Journal of Computer Science & Technology

Indexed in: SCIE, EI, INSPEC, CBST, DBLP, etc.

Sponsored by: ICT, CAS & CCF

JCST Papers

Only for academic and non-commercial use

Thanks for reading!

JCST WeChat Subscription Account

JCST URL: https://jcst.ict.ac.cn

SPRINGER URL: https://www.springer.com/journal/11390

E-mail: jcst@ict.ac.cn

Online Submission: https://mc03.manuscriptcentral.com/jcst

Twitter: JCST_Journal

LinkedIn: Journal of Computer Science and Technology

Chen YQ, Zhang T, Jiang XL *et al.* FedBone: Towards large-scale federated multi-task learning. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 39(5): 1040-1057 Sept. 2024. DOI: 10.1007/s11390-024-3639-x

FedBone: Towards Large-Scale Federated Multi-Task Learning

Yi-Qiang Chen (陈益强), *Fellow, CCF, Senior Member, IEEE*, Teng Zhang (张　腾)
Xin-Long Jiang (蒋鑫龙), *Member, CCF, IEEE*, Qian Chen (陈　前)
Chen-Long Gao (高晨龙), *Member, CCF*, and Wu-Liang Huang (黄武亮)

Beijing Key Laboratory of Mobile Computing and Pervasive Devices, Institute of Computing Technology Chinese Academy of Sciences, Beijing 100190, China

University of Chinese Academy of Sciences, Beijing 100190, China

E-mail: yqchen@ict.ac.cn; zhangteng19s@ict.ac.cn; jiangxinlong@ict.ac.cn; chenqian20b@ict.ac.cn; gaochenlong@ict.ac.cn huangwuliang19b@ict.ac.cn

Received August 3, 2023; accepted March 13, 2024.

Abstract Federated multi-task learning (FMTL) has emerged as a promising framework for learning multiple tasks simultaneously with client-aware personalized models. While the majority of studies have focused on dealing with the non-independent and identically distributed (Non-IID) characteristics of client datasets, the issue of task heterogeneity has largely been overlooked. Dealing with task heterogeneity often requires complex models, making it impractical for federated learning in resource-constrained environments. In addition, the varying nature of these heterogeneous tasks introduces inductive biases, leading to interference during aggregation and potentially resulting in biased global models. To address these issues, we propose a hierarchical FMTL framework, referred to as FedBone, to facilitate the construction of large-scale models with improved generalization. FedBone leverages server-client split learning and gradient projection to split the entire model into two components: 1) a large-scale general model (referred to as the general model) on the cloud server, and 2) multiple task-specific models (referred to as client models) on edge clients, accommodating devices with limited compute power. To enhance the robustness of the large-scale general model, we incorporate the conflicting gradient projection technique into FedBone to rectify the skewed gradient direction caused by aggregating gradients from heterogeneous tasks. The proposed FedBone framework is evaluated on three benchmark datasets and one real ophthalmic dataset. The comprehensive experiments demonstrate that FedBone efficiently adapts to the heterogeneous local tasks of each client and outperforms existing federated learning algorithms in various dense prediction and classification tasks while utilizing off-the-shelf computational resources on the client side.

Keywords federated learning, multi-task learning, split learning, heterogeneous task

1 Introduction

Federated learning[1] is a collaborative model training approach that allows multiple edge devices or institutions to participate in the training process while keeping their data local and confidential. In a typical federated learning pipeline, several edge clients independently train models using their local data. These clients then send their trained model parameters to a central cloud server. The server aggregates these parameters into a global model and distributes the updated model back to the clients. In this way, each client benefits from the knowledge learned by other clients without revealing their private data. In recent years, researchers have proposed various federated learning methods to address challenges such as

Regular Paper
Recommended by FL-IJCAI 2023
This work was partially supported by the Beijing Municipal Science and Technology Commission under Grant No. Z221100002722009, the National Natural Science Foundation of China under Grant No. 62202455, the Youth Innovation Promotion Association of Chinese Academy of Sciences (CAS), the Hunan Provincial Natural Science Foundation of China under Grant No. 2023JJ70034, and the Science Research Foundation of the CAS-Aier Joint Laboratory on Digital Ophthalmology under Grant No. SZYK202201.

handling non-independent and identically distributed (Non-IID) data[2], enhancing privacy protection to prevent data leakage[3]. While the majority of federated learning methods focus on classification tasks, some researches have extended federated learning to handle more complex tasks, for instance object detection[4], semantic segmentation[5], etc. These researches have contributed to the widespread adoption of federated learning in many real-world applications.

Due to the diversity of tasks, it is natural to encounter task heterogeneity among clients in practical federated learning scenarios. For instance, in a medical federated learning setting, different hospitals with the same kind of medical images may have distinct tasks based on their specific medical objectives. One hospital might focus on directly classifying diseases using medical images, while another hospital could be engaged in lesion detection tasks, where doctors/researchers in the hospital can label lesions with bounding boxes to identify locations of lesions. Additionally, another hospital might perform more advanced tasks like lesion segmentation, where doctors/researchers in the hospital may annotate the masks of lesions to compute lesion volumes. The federated learning setting of heterogeneous tasks is defined as federated hetero-task learning in B-FHTL[6], a federated learning benchmark with heterogeneous tasks. This federated learning setting holds considerable promise to unlock a broader range of applications for federated learning. It effectively encourages the participation of various clients with different learning objectives, promoting wider adoption of federated learning and facilitating beneficial collaborations.

Despite the setting definition, the federated multi-task learning (FMTL) of heterogeneous tasks remains a relatively unexplored research area. While considerable attention has been devoted to addressing data heterogeneity, most existing work assumes a consensus on the learning objective and limits the ability to effectively handle task heterogeneity[6]. However, B-FHTL[6] shows unsatisfactory benchmark results of existing FMTL methods, which prompts us to investigate a more tailored FMTL method for heterogeneous tasks. In centralized multi-task learning[7], the presence of data label for every task allows the simultaneous training of multiple tasks towards a complex backbone. Each task is then adapted using a task-aware output head model, enhancing the model's performance for individual tasks. In contrast, federated learning operates in a distributed way, where client data is privacy-sensitive and cannot be shared with other clients for task labeling. This data privacy constraint leads to data and task heterogeneity across clients, making it challenging to align learning objectives and perform aggregation. Moreover, edge clients, such as hospitals, may encounter limitations in computational resources, preventing them from training the full backbone on their own devices. The computational constraint adds another layer of complexity to the FMTL of heterogeneous tasks, necessitating efficient model partitioning and aggregation strategies to accommodate the diverse capabilities of edge clients. Therefore, the FMTL of heterogeneous tasks requires novel techniques to handle data and task heterogeneity while respecting resource limitations.

In light of these challenges, we propose FedBone, a hierarchical FMTL framework as shown in Fig.1, which takes advantage of the server-client split learning paradigm to enable the edge clients to participate in large-scale federated training with low memory footprints. The FedBone framework is designed to execute a multi-stage process for handling heterogeneous tasks which entail client-side data embedding, server-side universal model feature extraction, and client-side task-specific processing. Throughout the process, edge clients are only responsible for computing data embedding and propagating it to the cloud server for feature extraction of the large-scale general model. The resulting latent representations are then dispatched back to the clients to perform task output. In this way, the edge clients can utilize their limited computational resources to focus on task-specific learning, while benefiting from the knowledge shared by the general model. To enhance the general model's generalization, we introduce a gradient projection method and gradient rescaling based on historical gradient attention to reduce the negative impact of conflicting gradient components on gradient aggregation. The task output module on the client is tailored to specific task types but is generally concise due to the assumption of feature extraction having already been fulfilled. Latent representations extracted from the large-scale general model are usually low-level features for various tasks. In addition, we consider issues related to privacy and local environment heterogeneity. We design an approach that combines local differential privacy and the trusted execution environment to address various threats from the cloud server. Asynchronous federated optimization is incorporated to alleviate performance degradation caused by dis-

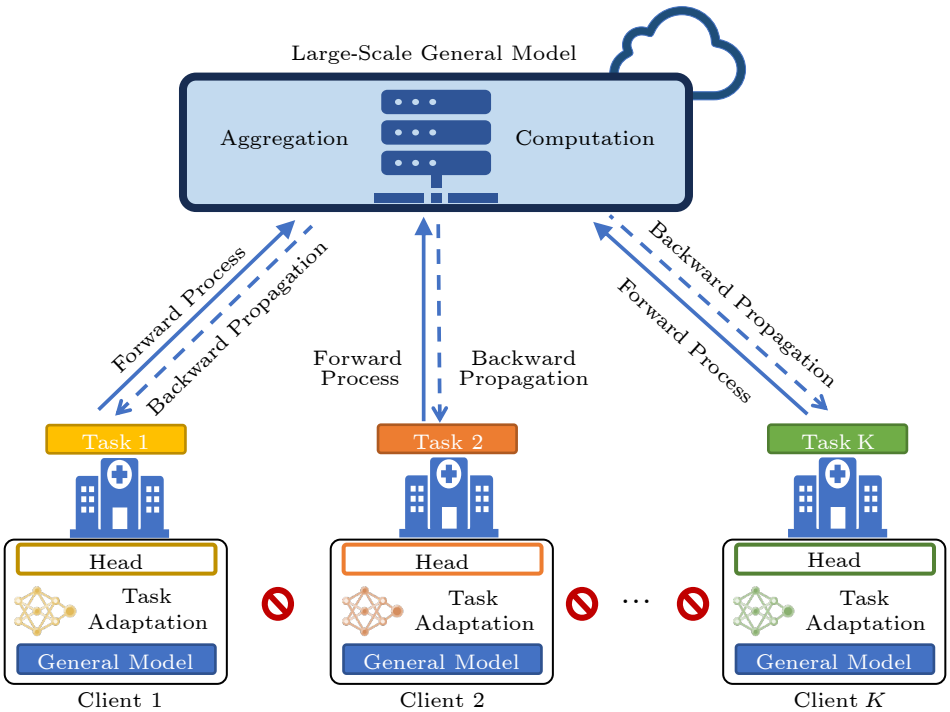

Fig.1. Overview of our proposed framework FedBone.

parities between network states and computational resources. Therefore, we propose a task adaptation module, which utilizes deformable convolutions and a self-attention mechanism to focus on low-level features in the task-specific region and perform task interactions. The proposed task adaptation module significantly improves downstream task performance in the experiments. Our main contributions can be summarized as follows.

• We propose FedBone, a novel FMTL framework via split learning for large-scale federated training on edge clients and heterogeneous task adaptation.

• We propose GPAggregation to alleviate optimization challenges of the general model posed by task heterogeneity among clients, which rescales client gradients with historical gradients attention and merges gradient conflict between clients.

• We design an approach that combines local differential privacy and the trusted execution environment to cover different server roles, tackling threats from semi-trusted and malicious servers while safeguarding sensitive information. Simultaneously, asynchronous optimization is incorporated to alleviate the performance issue caused by the mismatch between client states and computational resources.

• We conduct extensive experiments on three public multi-task datasets. The results show that our proposed FedBone outperforms the compared state-of-the-art federated learning algorithms in heterogeneous tasks with much smaller client resource requirements. The experiments on 13 real-world ophthalmic tasks reveal the potential capability of FedBone in real medical and healthcare applications with heterogeneous dense prediction and classification tasks.

2 Related Work

2.1 Federated Multi-Task Learning

Federated multi-task learning[8] was proposed to handle the statistical challenges of Non-IID data by training personalized models in a federated learning setting. Ditto[9] provides a multi-task learning objective for federated learning to enhance task personalization. The FATHOM framework[10] leverages the attention mechanism to extract input features and learn a shared temporal representation across different devices, thereby achieving knowledge transfer and performance improvement. FedICT[11] achieves personalized models for multi-task clients by using federated prior distillation and local knowledge adjustment.

In addition to addressing the Non-IID problem, FMTL has also expanded to accommodate various data modalities. FedMSplit[12] was proposed to use a dynamic multi-view graph structure to address the modality incongruity problem among sensor devices

and to promote local model relations through neighborhood message passing in the graph. The SpreadGNN[13] framework has solved the Non-IID problem of graph data and uses a dynamic multi-task optimization method to ensure model convergence. Among these FMTL methods, knowledge distillation based approaches and frameworks with hierarchical model structures can adapt to heterogeneity tasks with less modification. The B-FHTL[6] benchmark results show that FMTL methods could outperform traditional federated learning methods, but perform even worse than model trained using only the client's own data in the federated learning setting of heterogeneous tasks.

FedBone follows the FMTL framework, and further takes into account the computational limitations on edge clients and the central server.

2.2 Personalized Federated Learning

Another related topic is personalized federated learning. Personalized federated learning tries to solve the challenges of federated learning on heterogeneous data. There are two different strategies for personalized federated learning: personalization of the global model and individual personalized models. The former performs local adaptation for each client based on the trained global federated model to achieve personalized processing, while the latter trains individual personalized models on each client. In the data augmentation aspect, the self-balancing learning framework Astraea[14] uses z-score based data augmentation and mediator-based multi-client rescheduling to mitigate the impact of data distribution differences. In FedHome[15], each client performs personalized adaptation on a locally enhanced class-balanced dataset.

Some studies use the method of adding a local loss regularization term. For instance, FedProx[16] introduces an approximation term for the local subproblem, taking into account the dissimilarity between the global federated learning model and the local model to adjust the impact of local updates. FedCL[17] uses the regularization term of elastic weight consolidation (EWC) from the continual learning domain. Other methods like transfer learning[18], and meta-learning[19] are also used to improve the performance of the global shared model trained on heterogeneous data in federated learning. The personalized solutions for clients mainly include methods such as parameter decoupling[20], model interpolation[21], and clustering[22]. Specifically, the importance of parameters in FedCurv[23] is estimated by the Fisher information matrix, and a penalty step is performed to retain important parameters, which can reduce the catastrophic forgetting problem between multiple tasks.

The concept of personalized federated learning highlights the necessity of adaptation for local data distribution. Furthermore, FedRep[24] employs a hierarchical structure that learns a shared data representation model across clients and unique local heads for each client. However, existing methods fail to consider the potential for personalization in the event of task heterogeneity[6].

2.3 Foundation Models

The foundation model[25] is a deep learning paradigm in which a model is pre-trained on a large amount of unlabeled data, which can be adapted to various downstream tasks via transfer learning methods like fine-tuning. This paradigm has already been used in a variety of domains, including neural language processing[26], computer vision[27], etc. The pre-training of foundation models requires substantial data and computational power, typically conducted by large data centers.

Some existing work[28] has adopted federated learning to carry out the pre-training process, thereby reducing the resource burden on individual nodes. Most current research focuses on efficiently adapting a foundation model to downstream tasks using adapters[29]. Chen *et al.*[30] have extended the training of these adapters to federated learning. Offsite-tuning[31] provides a method based on knowledge distillation and simulators that can train an adapter without transmitting the entire foundation model, while achieving high performance on downstream tasks. FedKD[32] provides a method to update the foundation model from feature tasks, but since the parameter size of the distillation model is much smaller than the original foundation model, it enables direct training and updating of the foundation model by the clients.

In contrast, our primary goal is to develop a task-agnostic foundation model by incorporating additional layers that can successfully adjust to various downstream tasks. The construction of FedBone exemplifies large-scale models. This enables the foundation model to utilize the knowledge and insights acquired from diverse tasks, thus transforming it into an exceptionally efficient and adaptable feature extractor capable of extracting common patterns across multiple clients.

3 Method

3.1 Problem Formulation

We consider a set of federated clients $\mathcal{K} = \{1, 2, \ldots, K\}$, and each client $k \in \mathcal{K}$ has a local dataset $\mathcal{D}_k = \{(x_k, y_k)_i, i = 1, 2, \ldots, N_k\}$ and collaboratively trains models with other federated learning clients, with the goal of training personalized local models f_k that can adapt to the distinct local task. The goal is to solve the following optimization problems:

$$\forall k \in \mathcal{K}, \quad \min_{f_k \in \mathcal{F}} \mathcal{L}_k(f_k),$$

where $\mathcal{F}$ denotes the set of all personalized local models and $\mathcal{L}_k$ denotes the local loss function.

3.2 Overall Architecture

Our proposed framework FedBone aims to enable the participation of heterogeneous task clients in federated learning, thereby facilitating federated training of large-scale models. To achieve this, we adopt a split federated learning approach[33], which involves the computation of a large-scale general model on the cloud server and the lightweight computation of data embedding and task head output on the edge clients. FedBone aggregates large-scale general models using a task gradients projection method, which prevents gradient conflicts and improves model generalization performance, as opposed to the direct federated averaging aggregation methods. To enhance the performance of client local tasks, we introduce a task adaptation module, which comprises the deformable convolution and self-attention mechanism. These techniques adapt to irregularly shaped feature maps through deformable convolution and capture the task interaction features. The full framework is illustrated in Fig.2. In the following, we will outline the workflow of split FMTL and elaborate on the comprehensive design of federated aggregation via task gradient

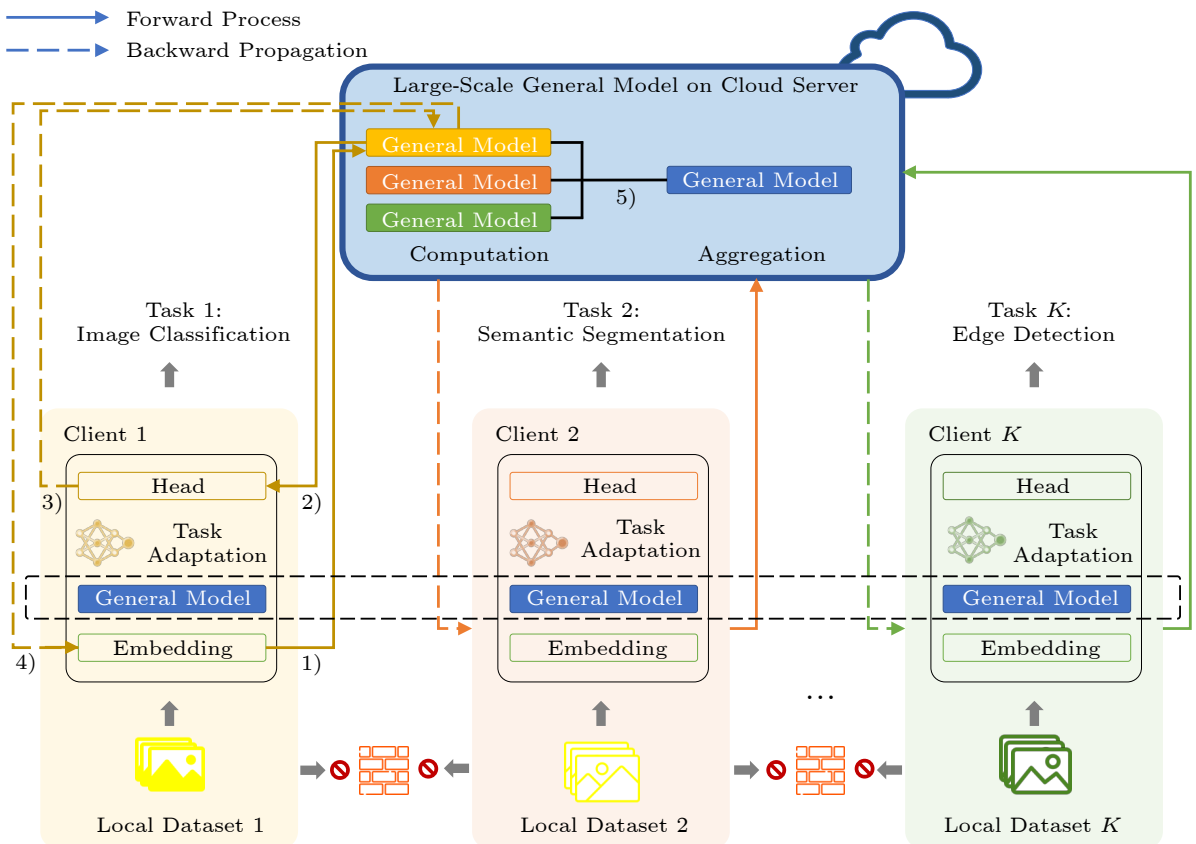

Fig.2. Workflow of FedBone. Clients perform patch embedding locally and 1) send embeddings to the cloud server for feature extraction using the general model, and the cloud server 2) sends extracted features back to clients. Clients complete the loss computation and 3) send backward intermediate results to the cloud server for backward propagation of the general model, and the cloud server 4) sends results back to clients. The clients can now update the local task adaptation module. For each client, the cloud server maintains a distinct general model, which is updated during every client's mini-batch. When all clients finish a local training epoch, the cloud server will 5) aggregate these general models to finalize one communication round.

projection and task adaptation module.

3.3 Split Federated Multi-Task Learning

FedBone follows the split learning[33] approach, but it only requires one cloud server for high-performance computation and model aggregation. All clients perform patch embedding[34] computations in parallel and then send the local results to the cloud server for feature extraction using a large-scale general model. After receiving client results, the cloud server responds to the client with general latent representations. Using these representations, clients sequentially complete forward propagation through the task adaptation module and then the task output head, and immediately begin backward propagation. After receiving the gradients of the general model, the cloud server stores them and then sends the subsequent gradients of the task adaptation module to the original client. Clients with complete gradients can now update the parameters of the local patch embedding, the task adaptation module, and the task output head. When all selected clients send the gradients of the general model, the cloud server aggregates the gradients and updates the parameters of the general model. The specific detail can be found in Algorithm 1.

In Algorithm 1, clients compute patch embedding $e(\cdot)$, task adaptation $l(\cdot)$, and task output head $o(\cdot)$ with parameters ζ, η, and ϕ, respectively. The patch embedding module transposes raw data patches to flatten patch embeddings with a single convolution operation. The task adaptation module is built with deformable convolution and multi-head self-attention, which will be described in more detail in Subsection 3.5. The task output head can vary for heterogeneous tasks, but it typically contains convolution, normalization, and deconvolution operations. Computation on clients yields relatively low resource requirements and can be conducted on low-power consumption edge devices. During clients update, the cloud server gradually gathers task gradients ∇_k^t for gradients aggregation and general model update subsequently.

3.4 Gradients Aggregation via Conflicting Gradients Projection

The cloud server conducts gradient aggregation for optimizing parameters of the general model, which could integrate the knowledge of all client tasks and improve the generalization capability of the general model. Learning multiple tasks simultaneously is a challenging optimization problem that can sometimes lead to poorer model performance[35].

Algorithm 1. FedBone

Input: client set $\mathcal{K}$ with local datasets $\mathcal{D}_k, \forall k \in \mathcal{K}$
Output: general model θ, client task-specific modules ζ, η, ϕ
Server initializes θ^0, $\forall$ client $k \in \mathcal{K}$ initializes ζ_k^0, η_k^0, ϕ_k^0
for round $t = 0, \ldots, T-1$ **do**
  for client $k = 1, \ldots, K$ **do**
    Client patch embedding $\boldsymbol{x}_{k,\,e}^t \leftarrow e(\boldsymbol{x}_k; \zeta_k^t)$
    Server feature extraction $\boldsymbol{x}_{k,\,h}^t \leftarrow f(\boldsymbol{x}_{k,\,e}^t; \theta^t)$
    $\partial \mathcal{L}_k / \partial f \leftarrow$ ClientUpdate$\boldsymbol{x}_{k,\,h}^t$
    $\nabla_k^t = (\partial \mathcal{L}_k / \partial f)(\partial f / \partial \theta^t)$
    Server sends $\partial \mathcal{L}_k / \partial e$ to client
    Client completes backward propagation
    Client optimizes ζ_k^{t+1}, η_k^{t+1}, ϕ_k^{t+1}
  end for
  Server gathers $\nabla_{\mathcal{K}}^t = \{\nabla_1^t, \nabla_2^t, \ldots, \nabla_K^t\}$
  $\nabla^t \leftarrow$ GPAggregation$\nabla_{\mathcal{K}}^t, \nabla^{t-1}$
  $\theta^{t+1} \leftarrow$ Optimizerθ^t, ∇^t
end for
function ClientUpdate($\boldsymbol{x}_{k,\,h}^t$)
  Task adaptation $\boldsymbol{x}_{k,\,l}^t \leftarrow l(\boldsymbol{x}_{k,\,h}^t; \eta_k^t)$
  Task output $\hat{y}_k^t \leftarrow o(\boldsymbol{x}_{k,\,l}^t; \phi_t^k)$
  Task specific loss computation $\mathcal{L}_k(\hat{y}_k^t, y_k)$
  $\partial \mathcal{L}_k / \partial f \leftarrow$ Backward propagation to task adaptation
  return $\partial \mathcal{L}_k / \partial f$
end function

In the federated learning scenario, things become even trickier, since most existing methods require access to raw data to build the relationship between tasks and determine the strategy for aggregating task gradients. To ease the need for raw data, one feasible approach is to attribute multi-objective optimization problems to the existence of gradient conflicts, describe them as gradients from different tasks conflicting with one another among tasks[36], and solve them by correcting the gradients. For gradients ∇_i from client i and gradients ∇_j from client j, we define the conflicting gradients as follows.

Definition 1 (Conflicting Gradient). *Defining the angle between client i gradients ∇_i and client j gradients ∇_j as ω_{ij}, ∇_i and ∇_j are conflicting gradients when $\cos \omega_{ij} < 0$.*

As shown in Fig.3(a), gradients ∇_i and ∇_j have a negative impact on each other, and direct aggregation will cause a reduction in final gradients. An intuitive idea can be projecting one gradient ∇_i onto the normal plane of another gradient ∇_j to eliminate the opposite component:

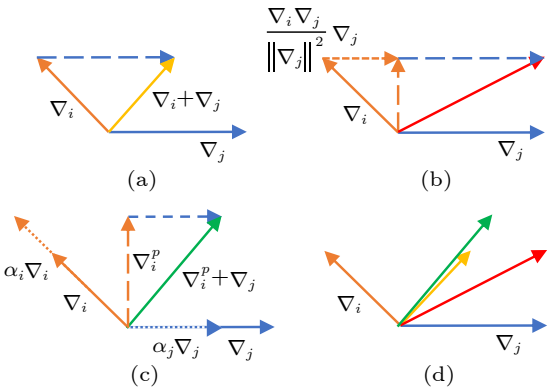

Fig.3. Gradients projection process of two task gradients ∇_i and ∇_j. (a) Two gradients with conflicting gradient directions are aggregated directly, which can lead to interference. (b) Gradients ∇_i are firstly projected onto the normal vector of the gradients ∇_j, and then they are aggregated. (c) Two gradients ∇_i and ∇_j are scaled with attention α, and continue the projection-aggregation procedure. (d) Red, yellow, and green lines represent the aggregated gradients of the three cases, respectively.

$$\nabla_i' = \nabla_i - \frac{\nabla_i : \nabla_j}{\|\nabla_j\|^2}\nabla_j,$$

where ":" denotes the Frobenius inner product, the projecting procedure is illustrated in Fig.3(b). The method works well when the general model converges towards flatter minima, but certain clients may fall into a sharp valley and the weight of the clients should be decreased in the aggregation procedure. To dampen the influence of clients which converge towards sharp minima, we propose a novel gradients aggregation method GPAggregation by the use of historical aggregated gradients. We rescale gradients by calculating the attention values of historical aggregated gradients. A simple example is shown in Fig.3(c): the gradient ∇_i is scaled by attention α_i and then projected onto the normal plane of scaled gradient ∇_j.

The gradient projection method is described in Algorithm 2. The task gradients ∇_k are scaled by attention mechanism with historical aggregated gradients ∇':

$$\nabla_k = \text{softmax}\left(\frac{\nabla_k \nabla'^{\mathrm{T}}}{d_\nabla}\right)\nabla_k.$$

Iterating through the gradients of all the other clients and projecting onto every normal plate, we now get the de-conflicted task gradients which can be used for average aggregation.

3.5 Heterogeneous Task Adaptation

The large-scale general model can extract latent representations with sufficient information for handling heterogeneous tasks on the client side. In centralized multi-task learning, similar task output header structures are used for different tasks to reduce the complexity of optimizing model parameters[7]. In an FMTL scenario, data distribution shifts more unevenly than centralized multi-task learning data. As a result, the latent representations by the general model are more generalized and decoupled from the specific distribution of client data. Relying solely on a lightweight task output head makes it challenging to extract further task-specific information from the general latent representations and apply it to accomplish tasks, leading to a more obvious distribution shift.

Algorithm 2. GPAggregation

Input: a set of current round task gradients $\nabla_{\mathcal{K}}$, previous round aggregated gradients ∇'
Output: aggregated gradients ∇

for $\nabla_k \in \nabla_{\mathcal{K}}$ **do**
  $\nabla_k = \textsc{Softmax}(\nabla_k \nabla'^T / d_\nabla)\nabla_k$
end for
store $\nabla_k^p \leftarrow \nabla_k, \forall \nabla_k \; in \; \nabla_{\mathcal{K}}$
for $\nabla_k \in \nabla_{\mathcal{K}}$ **do**
  for $\nabla_i \in \nabla_{\mathcal{K}} \setminus \nabla_k$ **do**
    if $\nabla_k^p : \nabla_i < 0$ **then**
      $\nabla_k^p = \nabla_k^p - (\nabla_k^p : \nabla_i / \|\nabla_i\|^2)\nabla_i$
    end if
  end for
end for
$\nabla = (\sum_k \nabla_k^p)/K$

Inspired by the successful deformable convolutional network[37] and convolutional Transformer joint structure[38], we propose a heterogeneous task adaptation module that adaptively captures unique receptive regions specific to each task and task interactions. The heterogeneous task adaptation module uses channel-wise pooling, spatial-wise sampling, and intra-task attention to learn relevant task-specific features. Utilizing the reconstructed feature representations enables the task output head to perform downstream tasks more effectively and efficiently.

The heterogeneous task adaptation module mainly consists of 1×1 convolution, deformable convolution, and self-attention mechanism, as shown in Fig.4. The module uses general latent representation $\boldsymbol{x}_h$ received from the cloud server which is initially fed into a linear layer to reduce the channel dimension. The feature map then employs 1×1 convolution to communicate between channels. Following the Gaussian error linear unit (GELU) activation, the resulting feature map is denoted as $\boldsymbol{x}_h'$.

Following [37], we first sample a regular grid $\mathcal{R}$

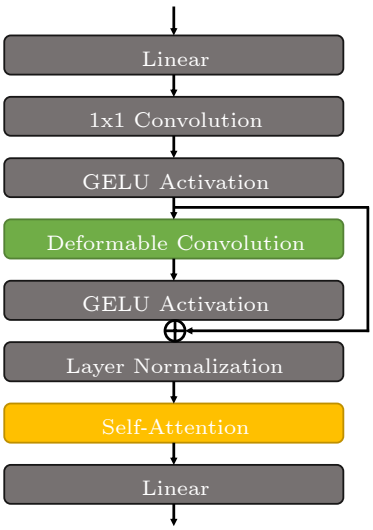

Fig.4. Illustration of task adaptation module.

over the input feature map $\boldsymbol{x}'_h$ and then the summation of sampled values weighted by $\boldsymbol{w}$. To generate the relative offsets with respect to the reference point $\boldsymbol{p}$, the full feature map $\boldsymbol{x}'_h$ is fed to the convolution operator to learn the corresponding offsets $\boldsymbol{\delta_p}$. For each location point $\boldsymbol{p}$ and a regular grid $\mathcal{R}$, the deformable convolution can be formulated as:

$$\boldsymbol{x}_d(\boldsymbol{p}) = \sum_{\boldsymbol{\delta_p}\in\mathcal{R}} \boldsymbol{w}(\boldsymbol{p}) \cdot \boldsymbol{x}'_h(\boldsymbol{p}+\boldsymbol{\delta_p}).$$

The output $\boldsymbol{x}_d$ is then projected into the queries ($\boldsymbol{x}_q$), keys ($\boldsymbol{x}_k$), and values ($\boldsymbol{x}_v$) of dimension d_k using linear transformation for the final self-attention. The self-attention mechanism calculates the attention weights by computing the dot product between the queries and keys, scaled by the square root of the dimension d_k. The softmax function is applied to normalize the attention weights. Finally, the values are weighted by the attention weights to obtain the output:

$$\boldsymbol{x}_t = \text{softmax}\left(\frac{\boldsymbol{x}_q \boldsymbol{x}_k^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{x}_v,$$

and $\boldsymbol{x}_q$, $\boldsymbol{x}_k$, and $\boldsymbol{x}_v$ are obtained by multiplying with learnable weight matrices $\boldsymbol{W}_q$, $\boldsymbol{W}_k$, and $\boldsymbol{W}_v$, respectively. The layer normalization $LN(\cdot)$ layer is applied before the self-attention:

$$\begin{aligned}\boldsymbol{x}_q &= LN(\boldsymbol{x}_d\boldsymbol{W}_q),\\ \boldsymbol{x}_k &= LN(\boldsymbol{x}_d\boldsymbol{W}_k),\\ \boldsymbol{x}_v &= LN(\boldsymbol{x}_d\boldsymbol{W}_v).\end{aligned}$$

Finally, a linear layer is utilized to enhance the features produced by the self-attention module, generating task-specific adapted features.

3.6 Asynchronous Algorithm

Synchronous federated learning methods require that every selected client returns updated results in each round. Due to the differences in computing capabilities, network bandwidth, latency, and data volume among clients, the time required for each round of updates from each client may vary significantly. Although the process of clients uploading local updates is asynchronous, the cloud server must wait for all participants to complete their updates before it can perform the aggregation. Consequently, the slowest client determines the training time per round in synchronous federated learning. In the FMTL scenarios with task heterogeneity, the time difference in client updates caused by these factors will be even greater due to the larger size of the general model.

Asynchronous federated learning[39] enables the cloud server to initiate the aggregation process after receiving at least a predetermined proportion of client updates. This provides flexibility in server aggregation, as it does not require all clients to complete local training and network uploading. For clients who lag behind, the cloud server measures the weights based on the asynchronous federated learning method once their local updates are uploaded. The cloud server then performs a weighted averaging aggregation for these outdated updates. Numerous asynchronous federated learning methods[40] have been proposed, offering more flexible aggregation strategies. However, these methods may lead to performance degradation as updates from certain clients might be discarded due to their relatively small weight in the aggregation strategy.

FedBone employs a split learning paradigm, whereby the general model is kept, avoiding the network latency issues caused by uploading the general model gradients from clients. However, the differences in computational capabilities among clients and the varied sizes of local datasets still result in the diversity of the total time for each round of local updates. Therefore, we propose an asynchronous FedBone algorithm that allows the aggregation of the general model after p clients have completed a round of local training. This idea is similar to FedBuff[41], but under the paradigm of split learning, every client retains some batches of general model update gradients on the cloud server. Hence, we set an update completion ratio λ, which represents the ratio of updated batches to total batches, and the cloud server aggregates gradient updates from both clients who

have accumulated Λ updates and the p fully completed clients. Before updating the general model, it is necessary to rescale the gradients of all participants involved in the aggregation using GradNorm[42] or similar methods. It is to ensure the balance of the gradients involved in the aggregation and maintain convergence stability. The updated general model parameters will be immediately implemented on the cloud server, and the ratio λ will be reset. In this way, the flexibility and scalability of federated learning are enhanced, while also optimizing the utilization of gradient updates from participants and heterogeneous tasks. The algorithm process is shown in Algorithm 3.

Algorithm 3. Asynchronous FedBone

Input: client set $\mathcal{K}$ with local datasets $\mathcal{D}_k, \forall k \in \mathcal{K}$

Output: general model θ, client task-specific modules ζ, η, ϕ

Server Side:

Server initializes θ^0, $\lambda_{\mathcal{K}} = \{\lambda_1, \ldots, \lambda_K\}$
for round $t = 0, \ldots, T-1$ **do**
  while any p clients have fully completed **do**
    Server records λ_k for each client
    Server gathers $\nabla^t_{\mathcal{PC}} = \{\nabla^t_1, \nabla^t_2, \ldots, \nabla^t_p\}$
    Server gathers $\nabla^t_{\mathcal{PR}} = \{\nabla^t_k | \lambda_k > \Lambda\}$
    $\nabla^t_{\mathcal{P}} = \nabla^t_{\mathcal{PC}} \cup \nabla^t_{\mathcal{PR}}$
    $\nabla^t \leftarrow \text{GPAggregation}(\nabla^t_{\mathcal{P}}, \nabla^{t-1})$
    $\theta^{t+1} \leftarrow \text{Optimizer}(\theta^t, \nabla^t)$
    Server resets $\lambda_{\mathcal{K}}$
  end while
end for

Client Side:

Do the same as synchronous FedBone

3.7 Privacy Threat and Protection

The general model employed by FedBone operates exclusively within the server-side infrastructure, effectively mitigating existing client-side privacy-related attacks. Consequently, our focus lies solely on attacks initiated from the server environment. To tackle this challenge, we follow a prevalent threat model that portrays the adversary as an honest-but-curious server. This implies that while the server complies with allowing federated learning algorithms to run according to their design, it actively attempts to infer sensitive information from the embeddings uploaded by clients.

In our defense strategy, we integrate trusted execution environments (TEEs)[43] within the server infrastructure to counter this identified threat. TEEs represent hardware extensions specifically designed to furnish both integrity and confidentiality assurances during security-sensitive computations conducted within an untrusted environment, without exposing the data or processing activities to the host system (comprising the kernel, hypervisor, etc.). Primarily, TEEs aim to resolve the challenge of executing secure remote computations on potentially untrustworthy machines, ensuring integrity and trustworthiness throughout. As a consequence, TEEs have garnered widespread adoption in privacy-preserving federated learning methods[44]. Within the TEE-augmented FedBone framework, the cloud server disseminates a public key generated by TEEs to clients. This key serves the purpose of encrypting local embeddings possessed by clients before transmitting them to the cloud server. Subsequently, the cloud server receives these encrypted embeddings and decrypts them within the confines of the TEE environment. As a result, client embeddings remain shielded from unauthorized inference, thus bolstering the overall security posture of the system.

While TEEs serve to prevent unauthorized access to client embeddings by an honest-but-curious server, it is imperative to acknowledge the susceptibility of TEEs to various side-channel attacks. Should a successful attack occur, the cloud server could potentially transform into a malicious entity, posing a severe threat to the security of the system. To fortify the protection of client embeddings in such precarious scenarios, we leverage the efficacy of local differential privacy (LDP)[45]. Incorporating LDP involves applying a zero-mean Gaussian noise mechanism to the client embeddings, thereby mitigating the risk of information leakage. This process is guided as follows:

$$\boldsymbol{x}_e = e(\boldsymbol{x}; \zeta) + \mathcal{N}(0,\ \sigma^2),$$

where $\mathcal{N}(0,\ \sigma^2)$ is a sample from the normal distribution with mean zero and variance σ^2. The value of σ^2 is determined by $\sigma^2 = 2s^2 \log(1.25/\delta)/\epsilon^2$, where s represents the sensitivity of the embedding module e and ϵ denotes the privacy budget. Consequently, the client embeddings, augmented with meticulously crafted noise, are transmitted to the cloud server for subsequent feature extraction. This strengthens the data's resilience against potential inference or exploitation, especially in scenarios involving compromised TEE security, while satisfying $(\epsilon,\ \sigma)$-LDP.

4 Experiments

4.1 Experimental Setup

We evaluate the performance of FedBone on two

multi-task dense prediction datasets and one large multi-task dataset, which contains both classification and dense prediction tasks. We compare it with a common federated learning method FedAvg[1], personalized federated learning methods FedProx[16] and pFedMe[46], and a multi-task federated learning method FedEM[47].

4.1.1 Datasets

We adopt three publicly accessible datasets to evaluate the performance of our proposed method FedBone, including NYUDv2[48], PASCAL-Context[49], and Taskonomy[7]. NYUDv2 contains 1 449 RGB images and provides dense labels for semantic segmentation, depth estimation, normal estimation, and boundary detection tasks. PASCAL-Context contains 10 180 training RGB images with dense labels for semantic segmentation, saliency estimation, normal estimation, and boundary detection tasks. Meanwhile, Chen *et al.*[50] provided extra human parts annotations for 3 589 images, which act as the labels for the human part segmentation task. The taskonomy dataset contains more than 4.5 million images from 537 building scenes, and we use the tiny split with 366 782 images and 30 building scenes. The taskonomy dataset has annotations for 26 tasks. We discard tasks with corrupt annotations and choose 10 tasks among them: object classification (OC), scene classification (SC), depth estimation with Euclidean depth (DE), depth estimation with z-buffer depth (DZ), surface normals (SN), principal curvature estimation (PC), edge detection in 2D (E2D) and 3D (E3D), and keypoint detection in 2D (K2D) and 3D (K3D). Details of these tasks can be found in [7], and we follow the preprocessing procedure from [51].

4.1.2 Implementation

We determine the number of clients by the tasks of each dataset. For each task in the dataset, we randomly split it into four clients with equal data volumes and partition the training and testing sets in an 8:2 ratio on the clients. For the federated learning methods used for comparison, we design a fully convolutional[52] task-specific output head for each task. For every federated learning method, we set communication rounds to 200. For the taskonomy dataset, since the images are taken from 30 building scenes, each scene naturally forms a data domain. As shown in Fig.5, the specified scenes all feature a view of the kitchen but with distinctive variations in the decoration style and interior arrangement. We consider each scene as representing a data domain and correspond it to a client, dividing them into 30 different data domain clients. These clients are evenly distributed across 10 tasks, ensuring that each task involves three clients conducting the same task.

Fig.5. Kitchen views from nine buildings in the taskonomy dataset. (a) Beechwood. (b) Benevolence. (c) Coffeen. (d) Cosmos. (e) Forkland. (f) Hanson. (g) Hiteman. (h) Lakeville. (i) Leonardo.

For FedBone, FedAvg, and FedEM, we use stochastic gradient descent (SGD)[53] as the optimizer. For FedProx and pFedMe, we have modified the SGD optimizer to fit the optimization process of the algorithm. The batch size is set to 16 and the learning rate is set to 0.001, scheduled to decay by a fraction of 0.1 every 50 epochs for NYUDv2 and PASCAL-Context, and 0.5 every 10 epochs for taskonomy. All our experiments are conducted on the PyTorch framework with eight NVIDIA A800 80 GB GPUs and 1 TB system memory.

4.1.3 Metrics

The chosen two dense prediction datasets comprise a total of five different types of tasks. For semantic segmentation tasks (including human part segmentation), we use mean intersection over union (mIoU) as the metric. For normal estimation tasks, mean error (mErr) is adopted, and for boundary detection tasks, optimal dataset scale F-measure (ods$F$)

is used. For the depth and saliency estimation, the root mean square error (RMSE) and the maximum F-measure (max$F$) are exploited, respectively.

For the taskonomy dataset, we use accuracy (Acc) as the metric of classification tasks, RMSE as the metric of depth estimation with Euclidean depth, depth estimation with z-buffer depth, principal curvature estimation, and keypoint detection in 2D and 3D. The mean Error (mErr) is still the metric of the surface normal estimation task. The mean absolute error (MAE) is employed as the metric of edge detection tasks in 2D and 3D as the labels are derived from a Canny edge detector output without non-maximum suppression, rather than binary edge annotations. Due to the inconsistency in the comparison of metrics such as mIoU, where higher values are preferred, and RMSE, where lower values are preferred, we employ upward (↑) and downward (↓) arrows to denote higher-is-better and lower-is-better metrics in our results, respectively. Additionally, the best results are highlighted in bold font for clarity.

4.1.4 *Backbones*

We employ Swin Transformer Small (Swin-S)[34] pre-trained on ImageNet-22K as the backbone for all experiments except the analysis of computational resource requirements. In order to accommodate the demand of model scale in the production environment, we use a larger model Swin Transformer Base (Swin-B)[34] as the backbone to analyze the computational and memory resources required by various federated learning methods on the client side.

4.2 Evaluation Results

Table 1 and Table 2 present the performance of FedBone on the NYUDv2 dataset, and Table 3 presents the performance on the PASCAL-Context dataset. These tables compare the performance of four different methods, namely FedAvg[1], FedProx[16], pFedMe[46], and FedEM[47], across multiple tasks. In certain tasks such as segmentation, human part, saliency, and bound, higher values indicate better performance, whereas in tasks like depth and normal, lower values indicate superior performance. Our method, FedBone, outperforms all the comparative methods in the segmentation, human part, and saliency tasks. For the bound task, FedBone is only 0.25% lower than the pFedMe method, which achieves the

Table 1. Comparison of Federated Learning Methods on Dataset NYUDv2 for Segmentation and Depth Tasks

Method	Segmentation (mIoU)↑					Depth (RMSE)↓				
	1	2	3	4	Avg.	1	2	3	4	Avg.
FedAvg[1]	38.97	38.29	**43.30**	37.48	39.51	0.428 3	0.529 4	0.557 1	0.717 0	0.558 0
FedProx[16]	29.25	31.92	31.68	24.66	29.38	0.484 6	0.606 9	0.615 5	0.789 2	0.624 1
pFedMe[46]	33.14	33.32	33.90	26.74	31.78	0.434 0	0.531 4	0.546 6	0.745 7	0.564 4
FedEM[47]	41.57	38.33	41.97	40.31	40.55	**0.402 3**	0.522 1	**0.534 6**	0.721 5	0.545 1
FedBone	**42.92**	**40.73**	43.22	**42.47**	**42.34**	0.459 4	**0.519 0**	0.540 7	**0.613 6**	**0.533 2**

Table 2. Comparison of Federated Learning Methods on Dataset NYUDv2 for Normal and Bound Tasks

Method	Normal (mErr)↓					Bound (odsF)↑				
	1	2	3	4	Avg.	1	2	3	4	Avg.
FedAvg[1]	27.01	26.52	25.64	28.40	26.89	62.57	62.23	63.67	59.86	62.08
FedProx[16]	27.68	27.33	26.12	28.75	27.47	61.01	61.27	62.70	58.15	60.78
pFedMe[46]	23.93	25.97	22.76	**27.50**	25.04	62.29	62.92	64.57	60.77	62.63
FedEM[47]	26.44	26.41	25.11	28.16	26.53	62.81	**63.91**	63.82	60.43	62.74
FedBone	**23.67**	**25.32**	**22.57**	27.78	**24.84**	**63.46**	63.74	**65.04**	**61.32**	**63.39**

Table 3. Comparison of Federated Learning Methods on Dataset PASCAL-Context

Method	Segment (mIoU)↑	HumanPart (mIoU)↑	Saliency (maxF)↑	Normal (mErr)↓	Bound (odsF)↑
FedAvg[1]	52.71	56.12	82.97	17.66	61.23
FedProx[16]	61.69	53.21	81.48	15.69	62.32
pFedMe[46]	59.16	57.04	80.90	15.67	**66.59**
FedEM[47]	51.10	53.79	82.15	19.64	59.27
FedBone	**62.74**	**58.09**	**84.36**	**15.13**	66.42

best performance. These results highlight its effectiveness and broad capabilities across diverse tasks. Conversely, for the depth and normal tasks, FedBone consistently achieves lower values, indicating its better performance compared with the comparative methods. These results highlight the effectiveness and generalization of FedBone across diverse tasks.

The experimental results on the taskonomy dataset are shown in Table 4. Due to the ensemble learning design of FedEM, the time consumed in each round of training greatly exceeds that of the other methods, and we do not observe converged results. With large-scale data, both FedProx and pFedMe achieve good performance. Meanwhile, FedBone achieves the best performance in nine out of ten tasks among them, which also indicates that FedBone can carry out preferable performance with large data sizes and generalize the general model to a variety of tasks. It is worth noting that there is substantial room for improvement in the performance of federated learning methods on object and scene classification tasks, which only account for two out of ten tasks in the taskonomy dataset but have 1 000 and 365 classes, respectively. This suggests that the model may be more biased towards optimizing tasks that make up a larger proportion, i.e., dense prediction tasks, and performs poorly on tasks that make up a smaller proportion.

Overall, FedBone exhibits clear advantages over the methods being compared in terms of average performance in spite of a few isolated clients, of which FedBone falls slightly short. The findings demonstrate the effectiveness of FedBone across multiple tasks, validating its potential as a robust method in FMTL of heterogeneous tasks.

4.3 Analysis

4.3.1 *Ablation Study*

We conduct an ablation study of FedBone to evaluate the contribution of each component and setting. The results are presented in Fig.6.

The baseline FedAvg is shown in the first row (Base), while the +GPA and +TA indicate the addition of the GPAggregation and heterogeneous task adaptation module to the baseline, respectively, and +GPT indicates the addition of both modules. The figure shows that compared with FedAvg, the addition of either the GPAggregation or task adaptation module results in improved performance, with the task adaptation module providing a more significant gain in all tasks except the normal estimation task. This finding supports that heterogeneity between different tasks is a critical factor to consider when applying federated learning methods across multiple tasks. The last bar of Fig.6 shows the performance of the proposed FedBone. By integrating both the GPAggregation and task adaptation module, FedBone achieves the best performance among the different settings evaluated.

4.3.2 *Effect of Differential Privacy Budgets*

With a fixed δ value of 0.1, we systematically adjust the parameter ϵ to investigate the impact of differential privacy budgets on performance, as depicted in Table 5. Owing to the inherent heterogeneity among tasks, the results on metrics subsequent to the application of uniform privacy budgets manifest in an inconsistent manner. Globally, there is a discernible tendency wherein metrics across all tasks exhibit a decrement as the privacy budget increases. Notably, the semantic segmentation and edge detection tasks demonstrate relatively consistent declines in metrics, maintaining a functional performance threshold when ϵ surpasses 0.1. Conversely, both surface normal prediction and depth prediction tasks exhibit an immediate and substantial decline in metrics following the implementation of differential privacy. Noteworthy is the considerably greater decline observed in the latter compared with the scenario without differential privacy. This substantiates the need for a tailored design of differential privacy budgets specific to heterogeneous tasks per client, ensuring a judicious trade-off

Table 4. Comparison of Federated Learning Methods on Dataset Taskonomy

Method	OC Acc% ↑	SC Acc% ↑	DE RMSE ↓	DZ RMSE ↓	SN mErr ↓	PC RMSE ↓	E2D MAE ↓	E3D MAE ↓	K2D RMSE ↓	K3D RMSE ↓
FedAvg[1]	24.51	17.79	0.064 7	0.064 7	13.49	0.878 3	0.073 3	0.039 9	0.134 5	0.111 1
FedProx[16]	27.36	24.25	0.068 4	0.068 3	12.91	0.853 8	0.071 5	0.033 6	0.150 4	0.107 4
pFedMe[46]	25.68	19.58	0.059 6	0.059 6	**12.52**	0.809 8	0.063 2	0.033 1	0.133 5	0.106 4
FedEM[47]	13.68	12.13	0.264 5	0.264 5	56.19	1.019 2	0.183 9	0.108 5	0.209 9	0.149 7
FedBone	**39.09**	**56.00**	**0.054 4**	**0.047 7**	12.88	**0.807 2**	**0.061 5**	**0.027 9**	**0.124 7**	**0.100 8**

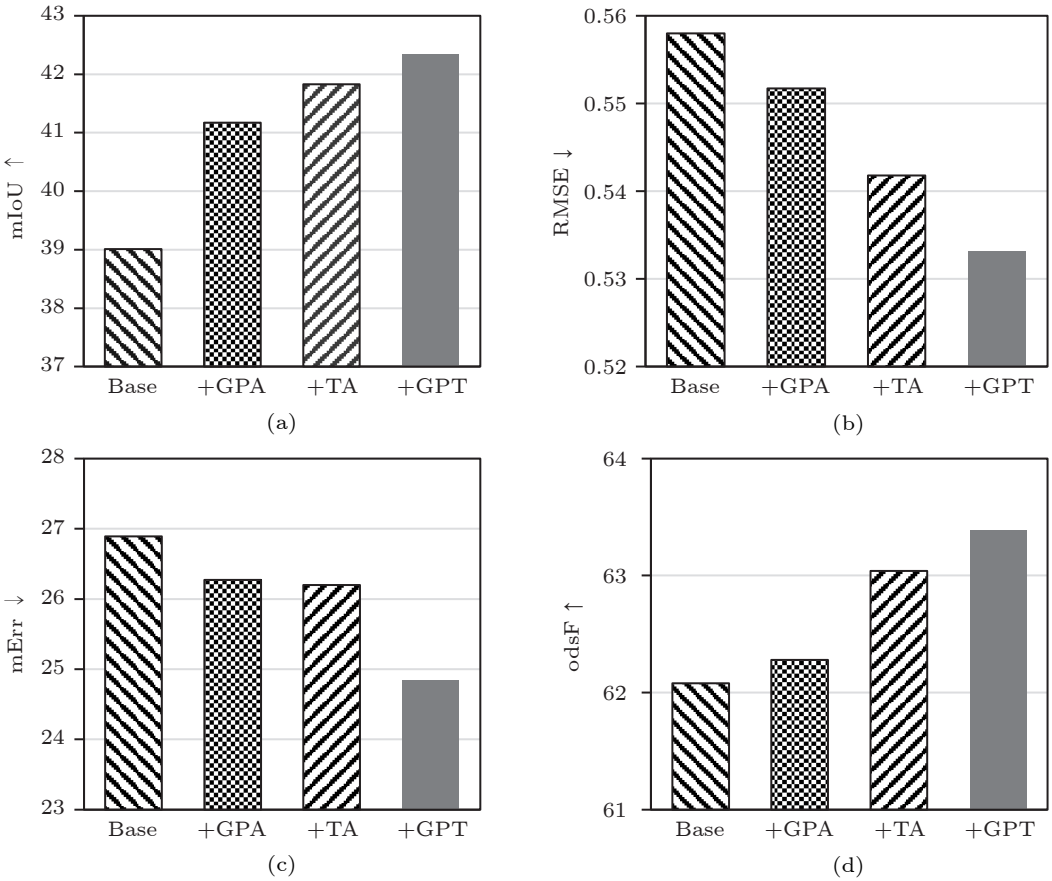

Fig.6. Ablation study results of FedBone. (a) Segmentation. (b) Depth. (c) Normal. (d) Bound.

Table 5. Comparison of Different Privacy Budgets' Effect

Privacy Budget	Segment mIoU↑	Depth RMSE↓	Normal mErr↓	Bound odsF↑
Original	42.34	0.533 2	24.84	63.39
$\epsilon = 0.500$	42.89	0.919 9	30.75	59.00
$\epsilon = 0.100$	38.33	0.942 6	32.02	53.28
$\epsilon = 0.050$	11.85	1.089 4	39.20	50.98
$\epsilon = 0.025$	11.76	1.093 3	42.62	43.46
$\epsilon = 0.001$	0.07	1.143 3	44.65	28.21

between task performance and preservation of privacy integrity.

4.3.3 *Client Resource Requirements*

We conduct an analysis of the computational and communication resource requirements to compare FedBone with other federated learning methods. In Table 6, federated learning methods FedAvg, FedProx, and pFedMe, which employ the fully convolutional task-specific head, have a total parameter number similar to that of FedBone. However, FedBone utilizes the split learning paradigm, which places most computations on the cloud server, and thus, the majority of parameters are not stored locally, resulting in a vast disparity in local computation and memory usage during training. FedEM implements ensemble learning and has triple the parameters of common federated learning methods. Nevertheless, the total memory usage is comparable since it trains sequentially in effect. In terms of network resource require-

Table 6. Computational Resources Required by Federated Learning Methods on the Client Side when Training the Swin-B Model

Method	Number of Parameters ($\times 10^6$)	GFLOPS (G)	Memory (GB)	Comm. Cost per Client (MB)
FedAvg[1]	95.04	123.30	33.32	1 648.44
FedProx[16]	95.04	123.30	33.68	1 648.44
pFedMe[46]	95.04	123.30	33.68	1 648.44
FedEM[47]	285.13	369.91	36.13	4 945.36
FedBone	**1.92** (+88.67)	**11.74** (+87.05)	**3.31** (+32.12)	**32.49**

Note: The + number in brackets means the required resources on the cloud server.

ments, FedBone, which adopts the split learning paradigm, has its communication load for each instance determined by the results of forward propagation for each batch of training data, which includes the patch embedding outputs uploaded by the clients and the general model outputs sent by the server, as well as the results of backward propagation, which includes the task adaptation gradients uploaded by the clients and the general model gradients sent by the server. In the experimental setup of this paper, it requires less than 50MB of traffic. In contrast, federated learning methods being compared require the upload and download of the complete general model parameters for each communication, and result in a significant increase in single communication load compared with FedBone.

4.4 Real-World Ophthalmic Tasks

To further investigate the effectiveness of our proposed method FedBone in real-world applications, we collect 12 912 color fundus images illustrated in Fig.7 and label the images according to ophthalmic diseases, including high myopia maculopathy (HMM), retinal vein occlusion (RVO), proliferative retinopathy (PR), diabetic macular edema (DME), pathological myopia (PM), hypertensive retinopathy (HR), glaucoma (G), macular epiretinal membrane (MEM), and macular hole (MH). We label images that show potential pathological changes but could not be diagnosed as any specific disease, as needing further examination (FE). The 10 diseases, combined with the normal fundus labels, form 10 binary classification (disease diagnosis) tasks. In addition to labeling for disease diagnosis, we also conduct labeling for two types of disease grading, i.e., age-related macular degeneration (AMD) grading and diabetic retinopathy (DR) grading. Together with Retinal-Lesions[54], a retinal lesion segmentation dataset, we build up a 13-task real-world ophthalmic dataset, and the comparison results are shown in Table 7.

All federated learning methods perform well on simple binary classification tasks in Table 7, as these ophthalmic diseases manifest discernible characteristics or distinct features observable in fundus images. Overall, personalized federated learning methods, including FedProx, pFedMe, and FedEM, perform better than the common federated learning method Fe-

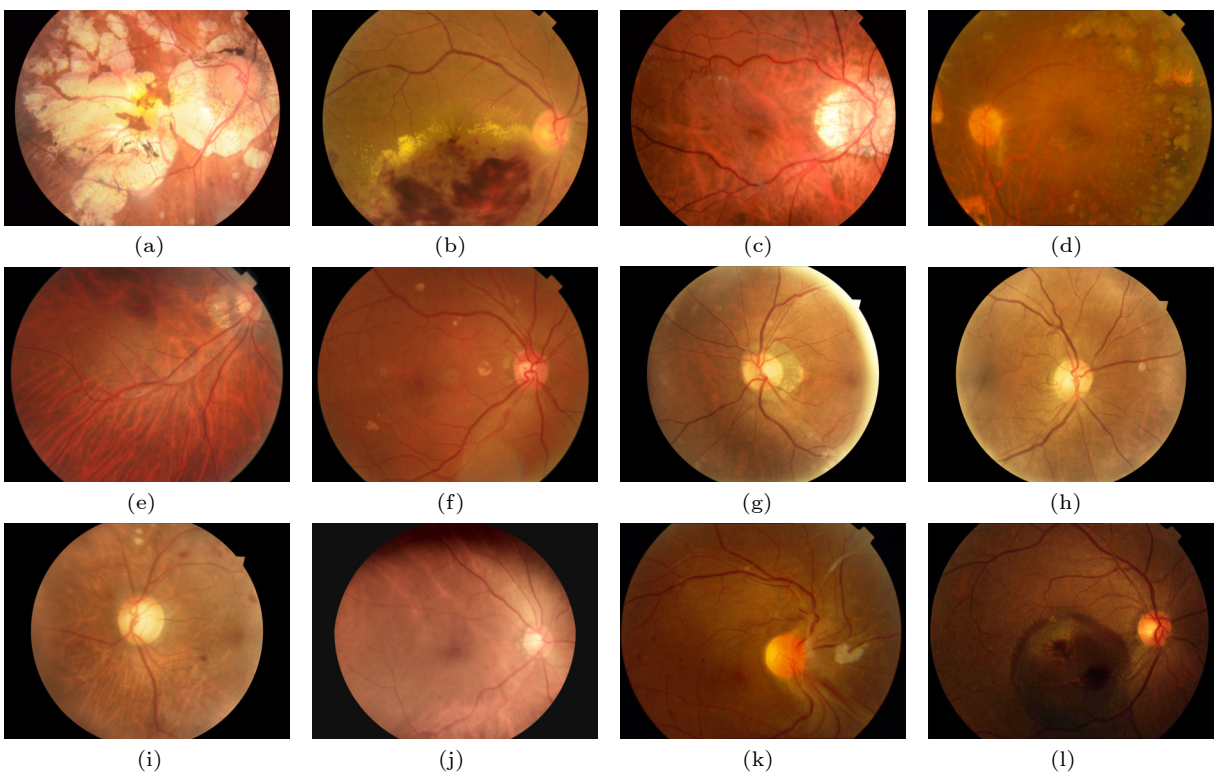

Fig.7. Sample fundus images from real-world ophthalmic dataset. (a) High myopia maculopathy (HMM). (b) Retinal vein occlusion (RVO). (c) Proliferative retinopathy (PR). (d) Diabetic macular edema (DME). (e) Pathological myopia (PM). (f) Hypertensive retinopathy (HR). (g) Glaucoma (G). (h) Macular epiretinal membrane (MEM). (i) Macular hole (MH). (j) Further examination (FE). (k) Age-related macular degeneration (AMD). (l) Diabetic retinopathy (DR).

Table 7. Comparison of Federated Learning Methods on Dataset Real Ophthalmic

Method	HMM	RVO	PR	DME	PM	FE	HR	G	MEM	MH	AMD	DR	LS
FedAvg[1]	94.33	93.72	96.07	93.52	91.27	78.27	94.35	94.22	92.25	94.56	79.63	63.86	49.71
FedProxFedAvg[16]	97.91	97.86	94.30	94.61	92.36	80.20	93.20	95.88	94.58	97.64	88.47	92.98	50.69
pFedMeFedAvg[46]	98.62	96.57	94.94	98.01	92.88	81.30	94.17	96.08	95.99	97.68	90.31	92.06	52.63
FedEMFedAvg[47]	98.79	**99.41**	96.14	96.22	91.25	83.30	95.23	96.22	95.92	97.03	90.15	**94.68**	54.21
FedBone	**98.87**	98.91	**99.24**	**99.13**	**93.71**	**84.57**	**95.90**	**96.75**	**96.15**	**98.93**	**91.18**	94.57	**55.82**

dAvg. Additionally, our proposed FedBone achieves the best performance on the vast majority of tasks, except the retinal vein occlusion (RVO) classification task, on which FedEM achieves the best performance with triple parameters. Fundus images with potential pathological changes may exhibit features indicative of many ophthalmic diseases. FedBone, employing GPAggreation, has effectively cultivated a versatile capability in extracting shared features across diverse ophthalmic diseases, thus demonstrating enhanced classification performance of the further examination (FE) task. Despite the somewhat imprecise grading criteria for diabetic retinopathy (DR), which engenders a degree of subjectivity in labels, FedBone remarkably exhibits a negligible performance disparity of less than 0.2% when compared with the superior-performing FedEM. For the ophthalmic semantic segmentation task LS, FedBone outperforms all the other federated learning methods, which shows the potential of FedBone in real medical scenarios.

5 Conclusions

In this paper, we proposed a novel federated multi-task learning framework FedBone via split learning for large-scale federated training on edge clients and gradient deconflicting aggregation for heterogeneous task adaptation. We further proposed an asynchronous variant of FedBone to mitigate the influence of clients' restricted network connectivity on federated learning aggregation, offering enhanced flexibility and scalability. The extensive experiments showed that FedBone outperforms existing federated learning algorithms in heterogeneous tasks with off-the-shelf computational resources on the client side. The real ophthalmic experiment also indicated a promising future in using FedBone for real medical and healthcare applications. In the future, we may further extend FedBone to encompass a broader range of tasks, especially in the emerging research area of large language models.

Conflict of Interest The authors declare that they have no conflict of interest.

References

[1] McMahan B, Moore E, Ramage D, Hampson S, Arcas B A Y. Communication-efficient learning of deep networks from decentralized data. In *Proc. the 20th International Conference on Artificial Intelligence and Statistics*, Apr. 2017, pp.1273–1282.

[2] Cao X J, Li Z H, Sun G, Yu H F, Guizani M. Cross-silo heterogeneous model federated multitask learning. *Knowledge-Based Systems*, 2023, 265: 110347. DOI: 10.1016/j.knosys.2023.110347.

[3] Mo F, Shamsabadi A S, Katevas K, Demetriou S, Leontiadis I, Cavallaro A, Haddadi H. DarkneTZ: Towards model privacy at the edge using trusted execution environments. In *Proc. the 18th International Conference on Mobile Systems, Applications, and Services*, Jun. 2020, pp.161–174. DOI: 10.1145/3386901.3388946.

[4] Liu Y, Huang A B, Luo Y, Huang H, Liu Y Z, Chen Y Y, Feng L C, Chen T J, Yu H, Yang Q. FedVision: An online visual object detection platform powered by federated learning. In *Proc. the 34th AAAI Conference on Artificial Intelligence*, Feb. 2020, pp.13172–13179. DOI: 10.1609/aaai.v34i08.7021.

[5] Miao J X, Yang Z X, Fan L L, Yang Y. FedSeg: Class-heterogeneous federated learning for semantic segmentation. In *Proc. the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp.8042–8052. DOI: 10.1109/CVPR52729.2023.00777.

[6] Yao L Y, Gao D W, Wang Z, Xie Y X, Kuang W R, Chen D Y, Wang H H, Dong C H, Ding B L, Li Y L. A benchmark for federated hetero-task learning. arXiv: 2206.03436, 2022. http://arxiv.org/abs/2206.03436, Jul. 2024.

[7] Zamir A R, Sax A, Shen W, Guibas L, Malik J, Savarese S. Taskonomy: Disentangling task transfer learning. In *Proc. the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2018, pp.3712–3722. DOI: 10.1109/CVPR.2018.00391.

[8] Smith V, Chiang C K, Sanjabi M, Talwalkar A. Federated multi-task learning. In *Proc. the 31st International Conference on Neural Information Processing Systems*, Dec. 2017, pp.4427–4437.

[9] Li T, Hu S Y, Beirami A, Smith V. Ditto: Fair and robust federated learning through personalization. In *Proc. the 38th International Conference on Machine Learning*,

Jul. 2021, pp.6357–6368.

[10] Chen Y J, Ning Y, Chai Z, Rangwala H. Federated multi-task learning with hierarchical attention for sensor data analytics. In *Proc. the 2020 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2020, pp.1–8. DOI: 10.1109/IJCNN48605.2020.9207508.

[11] Wu Z Y, Sun S, Wang Y W, Liu M, Pan Q Y, Jiang X F, Gao B. FedICT: Federated multi-task distillation for multi-access edge computing. *IEEE Trans. Parallel and Distributed Systems*, 2024, 35(6): 1107–1121. DOI: 10.1109/TPDS.2023.3289444.

[12] Chen J Y, Zhang A D. FedMSplit: Correlation-adaptive federated multi-task learning across multimodal split networks. In *Proc. the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Aug. 2022, pp.87–96. DOI: 10.1145/3534678.3539384.

[13] He C Y, Ceyani E, Balasubramanian K, Annavaram M, Avestimehr S. SpreadGNN: Decentralized multi-task federated learning for graph neural networks on molecular data. In *Proc. the 36th AAAI Conference on Artificial Intelligence*, Feb. 22 -Mar. 1, 2022, pp.6865–6873. DOI: 10.1609/aaai.v36i6.20643.

[14] Duan M M, Liu D, Chen X Z, Liu R P, Tan Y J, Liang L. Self-balancing federated learning with global imbalanced data in mobile systems. *IEEE Trans. Parallel and Distributed Systems*, 2021, 32(1): 59–71. DOI: 10.1109/TPDS.2020.3009406.

[15] Wu Q, Chen X, Zhou Z, Zhang J S. FedHome: Cloud-edge based personalized federated learning for in-home health monitoring. *IEEE Trans. Mobile Computing*, 2022, 21(8): 2818–2832. DOI: 10.1109/TMC.2020.3045266.

[16] Li T, Sahu A K, Zaheer M, Sanjabi M, Talwalkar A, Smith V. Federated optimization in heterogeneous networks. In *Proc. the 3rd Conference on Machine Learning and Systems (MLSys 2020)*, Mar. 2020, pp.429–450.

[17] Yao X, Sun L F. Continual local training for better initialization of federated models. In *Proc. the 2020 IEEE International Conference on Image Processing (ICIP)*, Oct. 2020, pp.1736–1740. DOI: 10.1109/ICIP40778.2020.9190968.

[18] Li D L, Wang J P. FedMD: Heterogenous federated learning via model distillation. arXiv: 1910.03581, 2019. http://arxiv.org/abs/1910.03581, Jul. 2024.

[19] Jiang Y H, Konečný J, Rush K, Kannan S. Improving federated learning personalization via model agnostic Meta learning. arXiv: 1909.12488, 2019. http://arxiv.org/abs/1909.12488, Jul. 2024.

[20] Liang P P, Liu T, Liu Z Y, Allen N B, Auerbach R P, Brent D, Salakhutdinov R, Morency L P. Think locally, act globally: Federated learning with local and global representations. arXiv: 2001.01523, 2020. http://arxiv.org/abs/2001.01523, Jul. 2024.

[21] Diao E N, Ding J, Tarokh V. HeteroFL: Computation and communication efficient federated learning for heterogeneous clients. In *Proc. the 9th International Conference on Learning Representations*, May 2021.

[22] Zhang X, Li Y C, Li W P, Guo K Y, Shao Y F. Personalized federated learning via variational Bayesian inference. In *Proc. the 39th International Conference on Machine Learning*, Jul. 2022, pp.26293–26310.

[23] Shoham N, Avidor T, Keren A, Israel N, Benditkis D, Mor-Yosef L, Zeitak I. Overcoming forgetting in federated learning on Non-IID data. In *Proc. the 2019 Workshop on Federated Learning for Data Privacy and Confidentiality*, Oct. 2019.

[24] Collins L, Hassani H, Mokhtari A, Shakkottai S. Exploiting shared representations for personalized federated learning. In *Proc. the 38th International Conference on Machine Learning*, Jul. 2021, pp.2089–2099.

[25] Bommasani R, Hudson D A, Adeli E *et al.* On the opportunities and risks of foundation models. arXiv: 2108.07258, 2021. http://arxiv.org/abs/2108.07258, Jul. 2024.

[26] Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I. Language models are unsupervised multitask learners. *OpenAI Blog*, 2019, 1(8): 9. https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf, Sept. 2024.

[27] Kirillov A, Mintun E, Ravi N, Mao H Z, Rolland C, Gustafson L, Xiao T T, Whitehead S, Berg A C, Lo W Y, Dollár P, Girshick R. Segment anything. arXiv: 2304.02643, 2023. http://arxiv.org/abs/2304.02643, Jul. 2024.

[28] Tian Y Y S, Wan Y, Lyu L, Yao D Z, Jin H, Sun L C. FedBERT: When federated learning meets pre-training. *ACM Trans. Intelligent Systems and Technology*, 2022, 13(4): 66. DOI: 10.1145/3510033.

[29] Houlsby N, Giurgiu A, Jastrzebski S, Morrone B, de Laroussilhe Q, Gesmundo A, Attariyan M, Gelly S. Parameter-efficient transfer learning for NLP. In *Proc. the 36th International Conference on Machine Learning*, Jun. 2019, pp.2790–2799.

[30] Chen C C, Feng X H, Zhou J, Yin J W, Zheng X L. Federated large language model: A position paper. arXiv: 2307.08925, 2023. http://arxiv.org/abs/2307.08925, Jul. 2024.

[31] Xiao G X, Lin J, Han S. Offsite-tuning: Transfer learning without full model. arXiv: 2302.04870, 2023. http://arxiv.org/abs/2302.04870, Jul. 2024.

[32] Wu C H, Wu F Z, Lyu L, Huang Y F, Xie X. Communication-efficient federated learning via knowledge distillation. *Nature Communications*, 2022, 13(1): Article No. 2032. DOI: 10.1038/s41467-022-29763-x.

[33] Thapa C, Arachchige P C M, Camtepe S, Sun L C. SplitFed: When federated learning meets split learning. In *Proc. the 36th AAAI Conference on Artificial Intelligence*, Feb. 22–Mar. 1, 2022, pp.8485–8493. DOI: 10.1609/aaai.v36i8.20825.

[34] Liu Z, Hu H, Lin Y T, Yao Z L, Xie Z D, Wei Y X, Ning J, Cao Y, Zhang Z, Dong L, Wei F R, Guo B N. Swin transformer V2: Scaling up capacity and resolution. In

Proc. the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2022, pp.11999–12009. DOI: 10.1109/CVPR52688.2022.01170.

[35] Rusu A A, Colmenarejo S G, Gülçehre Ç, Desjardins G, Kirkpatrick J, Pascanu R, Mnih V, Kavukcuoglu K, Hadsell R. Policy distillation. In *Proc. the 4th International Conference on Learning Representations*, May 2016.

[36] Yu T H, Kumar S, Gupta A, Levine S, Hausman K, Finn C. Gradient surgery for multi-task learning. In *Proc. the 34th International Conference on Neural Information Processing Systems*, Dec. 2020, pp. 5824–5836. DOI: 10.5555/3495724.3496213.

[37] Dai JF, Qi HZ, Xiong YW, Li Y, Zhang GD, Hu H, Wei YC. Deformable convolutional networks. In *Proc. the 2017 IEEE International Conference on Computer Vision*, Oct. 2017, pp.764–773. DOI: 10.1109/ICCV.2017.89.

[38] Xu Y Y, Yang Y B, Zhang L F. DeMT: Deformable mixer transformer for multi-task learning of dense prediction. In *Proc. the 37th AAAI Conference on Artificial Intelligence*, Fed. 2023, pp.3072–3080. DOI: 10.1609/aaai.v37i3.25411.

[39] Xie C, Koyejo S, Gupta I. Asynchronous federated optimization. In *Proc. the 12th Annual Workshop on Optimization for Machine Learning*, Dec. 2020.

[40] Imteaj A, Thakker U, Wang S Q, Li J, Amini M H. A survey on federated learning for resource-constrained IoT devices. *IEEE Internet of Things Journal*, 2022, 9(1): 1–24. DOI: 10.1109/JIOT.2021.3095077.

[41] Nguyen J, Malik K, Zhan H Y, Yousefpour A, Rabbat M, Malek M, Huba D. Federated learning with buffered asynchronous aggregation. In *Proc. the 25th International Conference on Artificial Intelligence and Statistics*, Mar. 2022, pp.3581–3607.

[42] Chen Z, Badrinarayanan V, Lee C Y, Rabinovich A. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proc. the 35th International Conference on Machine Learning*, Jul. 2018, pp.793–802.

[43] Sabt M, Achemlal M, Bouabdallah A. Trusted execution environment: What it is, and what it is not. In *Proc. 2015 IEEE Trustcom/BigDataSE/Ispa*, Aug. 2015, pp.57–64. DOI: 10.1109/Trustcom.2015.357.

[44] Kato F, Cao Y, Yoshikawa M. Olive: Oblivious federated learning on trusted execution environment against the risk of sparsification. *Proceedings of the VLDB Endowment*, 2023, 16(10): 2404–2417. DOI: 10.14778/3603581.3603583.

[45] Dwork C, Roth A. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 2014, 9(3/4): 211–407. DOI: 10.1561/0400000042.

[46] Dinh C T, Tran N H, Nguyen T D. Personalized federated learning with Moreau envelopes. In *Proc. the 34th International Conference on Neural Information Processing Systems*, Dec. 2020, Article No. 1796.

[47] Marfoq O, Neglia G, Bellet A, Kameni L, Vidal R. Federated multi-task learning under a mixture of distributions. In *Proc. the 35th International Conference on Neural Information Processing Systems*, Dec. 2021, pp.15434–15447.

[48] Silberman N, Hoiem D, Kohli P, Fergus R. Indoor segmentation and support inference from RGBD images. In *Proc. the 12th European Conference on Computer Vision*, Oct. 2012, pp.746–760. DOI: 10.1007/978-3-642-33715-4_54.

[49] Mottaghi R, Chen X J, Liu X B, Cho N G, Lee S W, Fidler S, Urtasun R, Yuille A. The role of context for object detection and semantic segmentation in the wild. In *Proc. the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2014, pp.891–898. DOI: 10.1109/CVPR.2014.119.

[50] Chen X J, Mottaghi R, Liu X B, Fidler S, Urtasun R, Yuille A. Detect what you can: Detecting and representing objects using holistic models and body parts. In *Proc. the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2014, pp.1979–1986. DOI: 10.1109/CVPR.2014.254.

[51] Chen Z T, Shen Y K, Ding M Y, Chen Z F, Zhao H S, Learned-Miller E G, Gan C. Mod-Squad: Designing mixtures of experts as modular multi-task learners. In *Proc. the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2023, pp.11828–11837. DOI: 10.1109/CVPR52729.2023.01138.

[52] Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. In *Proc. the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp.3431–3440. DOI: 10.1109/CVPR.2015.7298965.

[53] Robbins H, Monro S. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 1951, 22(3): 400. DOI: 10.1214/aoms/1177729586.

[54] Wei Q J, Li X R, Yu W H, Zhang X, Zhang Y P, Hu B J, Mo B, Gong D, Chen N, Ding D Y, Chen Y X. Learn to segment retinal lesions and beyond. In *Proc. the 25th International Conference on Pattern Recognition (ICPR)*, Jan. 2021, pp.7403–7410. DOI: 10.1109/ICPR48806.2021.9412088.

Yi-Qiang Chen received his Ph.D. degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, in 2003. He is currently a professor and the deputy director of ICT, CAS, Beijing. His research interests include artificial intelligence, pervasive computing, and human-computer interaction.

Teng Zhang received his B.S. degree in computer science from Xidian University, Xi'an, in 2015. He is currently pursuing his Ph.D. degree with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. His research interests include pervasive computing, multi-task learning, and federated learning.

Xin-Long Jiang received his Ph.D. degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, in 2018. He is currently an associate professor of the Research Center for Ubiquitous Computing Sytems at ICT, CAS, Beijing. His research interests include artificial intelligence, pervasive computing, and federated learning.

Qian Chen received his B.S. degree in computer science and technology from Tianjin Agricultural University, Tianjin, in 2016, and his M.E. degree in computer technology from Lanzhou Jiaotong University, Lanzhou, in 2020. He is currently pursuing his Ph.D. degree with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. His research interests include machine learning, federated learning, and privacy-preserving technology.

Chen-Long Gao received his Ph.D. degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, in 2023. He is currently an associate professor of the Research Center for Ubiquitous Computing Sytems at ICT, CAS, Beijing. His research interests include artificial intelligence, pervasive computing, and human-computer interaction.

Wu-Liang Huang received his B.S. degree from Sichuan University, Chengdu, in 2019. He is currently pursuing his Ph.D. degree with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. His research interests include pervasive computing, graph learning, and transfer learning.