and Systems, MLSys 2020, Austin, TX, USA, March 2-4, 2020, mlsys.org, 2020. URL: https://proceedings.mlsys.org/paper_files/paper/2020/hash/1f5fe83998a09396ebe6477d9475ba0c-Abstract.html.

[12] K. A. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, B. McMahan, T. V. Overveldt, D. Petrou, D. Ramage, J. Roselander, Towards federated learning at scale: System design, in: A. Tal-

walkar, V. Smith, M. Zaharia (Eds.), Proceedings of the Second Conference on Machine Learning and Systems, SysML 2019, Stanford, CA, USA, March 31 - April 2, 2019, mlsys.org, 2019. URL: https://proceedings.mlsys.org/paper_files/paper/2019/hash/7b770da633baf74895be22a8807f1a8f-Abstract.html.

[13] H. Wang, S. Sievert, S. Liu, Z. Charles, D. S. Papailiopoulos, S. J. Wright, ATOMO: communication-efficient learning via atomic sparsification, in: S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada, 2018, pp. 9872–9883.

[14] W. Zhang, X. Liu, S. Tarkoma, Fedgk: Communication-efficient federated learning through group-guided knowledge distillation, ACM Transactions on Internet technology 24 (2024) 1–21.

[15] X. Lian, C. Zhang, H. Zhang, C. Hsieh, W. Zhang, J. Liu, Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent, in: I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017, pp. 5330–5340.

[16] H. Tang, X. Lian, M. Yan, C. Zhang, J. Liu, D^2: Decentralized training over decentralized data, in: J. G. Dy, A. Krause (Eds.), Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018, volume 80 of *Proceedings of Machine Learning Research*, PMLR, 2018, pp. 4855–4863.

[17] M. Assran, N. Loizou, N. Ballas, M. G. Rabbat, Stochastic gradient push for distributed deep learning, in: K. Chaudhuri, R. Salakhutdinov (Eds.), Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, volume 97 of *Proceedings of Machine Learning Research*, PMLR, 2019, pp. 344–353.

[18] L. Luo, Y. Xiong, Y. Liu, X. Sun, Adaptive gradient methods with dynamic bound of learning rate, in: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019, OpenReview.net, 2019.

[19] Z. Lin, G. Qu, W. Wei, X. Chen, K. K. Leung, Adaptsfl: Adaptive split federated learning in resource-constrained edge networks, CoRR abs/2403.13101 (2024). URL: https://doi.org/10.48550/arXiv.2403.13101. doi:10.48550/ARXIV.2403.13101. arXiv:2403.13101.

[20] V. Smith, C. Chiang, M. Sanjabi, A. Talwalkar, Federated multi-task learning, in: I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017, pp. 4424–4434. URL: https://proceedings.neurips.cc/paper/2017/hash/6211080fa89981f66b1a0c9d55c61d0f-Abstract.html.

[21] F. Sattler, K. Müller, W. Samek, Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints, IEEE Trans. Neural Networks Learn. Syst. 32 (2021) 3710–3722.

[22] F. Hanzely, P. Richtárik, Federated learning of a mixture of global and local models, CoRR abs/2002.05516 (2020).

[23] C. Wu, F. Wu, L. Lyu, Y. Huang, X. Xie, Communication-efficient federated learning via knowledge distillation, Nature Communications 13 (2022) 1–8.

[24] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, K. Chan, Adaptive federated learning in resource constrained edge computing systems, IEEE J. Sel. Areas Commun. 37 (2019) 1205–1221.

[25] P. Vanhaesebrouck, A. Bellet, M. Tommasi, Decentralized collaborative learning of personalized models over networks, 2017. URL: https://arxiv.org/abs/1610.05202. arXiv:1610.05202.

[26] A. Lalitha, O. C. Kilinc, T. Javidi, F. Koushanfar, Peer-to-peer federated learning on graphs, CoRR abs/1901.11173 (2019). URL: http://arxiv.org/abs/1901.11173. arXiv:1901.11173.

[27] J. Liu, H. Xu, L. Wang, Y. Xu, C. Qian, J. Huang, H. Huang, Adaptive asynchronous federated learning in resource-constrained edge computing, IEEE Transactions on Mobile Computing 22 (2023) 674–690.

[28] Z. Chen, Y. Shen, M. Ding, Z. Chen, H. Zhao, E. G. Learned-Miller, C. Gan, Mod-squad: Designing mixtures of experts as modular multi-task learners, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023, IEEE, 2023, pp. 11828–11837.

[29] P. Zhou, Q. Lin, D. Loghin, B. C. Ooi, Y. Wu, H. Yu, Communication-efficient decentralized machine learning over heterogeneous networks, in: 2021 IEEE 37th International Conference on Data Engineering (ICDE), 2021, pp. 384–395. doi:10.1109/ICDE51399.2021.00040.

[30] O. J. Faqir, E. C. Kerrigan, D. Gündüz, Information transmission bounds between moving terminals, IEEE Communications Letters 24 (2020) 1410–1413.

[31] A. Ogulenko, I. Benenson, I. Omer, B. Alon, Bayesian estimate of position in mobile phone network, arXiv preprint arXiv:2007.12464 (2020).

[32] S. Wei, R. Guo, C. Cui, B. Lu, S. Dong, T. Gao, Y. Du, Y. Zhou, X. Lyu, Q. Liu, et al., Pp-shitu: a practical lightweight image recognition system, arXiv preprint arXiv:2111.00775 (2021).

[33] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, et al., Searching for mobilenetv3, in: Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 1314–1324.

[34] Y. A. Malkov, D. A. Yashunin, Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs, IEEE transactions on pattern analysis and machine intelligence 42 (2018) 824–836.

[35] L. Wang, K.-J. Yoon, Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks, IEEE transactions on pattern analysis and machine intelligence 44 (2021) 3048–3068.

[36] Y. Chen, W. Ding, J. Lai, Improving noisy student training on non-target domain data for automatic speech recognition, in: ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2023, pp. 1–5.

[37] X. Luo, H. Wang, D. Wu, C. Chen, M. Deng, J. Huang, X.-S. Hua, A survey on deep hashing methods, ACM Transactions on Knowledge Discovery from Data 17 (2023) 1–50.

[38] J. Zhang, Y. Peng, Ssdh: Semi-supervised deep hashing for large scale image retrieval, IEEE Transactions on Circuits and Systems for Video Technology 29 (2017) 212–225.

[39] X. Li, D. Hu, F. Nie, Deep binary reconstruction for cross-modal hashing, in: Proceedings of the 25th ACM international conference on Multimedia, 2017, pp. 1398–1406.

[40] A. Handojo, N. Pujawan, B. Santosa, M. L. Singgih, Job assignment problem on online transportation order using hungarian algorithm, in: 2022 International Conference of Science and Information Technology in Smart Administration (ICSINTESA), IEEE, 2022, pp. 183–187.

[41] X. Lian, W. Zhang, C. Zhang, J. Liu, Asynchronous decentralized parallel stochastic gradient descent, 2018. URL: `https://arxiv.org/abs/1710.06952`. `arXiv:1710.06952`.

[42] J. S. Hunter, The exponentially weighted moving average, Journal of quality technology 18 (1986) 203–210.

[43] H. Touvron, M. Cord, A. El-Nouby, P. Bojanowski, A. Joulin, G. Synnaeve, H. Jégou, Augmenting convolutional networks with attention-based aggregation, arXiv preprint arXiv:2112.13692 (2021).

[44] M.-H. Guo, C.-Z. Lu, Z.-N. Liu, M.-M. Cheng, S.-M. Hu, Visual attention network, Computational Visual Media 9 (2023) 733–752.

[45] K. Wu, J. Zhang, H. Peng, M. Liu, B. Xiao, J. Fu, L. Yuan, Tinyvit: Fast pretraining distillation for small vision transformers, in: European Conference on Computer Vision, Springer, 2022, pp. 68–85.

[46] L. Yuan, Q. Hou, Z. Jiang, J. Feng, S. Yan, Volo: Vision outlooker for visual recognition, IEEE transactions on pattern analysis and machine intelligence 45 (2022) 6575–6586.

[47] D. Han, S. Yun, B. Heo, Y. Yoo, Rethinking channel dimensions for efficient model design, in: Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition, 2021, pp. 732–741.

[48] S. Laine, T. Aila, Temporal ensembling for semi-supervised learning, arXiv preprint arXiv:1610.02242 (2016).

[49] X. Jia, S. Song, W. He, Y. Wang, H. Rong, F. Zhou, L. Xie, Z. Guo, Y. Yang, L. Yu, et al., Highly scalable deep learning training system with mixed-precision: Training imagenet in four minutes, arXiv preprint arXiv:1807.11205 (2018).