Contrastive Language–Image Pre-Training Model-based Semantic Communication Performance Optimization

Shaoran Yang*, Dongyu Wei*, Hanzhi Yu*, Zhaohui Yang†, Yuchen Liu‡, and Mingzhe Chen*§
*Department of Electrical and Computer Engineering, University of Miami, Coral Gables, FL, 33146, USA
†College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China
‡Department of Computer Science, NC State University, Raleigh, NC, 27695, USA
§Frost Institute for Data Science and Computing, University of Miami, Coral Gables, FL, 33146, USA
Emails: {sxy632, dongyu.wei, hanzhiyu, mingzhe.chen}@miami.edu, yang_zhaohui@zju.edu.cn, yuchen.liu@ncsu.edu

***Abstract*—In this paper, a novel contrastive language–image pre-training (CLIP) model based on semantic The communication framework is designed. Compared to a standard neural network (e.g., convolutional neural network) based semantic encoders and decoders that require joint training over a common dataset, Our CLIP model-based method does not require any training procedures, thus enabling a transmitter to extract data meanings of the original data without neural network model training, and the receiver to train a neural network for follow-up task implementation without the communications with the transmitter. Next, we investigate the deployment of the CLIP model-based semantic framework over a noisy wireless network. Since the semantic information generated by the CLIP model is susceptible to wireless noise and the spectrum used for semantic information transmission are limited; it is necessary to optimize CLIP jointly model architecture and spectrum resource block (RB) allocation to maximize semantic communication performance while considering wireless noise, the delay and energy used for semantic communication. To achieve this goal, we use a proximal policy optimization (PPO) based reinforcement learning (RL) algorithm to learn how wireless noise affects the semantic communication performance, thus finding optimal CLIP model and RB for each user. Simulation results show that our proposed method improves the convergence rate by up to 40%, and the accumulated reward by 4x compared to soft actor-critic.**

I. Introduction

With the rapid development of edge devices, such as advanced computing hardware, human intelligence-driven wireless applications with new communication requirements (e.g., data rate, resilience) have emerged. However, traditional communication systems, which primarily focus on bit-level data transmission, are struggling to meet these requirements [1]. Semantic communications seems a promising solution to meet these emerging application requirements. By leveraging shared knowledge between the transmitter and receiver, semantic communications enables the extraction and transmission of the meaning of data rather than the complete raw data. Thereby significantly enhancing communication efficiency and intelligence [2]. Despite its immense potential, deploying semantic communication over current wireless networks also faces several challenges. Including efficient semantic information extraction and representation, improving robustness to transmission errors in complex environments, and secure and private semantic communication system design [3].

This work was supported in part by the U.S. National Science Foundation under Grants ECCS-2434054, CNS-2440756, and CNS-2312138.

Recently, several existing works [5]-[8] have studied the use of deep learning for the extraction of semantic information and optimization of semantic communication performance. In particular, the authors in [4] proposed a semantic communication framework that models the data semantics using a knowledge graph and jointly optimizes semantic information extraction and wireless resource management. The authors in [5] used an attention mechanism-based reinforcement learning (RL) framework to optimize semantic information extraction and wireless resource allocation strategies. In [6], the authors designed a lightweight semantic communication system based on an integrated source and channel coding scheme, and applied model sparsification and quantization to reduce transmission latency. In [7], the authors developed a transfer learning-based semantic communication framework for task-unaware and dynamic task request users. However, most of these works require training the semantic encoder and decoder for a specific user with a target follow-up task (e.g., image regeneration or classification), which demands significant time and energy.

The main contribution of this work is a novel semantic communication framework that enables 1) a transmitter to extract data meanings of the original data without neural network model training, and 2) the receiver to train a neural network for follow-up task implementation without communication with the transmitter. In particular, we consider the use of a contrastive language–image pre-training (CLIP) model as a semantic encoder to extract the data meanings of the original data. Compared to standard neural network (e.g., convolutional neural network) based semantic encoders and decoders that require joint training over a common dataset, our CLIP model-based semantic encoder does not require any training procedures thus significantly reducing the time and computing power of devices for semantic encoder deployment. Then, we consider the deployment of the CLIP model-based semantic encoder and decoder over a large-scale network that consists of multiple users and one server. Since the semantic information generated by the CLIP models is susceptible to wireless noise and the spectrum used for semantic information transmission is limited, it is necessary to jointly optimize CLIP model selection and spectrum resource block allocation to maximize semantic communication performance while considering wireless noise, the delay and energy used for semantic communication. This problem is solved by a

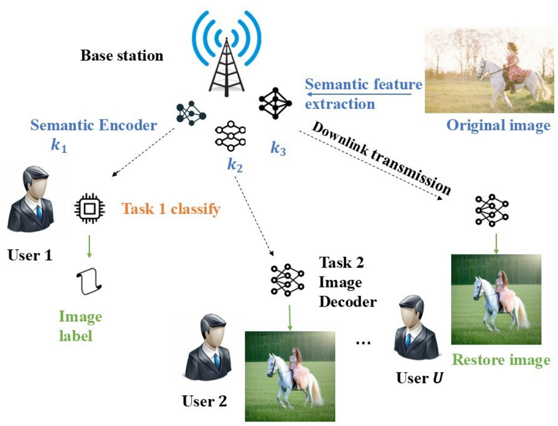

Fig. 1. Illustration of the proposed semantic communication framework.

proximal policy optimization (PPO) based RL algorithm. The simulation results show that our proposed method improves the convergence rate by up to 40%, and the accumulated reward by 4x compared to the soft actor-critic (SAC).

II. System Model

We consider a wireless network in which a base station (BS) transmits images to a set $\mathcal{U}$ of U users through semantic communication techniques, as shown in Fig. 1. In particular, the BS can select an appropriate semantic encoder to extract the feature vectors, called semantic information, from the original images according to the users' task requirements and wireless channel conditions. The users will use the received semantic information to implement the follow-up tasks (e.g., image regeneration or classification). Next, we first introduce our proposed semantic encoder and decoder. Then, we introduce the transmission and computing model for image processing and transmission. Finally, we explain our considered optimization problem.

A. Semantic Encoder

Our proposed semantic encoder is based on the CLIP model [8]. Compared to standard neural network (e.g., convolutional neural network) based semantic encoders and decoders that require joint training over a common dataset, our CLIP model-based semantic encoder does not require any training procedures, thus significantly reducing the time and computing power of devices for semantic encoder deployment. Next, we introduce the components of the CLIP model-based semantic encoder.

1) Image feature vector: The image feature extraction component of the proposed CLIP model-based semantic encoder consists of an input layer and a Transformer, as detailed below.

- Input layer: Let $\boldsymbol{x}_{\mathrm{i}}$ be the original image to be sent to the user in the BS. Each image $\boldsymbol{x}_{\mathrm{i}}$ will be divided into P patches using the patch method [9]. Then, each patch is flattened and projected into a n_{e}-dimensional embedding space with positional embeddings [10]. The n_{e}-dimensional embedding vectors from P patches will be concatenated to generate an input vector $\boldsymbol{x}_i'$.
- Vision transformer: The vision transformer is used to capture the image features. Here, we use a transformer instead of other neural networks since the transformer can use its self-attention mechanism to capture the dependencies between different parts of an image, thus enhancing the understanding of the image. Meanwhile, the transformer allows for the parallel processing of input image data, thus improving training and inference speed. The input vector $\boldsymbol{x}_i'$ is fed into a serialized transformer network that consists of n_{l} transformer layers. Each transformer layer has n_{h} paralleled attention heads. Each attention head h with its unique weights will independently extract features $\boldsymbol{f}_{\mathrm{h}}$ from the input vector $\boldsymbol{x}_i'$. The feature vectors $\boldsymbol{f}_{\mathrm{h}}$ outputted by all attention heads are then concatenated and processed through a feed-forward neural network (FNN). The output of the FNN is an image feature vector, which is also the considered semantic information.

2) Text feature vector: The text feature extraction component of the proposed CLIP-based semantic encoder consists of an input layer, a Transformer, and a global text representation, as detailed below:

- Input layer: The original text input by the user is tokenized. Then, each token is mapped to a fixed-dimensional embedding vector through an embedding layer. To preserve the sequential information of the tokens within the text, positional embeddings are added to the word embeddings, forming the final input vector.
- Text transformer: The generated input vector is fed into the text transformer for text feature extraction. The extraction process is similar to that of the vision transformer, where each layer utilizes parallel attention heads to extract corresponding feature vectors. These feature vectors are then concatenated and integrated to generate a text feature vector.

B. Image Decoder

Next, we introduce the use of semantic information (i.e., CLIP model output) for data classification and image regeneration.

1) Image classification: For image classification tasks, we assume that the image that needs to be classified is $\boldsymbol{x}$. Then, the CLIP model-based image classification process is summarized as follows:

- CLIP model for image label generation: For each category v of the images, we can generate a text vector "an image of a label v" and feed it into the CLIP model to generate the corresponding text feature vector $\boldsymbol{f}_v$. We assume that the dataset has N categories of images. Hence, we will finally generate N CLIP text feature vectors, i.e., $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_N$.
- Similarity calculation: To classify image $\boldsymbol{x}$, we first use the CLIP model to generate its image feature vector $\boldsymbol{f}_{\boldsymbol{x}}$. Then, we calculate the cosine similarity between the image feature vector $\boldsymbol{f}_{\boldsymbol{x}}$ and the text feature vectors $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_N$ of all labels.

- Image category determination: We assume that the cosine similarity between $\boldsymbol{f_x}$ and $\boldsymbol{f}_n$ is $\kappa\big(\boldsymbol{f_x}, \boldsymbol{f}_n\big)$. Then, the category of image $\boldsymbol{x}$ is determined by

$$\hat{y} = \underset{n \in \{1,\ldots,N\}}{\operatorname{argmax}} \ \kappa\big(\boldsymbol{f_x}, \boldsymbol{f}_n\big), \tag{1}$$

where $\hat{y}$ is the estimated category of the image $\boldsymbol{x}$.

2) Image regeneration: Our designed image regeneration decoder is based on a stable diffusion model, since a stable diffusion model can quickly generate high-resolution images while ensuring that the content accurately reflects user input [11]. A standard stable diffusion model is primarily used for text-to-image tasks (e.g., using text feature vectors to guide the generation of corresponding images). Here, we investigate the use of image feature vectors $\boldsymbol{f}_{\mathrm{h}}$ to guide the model to regenerate source images transmitted by the transmitter. Our designed stable diffusion model includes an image initialization module, a U-Net network [12], and a variational autoencoder, which are specified as follows.

- Image initialization module: The image initialization module is used to generate a latent space representation of an image. This latent space representation matrix serves as the basis for generating source images transmitted by the transmitter. During the training stage, the input of the image initialization module is the source image $\boldsymbol{x}_{\mathrm{i}}$, and a latent space representation $\boldsymbol{L}$ is generated through the Variational Autoencoder(VAE) encoding module. During the implementation stage, we generate a random Gaussian noise matrix $\boldsymbol{L}$ with dimensions of $64 \times 64 \times 4$, to approximate the latent space representation of an image.
- Denoising Diffusion Probabilistic Model(DDPM) scheduler: The DDPM scheduler controls the forward diffusion of the latent space representation $\boldsymbol{L}$, where noise is incrementally added to the latent space representation $\boldsymbol{L}$ over T time steps. and generates T noisy latent space representations $\hat{\boldsymbol{L}}_1, \ldots, \hat{\boldsymbol{L}}_T$. In the reverse diffusion process, DDPM controls how much noise is removed at each step corresponding to the forward process.
- U-Net: U-Net is used to generate the latent space representations of images by progressively denoising. During the training stage, the input of U-Net is are noisy latent space representation $\hat{\boldsymbol{L}}_t$ at the current time step, the source image $\boldsymbol{x}_{\mathrm{i}}$ as label, and the image feature vector (i.e., $\boldsymbol{f}_{\mathrm{h}}$). Guided by the image feature vector $\boldsymbol{f}_{\mathrm{h}}$, U-Net denoises the image at the current time step t to generate the noisy latent space representation $\hat{\boldsymbol{L}}_{t-1}$ for the next time step $t-1$. By repeating this process for T iterations, the model progressively removes noise from the image, ultimately obtaining the fully denoised high-quality latent space representation of the image $\boldsymbol{Z}$. During the inference process, the input of the U-Net are noisy latent space representation $\hat{\boldsymbol{L}}_t$ at the current time step, and the image feature vector (i.e., $\boldsymbol{f}_{\mathrm{h}}$).
- Variational autoencoder decoder: The VAE decoder is used to transform the output of U-Net to the source image that is transmitted by the transmitter. Hence, the input of VAE decoder is the latent space representations of image $\boldsymbol{Z}$ while the output is the regenerated source image $\hat{\boldsymbol{x}}_{\mathrm{i}}$.

C. Transmission Model

In our transmission model, the orthogonal frequency division multiple access (OFDMA) protocol is used for semantic information transmission from the transmitter to users. Assume the BS has a set of orthogonal downlink resource blocks (RB) that need to be allocated according to user requirements. For each user i , the RB allocation can be represented by an allocation vector $\boldsymbol{\alpha}_i = [\alpha_{i,1}, \ldots, \alpha_{i,q}, \ldots, \alpha_{i,Q}]$, where $\alpha_{i,q} \in \{0,1\}$ with $\alpha_{i,q} = 1$ indicating that RB q is allocated to user i, and $\alpha_{i,q} = 0$, otherwise. The transmission data rate of each user i is

$$c_i\left(\boldsymbol{\alpha}_i\right) = \sum_{q=1}^{Q} \alpha_{i,q} W \log_2\left(1 + \frac{P\phi_i}{I_q + WN_0}\right), \tag{2}$$

where $\alpha_{i,q}$ is the RB allocation index of user i, Q is the number of RB, W is the bandwidth of each RB, P is the transmission power of each user, I_q is the interference of RB q caused by the BSs in other service areas, and N_0 is the noise power spectral density. The channel gain between the BS and user i is $\phi_i = \gamma_i d_i^{-2}$, where γ_i is the Rayleigh fading parameter, and d_i is the distance between the BS and user i.

D. Time Consumption Model

1) Semantic Information Extraction Delay Model: The time required for the BS to extract the semantic information is

$$l_i^{\mathrm{B}}\left(k_i\right) = \frac{\omega^{\mathrm{B}} D_i^{\mathrm{X}} D_{k_i}^{\mathrm{M}}}{f^{\mathrm{B}}}, \tag{3}$$

where f^{B} is the frequency of the central processing unit (CPU) clock of each BS, ω^{B} is the number of CPU cycles required for computing data (per bit). D_i^{X} is the data size of the images X that the BS needs to extract semantic information, $k_i \in \{0,1,2\}$ is a CLIP model selection index, $D_{k_i}^{\mathrm{M}}$ is the size of the CLIP model selected by the BS.

2) Transmission Time: Given (3), the time that the BS transmits semantic information to user i is

$$l_i^{\mathrm{T}}\left(k_i, \boldsymbol{\alpha}_i\right) = \frac{D_{k_i}^{\mathrm{O}}}{c_i(\boldsymbol{\alpha}_i)}, \tag{4}$$

where $D_{k_i}^{\mathrm{O}}$ is the data size of semantic information (i.e., output of the CLIP model).

3) User Computing Model: Each user needs to use a decoder to recover the original images using the received semantic information. The time required for user i to process this task can be expressed as

$$l_i^{\mathrm{L}}\left(k_i\right) = \frac{\omega_i^{\mathrm{U}} D_{k_i}^{\mathrm{O}} D^{\mathrm{E}}}{f_i^{\mathrm{U}}}, \tag{5}$$

where f_i^{U} is the frequency of the CPU clock of user i, ω_i^{U} is the number of CPU cycles required for computing the data (per bit) of user i, D^{E} is the size of the decoder model selected by the user i.

4) Total time: Given (3),(4) and (5), the entire processing time can be expressed as

$$l_{total}\left(k_i,\boldsymbol{\alpha}_i\right)=l_i^{\mathrm{B}}\left(k_i\right)+l_i^{\mathrm{T}}\left(k_i,\boldsymbol{\alpha}_i\right)+l_i^{\mathrm{L}}\left(k_i\right), \tag{6}$$

E. Energy Consumption Model

Next, we introduce the energy consumption of the BS extracting and transmitting semantic information, and each user regenerating original images.

1) BS energy consumption: The energy consumption of the BS extracting semantic information for user i is expressed as

$$e_i^{\mathrm{B}}\left(k_i\right)=\zeta_{\mathrm{B}}\left(f^{\mathrm{B}}\right)^2 D_i^{\mathrm{X}} D_{k_i}^{\mathrm{M}}+P l_i^{\mathrm{T}}\left(k_i\right), \tag{7}$$

where ζ_{B} is the BS energy consumption coefficient. $\zeta_{\mathrm{B}}\left(f^{\mathrm{B}}\right)^2 D_i^{\mathrm{X}} D_{k_i}^{\mathrm{M}}$ is the energy consumption of the CLIP model extracting semantic information. $P l_i^{\mathrm{T}}\left(k_i\right)$ is the energy consumption of transmitting semantic information to user i.

2) User energy consumption: The energy consumption of user i regenerating the original images can be expressed as

$$e_i^{\mathrm{L}}\left(k_i\right)=\zeta_i\left(f_i^{\mathrm{U}}\right)^2 D_{k_i}^{\mathrm{O}} D^{\mathrm{E}}, \tag{8}$$

where ζ_i is the user device energy consumption coefficient.

Given (7) and (8), the total energy consumption of the BS and user i is

$$e_{total}\left(k_i\right)=e_i^{\mathrm{B}}\left(k_i\right)+e_i^{\mathrm{L}}\left(k_i\right), \tag{9}$$

F. Problem Formulation

Given the system model, our goal is to optimize the follow-up task performance (e.g., image regeneration or data classification) while meeting the delay and energy consumption requirements. This problem involves the optimization of RB allocation, the selection of the CLIP model-based encoder. Let $f\left(\boldsymbol{\alpha}_i,k_i\right)$ be the performance of the follow-up task of user i. In particular, if the follow-up task is image regeneration, $f\left(\boldsymbol{\alpha}_i,k_i\right)$ will be the CLIP Image-to-Image Similarity between the raw images and the images regenerated by the user. If the follow-up task is data classification, $f\left(\boldsymbol{\alpha}_i,k_i\right)$ is the classification accuracy, i.e., $f\left(\boldsymbol{\alpha}_i,k_i\right)=\min_{\boldsymbol{\alpha}_i,k_i}\sum_{i=1}^{U}\sum_c \mathbb{1}\{y_i=c\}\log\left(\hat{y}_i^c\right)$. Given these definitions, the optimization problem is formulated as

$$\min_{\boldsymbol{\alpha}_i,k_i}\sum_{i=1}^{U} f\left(\boldsymbol{\alpha}_i,k_i\right), \tag{10}$$

$$\text{s.t.}\quad \alpha_{i,q}\in\{0,1\},\quad k_i\in\{0,1,2\},\quad \forall i\in\mathcal{U},\,\forall q\in\mathcal{Q}, \tag{10a}$$

$$\sum_{q=1}^{Q}\alpha_{i,q}\leq 1,\ \forall i\in\mathcal{U}, \tag{10b}$$

$$\sum_{i=1}^{U}\alpha_{i,q}\leq 1,\ \forall q\in Q, \tag{10c}$$

$$l_{\text{total}}\left(k_i,\boldsymbol{\alpha}_i\right)\leq D, \tag{10d}$$

$$e_{\text{total}}\left(k_i\right)\leq E, \tag{10e}$$

where $\hat{\boldsymbol{x}}_i\left(k_i\right)$ is the image regenerated by user i, $\mathcal{Q}$ is the set of RBs that the BS can allocate to the users, D is the maximum delay allowed by the system, E is the maximum energy consumption allowed by the system for each data transmission. The constraints from (10a) to (10c) guarantee that each user can only occupy one RB and each RB can only be allocated to one user for semantic information transmission. The constraint in (10d) is a delay requirement of semantic information transmission. The constraint in (10e) is the energy consumption requirement for semantic communications.

III. Proposed Solution

To solve problem (10), we propose a proximal policy optimization (PPO) based RL algorithm algorithm. Compared to other RL algorithms, PPO has the advantages of high computational efficiency and stable convergence since it improves training robustness by clipping the objective function to prevent overly large policy updates. Next, we will first introduce the components of the proposed PPO algorithm and then explain its training process.

A. Components of the PPO Algorithm

Our proposed PPO model consists of the following components: 1) agent, 2) state, 3) action, 4) policy, 5) reward function, which are detailed as follows:

1) Agent: The agent is the BS which needs to determine the semantic encoder and the RB for each user in order to minimize the objective function in (10).

2) State: The state is used to describe the current network status under which the BS must determine the values of the variables k_i and $\boldsymbol{\alpha}_i$. Hence, a state of the BS includes: 1) interference over each RB, $\boldsymbol{I}=[I_1,\ldots,I_Q]$, 2) user location vector $\boldsymbol{p}=[\boldsymbol{p}_1,\ldots,\boldsymbol{p}_U]$, and 3) available RBs that can be allocated to the users, which is represented by a vector $\boldsymbol{\nu}=[\nu_0,\nu_1,\nu_2,...,\nu_Q]$ with $\nu_k\in\{0,1\}$ indicating whether RB k has been allocated to the users with $\nu_k=1$ indicating that RB k has been allocated to the users, otherwise, we have $\nu_k=0$. Given these definitions, each state of the BS at time slot t is $\boldsymbol{s}_t=[\boldsymbol{I},\boldsymbol{p},\boldsymbol{\nu}]$.

3) Action: The action of the BS is to determine the CLIP model used by the BS for semantic information extraction, and the appropriate RBs for image transmission. Hence, at time slot t, an action of the BS can be represented by a vector $\boldsymbol{a}_t=[k_i,\boldsymbol{\alpha}_i]$. Here, at each step, the BS only determines the semantic encoder and RB for only one user. Hence, the BS needs to implement U steps to determine the semantic encoder and RB for all U users.

4) Policy: The policy of the BS is the conditional probability of the BS choosing action $\boldsymbol{a}_t$ based on state $\boldsymbol{s}_t$. The policy is approximated by a deep neural network (DNN) parameterized by $\boldsymbol{\varphi}$. The input is the state $\boldsymbol{s}_t$, and the output is the action probability distribution. In our problem, the policy network describes the relationship among the semantic encoder model selection, RB allocation, transmission delay, energy consumption, and data transmission quality. The conditional probability of the BS taking action $\boldsymbol{a}_t$ in state $\boldsymbol{s}_t$ is $\boldsymbol{\pi}_{\boldsymbol{\varphi}}\left(\boldsymbol{a}_t|\boldsymbol{s}_t\right)$.

Algorithm 1 Training Process of the Proposed PPO Algorithm

Input: Image vector $\boldsymbol{f}_h$ required to transmit to each user, delay threshold D, energy consumption threshold E, and interference I_q of each RB.
Initialize: Parameters $\boldsymbol{\theta}^*$ generated randomly, semantic information extraction model, text recovery model, penalty coefficient λ, threshold τ, coefficient η.
repeat
  Store the policy $\boldsymbol{\pi}_{\boldsymbol{\theta}^*}(\boldsymbol{s}, \boldsymbol{a})$ and collect W trajectories $\mathcal{W} = \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_W\}$ using $\boldsymbol{\pi}_{\boldsymbol{\theta}^*}(\boldsymbol{s}, \boldsymbol{a})$.
  for $t = 1$ to T **do**
    Update the parameters of the policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})$ based on (14).
  end for
  Update the penalty coefficient λ.
until the objective function defined in (13) converges.

5) Reward: The reward function $r(\boldsymbol{s}_t, \boldsymbol{a}_t)$ is used to evaluate the state-action pairs during the entire RL implementation period from time slot 1 to U. Thus, the reward function of the BS at step t is

$$r(\boldsymbol{s}_t, \boldsymbol{a}_t) = \begin{cases} 0, & t = 1, \ldots, U-1, \\ \sum_{i=1}^{U} \sum_c \mathbb{1}\{y_i = c\} \log(\hat{y}_i^c) & \\ -\lambda_{\mathrm{D}} \sum_{i=1}^{U} \mathbb{1}_{\{l_{\text{total}}(k_i, \boldsymbol{\alpha}_i) > D\}} & \\ -\lambda_{\mathrm{E}} \sum_{i=1}^{U} \mathbb{1}_{\{e_{\text{total}}(k_i) > E\}}, & t = U, \end{cases} \tag{11}$$

where $\sum_{i=1}^{U} \sum_c \mathbb{1}\{y_i = c\} \log(\hat{y}_i^c)$ is the sum of the classification cross-entropy loss of all users with $\hat{y}_i^c$ being the estimated probability of data sample belonging to class c and $\mathbb{1}_{\{\}}$ is an indicator function. When $l_{\text{total}}(k_i, \boldsymbol{\alpha}_i) > D$, $\mathbb{1}_{(l_{\text{total}}(k_i, \boldsymbol{\alpha}_i) > D)} = 1$, otherwise, we have $\mathbb{1}_{(l_{\text{total}}(k_i, \boldsymbol{\alpha}_i) > D)} = 0$, λ_{D} is the delay penalty coefficient, and λ_{E} is the energy consumption penalty coefficient.

B. PPO Training

Next, we introduce the training process of the PPO method.

1) Training of the policy neural network: The expected reward optimized by the PPO algorithm is

$$\bar{A}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{a} \sim \boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})}(R(\boldsymbol{a}|\boldsymbol{s})) \simeq \frac{1}{W} \sum_{w=1}^{W} R(\boldsymbol{a}_w^*|\boldsymbol{s}) \frac{\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a}_w^*)}{\boldsymbol{\pi}_{\boldsymbol{\theta}^*}(\boldsymbol{s}, \boldsymbol{a}_w^*)} \tag{12}$$

To optimize the policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})$, we introduce a penalty term to control the difference between the new and old policies

$$\max_{\boldsymbol{\theta}} J(\boldsymbol{\theta}), \tag{13}$$

where $J(\boldsymbol{\theta}) = \bar{A}(\boldsymbol{\theta}) - \lambda f_{KL}(\boldsymbol{\pi}_{\boldsymbol{\theta}^*}(\boldsymbol{s}, \boldsymbol{a}), \boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a}))$ with λ being the penalty coefficient, and $f_{KL}(\boldsymbol{\pi}_{\boldsymbol{\theta}^*}(\boldsymbol{s}, \boldsymbol{a}), \boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a}))$ represents the Kullback–Leibler divergence (KLD), which measures the difference between the $\boldsymbol{\pi}_{\boldsymbol{\theta}^*}(\boldsymbol{s}, \boldsymbol{a})$ and $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})$ policies. During each iteration t, the policy $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{s}, \boldsymbol{a})$ is refined through the standard gradient ascent approach to minimize the total cross-entropy loss. The corresponding policy update rule is given by

$$\boldsymbol{\theta}^{(t)} \leftarrow \boldsymbol{\theta}^{(t-1)} + \delta \nabla_{\theta} J(\boldsymbol{\theta}), \tag{14}$$

where $\boldsymbol{\theta}^{(t)}$ is the parameters of the policy at iteration t, δ is the learning rate. By iteratively updating the policy until the proposed PPO algorithm converges, the policy for RB allocation and semantic encoder model selection that minimizes the task training loss can be obtained [13]. The training process of the proposed method is summarized in **Algorithm 1**.

TABLE I. Simulation Parameters

Parameters	**Value**	**Parameters**	**Value**
D (ms)	200	E (J)	20
Q	10	P (W)	0.2
α	2	N_0 (W/MHz)	4×10^{-15}
λ_D	1	λ_E	1
W (MHz)	20	f^{B} (GHz)	$2.5-3.5$
ω^{B} (cycles/bit)	$500-1000$	I_q (W)	$10^{-9}-10^{0}$
CLIP models		CLIP–ViT–B/32 CLIP–ViT–B/16 CLIP–ViT–L/14	

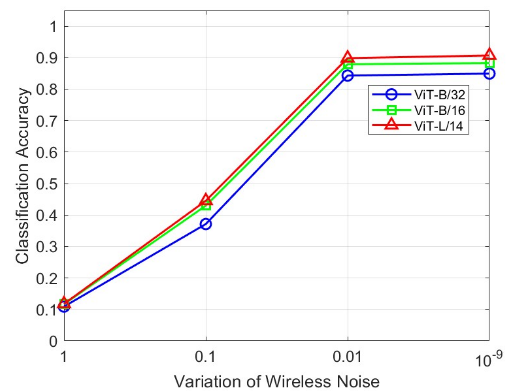

Fig. 2. The classification accuracy of different CLIP models under various noise conditions

IV. Simulation Results and Analysis

In our simulations, we consider a wireless network with one base station (BS) and $U = 5$ uniformly distributed users. We use the CLIP models in [8] and the stable diffusion model in [1]. The image dataset used to train our proposed PPO algorithm is Open Image V6 [14]. Other simulation parameters are shown in Table I. For comparisons, we compare the proposed method with SAC in [15] and Deep Q-Network(DQN).

Fig. 2 shows how the classification accuracy resulting from different CLIP models changes as the channel noise varies. From Fig. 2, we see that the CLIP-ViT-L/14 model achieves higher classification accuracy compared to the other two CLIP models. This is because the CLIP-ViT-L/14 model consists of more neural network parameters, thus extracting more features from the original image, and having stronger robustness against noise compared to the other two models.

Fig. 3 shows the images generated by our designed stable diffusion model using the text feature vectors extracted by different CLIP models. From this figure, we see that the image generated based on the semantic features extracted from CLIP-VIT-L/14 has a better reconstruction quality compared to the images generated using the semantic features extracted by the other two CLIP models. This is because CLIP-VIT-L/14 has a stronger model capacity, a larger receptive field, and a better text-vision alignment compared to the other two CLIP models. Hence, CLIP-VIT-L/14 can extract more image features to guide stable diffusion for generating higher-quality images.

Fig. 3. CLIP models for image regeneration.

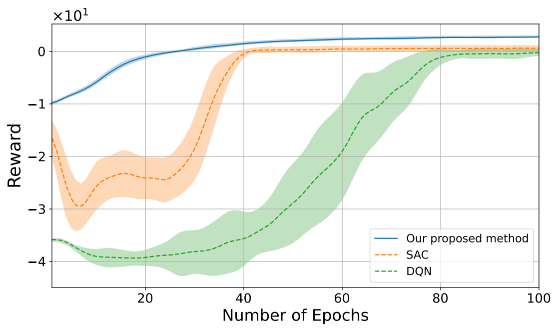

Fig. 4. Convergence of considered algorithms.

Fig. 4 shows the convergence of the considered algorithms during the training process. From this figure, we see that our proposed method can improve the convergence rate by up to 40%, and the accumulated reward by 4x compared to SAC when the number of epochs is over 100. This is because PPO improves the RL training robustness by clipping the objective function to prevent overly large policy updates.

V. Conclusion

In this paper, we have designed a novel CLIP model-based semantic communication framework. The proposed framework enables a transmitter to extract data meanings of the original data without neural network model training, and the receiver to train a neural network for follow-up task implementation without the communications with the transmitter. Then, we have investigated the deployment of the CLIP model-based semantic framework over a large-scale network that consists of multiple users and one server. Since the semantic information generated by the CLIP model is susceptible to wireless noise and the spectrum used for semantic information transmission is limited, it is necessary to jointly optimize the CLIP model architecture and spectrum resource allocation to maximize semantic communication performance while considering wireless noise, the delay and energy used for semantic communication. To achieve this goal, we have used a PPO-based RL algorithm to learn how wireless noise affects the semantic communication performance thus finding optimal CLIP model and RB for each user. Simulation results show that the designed semantic communication framework yields significant improvements in performance compared to existing methods.

References

[1] Y. Shi, K. Yang, T. Jiang, J. Zhang, and K. B. Letaief, "Communication-efficient edge AI: Algorithms and systems," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2167–2191, July 2020.

[2] X. Peng, Z. Qin, D. Huang, X. Tao, J. Lu, G. Liu, and C. Pan, "A robust deep learning enabled semantic communication system for text," in *Proc. the IEEE Global Communications Conference*, Rio de Janeiro, Brazil, December 2022, IEEE, pp. 2704–2709.

[3] T. M. Getu, G. Kaddoum, and M. Bennis, "Semantic communication: A survey on research landscape, challenges, and future directions," *Proceedings of the IEEE*, January 2025.

[4] P. Zhang, W. Xu, H. Gao, K. Niu, X. Xu, X. Qin, C. Yuan, Z. Qin, H. Zhao, J. Wei, et al., "Toward wisdom-evolutionary and primitive-concise 6G: A new paradigm of semantic communication networks," *Engineering*, vol. 8, pp. 60–73, January 2022.

[5] K. Niu, J. Dai, S. Yao, S. Wang, Z. Si, X. Qin, and P. Zhang, "A paradigm shift toward semantic communications," *IEEE Communications Magazine*, vol. 60, no. 11, pp. 113–119, August 2022.

[6] Y. Wang, M. Chen, T. Luo, W. Saad, D. Niyato, H. V. Poor, and S. Cui, "Performance optimization for semantic communications: An attention-based reinforcement learning approach," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 9, pp. 2598–2613, July 2022.

[7] H. Zhang, S. Shao, M. Tao, X. Bi, and K. B. Letaief, "Deep learning-enabled semantic communication systems with task-unaware transmitter and dynamic data," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 1, pp. 170–185, November 2022.

[8] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., "Learning transferable visual models from natural language supervision," in *Proc. International conference on machine learning*. PmLR, July 2021, pp. 8748–8763.

[9] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, December 2017.

[11] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proc. the IEEE/CVF conference on computer vision and pattern recognition*, New Orleans, Louisiana, USA, June 2022, pp. 10684–10695.

[12] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Medical Image Computing and Computer-Assisted Intervention*, Munich, Germany, Oct. 2015, Springer, pp. 234–241.

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

[14] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov, et al., "The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale," *International journal of computer vision*, vol. 128, no. 7, pp. 1956–1981, March 2020.

[15] D. Wei, X. Xu, Y. Liu, H. V. Poor, and M. Chen, "Optimizing model splitting and device task assignment for deceptive signal assisted private multi-hop split learning," *arXiv preprint arXiv:2507.07323*, July 2025.