Temporal Prompt Engineering for Generative Semantic Communication

Yiru Wang*†, Wanting Yang†, Zehui Xiong†, Yuping Zhao*
*Peking University, Beijing, China
†Singapore University of Technology and Design, Singapore
yiruwang@stu.pku.edu.cn, {wanting_yang, zehui_xiong}@sutd.edu.sg, yuping.zhao@pku.edu.cn

2024 IEEE 100th Vehicular Technology Conference (VTC2024-Fall) | 979-8-3315-1778-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/VTC2024-Fall63153.2024.10757628

***Abstract*—The rapidly evolving field of generative artificial intelligence technology has introduced innovative approaches for developing semantic communication (SemCom) frameworks, leading to the emergence of a new paradigm—generative AI-assisted SemCom (GSC). Benefiting from its strong ability to understand and generate high-quality content across various domains, this approach can effectively address the reconstruction limitations that challenge traditional SemCom systems. However, this architecture often suffers from high latency due to the complex processes involved in semantic extraction and generative semantic inference. To mitigate this issue, we propose a low-latency GSC framework, achieved by enabling the parallel execution of the transmitter's semantic extracting and the receiver's generating processes from a macro perspective. Furthermore, to attain more accurate and semantically aligned reconstruction, we design a temporal prompt engineering approach that utilizes reinforcement learning to sequence the temporal feature extraction steps at the transmitter. The results show that compared to the conventional GSC architecture, our designed framework can achieve a 52% reduction in residual task latency that extends beyond the fixed inference duration while only incurring an approximate 9% decrease in task score.**

***Index Terms*—Semantic communication, AI-generated content, generative AI, prompt engineering**

I. Introduction

THE forthcoming era of six-generation networks is poised to witness an unprecedented surge in data traffic, driven by emerging applications such as ultra-high-definition video streaming, augmented reality, virtual reality, and massive Internet of Things deployments. This escalation exacerbates the existing scarcity of spectrum resources, posing significant challenges for traditional communication paradigms. In this context, semantic communication (SemCom) presents a transformative solution, promising to tackle these challenges by focusing on the transmission of meaning and semantic content, thereby potentially reducing the required data payload and enhancing the overall efficiency of information exchange [1].

However, the widespread adoption of SemCom is hindered by several challenges. For example, the commonly used end-to-end models limit the SemCom frameworks' social acceptance and practicality in complex network environments [2]. Additionally, the predominantly uni-modal designs in current SemCom studies are often inadequate for scenarios involving multiple modalities or cross-modal interactions. Furthermore, the existing SemCom methods struggle to recover high-quality content when communication resources are severely constrained [3].

Recognizing the capacity of generative AI (GAI) for modal transformation and its exceptional ability to produce high-quality content across various domains, several studies have suggested integrating GAI to enhance the existing SemCom frameworks. To reduce the transmission payload, the authors in [4] combine GAI with semantic coding models and distill text prompts to facilitate efficient image delivery. However, due to the simplicity of extractors, many semantic attributes are lost during processing, affecting the fidelity of image reconstruction at the receiver. To enhance the reliability and efficiency of information transmission and reconstruction for vehicular networks, a GAI-based SemCom framework is introduced in [5], which utilizes diverse prompts for communication.

Nevertheless, it is essential to note that although GAI-assisted SemCom (GSC) demonstrates superiority in the aforementioned communication scenarios in [4], [5], it introduces additional computational latency. In a general GSC framework, to enhance communication reliability, the transmitter might employ multiple semantic extraction models to capture multiple types of semantic attributes, while the receiver relies on the received multiple prompts to reconstruct images based on an iterative denoising diffusion model. The cumulative delays in the entire computing process could potentially compromise the user experience, especially in applications where immediacy and real-time processing are critical.

To address the abovementioned issue, we introduce the Parallel Semantic Extraction and Generation based GSC (PGSC) with Temporal Prompt Engineering (TPE-PGSC), which can balance the overall latency and the quality of transmission. The contributions of this work are outlined as follows:

- To reduce the latency of GSC, we design a communication mechanism that incorporates parallel semantic extraction and generation. This design significantly reduces task latency, particularly in complex scenarios that necessitate the extraction and transmission of multiple types of semantic features.
- Building on the proposed PGSC framework, we introduce a reinforcement learning (RL)-guided temporal prompt

The work is supported by the National Key Research and Development Program of China under Grant 2018YFB1801403 and 2018YFB1801402. This work is also supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Future Communications Research & Development Programme. The work is also supported by the Ministry of Education, Singapore, under its SMU-SUTD Joint Grant (22-SIS-SMU-048), and SUTD Kickstarter Initiative (SKI 20210204).

engineering approach. This method aims to optimize the feature extraction sequence and align their semantic attributes with different phases of the receiver's denoising process, thereby enhancing the overall performance of the PGSC framework in terms of both reconstruction quality and system latency.

- Through simulations, the results indicate that our proposed TPE-PGSC approach can reduce latency by approximately 52% while only incurring a 9% loss in performance score compared to the conventional GSC.

II. PGSC: Framework and Key Components

In this study, we illustrate an image transmission scenario to demonstrate our method. Specifically, the receiver is set as an image renderer that produces images of varying levels of detail based on different tasks. The transmitter, on the other hand, is an end device with limited computational capabilities and resources, responsible for capturing the key features of the source images according to the receiver's request.

The PGSC framework can be divided into distinct modular functional components, including semantic extraction, wireless transmission, and diffusion-based reconstruction. In the following, we first introduce the semantic extraction and diffusion-based reconstruction processes. Subsequently, we compare the proposed PGSC framework with the conventional GSC framework regarding system latency and communication mechanisms.

A. Semantic Extraction

Similar to GSC, the extraction process is dedicated to deriving semantic features from the original source data. Assuming there are N_e extraction models implemented at the transmitter, each model is tasked with extracting semantic units that belong to a specific attribute, leveraging its specialized knowledge. Specifically, we denote each extraction result as a *semantic unit*, and refer to the collection of all semantic units from an image as a *prompt*.

We denote the source image as $\mathbf{s} \in \mathbb{R}^{c\times h\times w}$. The semantic extraction process of distilling the kth semantic unit $\mathbf{u}_k$ can be expressed by

$$\mathbf{u}_k = f_k\left(\mathbf{s};\boldsymbol{\psi}_k\right), \tag{1}$$

where $\boldsymbol{\psi}_k$ represents the trainable parameters of the extraction model for distilling $\mathbf{u}_k$, K represents the maximum number of semantic units and $\forall k \in \{0,1,...,K-1\}$.

B. Diffusion-based Reconstruction

Inspired by the significant success of diffusion models across a wide range of real-world generation tasks, the Stable Diffusion model [6] is commonly employed for image reconstruction at the receiver within the GSC framework [3], [4]. Its goal is to generate images that semantically adhere to the descriptions of the received text.

The stable diffusion model is built upon the denoising diffusion probabilistic model (DDPM) [7], which is capable of learning a distribution $p_\theta(\mathbf{x}_0)$ that approximates $q(\mathbf{x}_0)$. It can be treated as a latent variable model of the form $p_\theta(\mathbf{x}_0) = \int p_\theta(\mathbf{x}_{0:T})d\mathbf{x}_{1:T}$, where $\mathbf{x}_1,...,\mathbf{x}_T$ are called latents of the same dimensionality as the original data $\mathbf{x}_0 \sim q(\mathbf{x}_0)$. The joint distribution $p_\theta(\mathbf{x}_{0:T})$ is called the *reverse process*, which can be formulated as $p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$.

The learning process starts from a pure noise image $\mathbf{x}_N \sim \mathcal{N}(\mathbf{x}_N;\mathbf{0},\mathbf{I})$. The Markov chain with learned Gaussian transition can be expressed by $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1};\mu_\theta(\mathbf{x}_t,t),\boldsymbol{\Sigma}_\theta(\mathbf{x}_t,t))$. To facilitate the learning process, a *forward process* or *diffusion process* $q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$ is implemented, which is a fixed Markov chain that gradually adds Gaussian noise to the data according to a variance schedule $\beta_1,...,\beta_T$, as $q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t;\sqrt{1-\beta_t}\mathbf{x}_{t-1},\beta_t\mathbf{I}\right)$. Denote $\alpha_t = 1-\beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t}\alpha_s$, we can sample $\mathbf{x}_t$ at timestep t as $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t;\sqrt{\bar{\alpha}_t}\mathbf{x}_0,(1-\bar{\alpha}_t)\mathbf{I})$. Subsequently, the forward posteriors conditioned on $\mathbf{x}_0$ can be expressed by

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_{t-1};\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0),\tilde{\beta}_t\mathbf{I}\right), \tag{2}$$

where $\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t$ and $\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$. The target of the learning process of DDPM is to minimize the gap between $q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)$ and $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$. When the coefficient of variance is considered as a constant, this goal is equivalent to predict $\mu_\theta(\mathbf{x}_t,t)$ that as close as possible to $\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0)$. Denote $\mathbf{y}$ as the conditioning information, the loss function during training is formulated as the difference between the generated noise ε and predicted noise $\varepsilon_\theta(\mathbf{x}_0,t,\mathbf{y})$, formulated as follows [6]:

$$\mathcal{L} = \mathbb{E}_{t,\mathbf{y},\mathbf{x}_t,\varepsilon}\left[\left\|\varepsilon - \varepsilon_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon,t,\mathbf{y}\right)\right\|_2^2\right]. \tag{3}$$

During sampling, the noise estimation can be performed by classifier-free guidance [8], which is expressed by

$$\tilde{\varepsilon}_\theta(\mathbf{x}_t,\mathbf{y},t) = \varepsilon_\theta(\mathbf{x}_t,\mathbf{y},t) + w\cdot\left(\varepsilon_\theta(\mathbf{x}_t,\mathbf{y},t) - \varepsilon_\theta(\mathbf{x}_t,t)\right), \tag{4}$$

where w is the guidance scale.

As DDPM necessitates executing all consecutive denoising steps to produce the clean sample $\mathbf{x}_0$, it can pose challenges for real-time communication services. To reduce the generation latency, we adopt the sampling method of the denoising diffusion implicit model (DDIM) [9]. It improves upon DDPM by utilizing a non-Markovian inference process, which can be formulated as

$$\begin{aligned}\mathbf{x}_{t-1} =& \sqrt{\bar{\alpha}_{t-1}}\left(\frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\tilde{\varepsilon}_\theta(\mathbf{x}_t|\mathbf{y},t)}{\sqrt{\bar{\alpha}_t}}\right)\\ &+ \sqrt{1-\bar{\alpha}_{t-1}}\cdot\tilde{\varepsilon}_\theta(\mathbf{x}_t|\mathbf{y},t).\end{aligned} \tag{5}$$

C. Latency Discussion and Communication Mechanism

The system latency can be categorized into the following three components:

1) Semantic Extraction Latency: Due to limited computational resources, the extraction process proceeds sequentially, meaning that the kth semantic unit can only be extracted after the $(k-1)$th semantic unit has been obtained. Consequently,

the aggregate latency of the entire extraction procedure can be mathematically expressed as

$$\tau_{ext} = \sum_{k=0}^{K-1} \tau_k, \tag{6}$$

where τ_k is the latency for extracting the k-th semantic unit. In practice, τ_k hinges on the efficiency of each extraction model and the computational power available at the transmitter, with durations extending from milliseconds to seconds [10].

2) Wireless Transmission Latency: During transmission, where prompts are formatted as text, leveraging the capabilities of 5G technology can significantly reduce the transmission latency, τ_{trans}, to less than one millisecond [11].

3) Diffusion-based Reconstruction Latency: In the generation process, empowered by DDIM [9], we can skip some denoising steps, and the latency of the reconstruction model is determined by the set total number of denoising steps M, which can be formulated as

$$\tau_{gen} = \sum_{m=0}^{M-1} \tau_m, \tag{7}$$

where τ_m is the generation latency of the denoising step m and $m \in \{0, 1, ..., M-1\}$. In practice, τ_m depends on the computational resources of the receiver, varying from milliseconds to seconds [12].

Therefore, the system latency is dominated by the extraction and generation processes. A significant cumulative latency results when both the number of required semantic units and the total number of denoising steps are large.

In conventional GSC, the aforementioned processes operate in a temporally disjointed manner, resulting in cumulative latency. By contrast, in PGSC, we propose executing the semantic extraction and diffusion-based reconstruction in parallel. Specifically, the diffusion model's denoising process is initiated by the first received semantic units. In subsequent stages, newly arrived semantic units combine with previously arrived units to collectively guide the denoising process. A comparison of the communication mechanisms and system latency of the conventional GSC and our proposed PGSC is presented in Fig. 1.

III. RL-based Temporal Prompt Engineering

In the realm of GAI, prompt engineering plays a pivotal role by defining and refining the inputs given to AI models to produce desired outputs. This process is crucial as the specificity and structure of prompts significantly influence the accuracy, relevance, and quality of the generated content [13]. In the PGSC system, as the denoising process progresses and the noise gradually diminishes, the ability of the conditional prompt to modify the generated image also decreases. In light of this, we propose a TPE approach, which involves developing an RL model to align the feature extraction sequence with the conditioning denoising progress of the diffusion model. Its aim is to ensure the final generated images bear a close resemblance to the comprehensive prompt, while minimizing system latency as much as possible. The designs of the fundamental components are outlined in the following content.

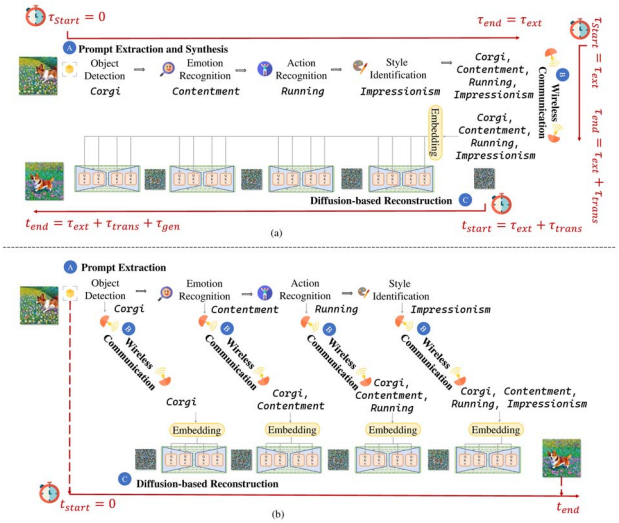

Fig. 1. Comparison of communication mechanisms and system latency. (a) Conventional GSC framework; (b) Proposed PGSC framework.

A. MDP Constructing

In constructing the RL model, we frame the interaction between the agent and the environment as a Markov Decision Process (MDP). The MDP can be defined as a three-component tuple $(\mathcal{S}, \mathcal{A}, r)$, where $\mathcal{S}$ is state space set, $\mathcal{A}$ is the action space set, and r is the reward function. To better illustrate the MDP in our study, we depict an episode in Fig. 2 and the detailed design of this framework is given as follows:

State: We include the task-related request in the agent's state, which contains the attribute and quantity of semantic units required by the receiver. To facilitate the learning of the RL model, we can employ one-hot encoding to preprocess the semantic attributes. Let $\mathbf{e}_{k,t}$ denote the encoded N_e-length one-hot vector for the k-th required semantic attribute at step t, where the n_e-th position is set to 1, corresponding to the activation of the n_e-th extraction model to obtain the relevant semantic unit. When the requested number of semantic units is less than K, we fill the remaining positions in the one-hot encoding with zero vectors. Subsequently, the entire encoded one-hot matrix for the whole prompt can be expressed by $\mathbf{E}_t = [\mathbf{e}_{0,t}, \ldots, \mathbf{e}_{K-1,t}]$. Since the residual noise levels vary at different denoising steps, and the allowable space for modifications also changes, we include the conditional denoising step of the diffusion model d_t as part of its state, which indicates the starting denoising step from which the semantic units selected at time t can guide. When d_t reaches the designated maximum number of denoising steps, the episode terminates. To simplify the learning process of RL models, we divide the inference process into multiple segments at equal intervals. At the beginning of each inference segment, only newly arrived semantic units can be integrated with previously arrived ones to guide the denoising. To distinguish whether the semantic units are still required by the receiver at step t, we also design a binary K-length vector $\mathbf{c}_t$ as an indicator. When an element of this vector is set to 0, it indicates that the corresponding semantic unit has either been transmitted or the associated one-hot attribute vector is a zero vector. The element is set to

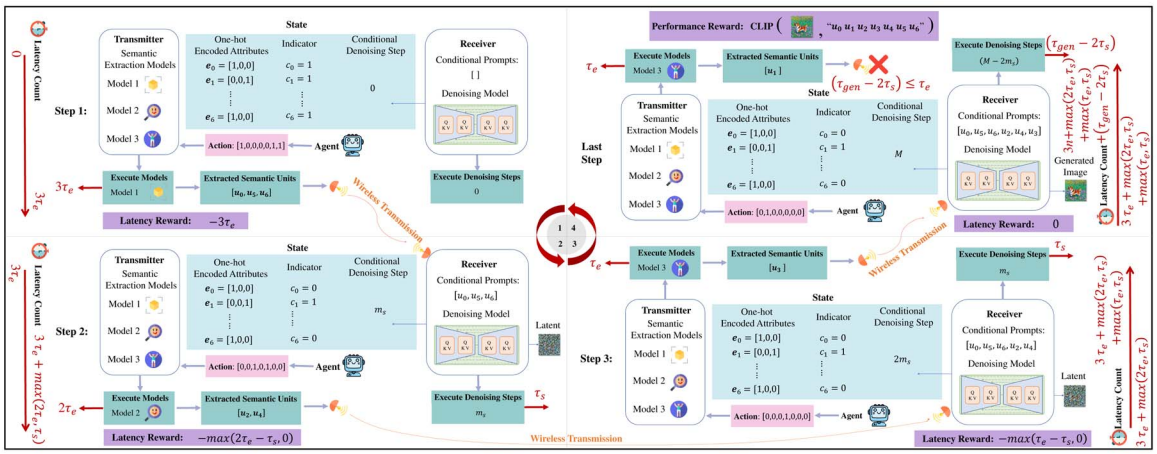

Fig. 2. Illustration of the MDP within a single episode.

1 otherwise. Therefore, the corresponding state observed by the agent at time step t is denoted as $\mathbf{s}_t = [\mathbf{E}_t, \mathbf{c}_t, d_t]$.

Action: The action executed by the agent at time t is denoted by $\mathbf{a}_t = [a_0, ..., a_{K-1}]$, where $a_k \in \{0, 1\}$ for $k = 0, ..., K-1$. The practical implication of this action is that, when the agent selects certain semantic attributes, the corresponding extraction models that distill the semantic units with the selected attributes are executed at this RL step.

Reward: Aligned with the objectives of TPE, the reward is a composite of two components: the system's *performance reward* and the *latency reward*. Similar to [3], we adopt the the CLIP score [14] as the performance reward in our work, which can only be measured at the end of an episode. This metric assesses the semantic similarity between the generated images and the corresponding text prompts within the embedding space of CLIP model [15], which can be expressed by

$$r_p = \text{CLIP-S} = w_1 \cdot \max\left(\cos\left(\mathbf{h}, \mathbf{v}\right), 0\right), \tag{8}$$

where $\mathbf{h}$ represents the textual CLIP embedding, $\mathbf{v}$ denotes visual CLIP embedding and $w_1 = 1$ in our work.

In the PGSC framework, system latency is determined by the degree of parallel execution between semantic extraction and generation processes. Specifically, in the first step of each episode, the receiver has not yet received any prompts, and thus, denoising has not yet started. The latency of this step is entirely determined by the execution time of the selected extraction model. From the second step onward, the receiver can condition on the received prompts, and extraction at the transmitter and generation at the receiver start simultaneously. The latency of these steps is determined by the maximum latency of either the transmitter's semantic extraction or the receiver's segmented denoising. Thus, instead of calculating system latency at the conclusion of one episode, we can expedite the learning process by defining the latency reward as the additional extraction delay relative to the segmented denoising duration at each step. Therefore, the combined reward obtained at step t can be formulated as

$$r_t = \begin{cases} -\tau_e \cdot n_t & t = 0, \\ -\max\left(\left(\tau_e \cdot n_t - \tau_s\right), 0\right) & 1 \leqslant t < L-1, \\ -\max\left(\mathbf{1}_{nor} \cdot \left(\tau_e \cdot n_t - \tau_s\right), 0\right) + r_p & t = L-1, \end{cases} \tag{9}$$

where τ_e denotes the latency for extracting a single semantic unit, which is assumed to be consistent across all semantic units. n_t denotes the number of selected semantic units at RL step t, τ_s is time delay for executing the segmented denoising steps. L denotes the number of steps in one episode, which varies with each episode. Its value depends on the number of semantic units required for transmission in each round and the progress of model learning. The index factor, $\mathbf{1}_{nor}$, equals 1 if the distilled semantic units at this step can be received before the conclusion of the denoising process, and 0 otherwise.

Finally, the long-term reward for the agent can be calculated as the sum of the discounted rewards obtained at each step in an episode, which is expressed by

$$r = \sum_{t=0}^{L-1} \gamma^t \cdot r_t, \tag{10}$$

where γ is a discount factor.

B. Learning Principle

Building on the MDP framework described above, we propose utilizing the Proximal Policy Optimization (PPO) algorithm [16] to identify the optimal strategy for dispatching prompt words. This policy gradient method directly updates a stochastic policy neural network to model the action distribution for a given state.

To efficiently leverage experiences gathered under an old policy $\pi_{\boldsymbol{\varphi}'}\left(\mathbf{a}_t|\,\mathbf{s}_t\right)$ for estimating potential performance under a new policy $\pi_{\boldsymbol{\varphi}}\left(\mathbf{a}_t|\,\mathbf{s}_t\right)$, we can define the probability ratio between the new policy and the old policy as $r_t\left(\boldsymbol{\varphi}\right) = \frac{\pi_{\boldsymbol{\varphi}}\left(\mathbf{a}_t|\mathbf{s}_t\right)}{\pi_{\boldsymbol{\varphi}'}\left(\mathbf{a}_t|\mathbf{s}_t\right)}$. Subsequently, the clipped surrogate objective in PPO algorithm is formulated by

$$L_t^{\text{clip}}(\boldsymbol{\varphi}) = \hat{\mathbb{E}}_t\left[\min(r_t(\boldsymbol{\varphi})\hat{A}_t, \text{clip}(r_t(\boldsymbol{\varphi}), 1-\eta, 1+\eta)\hat{A}_t)\right], \tag{11}$$

where clip $(\cdot)$ is the clip function which prevents the new policy from going far away from the old policy by limiting the probability ratio to the interval $[1-\eta, 1+\eta]$. $\hat{A}_t$ represents the estimated advantage function, given by $\hat{A}_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \ldots + (\gamma\lambda)^{T-t+1}\delta_{T-1}$, where γ is the discount factor and λ is the trace-decay parameter. $\delta_t = r_t + \gamma V_{\boldsymbol{\phi}}(\mathbf{s}_{t+1}) - V_{\boldsymbol{\phi}}(\mathbf{s}_t)$ represents the temporal difference error, and $V_{\boldsymbol{\phi}}(\mathbf{s}_t)$ is the state value function output by the critic network with parameter $\boldsymbol{\phi}$. The critic network is updated by gradient descent, i.e. $\boldsymbol{\phi} = \boldsymbol{\phi} - \alpha\nabla_{\boldsymbol{\phi}} L(\boldsymbol{\phi})$, where $L(\boldsymbol{\phi})$ is the loss function for the critic network, which is a mean squared error function defined as

$$L(\boldsymbol{\phi}) = \mathbb{E}\left[\left(\sum_{\tau=0}^{\infty}\gamma^{\tau} r_{t+1+\tau} - V_{\boldsymbol{\phi}}(\mathbf{s}_t)\right)^2\right]. \tag{12}$$

Moreover, to encourage exploration and prevent the agent from falling into a suboptimal situation, we incorporate the entropy of the probability distribution into the loss function to increase the policy's exploration of actions. Finally, the loss function for the new policy network can be expressed by

$$L_t^{\text{PPO}}(\boldsymbol{\varphi}) = L_t^{\text{clip}}(\boldsymbol{\varphi}) + w_2 \cdot S_{\pi_{\boldsymbol{\varphi}}}(\mathbf{s}_t), \tag{13}$$

where $S_{\pi_{\boldsymbol{\varphi}}}(\mathbf{s}_t)$ is the entropy of the new policy network at time step t and w_2 denotes the entropy coefficient.

C. Invalid Action Masking

To accelerate the learning process, we adopt the invalid action masking technique [17] to exclude actions that fail to contribute to the enhancement of performance reward. Specifically, we identify two types of non-contributory actions: 1) Actions intended to distill semantic units associated with a zero-vector in their one-hot attribute representation; and 2) Actions aimed at distilling semantic units that have already been extracted and transmitted. We set the value of logits associated with these actions to $-\infty$ and normalize the probabilities of all the actions by

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=0}^{N_{\text{A}}} e^{z_j}}, \tag{14}$$

where z_i the model's output probability distribution of action $\mathbf{a}_i$. N_{A} denotes the total number of actions in the action set, which equals to 2^K in our work. The probability of sampling invalid actions is equal to 0 as $\lim_{z\to-\infty} e^z = 0$.

IV. Evaluation

A. Setup

1) Datasets: We employ the DIFFUSIONDB dataset [18] for both training and testing purposes. This dataset stands as the first extensive text-to-image prompt repository, exhibiting a wide array of syntactic and semantic attributes. We categorize each text word's semantic attribute into five types: NOUN, ADJECTIVE, STYLE[1], VERB, and OTHERS. We limit the length of the prompts to between 6 and 12 words, which reduces the dataset to approximately 0.2 million usable prompts. This dataset is divided into training and testing sets following a 9:1 ratio.

2) Experimental Settings: We employ Stable Diffusion v1-4[2] as the generation model at the receiver and base our optimization strategies on it. As analyzed in II-C, the wireless transmission delay is negligible for text prompt transmission. Therefore, we exclude this delay from our experiments. Additionally, we assume perfect transmission to eliminate the impact of transmission errors on system performance. The experiments are conducted on a server with an NVIDIA RTX 4090 GPU with 24 GB of memory. The operating system is Ubuntu 20.04 with Pytorch 2.0.1.

3) Hyperparameters: We set the total number of denoising steps M at 60 and segment the whole process every 10 denoising steps. We equate the extraction latency of distilling one text word from the source image to 5 denoising steps. The learning rate is established at 0.009. The entropy coefficient w_2 is set at 0.01, the discount factor γ is set at 0.99 and the η parameter in the clip function is fixed at 0.2.

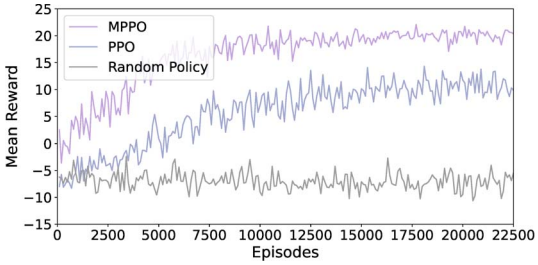

Fig. 3. Learning performance of MPPO, PPO and random policy.

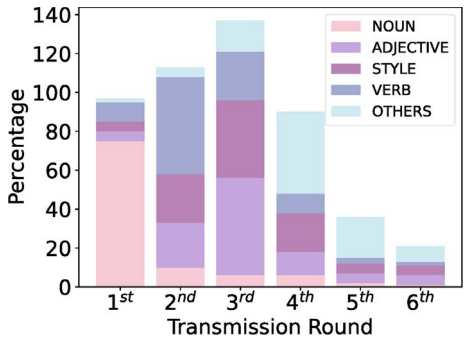

Fig. 4. Statistical distribution of semantic prompt words sent across different time sequences.

B. Results Analysis

The learning curves of the agents under different settings are shown in Fig. 3. We compare our invalid action masking-assisted PPO (MPPO) method against the conventional PPO method without masking and the random policy. It can be observed that our proposed MPPO algorithm offers notable enhancements compared to the one without masking in both convergence speed and the final values achieved. Subsequently,

[1]In our work, the style-related text words comprise 51 types of styles as detailed in [19] and words with proper noun properties.

[2]https://huggingface.co/CompVis/stable-diffusion-v1-4

Fig. 5. Visual demonstration and comparison of denoising processes: (a) Conventional GSC, conditioned on the entire prompt throughout the generation process; (b) PGSC, conditioned on prompt words received in a random sequence over time; (c) TPE-PGSC, conditioned on prompt words received in a sequence guided by TPE; (d) TPE-PGSC, with changes in an action-related prompt word; (e) TPE-PGSC, incorporating a specific style. The residual task latency is defined as the portion of task latency that extends beyond the fixed inference duration and is measured in denoising steps.

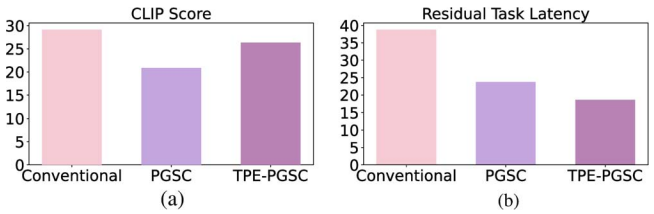

Fig. 6. Numerical results: (a) System performance, measured by CLIP score; (b) Residual task latency, measured in denoising steps.

we analyze the semantic characteristics of prompt words sent in chronological order under the MPPO method, as illustrated in Fig. 4[3]. Statistical analyses reveal that the MPPO algorithm preferentially initiates the transmission of nouns. Consequently, extraction models that target noun retrieval, such as those used for object and scene recognition, should be prioritized in execution. During the second round of transmission, action-related words are more likely to be selected, corresponding to the outputs from action detection models. During the third round of transmission, words related to style and adjectives, which share similar characteristics, are predominantly selected. This indicates that the execution of perceptual extraction models, such as those used for sentiment analysis and style detection, can be appropriately delayed. Words of other types are typically chosen for transmission at later stages due to their lesser contribution to the semantic information of the generated images. Finally, we conduct a comparative analysis of the conventional GSC, PGSC with random prompt engineering, and TPE-PGSC frameworks through both visual and numerical demonstrations presented in Fig. 5 and Fig. 6. The visual and numerical results of the PGSC reveal that while the direct implementation of PGSC is effective to reduce the system latency, it also leads to a large degree of performance loss. Through the application of TPE, we can enhance the performance of the PGSC system, achieving a 52% reduction in residual task latency while only incurring an approximate 9% decrease in CLIP score compared to the conventional GSC framework.

[3]Note that to eliminate the impact of different proportions of prompt word types in the dataset on the results, we compute the statistics based on their respective proportions over the time series. Consequently, the sum of their proportions at any given moment may exceed 100%.

V. Conclusion

In this paper, we proposed the TPE-PGSC framework, designed to balance system latency and transmission quality within the GSC context. Specifically, we devised a communication mechanism that integrates parallel semantic extraction and generation. We also implemented an RL-guided temporal prompt engineering method at the transmitter to sequence the feature extraction steps according to the distilled semantic attributes. Simulation results validated the effectiveness of our proposed framework and demonstrated its significant potential for reducing system latency.

References

[1] W. Yang *et al.*, "Semantic communications for future internet: Fundamentals, applications, and challenges," *IEEE Commun. Surveys and Tutorials*, vol. 25, no. 1, pp. 213–250, 1st Quart., 2023.

[2] W. Yang, Z. Xiong, Y. Yuan, W. Jiang, T. Q. S. Quek, and M. Debbah, "Agent-driven generative semantic communication for remote surveillance," arXiv preprint arXiv:2404.06997, 2024.

[3] Y. Wang, W. Yang, Z. Xiong, Y. Zhao, T. Q. S. Quek, and Z. Han, "Harnessing the power of AI-generated content for semantic communication," *IEEE Network*, 2024, to appear.

[4] L. Xia, Y. Sun, C. Liang, L. Zhang, M. A. Imran, and D. Niyato, "Generative AI for semantic communication: Architecture, challenges, and outlook," arXiv preprint arXiv:2308.15483, 2024.

[5] R. Zhang *et al.*, "Generative AI-enabled vehicular networks: Fundamentals, framework, and case study," *IEEE Network*, vol. 38, no. 4, pp. 259–267, 2024.

[6] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, New Orleans, LA, USA, Jun. 2022, pp. 10 684–10 695.

[7] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[8] J. Ho and T. Salimans, "Classifier-free diffusion guidance," arXiv preprint arXiv:2207.12598, 2022.

[9] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," arXiv preprint arXiv:2010.02502, 2020.

[10] G. Jocher, A. Chaurasia, and J. Qiu, "YOLO by Ultralytics," Jan. 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[11] F. Ian, "Benchmarking the Global 5G Experience — June 2023," Jun. 2023. [Online]. Available: https://www.opensignal.com/2023/06/30/benchmarking-the-global-5g-experience-june-2023

[12] A. Jon, "Stable Diffusion LoRA Training – Consumer GPU Analysis," Dec. 2023. [Online]. Available: https://www.pugetsystems.com/labs/articles/stable-diffusion-lora-training-consumer-gpu-analysis/

[13] Y. Liu *et al.*, "Optimizing mobile-edge AI-generated everything (AIGX) services by prompt engineering: Fundamental, framework, and case study," *IEEE Network*, 2024, to appear.

[14] J. Hessel, A. Holtzman, M. Forbes, R. L. Bras, and Y. Choi, "Clipscore: A reference-free evaluation metric for image captioning," arXiv preprint arXiv:2104.08718, 2021.

[15] A. Radford *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[16] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv preprint arXiv:1707.06347, 2017.

[17] S. Huang and S. Ontañón, "A closer look at invalid action masking in policy gradient algorithms," arXiv preprint arXiv:2006.14171, 2020.

[18] Z. J. Wang, E. Montoya, D. Munechika, H. Yang, B. Hoover, and D. H. Chau, "Diffusiondb: A large-scale prompt gallery dataset for text-to-image generative models," arXiv preprint arXiv:2210.14896, 2022.

[19] V. Liu and L. B. Chilton, "Design guidelines for prompt engineering text-to-image generative models," in *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, 2022, pp. 1–23.