QoS-Driven Hybrid Inference Scheme for Generative Diffusion Models in MEC-Enabled AI-Generated Content Networks

Xinyi Zhuang, Jiaqi Wu, Hongjia Wu, Ming Tang, and Lin Gao

***Abstract*—AI-Generated Content (AIGC) based on Generative Diffusion Models (GDMs) is revolutionizing content creation and promoting substantial advancements in domains like autonomous driving and robotics. Leveraging progress in Mobile Edge Computing (MEC) and model compression techniques, GDMs are increasingly being deployed on Edge Servers (ESs) and User Equipments (UEs), which typically face resource limitations. In such MEC-enabled scenarios, designing an efficient inference scheme for GDMs still remains a significant challenge, due to the resource constraints on ESs and UEs as well as the personalized demands of AIGC users. In this work, we propose a novel *hybrid inference scheme*, which consists of two stages: public prompt generation and common-to-personalized inference. In the first stage, a Large Language Model (LLM) is adopted to generate public prompts derived from the common features of users' personal prompts. In the second stage, a common inference phase based on public prompts is first executed for all users (to produce common intermediate results), and then a personalized inference phase based on each user's personal prompts is performed for each individual user (to generate final contents). Clearly, by introducing the common inference phase, the total inference steps can be significantly reduced. In such a scheme, we further study a hybrid inference optimization problem to optimize both common and personalized inference steps, aiming to maximize the total Quality of Service (QoS), while minimizing delay and energy consumption. Simulation results show that our proposed scheme significantly outperforms existing benchmarks, with the performance gains ranging from 12.6% to 102.2%.**

I. Introduction

A. Background and Motivations

AI-Generated Content (AIGC) has recently garnered significant attention for its transformative impact on controllable content generation. As the powerful "engine" of AIGC, Generative Diffusion Models (GDMs) have demonstrated exceptional capabilities in synthesizing high-quality images, videos, and a wide array of other content types. These capabilities grant GDMs significant potential for numerous practical applications (e.g., autonomous driving, autonomous agents, robotics, and artistic expression [1]), paving the way for advancements in 6G communications. Technically, GDMs employ a stepwise denoising process to progressively transform simple noise distributions (e.g., Gaussian Distribution) into complex data distributions. This process involves a series of small and reversible *inference steps*. At each step, GDMs estimate and remove noise, gradually guiding the data toward its original high-dimensional structure. Through this iterative refinement, GDMs can generate high-quality and creative data samples that closely match the characteristics of the training data. Leveraging the powerful capabilities of GDMs, many advanced commercial models have been developed and released, including Stable Diffusion[1] and DALL-E[2].

Traditionally, GDMs are often deployed on *cloud servers* to deliver large-scale AIGC services, which demonstrates remarkable capabilities in generating diverse and coherent visual content. However, accessing AIGC services from remote cloud servers poses challenges for users due to server vulnerability and latency issues, rendering them less suitable for real-time applications. To this end, *Mobile Edge Computing* (MEC) has emerged as a crucial enabling technology for AIGC services, by pushing computation workload down to Edge Servers (ESs) and User Equipments (UEs) that approximate to end users [1]. While recent advancements in model compression techniques have enabled GDMs to transition towards both ESs and UEs, a substantial gap still exists in the development of efficient inference schemes for MEC-enabled AIGC networks. This motivates us to explore the following question:

Question 1. *How to design an efficient inference scheme in resource-constrainted MEC-enabled AIGC networks, by considering the new features and capabilities of GDMs?*

One promising approach for Question 1 is to enable GDMs to share part of the inference process by leveraging the similarity in different users' intentions. However, most existing works focused only on the task offloading and resource allocation (e.g., [2]–[4]), often overlooking the impact of the similarities among users' intentions, and instead assuming that each user follows an independent workflow. To exploit the similarity in user intentions, some researchers studied the collaborative inference schemes (e.g., [5], [6]), where the system randomly

This work was supported in part by the National Key Research and Development Program of China (Grant No. 2021YFB2900300), the National Natural Science Foundation of China (Grant No. 61972113), the Natural Science Foundation of Guangdong Province (Grant No. 2024A1515010178), Shenzhen Science and Technology Program (Grant No. KQTD20190929172545139, GXWD20231129103946001, and ZDSYS20210623091808025), and Guangdong Basic and Applied Basic Research Foundation (Grant No. 2021B1515120008).

X. Zhuang, J. Wu, and L. Gao are with the School of Electronics and Information Engineering and the Guangdong Provincial Key Laboratory of Aerospace Communication and Networking Technology, Harbin Institute of Technology, Shenzhen, China. H. Wu is with the Department of Mathematics and Information Technology, The Education University of Hong Kong, N.T., Hong Kong. M. Tang is with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. Email: gaol@hit.edu.cn. *(Corresponding Author: Lin Gao)*

[1] Available at: https://stability.ai/stable-image

[2] Available at: https://openai.com/index/dall-e-3

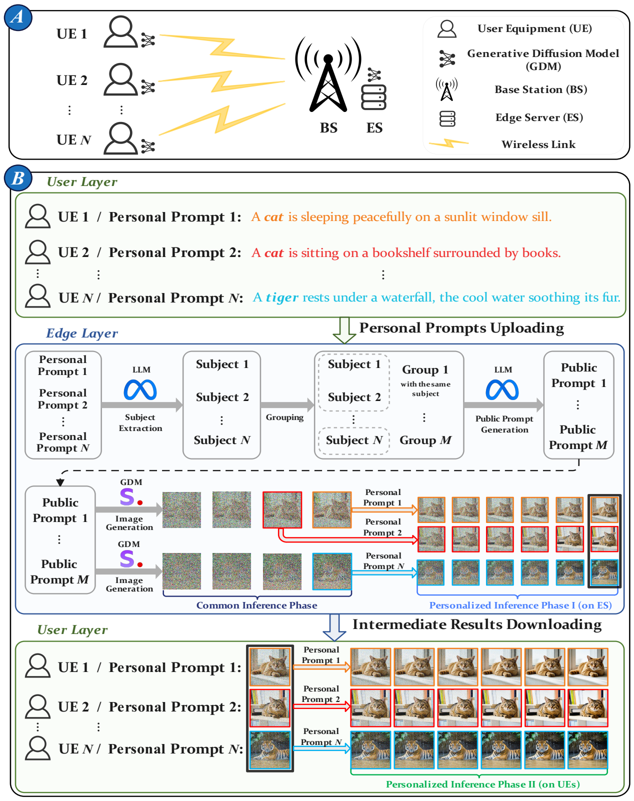

A
UE 1
UE 2
UE N
BS
ES
User Equipment (UE)
Generative Diffusion Model (GDM)
Base Station (BS)
Edge Server (ES)
Wireless Link
B
User Layer
UE 1 / Personal Prompt 1: A cat is sleeping peacefully on a sunlit window sill.
UE 2 / Personal Prompt 2: A cat is sitting on a bookshelf surrounded by books.
UE N / Personal Prompt N: A tiger rests under a waterfall, the cool water soothing its fur.
Personal Prompts Uploading
Edge Layer
Personal Prompt 1
Personal Prompt 2
Personal Prompt N
LLM
Subject Extraction
Subject 1
Subject 2
Subject N
Grouping
Group 1
with the same subject
Group M
Public Prompt Generation
Public Prompt 1
Public Prompt M
GDM
Image Generation
Personal Prompt 1
Personal Prompt 2
Personal Prompt N
Common Inference Phase
Personalized Inference Phase I (on ES)
Intermediate Results Downloading
User Layer
UE 1 / Personal Prompt 1:
UE 2 / Personal Prompt 2:
UE N / Personal Prompt N:
Personalized Inference Phase II (on UEs)

Fig. 1. **Part A**: An illustration of MEC-enabled AIGC networks. **Part B**: Workflows of our proposed hybrid inference scheme.

selects a user to execute the inference process, while other users benefit from the same intermediate results. However, random selection and uniform sharing cannot effectively address the diverse intentions of users in the system.

To overcome these challenges, we conduct extensive real data experiments, which indicate that a limited number of inference steps can be shared among different users. In addition, increasing the number of shared steps tends to introduce excessive biased semantic information, which can compromise the accuracy and integrity of the generated contents. This limitation presents a significant opportunity for improvement, especially in scenarios with numerous users, where efficient inference schemes could greatly enhance the network performance.

B. Solution and Contributions

In this work, we propose an efficient and Quality of Service (QoS)-driven *hybrid inference scheme* for MEC-enabled AIGC networks, which consists of two sequential stages: *public prompt generation* and *common-to-personalized inference*. In the first stage, a Large Language Model (LLM) is adopted to analyze and extract the common features from users' personal prompts. Similar prompts are clustered into groups to generate corresponding public prompts. In the second stage, a common inference phase based on public prompts is first executed for all users to generate a series of common intermediate results. For each user, a suitable result is then selected based on the similarity between the public and personal prompts to proceed with a personalized inference phase. This approach allows users with higher similarity to perform fewer inference steps. As a result, users can maintain acceptable QoS while significantly reducing resource consumption.

To accurately characterize the QoS for users, we define a Hybrid Inference Quality Metric (HIQM), derived from real data experiments, to capture the misalignment between users' intentions and the generated contents. This new metric considers the combined effects of common inference steps and the similarity between public and personal prompts. Based on this new metric, we formulate a hybrid inference optimization problem that seeks to optimize both common and personalized inference steps, with the goal of maximizing overall QoS while minimizing delay and energy consumption. We further develop a Deep Reinforcement Learning (DRL) algorithm based on the Independent Proximal Policy Optimization (IPPO) framework to solve the optimization problem. In summary, the key contributions are as follows:

- ***Novel Hybrid Inference Scheme:*** We propose a novel efficient and QoS-driven hybrid inference scheme consisting of public prompt generation and common-to-personalized inference. By introducing the common inference phase, users can custmoize and simplify the inference process to reduce resource consumption while maintaining acceptable QoS.
- ***New Data-Driven Performance Metric*** We propose a new performance metric called HIQM, derived from real data experiments, which can characterize the combined effects of common inference steps and the similarity between public and personal prompts.
- ***Joint Optimization Problem and Solution:*** We formulate a hybrid inference optimization problem that seeks to optimize both common and personalized inference steps jointly, and develop an IPPO-based DRL algorithm to solve the problem. Simulation results demonstrate that our proposed scheme significantly outperforms existing benchmarks in the literature, with the performance gains ranging from 12.6% to 102.2%.

II. System Model

A. Network Model

As shown in Fig. 1 (Part A), we consider an MEC-enabled AIGC network, which consists of a set $\mathcal{N} \triangleq \{1, \cdots, N\}$ of N UEs (i.e., AIGC users), each with limited computing resources, and one BS equipped with an ES that offers powerful computing resources to support AIGC services. UEs generate personal prompts to request AIGC services for content generation. By leveraging model compression techniques (e.g., distillation, quantization, and pruning [6]), GDMs can be deployed on both UEs and the ES.

As shown in Fig. 1 (Part B), we propose an efficient and QoS-driven hybrid inference scheme in the MEC-enabled AIGC network. Specifically, at the user layer, UEs generate their personal prompts, which are uploaded to the edge layer for further processing. At the edge layer, in the first stage, the LLM agent extracts key subjects from these prompts and clusters similar prompts into groups to generate public

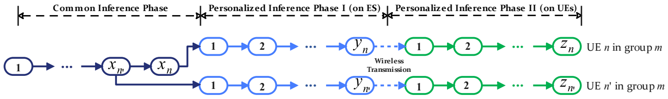


Fig. 2. An illustration of the common-to-personalized inference process, where UE n and n' belong to the same group m.

prompts. In the second stage, the common-to-personalized inference includes three phases:

- *Common Inference Phase:* Each public prompt is processed through the GDM to generate a series of common intermediate results. These results are shared across all UEs associated with the corresponding group.
- *Personalized Inference Phase I (on ES):* A selected common intermediate result (also with a specific personal prompt) is re-injected into the GDM to generate a personalized intermediate result. This result is the refined output for individual UE needs.
- *Personalized Inference Phase II (on UEs):* Each personalized intermediate result is processed locally after wireless transmission to meet personal requirements, outputting the final generated content.

B. Hybrid Inference Scheme

As discussed in Section II-A, the hybrid inference scheme consists of two stages, including public prompt generation and common-to-personalized inference.

1) Public Prompt Generation: As shown in Fig. 1 (Part B), UE $n \in \mathcal{N}$ generates its personal prompt $\mathbf{p}_n^{\text{per}}$, which is subsequently uploaded to the ES. After all UEs have completed the uploading process, the LLM agent[3] extracts key subjects from the collective set of personal prompts $\mathbf{P}^{\text{per}} = \{\mathbf{p}_n^{\text{per}} \mid n \in \mathcal{N}\}$ on the ES. Following this, the UEs (or the personal prompts) are grouped according to the same key subject. Without loss of generality, let the set $\mathcal{N}_m$ denote the group of UEs that share the same key subject, where $m \in \mathcal{M} \triangleq \{1, \cdots, M\}$. Thus, we have $\bigcup_{m \in \mathcal{M}} \mathcal{N}_m = \mathcal{N}$. The LLM agent further analyzes the personal prompts $\mathbf{P}_m^{\text{per}} = \{\mathbf{p}_n^{\text{per}} \mid n \in \mathcal{N}_m\}$ for group m to identify common features, which are then used to generate a corresponding public prompt $\mathbf{p}_m^{\text{pub}}$.

2) Common-to-Personalized Inference: As shown in Fig. 2, the common inference phase utilizes the GDM with the public prompt $\mathbf{p}_m^{\text{pub}}$ to perform $x_m^{\max} = \max\{x_n, n \in \mathcal{N}_m\}$ inference steps, and generate a corresponding series of common intermediate results on the ES. Let $x_n \in \{0, \cdots, x_{\max}\}$ denote the number of inference steps performed on the ES for UE n in this phase, which is corresponding to a specific common intermediate result. The personalized inference phase I (on ES) selects one of the above results for UE n of group m. The selected result (also with the personal prompt $\mathbf{p}_n^{\text{per}}$) is subsequently re-injected into the GDM to perform $y_n \in \{y_{\min}, \cdots, y_{\max}\}$ inference steps to generate the personalized intermediate result on the ES. Upon receiving the personalized intermediate result, the personalized inference phase II (on UEs) performs $z_n \in \{z_{\min}, \cdots, z_{\max}\}$ inference steps to output the final generated content on UE n. Here, we refer to x_n and $(y_n + z_n)$ as the common and personalized inference steps, respectively.

In the common inference phase, the GDM utilizes the public prompt to generate a series of common intermediate results. The computation delay[4] for group m can be calculated as:

$$T_m^{\text{C}} = \frac{x_m^{\max} \cdot \xi^{\text{ES}}}{f_m^{\text{ES}}}, \tag{1}$$

where ξ^{ES} is the number of cycles required for each inference step on the ES, and f_m^{ES} is the computing frequency allocated to group m. According to [8], the energy consumption in this phase can be calculated as:

$$E_m^{\text{C}} = \kappa^{\text{ES}} \cdot x_m^{\max} \cdot \xi^{\text{ES}} \cdot (f_m^{\text{ES}})^2, \tag{2}$$

where κ^{ES} is the ES's effective switched capacitance.

In the personalized inference phase I (on ES), the selected common intermediate result (also with a specific personal prompt) is re-injected into the GDM to generate the personalized intermediate result on the ES. The computation delay for UE n performed on the ES can be calculated as:

$$T_n^{\text{P-E}} = \frac{y_n \cdot \xi^{\text{ES}}}{f_n^{\text{ES}}}, \tag{3}$$

where f_n^{ES} is the computing frequency allocated to UE n. Similarly, the energy consumption in this phase can be calculated as:

$$E_n^{\text{P-E}} = \kappa^{\text{ES}} \cdot y_n \cdot \xi^{\text{ES}} \cdot (f_n^{\text{ES}})^2. \tag{4}$$

In the personalized inference phase II (on UEs), upon receiving the personalized intermediate result via wireless transmission, the UE continues the inference process to output the final content. The transmission delay can be calculated as:

$$T_n^{\text{T}} = \frac{n_{\text{h}} \cdot n_{\text{w}} \cdot d_{\text{h}} \cdot \varrho}{r_n \cdot \tau}, \tag{5}$$

where $(n_{\text{h}} \cdot n_{\text{w}})$ is the total number of tokens in the latent space, d_{h} is the dimension of each token, ϱ is the number of bits required to represent the information per token per dimension. τ is the compression ratio, and r_n is the downlink transmission rate, which can be calculated as:

$$r_n = b_n \cdot \log_2\left(1 + \frac{p_n \cdot g_n}{\sigma_0^2 \cdot b_n}\right), \tag{6}$$

where b_n and p_n are the bandwidth and transmit power, respectively, g_n is the channel gain, and σ_0^2 is the noise power. The energy consumption of wireless transmission can be calculated as:

$$E_n^{\text{T}} = \frac{p_n \cdot n_{\text{h}} \cdot n_{\text{w}} \cdot d_{\text{h}} \cdot \varrho}{r_n \cdot \tau}. \tag{7}$$

The computation delay of UE n can be calculated as:

$$T_n^{\text{P-U}} = \frac{z_n \cdot \xi_n^{\text{UE}}}{f_n^{\text{UE}}}, \tag{8}$$

where ξ_n^{UE} is the number of cycles required for each inference step on UE n, and f_n^{UE} is the available computing frequency of UE n. Similarly, the corresponding energy consumption in this phase can be calculated as:

$$E_n^{\text{P-U}} = \kappa^{\text{UE}} \cdot z_n \cdot \xi_n^{\text{UE}} \cdot (f_n^{\text{UE}})^2, \tag{9}$$

where κ^{UE} is the UE's effective switched capacitance.

[3] The LLM agent employs a lightweight model, such as Llama 3.2 [7], which can be executed in batches at a very high speed.

[4] For group m, the ES first perfoms $x_m^{\max}$ inference steps. Then, each UE in group m selects a suitable common intermediate result for further processing.

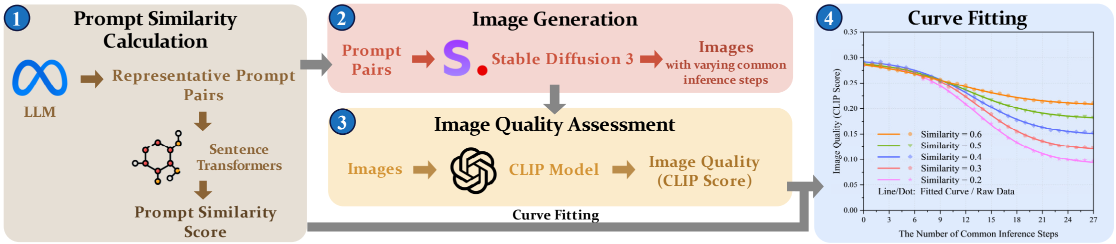


Fig. 3. An overview of the experimental process and the corresponding fitted curve based on real data.

III. Hybrid Inference Quality Metric

A. *Experimental Design and Key Findings*

We configure our experimental environment on an Ubuntu 20.04 system, with an Intel Xeon Platinum 8358 CPU and an NVIDIA RTX A800 GPU. As shown in Fig. 3, the detailed experimental process is as follows. The process begins with *Prompt Similarity Calculation* (Step 1). Inspired by [9], we first leverage advanced LLMs (e.g., Llama) to generate representative prompts, which are then randomly organized into multiple pairs. Next, we use the sentence-transformer model [10] to calculate the similarity score between pairs of prompts. The prompt similarity score can be calculated as:

$$\phi = \mathcal{C}\left(\text{emb}_{\text{S-T}}\left(\mathbf{p}^{(1)}\right), \text{emb}_{\text{S-T}}\left(\mathbf{p}^{(2)}\right)\right), \tag{10}$$

where $\mathcal{C}(\cdot,\cdot)$ is the cosine similarity function, and $\text{emb}_{\text{S-T}}(\cdot)$ is the prompt embedding. $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$ are a random pair of prompts, where $\mathbf{p}^{(1)}$ can be treated as the public prompt used to generate the common intermediate result, and $\mathbf{p}^{(2)}$ can be treated as the personal prompt reflecting the real intention. Next, in *Image Generation* (Step 2), we use Stable Diffusion 3 Medium [11] as an exemplified GDM to generate images with varying common inference steps. We finally end up with a total of $27,000$ images, each with a resolution of 1024×1024, leading to the following three key findings:[5]

- ***F1)*** As the number of common inference steps increases, the semantic alignment between the personal prompt and the generated image gradually decreases.
- ***F2)*** As the number of common inference steps increases, the generated image normally maintains satisfactory detail (but unsatisfactory semantic alignment).
- ***F3)*** Compared to a pair of prompts with low similarity, the impact of the number of common inference steps is less pronounced for prompts with higher similarity.

Based on the above findings, the number of common inference steps primarily affect the semantic information of the generated image, and this effect is correlated with the similarity between the public and personal prompts. Hence, in *Image Quality Assessment* (Step 3), we use the pre-trained ViT-L/14 CLIP (Contrastive Language-Image Pre-Training) model [12] to calculate the similarity between the personal prompt and the generated image, called the CLIP score. The CLIP score can be calculated as follows:

$$S_{\text{CLIP}} = \mathcal{C}\left(\text{emb}_{\text{CLIP}}\left(\mathbf{p}^{(2)}\right), \text{emb}_{\text{CLIP}}\left(\mathbf{I}\right)\right), \tag{11}$$

where $\text{emb}_{\text{CLIP}}(\cdot)$ is the CLIP embedding, and $\mathbf{I}$ is the generated image with certain common inference steps.

B. *Hybrid Inference Quality Metric (HIQM) Function*

In *Curve Fitting* (Step 4), the dots represent the raw data (the average image quality based on a certain number of common inference steps and also a certain similarity). Obviously, the image quality initially decreases slowly, then more rapidly, and finally stabilizes as the number of common inference steps increases. Therefore, we use a modified sigmoid function to capture the relationship between the number of common inference steps and image quality. Based on this, the HIQM function can be defined as follows:

$$\hat{Q}^{\phi}(x) = \frac{L^{\phi}}{1+\exp(K^{\phi}\cdot(x-B^{\phi}))} + D^{\phi}, \tag{12}$$

where the independent variable x is the number of common inference steps. L^{ϕ}, K^{ϕ}, B^{ϕ}, and D^{ϕ} are coefficients related to the similarity ϕ between the public and personal prompts. The coefficients $(L^{\phi} + D^{\phi})$ and D^{ϕ} represent the maximum and minimum values of the image quality, respectively.

Specifically, when $x = 0$, the image quality for different similarity curves are expected to be roughly the same because the common inference phase is not introduced in this case. However, when $x = 27$ (with only the common inference phase), the coefficient D^{ϕ} increases with similarity, indicating that even images generated from pure public prompt-based inference can achieve a reasonable level of semantic alignment. The coefficients K^{ϕ} and B^{ϕ} represent the steepness of the curve and the location of the inflection point, respectively. Specifically, as similarity increases, the function becomes progressively flatter. This indicates that with the same number of common inference steps, images with higher similarity can maintain better quality.

IV. Optimization of Hybrid Inference Scheme

A. *Problem Formulation*

As discussed in Section II, the service delay of UE n can be calculated as:

$$T_n = \sum_{m=1}^{M}\left(\gamma_{n,m}\cdot T_m^{\text{C}}\right) + T_n^{\text{P-E}} + T_n^{\text{T}} + T_n^{\text{P-U}}, \tag{13}$$

where $\gamma_{n,m} = 1$ indicates that UE n is clustered into group m, and vice versa. Hence, we have $\sum_{m=1}^{M}\gamma_{n,m} = 1, \forall n \in \mathcal{N}$. The energy consumption of UE n can be calculated as:

$$E_n = E_n^{\text{P-E}} + E_n^{\text{T}} + E_n^{\text{P-U}}. \tag{14}$$

As illustrated in Section III, we use the HIQM function $\hat{Q}^{\phi}$ as the UE's QoS $Q_n^{\phi_n}$. Thus, the total QoS Q, total service delay T, and total energy consumption E of all UEs can be calculated as:

$$Q = \sum_{n=1}^{N} Q_n^{\phi_n}, \quad T = \sum_{n=1}^{N} T_n, \tag{15}$$

[5] Our code is available in: https://github.com/iimxinyi/Inference-Sharing

$$E = \sum_{m=1}^{M} E_m^{\mathrm{C}} + \sum_{n=1}^{N} E_n. \tag{16}$$

For notational convenience, we introduce the following notations: $\mathbf{X} \triangleq (x_n,\ n \in \mathcal{N})$, $\mathbf{Y} \triangleq (y_n,\ n \in \mathcal{N})$, and $\mathbf{Z} \triangleq (z_n,\ n \in \mathcal{N})$. Based on this, the hybrid inference optimization problem can be defined as follows:

$$\begin{aligned}
\mathbb{P}:\ \max_{\mathbf{X},\mathbf{Y},\mathbf{Z}}\ & \varpi_1 U - \varpi_2 T - \varpi_3 E && \text{(17a)}\\
\text{s.t.}\ & x_n \in \{0, \cdots, x_{\max}\}, \quad \forall n \in \mathcal{N} && \text{(17b)}\\
& y_n \in \{y_{\min}, \cdots, c_{\max}\}, \quad \forall n \in \mathcal{N} && \text{(17c)}\\
& z_n \in \{z_{\min}, \cdots, q_{\max}\}, \quad \forall n \in \mathcal{N} && \text{(17d)}\\
& x_n + y_n + z_n = T_{\max}, \quad \forall n \in \mathcal{N}, && \text{(17e)}
\end{aligned}$$

where ϖ_1, ϖ_2, and ϖ_3 are weight factors. Constraint (17b), (17c), and (17d) give the range of inference steps in different phases. Constraint (17e) sets the total number of inference steps as a pre-determined constant. Notably, the value of $T_{\max}$ is sampler-specific and represents the number of inference steps required to achieve satisfactory results.

To make real-time decisions, we utilize the DRL algorithm to solve the formulated problem. However, the state-action space grows exponentially by the number of agents (UEs). This curse of dimensionality poses challenges to the traditional centralized execution scheme. Besides, since part of the inference process is performed on the ES side, the decision-making of each agent needs to be uploaded to the ES. This process introduces unnecessary communication overhead, posing challenges to the decentralized execution scheme.

B. Algorithm Design

To address the above challenges, we introduce the centralized control widely used for multi-agent systems as in [13]–[15]. Specifically, the ES acts as a centralized controller, hosting the neural networks of all agents (UEs). At each slot, the ES obtains each agent's individual observation, then makes and executes the corresponding decisions for each UE in a parallel way. Following the above idea, we develop a Hybrid Inference Decision (HID) algorithm, which employs the IPPO [16] approach with advantage normalization [17] and reward scaling [18] techniques to solve the problem in a centralized manner. Next, we provide details of the HID algorithm:

1) Observation: The observation consists of two parts: resource status and prompt similarity. Here, we define the observation for UE n at slot i as:

$$\mathbf{o}_n^{(i)} = (f_n^{\mathrm{UE},(i)}, \phi_n^{(i)}). \tag{18}$$

2) Action: The action consists of two parts: the number of inference steps in common and personalized inference phases. Here, we define the action for UE n at slot i as:

$$\mathbf{a}_n^{(i)} = (x_n^{(i)}, y_n^{(i)}). \tag{19}$$

3) Reward: The reward at slot i is defined as:

$$r^{(i)} = \varpi_1 U - \varpi_2 T - \varpi_3 E. \tag{20}$$

Based on the above, we summarize the HID algorithm in Algorithm 1. At each slot, the ES obtains individual observations and chooses corresponding actions for all agents (UEs) (lines 4-5). Then, the ES performs the common inference phase and personalized inference phase I based on all actions (line 6). After wireless transmission, the personalized inference phase II is performed on the UE based on Eq. (17e). Once all content has been generated, the ES obtains the reward and next individual observations, and stores the experiences in corresponding replay buffers (lines 7-8). Finally, the actor network can be updated according to the following objective[6]:

$$\begin{aligned}
&\arg\max_{\boldsymbol{\Theta}_n} \Bigg\{ \tfrac{1}{DI} \textstyle\sum_{d=1}^{D} \sum_{i=1}^{I} \min \Bigg\{ \tfrac{\pi_{\boldsymbol{\Theta}_n}(\mathbf{a}_n^{(d,i)} | \mathbf{o}_n^{(d,i)})}{\pi_{\boldsymbol{\Theta}_n^{\mathrm{old}}}(\mathbf{a}_n^{(d,i)} | \mathbf{o}_n^{(d,i)})} \hat{A}_n^{(d,i)}, \\
&\quad \mathrm{clip}\Bigg(\tfrac{\pi_{\boldsymbol{\Theta}_n}(\mathbf{a}_n^{(d,i)} | \mathbf{o}_n^{(d,i)})}{\pi_{\boldsymbol{\Theta}_n^{\mathrm{old}}}(\mathbf{a}_n^{(d,i)} | \mathbf{o}_n^{(d,i)})} \hat{A}_n^{(d,i)}, 1 - \varepsilon, 1 + \varepsilon \Bigg) \hat{A}_n^{(d,i)} \Bigg\} \Bigg\},
\end{aligned} \tag{21}$$

where $\hat{A}_n^{(d,i)}$ is the generalized advantage estimation, D is the batch size, and $\pi_{\boldsymbol{\Theta}_n}$ is the actor network for UE n.

Algorithm 1: HID Algorithm

```
1 Initialize actor networks $\pi_{\boldsymbol{\Theta}_n}$, critic networks $V_{\boldsymbol{\Phi}_n}$ and
    replay buffers $\mathcal{D}_n, \forall n$;
2 for each episode $e = 1, \cdots, E$ do
3     for each slot $i = 1, \cdots, I$ do
4         Obtain individual observations $\mathbf{o}_n^{(i)}, \forall n$;
5         Choose corresponding actions $\mathbf{a}_n^{(i)}, \forall n$;
6         Execute actions for all agents simultaneously;
7         Obtain the reward $r^{(i)}$ and next individual
            observations $\mathbf{o}_n^{(i+1)}, \forall n$;
8         Store the experiences $(\mathbf{o}_n^{(i)}, \mathbf{a}_n^{(i)}, r^{(i)}, \mathbf{o}_n^{(i+1)})$ in
            corresponding replay buffers $\mathcal{D}_n, \forall n$;
9     Update $\pi_{\boldsymbol{\Theta}_n}$ and $V_{\boldsymbol{\Phi}_n}$, $\forall n$, by Eq. (21) and [16];
```

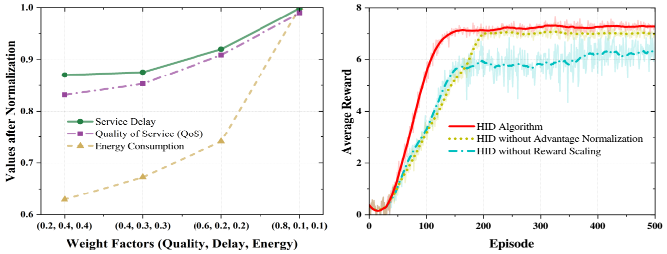
V. Simulation Results

In this section, we provide simulation results to demonstrate the effectiveness of the *HID algorithm* and the superiority of our proposed *hybrid inference scheme*. For simplicity, we next abbreviate our proposed scheme as HIS. The basic parameters are listed in Table I. For comparisons, we introduce the following three inference schemes as benchmarks:

- *Public Prompt Generation-Free Inference Scheme (PPGFIS)* [6]: This scheme employs the knowledge graph for grouping. In each group, a randomly selected personal prompt can be designated as the public prompt.
- *Grouping-Free Inference Scheme (GFIS)* [5]: This scheme treats all UEs as a single group, from which a personal prompt is randomly selected as the public one.
- *Independent Inference Scheme (IIS)* [4]: This scheme does not introduce the common inference phase.

1) The effectiveness of our proposed HID algorithm: As illustrated in Fig. 4(a), as the weight factor of the QoS increases, the values of the QoS, service delay, and energy consumption also rise, indicating that our proposed HID algorithm tends to achieve higher QoS with less common inference steps. This trend reveals that the HID algorithm

[6] For more details regarding the objective function of the critic network, please refer to [16].



(a) (b)

Fig. 4. (a) Comparison of the QoS, service delay, and energy consumption in different weight factors; (b) Comparison of different algorithms: HID algorithm, HID algorithm without advantage normalization, and HID algorithm without reward scaling.

TABLE I
PARAMETER SETTINGS

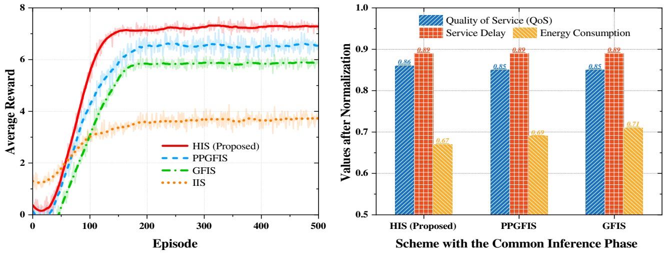
Parameter	Value
N, M, $x_{\max}$, $y_{\min}$, $y_{\max}$	50, 2, 13, 0, 16
ξ^{ES}, ξ^{UE}, κ^{ES}	15 Mcycles, 14 Mcycles, 10^{-27}
f_m^{ES}, f_n^{ES}, f_n^{UE}	2 GHz, 2 GHz, [1.4, 1.6] GHz
n_{h}, n_{w}, d_{h}, ϱ, κ^{UE}	128, 128, 16, 16, 10^{-27}
Learning Rate, Batch Size	0.0005, 128
Hidden Layers, Optimizer	[128, 128, 128], Adam

is adaptable to both QoS-sensitive and delay-energy-sensitive AIGC networks. Fig. 4(b) demonstrates the effect of two techniques on the performance of the HID algorithm. The absence of advantage normalization leads to higher variance in advantage estimates, which amplifies noise and results in slower convergence. Similarly, without reward scaling, the HID algorithm struggles to balance reward values, leading to instability and fluctuating performance.

2) The superiority of our proposed hybrid inference scheme (HIS): As depicted in Fig. 5(a) and Fig. 5(b), our proposed HIS outperforms other inference schemes. Several key conclusions can be drawn: (i) Taking advantage of the public prompt generation stage, the HIS achieves a superior balance among the QoS, service delay, and energy consumption. It improves the reward value by 12.6%, 24.1%, and 102.2% compared to the PPGFIS, GFIS, and IIS, respectively. (ii) Compared to the PPGFIS and GFIS, the HIS substantially reduces the energy consumption while increasing the QoS. Especially when the number of UEs is very high, resource consumption can be reduced by generating public prompts to share variable part of the inference process. (iii) Compared to IIS, the HIS can significantly improve the network performance by reducing the energy consumption, making it highly suitable for large-scale resource-constrained AIGC networks.

VI. Conclusion

In this work, we designed an efficient and QoS-driven hybrid inference scheme for MEC-enabled AIGC networks. First, we proposed a data-driven hybrid inference quality metric to effectively characterize the QoS. Second, we formulated the joint optimization problem, and then employed the IPPO approach to balance the trade-offs among the QoS, service delay, and energy consumption. Finally, Simulation results demonstrate that our proposed scheme significantly outperforms existing benchmarks in the literature, with the performance gains ranging from 12.6% to 102.2%.



(a) (b)

Fig. 5. (a) Reward curves of several inference schemes; (b) Values of the QoS, service delay, and energy consumption for several inference schemes with the common inference phase.

References

[1] M. Xu *et al.*, “Unleashing the power of edge-cloud generative AI in mobile networks: A survey of AIGC services,” *IEEE Commun. Surv. Tutor.*, vol. 26, no. 2, pp. 1127-1170, Secondquarter 2024.

[2] H. Liu, J. Wu, X. Zhuang, H. Wu, and L. Gao, “Joint communication and computation scheduling for MEC-enabled AIGC services based on generative diffusion model,” in *Proc. Int. Symp. Model. Optim. Mob., Ad Hoc, Wirel. Networks (WiOpt)*, Seoul, Republic of Korea, Oct. 2024, pp. 345-352.

[3] X. Zhuang, J. Wu, H. Wu, T. Zhang, and L. Gao, “Joint optimization of model inferencing and task offloading for MEC-empowered large vision model services,” in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, London, United Kingdom, May 2025, pp. 1-10.

[4] H. Du *et al.*, “Diffusion-based reinforcement learning for edge-enabled AI-generated content services,” *IEEE Trans. Mob. Comput.*, vol. 23, no. 9, pp. 8902-8918, Sep. 2024.

[5] H. Du *et al.*, “User-centric interactive AI for distributed diffusion model-based AI-generated content,” 2023. *arXiv:2311.11094*.

[6] G. Xie *et al.*, “GAI-IoV: Bridging generative AI and vehicular networks for ubiquitous edge intelligence,” *IEEE Trans. Wireless Commun.*, vol. 23, no. 10, pp. 12799-12814, Oct. 2024.

[7] Meta. “Llama-3.2-3B.” huggingface.io. Accessed: Oct. 1, 2024. [Online.] Available: https://huggingface.co/meta-llama/Llama-3.2-3B

[8] J. Zhao, L. Qian, and W. Yu, “Human-centric resource allocation in the metaverse over wireless communications,” *IEEE J. Sel. Areas Commun.*, vol. 42, no. 3, pp. 514-537, Mar. 2024.

[9] F. Fan, C. Luo, W. Gao, and J. Zhan, “AIGCBench: Comprehensive evaluation of image-to-video content generated by AI,” *BenchCouncil Trans. Benchmarks Stand. Eval.*, vol. 3, no. 4, pp. 1-11, Dec. 2023.

[10] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence embeddings using siamese BERT-networks,” in *Proc. Conf. Empir. Methods Nat. Lang. Process. Int. Jt. Conf. Nat. Lang. Process. (EMNLP-IJCNLP)*, Hong Kong, China, Nov. 2019, pp. 3982-3992.

[11] Stability AI. “stable-diffusion-3-medium.” huggingface.io. Accessed: Oct. 1, 2024. [Online.] Available: https://huggingface.co/stabilityai/stable-diffusion-3-medium/tree/main

[12] A. Radford *et al.*, “Learning transferable visual models from natural language supervision,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, Virtual, Online, Jul. 2021, pp. 8748-8763.

[13] A. Mostaani, T. X. Vu, S. Chatzinotas, and B. Ottersten, “Centralized control of a multi-agent system via distributed and bit-budgeted communications,” in *Proc. Wireless Commun. Networking Conf. (WCNC)*, Glasgow, United Kingdom, Mar. 2023, pp. 1-6.

[14] J. Wu, J. Luo, C. Jiang, and L. Gao, “A multi-agent deep reinforcement learning approach for multi-UAV cooperative search in multilayered aerial computing networks,” *IEEE Internet Things J.*, vol. 12, no. 5, pp. 5807-5821, Mar. 2025.

[15] J. Wu, M. Tang, C. Jiang, L. Gao, and B. Cao, “Cloud-edge-end collaborative task offloading in vehicular edge networks: A multi-layer deep reinforcement learning approach,” *IEEE Internet Things J.*, vol. 11, no. 22, pp. 36272-36290, Nov. 2024.

[16] C. S. D. Witt *et al.*, “Is independent learning all you need in the StarCraft multi-agent challenge?,” 2020, *arXiv:2011.09533*.

[17] G. Tucker *et al.*, “The mirage of action-dependent baselines in reinforcement learning,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, Stockholm, Sweden, Jul. 2018, pp. 7985-7994.

[18] L. Engstrom *et al.*, “Implementation matters in deep policy gradients: A case study on PPO and TRPO,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Addis Ababa, Ethiopia, Apr. 2020, pp. 1-9.