An Edge-Cloud Collaboration Framework for Generative AI Service Provision with Synergetic Big Cloud Model and Small Edge Models

Yuqing Tian, *Student Member, IEEE,* Zhaoyang Zhang, *Senior Member, IEEE,*
Yuzhi Yang, *Student Member, IEEE,* Zirui Chen, *Student Member, IEEE,* Zhaohui Yang, *Member, IEEE,*
Richeng Jin, *Member, IEEE,* Tony Q. S. Quek, *Fellow, IEEE,* and Kai-Kit Wong, *Fellow, IEEE*

***Abstract*—Generative artificial intelligence (GenAI) offers various services to users through content creation, which is believed to be one of the most important components in future networks. However, training and deploying big artificial intelligence models (BAIMs) introduces substantial computational and communication overhead. This poses a critical challenge to centralized approaches, due to the need of high-performance computing infrastructure and the reliability, secrecy and timeliness issues in long-distance access of cloud services. Therefore, there is an urging need to decentralize the services, partly moving them from the cloud to the edge and establishing native GenAI services to enable private, timely, and personalized experiences. In this paper, we propose a brand-new bottom-up BAIM architecture with synergetic big cloud model and small edge models, and design a distributed training framework and a task-oriented deployment scheme for efficient provision of native GenAI services. The proposed framework can facilitate collaborative intelligence, enhance adaptability, gather edge knowledge and alleviate edge-cloud burden. The effectiveness of the proposed framework is demonstrated through an image generation use case. Finally, we outline fundamental research directions to fully exploit the collaborative potential of edge and cloud for native GenAI and BAIM applications.**

***Index Terms*—Generative AI, big AI model, edge-cloud collaboration.**

I. Introduction

Generative artificial intelligence (GenAI) is an automated methodology that explores data structures and features to generate content resembling human-created material [1]. GenAI interacts with users to offer personalized services, including the generation of images, text, and videos. The evolution of GenAI, such as large language model (LLM) like GPT-4, image generation model like Dall-E 3, audio generation model like AudioPaLM, cross-modal model like Ferret-UI, enhances the quality of service (QoS) and the quality of experience (QoE) in various tasks.

However, the emergent abilities in large-scale GenAI, come at the cost of prohibitive computational and communication resource consumption when operated as centralized cloud service. Meanwhile, the sixth generation (6G) communication network is shifting from connected intelligence to collaborative intelligence [2], where the big artificial intelligence model (BAIM) [3] and small edge models collaborate for service provision. In the anticipated system, the cloud server maintains a unified BAIM by integrating small edge models with diverse tasks. After training, the enhanced BAIM can be partitioned into small models corresponding to the tasks, facilitating edge deployment and enabling the delivery of high-performance, low-latency native GenAI services. The term "native AI" refers to seamlessly embedding AI across the entire network infrastructure. Similarly, we use "native GenAI" to denote AI-driven generative service capabilities within 6G networks.

To tackle issues such as adaptability, knowledge acquisition, and overhead for centralized learning, a scalable BAIM architecture with a distributed model training paradigm is required.

1) Multi-task and Cross-Scenario Adaptability: The unified BAIM needs to effectively address the demands of all users. To manage the continuous increase in users, service diversity, and application complexity, the BAIM architecture should be scalable and capable of handling diverse tasks. Additionally, in real-world systems, edges exhibit heterogeneity in communication, computation, and storage capabilities. Furthermore, connections between nodes are unstable, so edge nodes may join or leave midway. Therefore, BAIM must remain adaptable to heterogeneous model aggregation and dynamic networks across different scenarios.

2) Large-Scale Knowledge Acquisition: BAIM has outperformed smaller models in various domains with ample training data. In 6G networks, data generated by individual devices is often not enough to train a high-quality model. To gather them together, edge models could extract local intelligence and transfer it to the center. This large-scale knowledge acquisition, powered by edge intelligence, forms the foundation for developing high-quality BAIMs. It ensures a comprehensive understanding of various scenarios by integrating global

This work was supported in part by National Natural Science Foundation of China under Grants U20A20158 and 62394292, Ministry of Industry and Information Technology under Grant TC220H07E, National Key R&D Program of China under Grant 2020YFB1807101, Zhejiang Provincial Key R&D Program under Grant 2023C01021, and the Fundamental Research Funds for the Central Universities. (*Corresponding Author: Zhaoyang Zhang*)

Y. Tian (e-mail: tianyq@zju.edu.cn), Z. Zhang (e-mail: ning_ming@zju.edu.cn), Y. Yang (e-mail: yuzhi_yang@zju.edu.cn), Z. Chen (e-mail: ziruichen@zju.edu.cn), Z. Yang (e-mail: yang_zhaohui@zju.edu.cn) and R. Jin (e-mail: richengjin@zju.edu.cn) are with College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China, and also with Zhejiang Provincial Key Laboratory of Info. Proc., Commun. & Netw. (IPCAN), Hangzhou 310027, China.

T. Q. S. Quek (email: tonyquek@sutd.edu.sg) is with the ISTD Pillar, Singapore University of Technology and Design (SUTD), Singapore 487372, and also with the SUTD-ZJU IDEA Center of Network Intelligence, Singapore 487372.

K. Wong (e-mail: kai-kit.wong@ucl.ac.uk) is with the Department of Electronic and Electrical Engineering, University College London, UK.

This article has been accepted for publication in IEEE Network. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MNET.2024.3420755

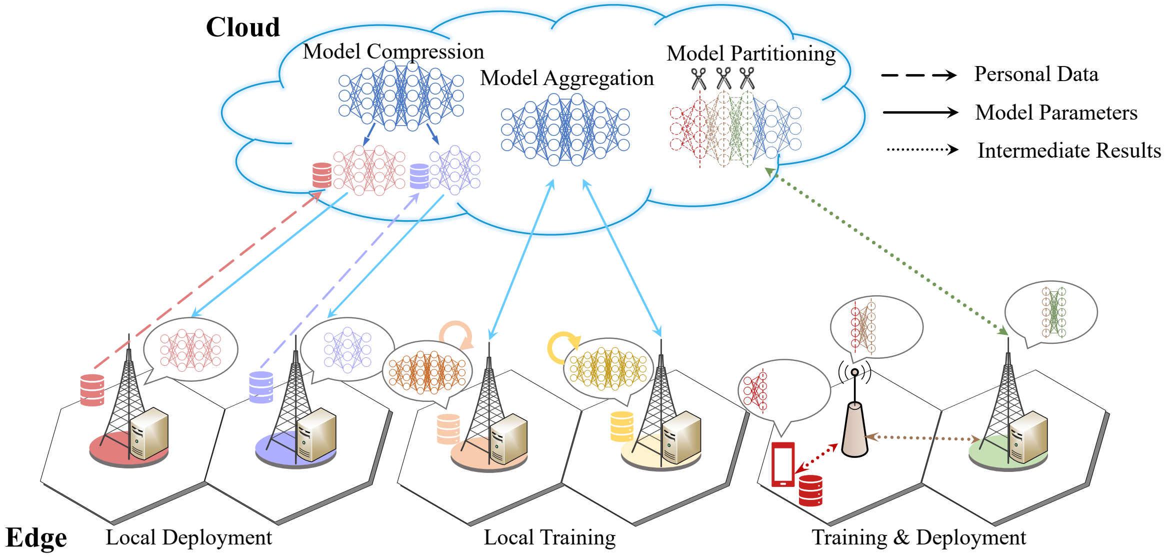


Fig. 1. Three frameworks for training and deploying AI models with cloud-edge collaboration. Data/model distribution method involves sharing data or models, often employing model compression techniques like KD. Model aggregation merges edge-trained models in an iterative process within the cloud. Model partitioning involves the joint training and deployment of models by splitting them across different nodes.

insights and resulting in more reliable decision-making.

3) Massive Cloud Overhead for Centralized Learning: Centralized BAIM training faces increasing demands for data storage, model parameter caching, and computational costs. This growing demand puts a strain on the capacity of central servers, especially when frequent interactions with users amplify the communication load. Simultaneously, the edge network offers substantial computational resources for model training. By shifting focus from a centralized to a distributed manner, BAIM training harnesses the computational resources provided by the edge network, resulting in a more environmentally friendly and cost-effective solution.

Moreover, utilizing edge services for the distributed deployment of GenAI models is another research focus. Edge services, in this context, refers to technologies allowing data processing to be done closer to the source rather than relying centrally on cloud servers. These are pivotal as they significantly reduce latency and bandwidth usage, enhancing system efficiency and user experience. In this way, the system could deliver secure, timely, and personalized services.

1) Data Security: Many GenAI services, such as autonomous driving and remote health care, require the collection of real user data. Centralized cloud computing requires users to upload all data to the cloud, raising privacy concerns. Deploying native GenAI close to or directly at the data source enables storing data on local servers or user devices, alleviating the need to share sensitive data.

2) Timeliness of Response: In contrast to discriminative AI, GenAI generates a great amount of data in response to user requests. Cloud services, relying on long-distance transmission, may suffer from significant latency when delivering these data to users. Native GenAI, with efficient short-range communications (e.g., wireless local area network), can enable high throughput and low-latency tasks.

3) Personalized Services: In response to user requests, edge servers can download a lightweight version of GenAI from the cloud with the necessary functionality, which can be further fine-tuned locally. Additionally, by grouping users with similar service requirements, the edge can maintain multiple models dedicated to efficiently handling various tasks and applications.

To enhance the QoE and QoS for users in 6G networks, it is crucial to simultaneously leverage BAIMs [3] and edge services [4]. This paper proposes a collaborative scheme that integrates native GenAI with cloud-based BAIM, offering a potential solution. Specifically, we analyze current AI training and deployment strategies in edge-cloud collaboration, demonstrating their limitations, and then summarize the challenges that restrict the distributed training of BAIMs and the deployment of native GenAI. In this context, we propose a bottom-up BAIM architecture, along with a distributed training framework and a task-oriented deployment solution. We illustrate the framework's impact through an image generation case study and outline future research directions for maximizing native GenAI and BAIM collaboration.

II. Overview of Model Training and Deployment with Edge-Cloud Collaboration

In this section, we provide an overview of AI model training and deployment frameworks with edge-cloud collaboration, as explored in the 3rd Generation Partnership Project (3GPP) SA1 Release 18 [5]. As illustrated in Fig. 1, these distributed AI frameworks include data/model distribution and sharing, typically employing model minimalism and model compression techniques like knowledge distillation (KD), model aggregation (e.g., federated learning, FL), and model partitioning (e.g., split learning, SL). We compare these frameworks with our proposed bottom-up BAIM architecture in Table I, highlighting existing limitations and summarizing challenges that hinder the distributed training and deployment of BAIMs.

TABLE I. Distributed Model Training and Deployment Frameworks

(a) Model Configuration and Deployment Phase

Distributed Paradigms	Basic Structure	Model Size		Deployed Model	Transmission Content in Deployment Phase	Inference Latency and Cost	
		Cloud	Edge			Comput.	Commun.
Knowledge Distillation	Teacher-Student	Big	Adaptive	Independent and Personalized	-	Low	-
Federated Learning	Top-down	Big	Big	Independent	-	High	-
Split Learning	Multi-Partition	Adaptive	Adaptive	Dependent and Cooperative	Intermediate Results	Medium	High
Ours	Bottom-up	Big	Adaptive	Independent and Personalized	-	Low	-

(b) Training Phase

Distributed Paradigms	Data		Training Method		Transmission Content in Training Phase	Training Latency and Cost	
	Cloud	Edge	Cloud	Edge		Comput.	Commun.
Knowledge Distillation	Personal Data	Personal Data	From Scratch	With Knowledge Logits	Personal Data	High	High
Federated Learning	-	Personal Data	Averaging	From Scratch	Model Parameters	High	High
Split Learning	Labels / -	Personal Data	From Scratch	From Scratch	Intermediate Results	Medium	High
Ours	Common Data	Personal Data	Fine-tuning	From Scratch	Model Parameters	Low	Low

A. Data/Model Distribution and Sharing

This framework enables model training using raw training data shared from edge nodes to the cloud, followed by distribution of well-trained models for inference tasks. To manage computational and communication limitations on edge nodes during inference, it employs model efficiency techniques like model compression and minimalism. Model minimalism designs and trains simpler models to enhance efficiency, while compression reduces post-training model size to boost runtime speed. Model compression involves various mature techniques, such as pruning, quantization, low-rank approximation, and KD. Specific challenges in compressing GenAI models include preserving generative distribution and diversity. Taking KD as an example, layer-wise KD compresses the teacher model into a student by mimicking the hidden representations at each intermediate layer but may lead to misfit due to smaller capacity. The task-aware distillation method proposed in [6] integrates task-aware filters to align hidden representations between student and teacher models, ensuring the preservation of generative distribution. These filters selectively extract relevant knowledge for the target task, thereby preserving diversity in the student model.

While model compression significantly improves deployment efficiency in resource-constrained environments, it also introduces several issues, including information loss, poor generalizability, and increased training costs. The training phase introduces a greater burden on the central server and fail to address the problem of distributed data sources.

B. Model Aggregation

Model aggregation is a crucial mechanism for integrating information from edge models into a global model. In the FL framework, the cloud server initializes a model and distributes it to each edge node. Each edge node then conducts model training using its local data. Subsequently, the model parameters are collected and aggregated on the cloud server to formulate a global model. It ensures data privacy and security by not sharing raw data among nodes. However, FL often involves multiple rounds of model parameter exchange, which can incur prohibitive communication and computational overhead for large-scale GenAI models. Therefore, FL has evolved to prioritize fine-tuning techniques such as parameter efficient fine-tuning (PEFT), prompt tuning (PT), and instruction tuning (IT) to better accommodate the demands of large-scale models.

PEFT operates through two stages: model pre-training and model sparsification, aiming to minimize model size while preserving performance, thereby significantly reducing communication overhead. For instance, as a federated transformer fine-tuning framework, FedPEFT freezes model weights and adjusts systematic errors for downstream tasks. PT involves fine-tuning soft prompts without altering pre-trained BAIMs. FedPrompt efficiently communicates and aggregates federally generated prompts, significantly reducing costs and improving global model performance. IT trains the model using pairs of input-output instructions, enhancing its understanding and application of instructions in various scenarios. In an FL application, FedIT explores IT on LLaMA-7B, demonstrating its efficacy.

While these advancements address the communication and computational challenges of FL, their application is limited by the necessity for each edge node to maintain a large-scale model, placing substantial demands on edge node resources and thereby hindering widespread adoption [7].

C. Model Partitioning

Model partitioning, often referred to as SL, is another method for distributing computational tasks of models. To accommodate large-scale GenAI models, model partitioning framework could adapt network management, for efficient edge model caching, training, and inference. In an SL system, the model's structure and parameters are divided into multiple partitions and computed by different nodes within a communication network. This helps balance the computational load across multiple nodes. Moreover, SL does not require users to share their original data, instead, they exchange intermediate results or labels. This approach ensures privacy protection and reduces the communication bandwidth required for exchanging original data [8]. Generally, SL can be employed both during the model deployment phase and in the model training process.

In the model deployment phase, it is crucial to establish topological partitioning of the well-trained model, considering the following aspects. *1) Edge-cloud node information:* This involves considering each node's communication, computational, and storage capabilities. Such information is crucial

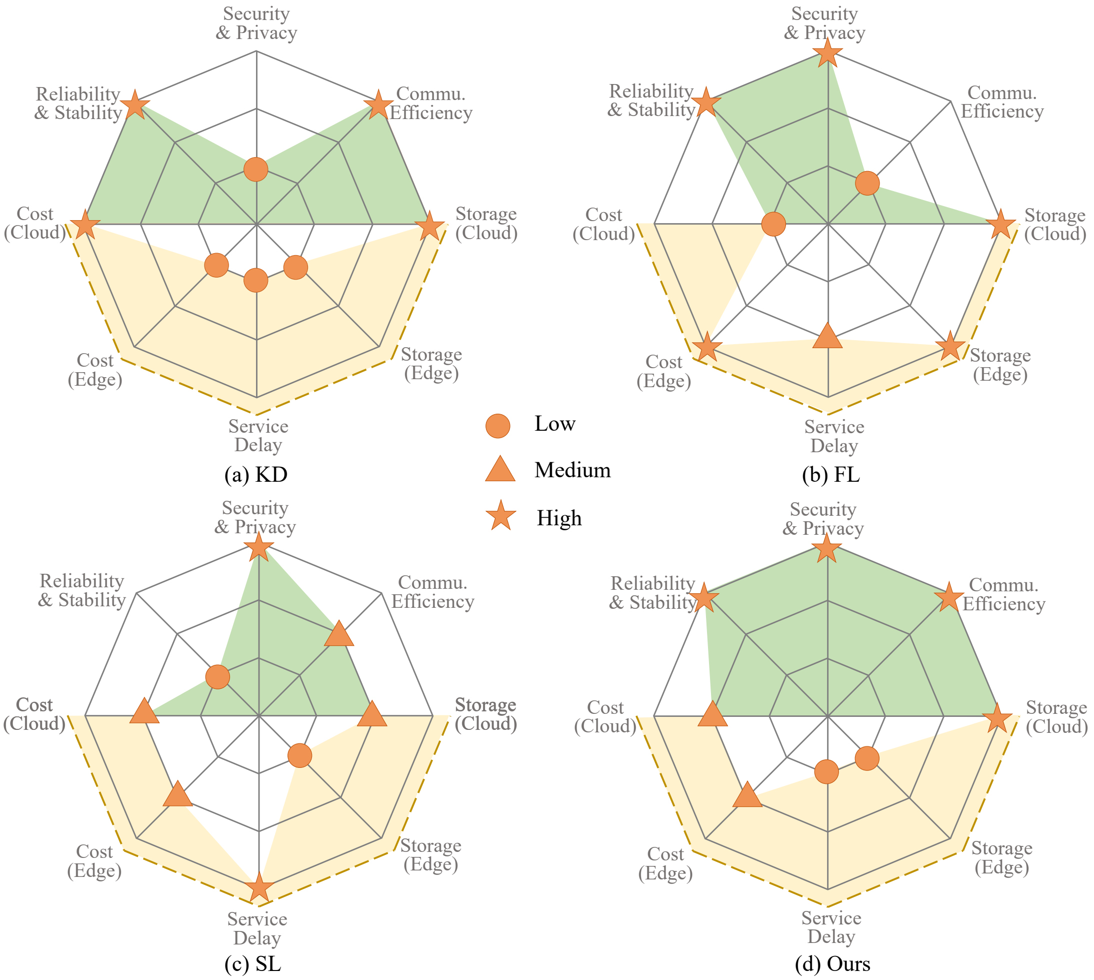


Fig. 2. KPI radar chart of four distributed frameworks over edge-cloud networks.

for determining the optimal distribution and execution of the model across the network. *2) The size of each layer's output:* Checking the size of the data generated by each layer is necessary. This helps determine the amount of data that needs to be sent when splitting the model at a specific layer for a given input. *3) Trade-off between communication and computational cost:* To reduce communication cost by splitting at layers with smaller outputs, more computation on less powerful devices is often needed. Therefore, achieving a compromise through the loss function is crucial for getting a suitable partitioning solution. Previous work employed a joint model split and neural architecture search method to determine the partitioned model [9]. This ensures optimal task performance and guaranteed latency within a given communication network.

Engaging in SL during model training and designing communication strategies for data transmission can mitigate the decline in model performance caused by imperfect communication. By combining the over-the-air computation framework with SL and leveraging the reciprocity of wireless channels, data transmission can be integrated seamlessly into the computation process between model layers. This integration helps reduce the resource expenditure during transmission [10]. Additionally, combining FL with SL leverages data from various edge nodes. Through cloud-edge collaboration, this approach effectively reduces the computational load of edge nodes, enhancing the overall efficiency of the network [11].

D. The Key Performance Indicators (KPIs)

Service provision to users within the edge-cloud collaborative framework requires careful consideration of various KPIs. Fig. 2 displays six KPIs, including service delay, cost, storage, reliability & stability, security & privacy, and communication efficiency. Additionally, in a distributed architecture, cost and storage are considered separately for edge and cloud. There is a trade-off between edge and cloud, so the cost and storage on the cloud are placed on the midline of the radar chart, contributing to the evaluation of both system characteristics (green area) and overhead savings (yellow area).

As shown in Fig. 2(a), since KD requires users to upload data to the center for model training, it damages the security of the system. Fig. 2(b) shows that FL brings huge overhead to the edge due to edge model training, as well as communication inefficiency caused by multiple model transmissions. Fig. 2(c) shows that SL causes high service delays due to the transmission of intermediate results and unreliability caused by relying on node connectivity. Fig. 2(d) is the solution we proposed, which has good performance in various KPIs.

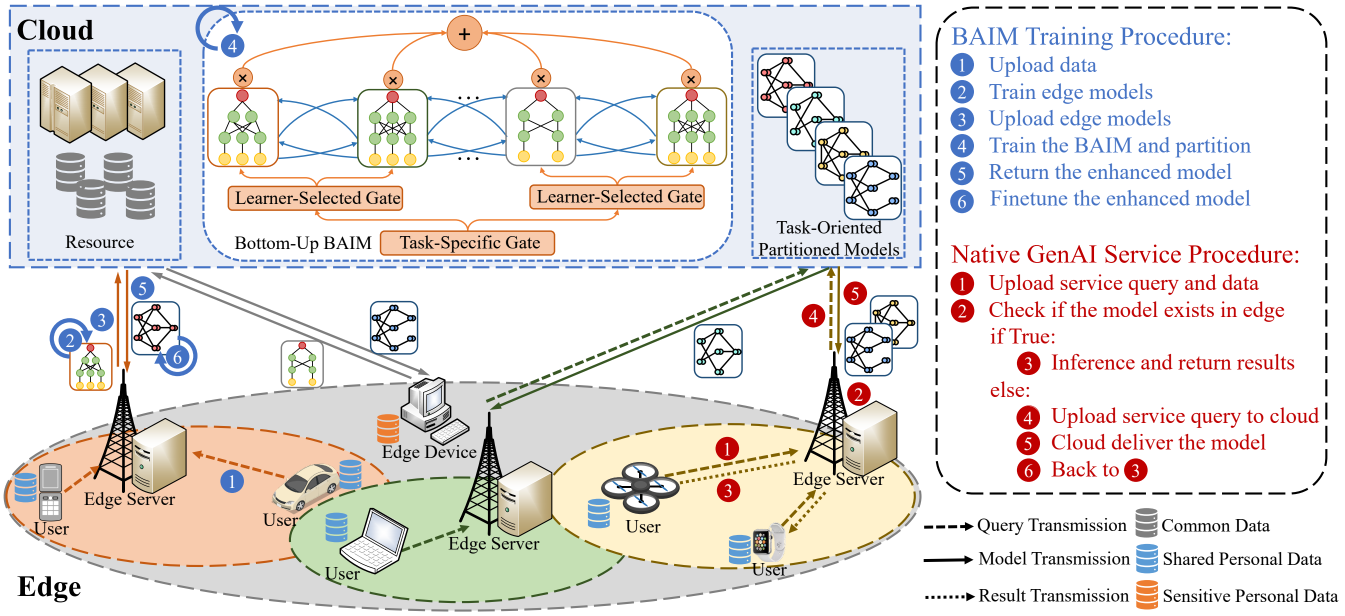


Fig. 3. The workflow of our proposed framework with BAIM training and native GenAI service procedures.

III. A Bottom-Up BAIM Architecture: Distributed Training and Task-Oriented Deployment

Delivering services in 6G networks requires a comprehensive bottom-up architecture due to node diversity and the complexities of multi-task services. Given variations across nodes, it’s more effective for nodes to autonomously determine model architecture for local training. Training multiple single-task edge models and integrating them in the cloud better meets diverse user demands compared to a top-down approach. The synergetic big cloud model and small edge models ensure that both single-task and multi-task scenarios are efficiently managed. The edge models can be fine-tuned for specific tasks based on local data, providing the benefits of single-task specialization, while the cloud-based BAIM ensures comprehensive learning and knowledge transfer across multiple tasks and edge nodes, embodying the essence of multi-task learning and pre-training.

In this section, we introduce the bottom-up BAIM architecture, which leverages edge-cloud collaboration for distributed training and task-oriented deployment. We first outline the workflow of the framework, encompassing the training process of the BAIM and the lifecycle process of native GenAI services. Subsequently, we describe the architecture, emphasizing its intricate design that enables distributed pre-training and a naturally partitioned deployment scheme. Then we explore its training process in the cloud, which is crucial for generalization with few training data. Finally, we present a deployment strategy based on the task-specific partitioning that empowers native GenAI to dynamically deploy the BAIM on edge nodes. This allows users to obtain performance enhancements of the BAIM and the improved QoE provided by edge services.

A. Workflow of the Framework

We depict the workflow of the proposed framework in Fig. 3, including the training process and service provision.

1) BAIM Training Process: Firstly, users upload shared personal data to the edge, constructing local datasets. Users with privacy-sensitive data act as edge devices, maintaining sensitive personal datasets. Secondly, edge nodes, considering their capabilities and user scale, initialize generative models for respective tasks and train them based on local datasets. Due to distinctive edge characteristics, the trained models may exhibit heterogeneous architectures and features. Next, edge nodes upload the trained models, enabling the cloud to obtain multiple edge models for multi-task and multi-modal learning. The cloud orchestrates edge models with gating neural networks and establishes linear projection connections between stages of different edge models, thereby constructing the bottom-up BAIM architecture. Then, the entire BAIM undergoes training based on the cloud common dataset, achieving superior performance across multiple tasks. Subsequently, the BAIM is easily partitioned based on tasks, yielding compact task-specific models to the edge. Finally, edge nodes can perform personalized fine-tuning on the returned lightweight models using their local datasets.

2) Native GenAI service lifecycle: Firstly, users submit queries to the edge based on their needs, uploading the required service data. Then, the edge checks its local toolbox for requested models. If found, it directly performs inference on user data and returns the results. Otherwise, it requests and downloads the corresponding model from the cloud. If the user service involves sensitive personal data, users can directly acquire the corresponding task model from the cloud.

B. Architecture of the Bottom-Up BAIM

In the communication systems, the centralized architecture of BAIM poses limitations on acquiring high-quality user data. Inspired by Pathways [12] and mixture of experts (MoE) [13], we propose a bottom-up BAIM Architecture. This architecture maximizes the utilization of user data and expert

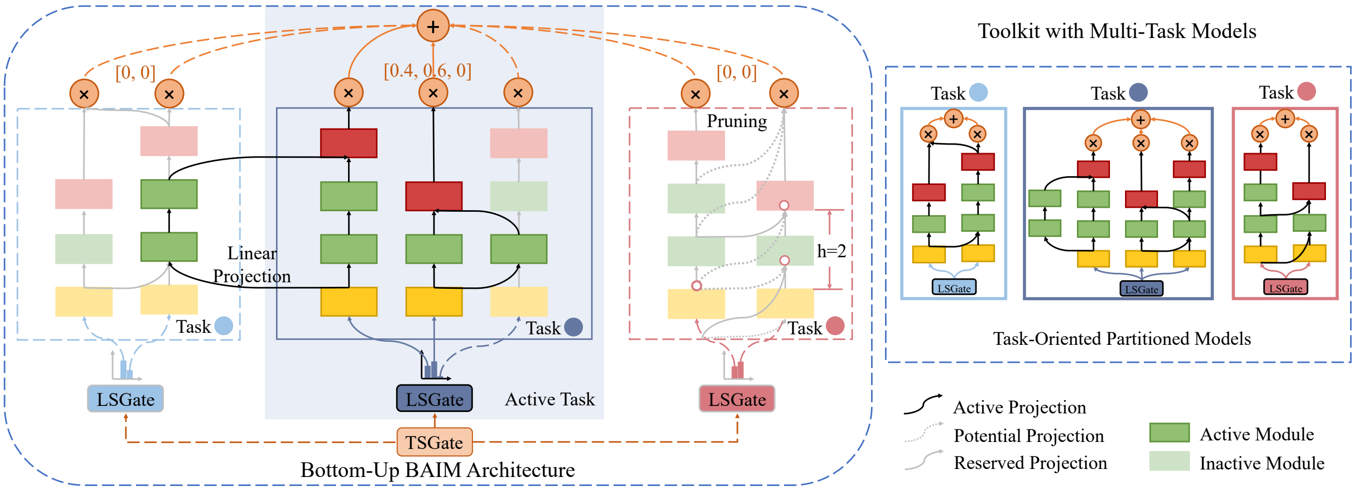


Fig. 4. The bottom-up BAIM architecture and task-oriented partitioned models in the toolkit, involving three tasks. The second task is currently selected by TSGate. Dark modules are executed, including top-k ($k = 2$) learners chosen by LSGate and modules with linear projection connections to these learners while light modules are inactive in the current round. In the third task, gray dashed lines denote the initial potential linear projection (connection height $h = 2$) originating from the first learner. During training, pruning filters and reserves a proportion of them, depicted as gray solid lines.

knowledge extracted by edge models. Pathways, introduced by Google Mind, represents a next-generation network architecture characterized by multi-tasking, multi-modality, and sparse activation. It is believed that a unified model should be able to expedite learning using existing skills by activating corresponding modules. MoE, purportedly the fundamental structure of GPT-4, combines multiple experts using gating neural networks, enabling adaptive expert output combination. This design harnesses knowledge from diverse experts while mitigating computational demands through sparse gating. We employ edge models as MoE experts, modularizing the models by establishing linear connections between them. This forms a multi-task, multi-modal, and sparsely activated hierarchical BAIM architecture, as illustrated in Fig. 4.

1) Multi-Tasking and Gating Network: The hierarchical gating network (HierGate) comprises M learner-selected gates (LSGates) and a task-specific gate (TSGate), allowing it to organize multiple tasks and handle multimodal inputs. The cloud categorizes N heterogeneous models from edge nodes into task-specific groups, forming M learner squads. Within each group, experts are arranged in parallel, combined through an LSGate. The TSGate is employed to govern the execution of various tasks by routing inputs to the corresponding task. LSGate selects the top K learners most suited to the input, assigning them individual weights. The value of K determines the number of activated learners for a specific task, thereby influencing the computational cost. After selecting a task, learners from other tasks do not need to activate the entire model. HierGate achieves efficient structural sparsity, producing an N-dimensional vector segmented into M sections, with only K dimensions being non-zero. This represents the proportion of outputs from N learners for a specific task, effectively utilizing diverse knowledge from different learners within the same task.

2) Modularization and Linear Projection: Different from typical MoE models that connect learners solely through gating networks, we suggest organizing learners into modules and establishing linear projection connections among them to facilitate knowledge sharing. Since learners in the same squad are likely to benefit from each other's expertise, and there could be information associations among learners from different tasks, we create linear connections among N learners following specific rules. Our rule can be explained as follows: Take the features produced by a learner at stage i, perform a linear projection to convert them to the input dimension of other learners at stage j (where $0 \leq j - i \leq h$), and add the result to the original input of stage j. The hyper-parameter $h \geq 0$ controls the initial connection density, based on the assumption that layers close in depth process original input to a similar extent. Importantly, connections are established from shallow to deep layers, avoiding the formation of cyclic model structures. Moreover, the dependency among learners can vary significantly. Certain tasks exhibit clear and strong relationships, benefiting from shared features, while others show weaker relations, with shared features less evident. To address this, during model training, we employ pruning to iteratively filter and preserve essential connections. This method reduces the model's parameter size while fostering stable feature-sharing relationships within the model, facilitating adaptive knowledge propagation across tasks.

C. Training BAIM in the Cloud

For the above model architecture, trainable parameters include gate neural networks, linear connections for feature projection, and each individual learner. Here we introduce three training strategies for BAIM:

1) Fine-tuning Strategy: All trainable parameters undergo updates. The uploaded edge models serve as initialization for fine-tuning. This comprehensive adjustment ensures the entire model converges to an optimized configuration, leveraging the knowledge embedded in the locally trained edge models.

2) Freezing Strategy: Keep the uploaded edge models unchanged, only update the connections between models and the gate network. This maintains the individuality of each learner throughout the training process, serving as static contributors to the overall model.

3) Scratch Strategy: Train all parameters starting from random initialization. This approach allows for a thorough evolution of the entire model architecture, emphasizing the independence of the unified BAIM from pre-existing knowledge encapsulated in the uploaded edge models.

The choice of a training strategy depends on various factors, including specific goals, available resources, and data characteristics. Fine-tuning strategy enhances existing models, freezing strategy preserves valuable learnings, and scratch strategy creates a unified model from scratch. In practice, a combination of strategies may be preferred at different stages, adapting to the evolving project requirements to optimize resource usage while maximizing BAIM performance.

In addition, it is crucial to consider the self-maintenance of th framework that needs continuous evolution and adaptation to changing conditions. This involves the following three approaches:

1) Continual Learning: Continual learning refers to the model acquiring new capabilities without forgetting original tasks. Unlike multitask learning [14], where all tasks are learned simultaneously, continual learning involves a gradual increase in tasks. In communication networks, as the number of users served by edge nodes grows, edge nodes continuously upload models to participate in the aggregation of the unified model. Our bottom-up BAIM is a scalable architecture that can fine-tune the gate to learn new tasks gradually.

2) Model-Level Pruning: Model-level pruning involves trimming sub-models from a large model. In the scalable BAIM, as the number of sub-models increases, the quantity of learners for each task grows continuously. Simultaneously, some poorly performing learners are rarely or almost never activated by LSGate. For these learners, in model-level pruning, the parts that do not assist other learners can be removed, and the modules that assist other learners are directly merged into the corresponding learners. This ensures the storage efficiency and computational effectiveness of BAIM.

3) Few-Shot Learning: Few-shot learning refers to training models to generate qualified context or make accurate predictions when provided with very limited examples [15]. For BAIM, the cloud data is typically generic public data, and for different tasks, there may be only a few shot samples or even zero shots. BAIM needs to adjust model parameters with limited samples to ensure good performance across various tasks. This requires thoroughly exploring the correlations between different tasks and leveraging the knowledge shared among learners.

D. Native GenAI with Task-Oriented Deployment

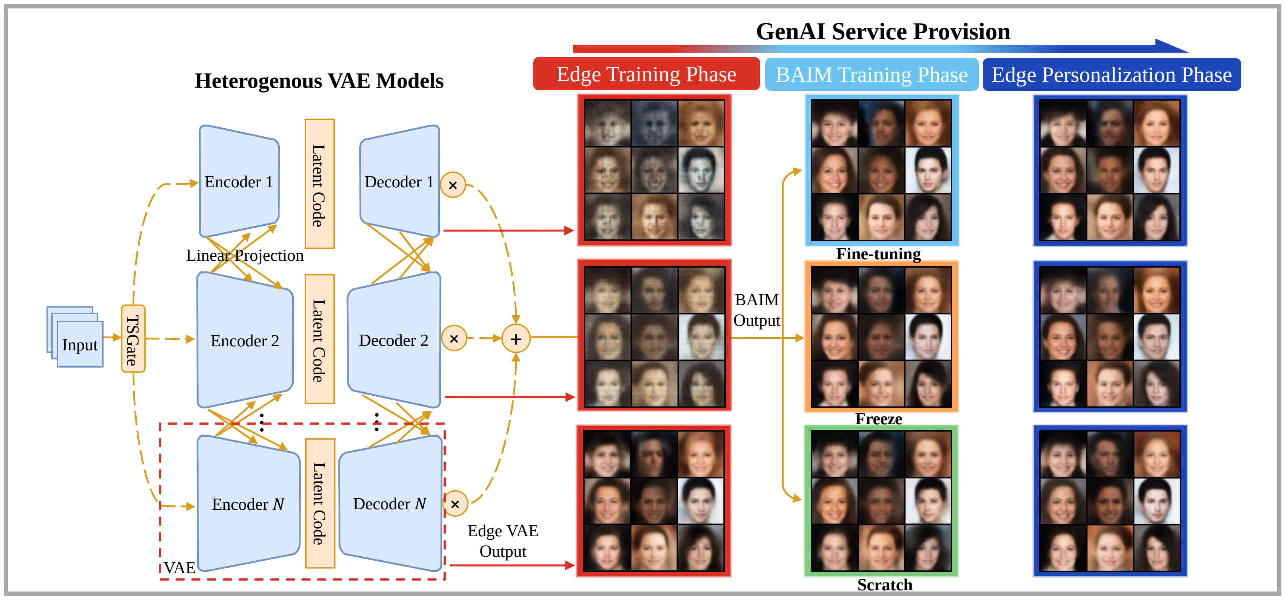
In addition to the unified multi-task BAIM obtained on the central server, the framework can also implement model compression and model partitioning to obtain enhanced performance for different tasks. These compact and lightweight models can be deployed to edge nodes, providing users with native GenAI services. Model compression typically results in a loss of model performance or requires additional fine-tuning. However, our proposed architecture has unique properties that allow the decomposition of the model into compact models for corresponding tasks based on TSGate without sacrificing performance. Linear connections are replicated in new models, as shown in Fig. 4. The resulting structure for each task is a MoE architecture, each having its own LSGate for selecting learners conditioned on the input, demonstrating excellent generalization capabilities for the task. As a result, M compact models can be obtained as needed, meeting user service requirements. This approach enables edge services to achieve performance comparable to the cloud while harnessing the advantages of edge services.

IV. A Case Study: Image Generation Service Provision

In this section, we demonstrate a typical image generation service with variational autoencoder (VAE) models with our framework. Our approach is compared with FL, with all training and evaluation procedures conducted on a standardized dataset derived from CelebA. In FL, for fairness, homogenous models are trained at both edge nodes and the cloud. Each of the 10 edge nodes holds local data, while the cloud possesses another part of dataset. After each epoch, edge nodes upload models to the cloud for aggregation. The cloud then trains the aggregated model for one epoch, repeating this cycle for 100 times. In contrast, our approach entails independently training heterogeneous models at the edge for 100 epochs. These models are then uploaded to the cloud for integration into BAIM, followed by an additional 100 epochs training. Thus, both frameworks undergo an equal total number of training epochs. Finally, the well-trained BAIM is distributed to the edge nodes.

Fig. 5 (b) presents a comparison of the three training approaches of our proposed BAIM architecture with FL, including the sizes of trainable parameters, number of training rounds, training FLOPs per epoch, for both cloud and edge nodes, as well as communication volumes for uploads and downloads (involving model size and communication rounds). This comparison highlights that our approach exhibits lower communication overhead. Moreover, although we have higher computational FLOPs at the cloud, the costs at the edge nodes are greatly reduced, which shows the effect of allocating the workload properly in edge-cloud collaboration.

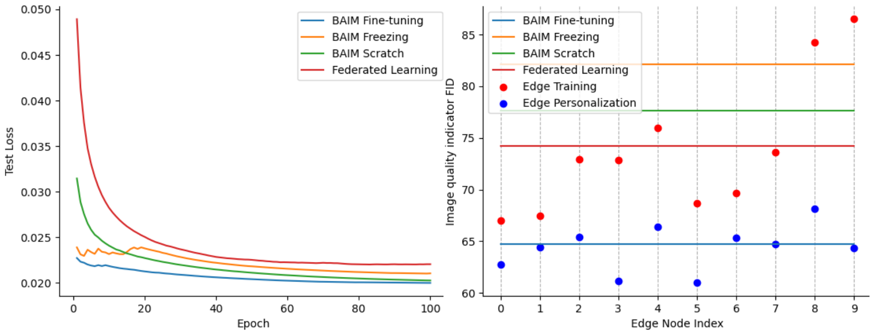
The testing samples of three phases in our approach are illustrated in the three columns of images in Fig. 5 (a), respectively. Fig. 5 (c) shows the convergence of the testing loss under the three training strategies for BAIM and FL. The fine-tuning strategy exhibits the best convergence and performance, while the freeze strategy initially oscillates for the first several epochs before converging, and FL performs the worst. Due to random initialization, the scratch strategy starts with a high initial loss, experiences a rapid decrease, and reaches the middle loss level. This is attributed to the lack of knowledge extracted by the edge model, preventing it



(a) Visualization of the process and results of small edge models constituting the cloud BAIM

		Ours Fine-tuning	Ours Freezing	Ours Scratch	Federated Learning
Trainable Parameters	Cloud	42.1 M	13.9 M	42.1 M	13 M
	Each Edge	(0.4 ~ 13) M			13 M
Commu. Volume (Model Size×Times)	Upload	(1.6 ~ 52) × 1 MB			52 × 100 MB
	Down load	168.4 × 1 MB			52 × 100 MB
Training Rounds	Cloud	100 epochs			100 epochs
	Each Edge	100 epochs			100 epochs
Comp. FLOPs (per epoch)	Cloud	627 G	207 G	627 G	92.5 G
	Each Edge	6.7 G ~ 92.5 G			92.5 G

(b) Comm. and Comp. cost comparison



(c) Convergence of cloud BAIM and FL

(d) Generated image quality

Fig. 5. The case study on image generation service provision.

from achieving the performance of fine-tuning despite having the same model size.

Fréchet Inception Distance (FID) is a widely used metric for evaluating the quality of generated images. It measures the similarity between the distribution of real images and generated images. A lower FID implies that the generated images closely match the real ones. Fig. 5 (d) displays the FID of images produced by models applying three distinct BAIM training strategies and FL (represented by four horizontal lines). Additionally, it presents the FID of images generated at 10 edge nodes initially trained and subsequently personalized after BAIM deployment (denoted by two types of scatter points). Notably, the fine-tuning strategy proves effective, significantly improving image quality compared to the original edge model.

V. Challenges and Potential Research Opportunities

Our proposed framework enables efficient service provision in 6G communication networks. However, it introduces challenges such as data management, model fusion scheme design, and node management, which demand attention.

A. Data Management

We address data privacy concerns by distinguishing between sensitive personal data, shared personal data, and common data. Moving forward, a comprehensive solution for data management and generation is yet to be developed.

1) Secure Data Management Scheme: Vigorous safeguards should be established to protect data during storage and transmission, including end-to-end data encryption and enhanced identity verification mechanisms. Additionally, desensitization techniques for cloud-side common data to minimize the risk of information leakage are also expected.

2) Substituting User Raw Data with Synthetic Data: This involves the application of differential privacy techniques, generative adversarial networks (GANs), and data perturbation methods to generate synthetic data with authentic data features. In light of the advancements in artificial intelligence generated content (AIGC), synthetic data could serve as a more secure and reliable alternative for training data.

B. Model Fusion Scheme

During the model training phase, designing more promising model fusion strategies and asynchronous update mechanisms

This article has been accepted for publication in IEEE Network. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/MNET.2024.3420755

could lead to improvements in reliability and efficiency.

1) Optimizing Heterogeneous Architecture Fusion Strategies: Edge models vary in structure concerning depth and width, as well as in architectures. For efficient multi-task learning with these diverse models, it's crucial to improve heterogeneous architecture fusion strategies, including projection methods, connecting rules and pruning strategies.

2) Designing Asynchronous Update Mechanisms: Asynchronous update systems allow immediate uploads from each edge node upon calculation completion, reducing waiting times. With independent edge training, the cloud continually receives and integrates these models into the BAIM. A well-designed asynchronous update mechanism is essential to balance the BAIM's staleness with computational cost.

C. Node Management

Node management involves the flexible monitoring, adjustment, and coordination of changes. Effective management enhances system stability and reliability, reducing performance declines caused by node anomalies.

1) Adapting Dynamic Edge Networks: Edge networks in practical systems are dynamic, leading to possible instability in edge nodes, including frequent access and disconnection. Systems must adapt to adding new nodes, handling failures, and responding to changes in node states. Given that edge nodes are often distributed and mobile, the system must effectively address issues like network delays, data loss, and node availability changes.

2) Addressing Security Threats: In open cloud-edge systems, malicious node attacks are inevitable. These include dishonest nodes sabotaging model performance with false training results or disrupting operations through denial-of-service (DoS) attacks. Security measures should involve tamper and anomaly detection, trust assessment, and cross-verification of node updates against historical behavior or peer updates.

VI. Conclusion

The synergies between edge-native GenAI and cloud-based BAIMs emerge as a crucial component in 6G communication networks, promising elevated QoE and QoS. The presented framework focus on mitigating challenges associated with BAIMs, and showcases its effectiveness in an image generation task. Furthermore, comprehensive research directions are delineated to unlock the full spectrum.

References

[1] M. Xu, H. Du, D. Niyato, *et al.*, "Unleashing the power of edge-cloud generative AI in mobile networks: A survey of AIGC services," *arXiv preprint arXiv:2303.16129*, 2023.

[2] X. Chen, Z. Guo, X. Wang, *et al.*, "Foundation model based native ai framework in 6g with cloud-edge-end collaboration," *arXiv preprint arXiv:2310.17471*, 2023.

[3] Z. Chen, Z. Zhang, and Z. Yang, "Big AI models for 6G wireless networks: Opportunities, challenges, and research directions," *arXiv preprint arXiv:2308.06250*, 2023.

[4] Y. Xiao, G. Shi, Y. Li, *et al.*, "Toward self-learning edge intelligence in 6G," *IEEE Communications Magazine*, vol. 58, no. 12, pp. 34–40, 2020. DOI: 10.1109/MCOM.001.2000388.

[5] X. Lin, "Artificial intelligence in 3GPP 5G-Advanced: A survey," *arXiv preprint arXiv:2305.05092*, 2023.

[6] C. Liang, S. Zuo, Q. Zhang, *et al.*, "Less is more: Task-aware layer-wise distillation for language model compression," in *International Conference on Machine Learning*, PMLR, 2023, pp. 20 852–20 867.

[7] M. Chen, D. Gündüz, K. Huang, *et al.*, "Distributed learning in wireless networks: Recent progress and future challenges," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3579–3605, 2021. DOI: 10.1109/JSAC.2021.3118346.

[8] W. Xu, Z. Yang, D. W. K. Ng, *et al.*, "Edge learning for B5G networks with distributed signal processing: Semantic communication, edge computing, and wireless sensing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 17, no. 1, pp. 9–39, 2023. DOI: 10.1109/JSTSP.2023.3239189.

[9] Y. Tian, Z. Zhang, Z. Yang, *et al.*, "JMSNAS: Joint model split and neural architecture search for learning over mobile edge networks," in *2022 IEEE International Conference on Communications Workshops*, IEEE, 2022, pp. 103–108.

[10] Y. Yang, Z. Zhang, Y. Tian, *et al.*, "Over-the-Air split machine learning in wireless MIMO networks," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 4, pp. 1007–1022, 2023. DOI: 10.1109/JSAC.2023.3242701.

[11] J. Li, L. Lyu, D. Iso, *et al.*, "MocoSFL: Enabling cross-client collaborative self-supervised learning," in *The Eleventh International Conference on Learning Representations*, 2022.

[12] Google, "Introducing Pathways: A next-generation AI architecture," https://blog.google/technology/ai/introducing-pathways-next-generation-ai-architecture/.

[13] B. Mustafa, C. Riquelme, J. Puigcerver, *et al.*, "Multimodal contrastive learning with LiMoE: The language-image mixture of experts," *Advances in Neural Information Processing Systems*, vol. 35, pp. 9564–9576, 2022.

[14] Z. Chen, Y. Shen, M. Ding, *et al.*, "Mod-squad: Designing mixtures of experts as modular multi-task learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 11 828–11 837.

[15] Z. Lin, S. Yu, Z. Kuang, *et al.*, "Multimodality helps unimodality: Cross-modal few-shot learning with multimodal models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 19 325–19 337.

Yuqing Tian [StM] (tianyq@zju.edu.cn) received her B.S.Eng. degree in information engineering from Zhejiang University, Hangzhou, China, in 2020. She is currently pursuing her Ph.D. in information and communication engineering with Zhejiang University under the supervision of Prof. Zhaoyang Zhang. Her current research interests include machine learning for wireless networks, distributed algorithms, and neural architecture search.

Zhaoyang Zhang [SM] (ning_ming@zju.edu.cn) received the Ph.D. degree from Zhejiang University, Hangzhou, China, in 1998. He is currently a Qiushi Distinguished Professor with Zhejiang University. He has authored more than 400 international journals and conference papers, and is the co-recipient of IEEE Leonard G. Abraham Prize 2024 and IEEE ICC 2019 and IEEE GLOBECOM 2020 Best Paper Awards. His current research interests are mainly focused on AI-empowered communications and networking, integrated sensing, computing and communication, etc. He was awarded the National Natural Science Fund for Distinguished Young Scholar by NSFC in 2017. He is the Chair of Wireless AI Task Group of China IMT-2030 (6G) Promotion Group and the Vice Chair of IEEE ComSoc Nanjing Chapter. He served as Editor for journals like IEEE TWC and TCOM, etc., and as General Chair, TPC Co-Chair, Symposium Co-Chair or Keynote Speaker for more than 10 international conferences.

Yuzhi Yang [StM] (yuzhi_yang@zju.edu.cn) received his B.S.Eng. degree in information engineering from Zhejiang University, Hangzhou, China, in 2019. He is currently pursuing his Ph.D. in information and communication engineering with Zhejiang University under the supervision of Prof. Zhaoyang Zhang. His current research interests include machine learning for wireless networks, distributed algorithms, and massive MIMO.

Zirui Chen [StM] (ziruichen@zju.edu.cn) received his B.S.Eng. degree in information engineering from Zhejiang University, Hangzhou, China, in 2021, where he is currently pursuing the Ph.D. degree in information and communication engineering under the supervision of Prof. Zhaoyang Zhang. His current research interests include AI-empowered communications and massive MIMO.

Zhaohui Yang [M] (yang_zhaohui@zju.edu.cn) received his Ph.D. degree in communication and information system with the National Mobile Communications Research Laboratory, Southeast University, Nanjing, China, in 2018. He is currently a ZJU Young Professor with the Zhejiang Key Laboratory of Information Processing Communication and Networking, College of Information Science and Electronic Engineering, Zhejiang University. His research interests include joint communication, sensing, and computation, federated learning, and semantic communication.

Richeng Jin [M] (richengjin@zju.edu.cn) received the B.S. degree in information and communication engineering from Zhejiang University, Hangzhou, China, in 2015, and the Ph.D. degree in electrical engineering from North Carolina State University, Raleigh, NC, USA, in 2020. He was a Postdoctoral Researcher in electrical and computer engineering at North Carolina State University, Raleigh, NC, USA, from 2021 to 2022. He is currently a faculty member of the department of information and communication engineering with Zhejiang University, Hangzhou, China. His research interests are in the general area of wireless AI, game theory, and security and privacy in machine learning/artificial intelligence and wireless networks.

Tony Q. S. Quek [F] (tonyquek@sutd.edu.sg) received the B.E. and M.E. degrees in electrical and electronics engineering from the Tokyo Institute of Technology in 1998 and 2000, respectively, and the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology in 2008. Currently, he is a Cheng Tsang Man Chair Professor with the Singapore University of Technology and Design (SUTD) and a ST Engineering Distinguished Professor. He is the Director of the Future Communications Research and Development Programme, the Head of ISTD Pillar, and the Deputy Director of the SUTD-ZJU IDEA. His current research interests include wireless communications and networking, network intelligence, non-terrestrial networks, open radio access networks, and 6G.

Kai-Kit Wong [F] (kai-kit.wong@ucl.ac.uk) received the B.Eng., M.Phil., and Ph.D. degrees in electrical and electronic engineering from The Hong Kong University of Science and Technology, Hong Kong, in 1996, 1998, and 2001, respectively. After graduation, he took up academic and research positions at The University of Hong Kong, Lucent Technologies, BellLabs, Holmdel, the Smart Antennas Research Group of Stanford University, and the University of Hull, U.K. He is currently the Chair of wireless communications with the Department of Electronic and Electrical Engineering, University College London, U.K. His current research interests include centers around 6G and beyond mobile communications. He is a fellow of IET. He served as the Editor-in-Chief for IEEE Wireless Communications Letters from 2020 to 2023.