Wiley
International Journal of Intelligent Systems
Volume 2025, Article ID 3750087, 22 pages
https://doi.org/10.1155/int/3750087

WILEY

Research Article

Federated Learning for Semantic Communication Based on CNNs and Transformer

Shufeng Li,[1] **Yujun Cai**,[1] **Zhaokai Deng**,[1] **Xinran Ba**,[1] **Qinghe Zheng**,[2] **Xinruo Zhang**,[3] **and Baoxin Su**[1]

[1]*The State Key Laboratory of Media Convergence and Communication, Communication University of China, Beijing 100024, China*
[2]*School of Intelligent Engineering, Shandong Management University, Jinan 250357, China*
[3]*School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, UK*

Correspondence should be addressed to Shufeng Li; lishufeng@cuc.edu.cn

Received 4 May 2024; Revised 4 May 2024; Accepted 9 July 2025

Academic Editor: Alexander Hošovský

This study focuses on the latest research advancements in the field of semantic communication. Traditional communication systems prioritize the transmission of raw data, whilst semantic communication emphasizes conveying the meaning represented by the data. However, the extracted semantic information is often ambiguous and subject to subjective evaluation. To address this problem, this study proposes a model that combines a convolutional neural network (CNN) with a Transformer, called DeepSC-CT. The model utilizes a CNN to extract semantic information from the data, followed by a Transformer model to capture spatial relationships and contextual information within the semantic content. We utilize federated learning to train the model and propose an adaptive aggregation algorithm to accelerate the convergence process. Moreover, we expand the single-modality semantic communication model to encompass multiple modalities, such as texts, audio, and images. Furthermore, this study introduces a learnable position-encoding method for the Transformer. The experimental results and visual effects of audio and image restoration demonstrate that the proposed method exhibits impressive performance and that the proposed model shows robust data restoration capabilities under various signal-to-noise ratio conditions.

Keywords: convolutional neural networks; federated learning; semantic communication; Transformer

1. Introduction

Traditional communication systems are transmission pipelines in which data are collected at the transmitter and reconstructed at the receiver. With the advent of the interconnected intelligent era [1], the wide deployment of devices has generated an unprecedented amount of multimodal data for various tasks, creating a new bottleneck in traditional communications and limiting overall performance. The problem can be solved using two approaches: (1) advancement of hardware to bolster system throughput and speed, evidenced by innovations in high-frequency communication bands, including millimeter-wave/terahertz bandwidths [2, 3], the integration of massive antenna arrays [4], and the utilization of reconfigurable intelligent surfaces [5] and (2) software improvements to optimize the use of communication resources, such as data compression [6], resource reuse [7], and the burgeoning field of semantic communications [8]. This study primarily focuses on semantic communication, an emerging communication paradigm that has advantages in handling massive amounts of data.

In traditional communication, the focus is on transmitting raw data whereas semantic communication is aimed at conveying the meaning expressed in the data (i.e., semantic information) to save communication resources. The

purpose of studying semantic communication is to address the semantic and pragmatic problems in communication systems, as defined by Weaver [9]. Existing research primarily falls into two directions: first, the subjective semantic information extraction problem, that is, how to effectively extract the semantic information of data, where scholars start from deep learning methods. Second, addressing problems at the semantic level, that is, how to convey content meaning accurately. Researchers start from the transmission requirements of communication systems and study the measurement of semantic information. For data restoration tasks, semantic communication typically involves extracting the global semantic information behind the data and restoring it based on the received semantic information. This task aims to achieve error-free data transmission under various signal-to-noise ratios to ensure the integrity and accuracy of the information, which is crucial for maintaining the reliability of data transmission systems, particularly in high-noise environments.

Meanwhile, federated learning (FL), as a form of distributed learning technology, has garnered attention for its ability to effectively safeguard user privacy. During the process of FL, only gradient parameters or weights are transmitted between devices and the server, while raw data remain locally stored. FL ensures that data remain on local devices and reduce the amount of data that needs to be transmitted during model training, while semantic communication transmits only processed semantic information and reduces the data volume during information transmission, thereby providing dual protection for user privacy and reducing the overhead of communication.

1.1. Related Works. Researchers have proposed a series of feasible semantic communication methods for different sources to address the problem of extracting subjective semantic information. Literature [10] introduces a cutting-edge architecture, referred to as the semantic-enhanced multimodal fusion network, specifically designed to unearth shared characteristics between events, which significantly advances the accuracy of fake news identification. Xie et al. introduced a text transmission methodology known as deep learning-based semantic communication (DeepSC), utilizing Transformer technology to identify semantic information at the sentence level for the first time [11]. In addition, they developed a streamlined semantic communication framework conducive to incorporating Internet of Things (IoT) devices into distributed networks. Weng et al. extended DeepSC to voice signal transmission by designing a semantic encoder–decoder DeepSC-S based on the attention mechanism [12]. Erdemir et al. concentrated their studies on a pioneering deep imaging transmission framework within the semantic communication domain, striving for simultaneous refinement of semantic and channel coding [13]. However, this method is only for images and has not been used for other data types. For image source transmission, Lee et al. proposed joint source-channel coding (JSCC) based on a convolutional neural network (CNN) to achieve image transmission in wireless channels while optimizing the semantic encoder–decoder to improve the image transmission performance [14]. Wu et al. proposed an ingenious semantic communication framework aimed at the simultaneous conveyance and categorization of images, where the terminal is capable of instantaneously delivering the classification outcomes [15]. Nevertheless, the semantic segmentation-based system has a high complexity, especially for high-definition images. Zhang et al. proposed FedPM, which enhances deep CNN models by leveraging distributed remote sensing data through prototype matching, regularizes local training, and reduces distribution divergence, while an attention-weighted aggregation scheme optimizes global model updates and sparse ternary compression minimizes communication costs [16]. Li et al. proposed FedTP, a Transformer-based FL framework that generates personalized self-attention projection matrices by learning a hypernetwork on the server, thereby improving the scalability and generalization ability of the model [17].

However, designing and implementing efficient semantic communication systems face unique challenges stemming from the nature of semantic information itself. Specifically, the nonstatistical characteristics and vagueness of semantic information distinguish it significantly from traditional information processing paradigms. The nonstatistical characteristics and vagueness of semantic information are represented from the perspective of information philosophy. Floridi et al. proposed using logical probability instead of statistical probability to describe deviations in the correctness and wrongness of semantic information and other nonstatistical problems [18]. Popper proposed using logical probability and information criteria to test semantic information and provided a mathematical description of logical probability [19]. Zadeh introduced fuzzy set theory and fuzzy events to describe the vagueness of semantic information [20, 21]. Luca et al. proposed a formula for fuzzy information entropy to measure the information content of fuzzy events.

Regarding the transmission of multimodal data, Xie et al. explored a visual question-answering scenario where one participant transmits a textual question while another sends a corresponding image. Although they introduced a multiuser semantic communication system specifically for such multimodal exchanges, their proposed model has a slightly higher computational complexity in the Chinese transmission task [22]. Furthermore, they extended their work to include a variety of user and task scenarios, unveiling a Transformer-based semantic communication framework. This framework demonstrated its effectiveness across several applications, including machine translation, image retrieval, and visual question-answering tasks [23]. Farsad pioneered a unified approach to source-channel coding for textual data, achieving compaction of sentences into uniform bit sequences within straightforward communications channels [24]. A novel system for the semantic transmission of images was investigated by Bourtsoulatze et al. focusing on the simultaneous enhancement of semantic content and channel coding [25]. Building upon Bourtsoulatze's foundation, Kurka et al. leveraged feedback from the communication channel to refine the fidelity of image reconstructions [26].

In recent years, scholars have conducted preliminary studies on the combination of FL and semantic communication to enhance the generalization and edge computing ability of the model. Yang et al. [27] proposed that FL models can be used to design channel encoders and decoders for semantic communication and that semantic information extraction from text or audio can be robust against noise. Tong et al. [28] proposed a wave-to-vector architecture-based autoencoder using CNN to extract semantic information from audio signals, while FL is implemented to improve the accuracy of semantic information extraction further. Hao et al. [29] innovated a federated semantic learning architecture aimed at cooperative training of semantic channel encoders on a multitude of devices, overseen by semantic channel decoders at a base station. While primarily addressing robustness against channel noise and efficient semantic information extraction, its formulation with information bottleneck principles offers a unique approach to federated semantic learning. This framework presents a more pragmatic solution for task-oriented communications compared with conventional distributed learning client-side approaches. Concurrently, Umberto et al. [30] exhibited unprecedented accuracy and rapid convergence in image classification and semantic segmentation tests. This is achieved by employing an FL model, which is honed through maximum margin training methods, but it may not be feasible due to communication complexity and may violate privacy requirements.

1.2. Contributions and Organization. As mentioned in Subsection A, the state-of-the-art methods either apply to a single modality or have a high complexity. Hence, it is necessary to adopt an approach that applies to different modalities while yielding low communication and computational complexity. In this paper, we propose an FL-assisted semantic communication model using a CNN and multiflow Transformers to address the aforementioned challenges, and the architecture is evaluated in various modalities, surpassing the state-of-the-art in all the tested modalities. The main research content includes the following: first, the proposal of a new position-encoding method that helps Transformers better capture contextual relationships; second, the optimization of the performance of single modality semantic communication models, focusing on combining the current mature deep learning models to enhance overall performance. The main contributions of this study are as follows:

1. This study introduces a novel approach that synergizes CNN and Transformer, known as DeepSC-CT. The CNN functions as a detector of semantic elements, grasping the spatial features within the data, which translates them into a cohesive sequence of embeddings. This sequence is subsequently processed through the Transformer to capture long-distance dependencies in the semantic information. By integrating these components, the system enhances its ability to interpret the intricacies of spatial and contextual data, leading to a significant improvement in model performance.
2. An innovative adaptive position-encoding method is designed, allowing dynamic adaptation based on the specific tasks and characteristics of the dataset, thereby more effectively capturing positional information in the sequence. Based on extensive simulation results, the proposed model outperforms traditional deep learning models and enhances the robustness of the system under low signal-to-noise ratio conditions. Compared with the existing literature [31], we extend the system model to several modalities, in which the data distributions have a great difference among modalities.
3. We integrate FL with semantic communication since FL enables model training across multiple devices while preserving privacy. We propose an adaptive weight-averaged training and aggregation algorithm. The core mechanism uses the accuracy of each client model in the training dataset as a weight to adjust its influence on the average global model parameters. Simulation results demonstrate that the proposed algorithm significantly outperforms the classic FedAvg algorithm and shows competitive or superior performance compared with leading approaches that integrate FL with semantic communication, such as FedSem [29] (which represents a recent advance in federated semantic learning) and FedMargin [30] (focused on maximum margin training) in terms of both robustness and communication efficiency.

The remainder of this paper is organized as follows. Section 2 introduces the system model. Section 3 presents numerical results to demonstrate the performance of the proposed model. Section 4 concludes the paper.

2. System Model

In this section, the overall architecture of the model is presented, as illustrated in Figure 1. This design combines CNN's ability to extract local features of images with Transformer's advantages in processing global information, enabling the model to better understand and generate content in different modalities. The client devices collaboratively learn a global and robust encoder by uploading local encoder model updates to the FL Server, which then performs model aggregation and distributes the aggregated global encoder model back to the clients. The primary objective of the framework is to reconstruct the original input data as accurately as possible at the decoder output. This is evidenced by the "Reconstruction Loss Calculation" module shown on the right, which computes the mean squared error (MSE) between the reconstructed output and original input, consistent with a reconstruction task aiming to minimize error. The rest of this section is structured as follows: Subsection A shows the preprocessing of each modality of the data. Subsections B–D provide a detailed description of the semantic encoding part of the model. Subsection E introduces the FL training process. The decoding part of the model is essentially the inverse of encoding, with the only difference being that for the text modality, and a linear layer is used at the last stage of restoration.

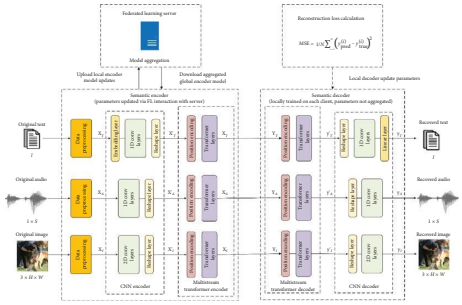


Figure 1: Framework of the proposed semantic communication systems based on DeepSC-CT.

2.1. Preliminaries. In deep learning models, data preprocessing is a key step that can effectively solve the problem of data format inconsistency and improve the performance of the model. The data preprocessing methods for texts, audio, and images are introduced as follows:

1. For text data, vocabulary construction is the key to data preprocessing. In the subsequent FL setup, we employ a predefined global vocabulary that is shared among all clients and the central server. This vocabulary is established before the commencement of the federated training, adopted from a standard pre-trained language model, that is, Word2Vec, and includes its special tokens. During the data preprocessing stage on each client's local data, text is tokenized into sentences and then words. Each word is subsequently converted into its corresponding index using this shared global vocabulary. If a word from a client's local data is not found in the global vocabulary, it is mapped to the "0" (unknown words) index. To ensure uniform input sequence length for the model, sentences are padded with the "1" index (or truncated) to a predefined maximum length. This process results in each sentence being represented as a one-dimensional tensor of indices.
2. For audio data, the key to data preprocessing is to convert the audio data into tensors. Initially, raw audio data are read from the audio files. Raw audio typically exists in waveforms, meaning it is a digital representation of the sound wave intensity over time. These data are typically stored as discrete sample points, each representing the sound wave intensity at a specific time. If the original audio is stereo (dual channel) or multichannel, it usually needs to be converted into mono. This can be achieved by averaging multiple channels or selecting only one channel. This operation is necessary because the audio dataset may contain mono, stereo (dual channel), and multichannel audio. This step can be omitted for audio that is already mono. To ensure the consistency of the input data, all audio is converted into the same format. Since neural networks require fixed-size inputs, it is necessary to ensure that each audio sample has a consistent length. If an audio sample is shorter than the set value, silence (i.e., zero padding) is added to reach the preset length. Length-standardized audio samples are then converted into tensors, which is the standard data format for deep learning models.
3. For image data, the key preprocessing steps include adjusting the image size, converting the data formats, and standardizing the pixel values of the images. First, all images are resized to a uniform size, which is crucial to ensure that the neural network can effectively process all images from the dataset. The image data are then converted from their original format into a tensor format, which is the standard data type for deep learning models. For color images containing three color channels (red, green, and blue), the pixel values of each channel are typically between 0 and 255. Each pixel value can be converted into a floating-point number by dividing it by 255. This step standardizes the input data; normalizing pixel values to a standard range (e.g., 0.0–1.0) helps maintain the consistency of input features, making the model easier to learn. Finally, the images are standardized, adjusting the pixel values to achieve a specific mean and standard deviation. This step helps the model converge faster and improves its image processing performance.

2.2. CNN Encoder. As shown in Figure 1, after data preprocessing, data from different modalities are passed through a CNN to extract semantic information. This subsection provides a detailed introduction to the semantic information extraction process for data obtained from various modalities.

1. Once the text data are preprocessed and converted into sequences of word indices (representing a one-dimensional tensor of indices), the next step is to transform these indices into dense vector representations suitable for a one-dimensional CNN model. This transformation is handled by an embedding layer. Specifically, in the model initialization phase, an embedding layer is designed that utilizes static, pretrained word embeddings. For the pretrained word embeddings, we utilized a Word2Vec model. While standard Word2Vec models often employ higher dimensions, we intentionally selected a relatively lower embedding dimension for the resource constraints on edge devices. In the context of FL, particularly with resource-constrained client devices, a smaller embedding dimension significantly reduces the model size, memory footprint, and computational overhead per word processing. This is crucial for deployment on low-power, edge-based systems, which is a primary focus of this work. The embedding layer performs a lookup table operation: for each word index in the input sequence, it retrieves its corresponding pretrained embedding vector. These embedding vectors remain fixed throughout the training of our model, serving purely as a feature lookup. These embedding vectors remain fixed throughout the training of our model, serving purely as a feature lookup. The output of this embedding layer is a two-

dimensional tensor with shape (sequence_length and embedding_dimension). This tensor then serves as the input for the subsequent one-dimensional CNN layer, which performs convolutional operations along the sequence_length dimension, treating the "embedding_dimension" as input channels. This approach enables the model to capture complex semantic relationships between words. The mathematical expression for the embedding matrix of the embedding layer is

$$\mathbf{E} = \begin{bmatrix} x_{11} & x_{21} & \cdots & x_{v1} \\ x_{12} & x_{22} & \cdots & x_{v2} \\ \vdots & \vdots & & \vdots \\ x_{1m} & x_{2m} & \cdots & x_{vm} \end{bmatrix}, \tag{1}$$

where v represents the number of words in the vocabulary, and m represents the embedding dimension for each index in the embedding layer. Assume that one sentence in the text, when converted into a tensor, is represented as $\mathbf{x}_T = [x_a, x_b, \ldots, x_u]$, where x_a, x_b, and x_u, respectively, represent the indices of the words in the vocabulary, while a, b, and u indicate, respectively, the different positions of the corresponding indices in the vocabulary. After the lookup table operation, each index in the tensor is converted into the corresponding column vector in the embedding matrix. The mathematical expression for the transformed tensor $\mathbf{x}_T$ after this conversion is

$$\mathbf{x}_T = \begin{bmatrix} x_{a1} & x_{b1} & \cdots & x_{u1} \\ x_{a2} & x_{b2} & \cdots & x_{u2} \\ \vdots & \vdots & & \vdots \\ x_{am} & x_{bm} & \cdots & x_{um} \end{bmatrix}. \tag{2}$$

In this process, the obtained tensor is represented by $\mathbf{x}_T \in \mathbb{R}^{m \times u}$, where u is the preset sequence length. Subsequently, the model sets up multiple one-dimensional convolutional layers, whose task is to extract local features from the above tensor. This type of convolutional layer is particularly suitable for processing sequence data because it can focus on key information in a sequence while retaining the sequence order. Each convolutional layer is followed by a batch normalization layer and a rectified linear unit (ReLU) activation function, which helps improve the learning efficiency and stability of the model as well as enhance its nonlinear expressive capability. Following multiple combined layers, the number of feature channels of the tensor gradually increases with deeper layers, while the spatial dimension (sequence length) gradually decreases due to convolution and pooling operations. This design allows the network to extract increasingly abstract and comprehensive features from the input text. Finally, the resulting tensor is passed through a reshaping layer or fully connected layers for subsequent processing. This results in a semantic information tensor from $\mathbf{x}_T' \in \mathbb{R}^{u' \times m'}$ for the text modality, where u' indicates the sequence dimension, and m' indicates the feature dimension.

2. Once the two-dimensional tensor of the audio data is obtained, by segmenting the audio into overlapping frames and stacking them to form a tensor of shape (number_of_frames and frame_length), it can directly serve as an input for a one-dimensional CNN. The 1D convolution operation is thus applied along the frame_length dimension. The subsequent convolution and reshaping steps are essentially the same as those used for the text data. The first dimension gradually increases whereas the second dimension l_A decreases progressively. Upon passing through the reshaping layer, which swaps the two dimensions, the resulting tensor for the audio modality takes the form of the semantic information tensor $\mathbf{x}_A' \in \mathbb{R}^{l_A' \times c_A}$, where c_A represents the output dimension after passing through the CNN, which acts as the feature dimension of the semantic information tensor, and l_A' serves as the sequence dimension of the semantic information tensor. Following these operations, the semantic information tensor of the audio data extracted by the audio modality semantic information extraction module can be used as the input to the Transformer.
3. After preprocessing the aforementioned image modality data, based on the number of image channels and the processed image size, its tensor form of image modality data can be represented as $\mathbf{x}_I \in \mathbb{R}^{3 \times h \times w}$, where h and w represent the height and width dimensions of the image size, respectively. Subsequently, the model sets up multiple two-dimensional convolutional layers suitable for image feature extraction. These layers effectively capture the spatial relationships in the image data by applying convolutional kernels across the height and width dimensions of the image and extracting local features such as edges, textures, and shapes. Similar to the semantic information extraction modules for the first two modalities, each convolutional layer is followed by a batch normalization layer and a ReLU activation function. Following multiple combined layers, the channel count of tensor $\mathbf{x}_I$ gradually increases, while the spatial dimensions h and w decrease progressively. The resulting semantic information tensor takes the form of $\mathbf{x}_I \in \mathbb{R}^{c_I \times h' \times w'}$, where c_I indicates the output dimension of the channel count after passing through the CNN and acts as the feature dimension in the semantic information tensor. The reshaping layer in the semantic information extraction module of image modality differs from the two aforementioned modules. In this module, the reshaping layer first needs to multiply h' and w' in the semantic information tensor and set the result to l_I', which acts as the sequence dimension in the semantic information tensor. The reshaping layer then adjusts the positions of the two dimensions, ultimately resulting in a semantic information tensor from $\mathbf{x}_I' \in \mathbb{R}^{l_I' \times c_I}$ for the image modality tensor.

2.3. Positional Encoding. Position encoding is a critical mechanism that provides a model with information regarding the positions of elements in sequential data. This is crucial for understanding the overall structure and context of the model. There are mainly two kinds of methods to encode the positional representations in Transformer models. Absolute methods assign a unique position identifier to each input token, ranging from 1 up to the maximum sequence length. Each position is represented by a distinct encoding vector. This encoding vector is subsequently integrated with the input token, allowing the model to incorporate positional information. Relative position methods capture the distances between input elements and learn the relationships between pairs of tokens. This is typically achieved using a look-up table with parameters that can be learned, which interact with queries and keys within self-attention mechanisms [32]. The semantic information weighting module mentioned earlier uses a Transformer module, whose core is the self-attention mechanism. This mechanism enables the model to simultaneously focus on all elements in the sequence and process information based on their relationships. The relationship between position encoding and the Transformer module is vital. As the self-attention mechanism of the Transformer does not directly process the positional information of the elements, position encoding is required to supplement this. Without position encoding, the Transformer model is unable to distinguish the order of the elements in the sequence, which is unacceptable for most sequence-processing tasks. By adding position encoding to each element of the input sequence, the Transformer can effectively manage tasks with sequential dependencies.

Through the semantic information extraction modules for the various modalities introduced in the previous section, we obtained semantic information tensors for each modality. For ease of subsequent discussion, this is abbreviated as the mathematical expression $\mathbf{X} \in \mathbb{R}^{L\times C}$, where L represents the sequence dimension of the semantic information tensor and C represents the feature dimension of the semantic information tensor.

The basic idea of traditional position encoding entails assigning a unique code to each position in a sequence so that the model can differentiate inputs from different positions. These codes are usually generated by mathematical functions, ensuring that the code of each position is unique and distinguishable from those of other positions. In the Transformer model, traditional position encoding is implemented using sine and cosine functions. For each position pos $\in [0, L-1]$ and each dimension $i \in [0, C-1]$, the value of each dimension in the traditional position encoding is calculated using the following equation.

$$\mathbf{PE}(\text{pos}, i) = \begin{cases} \sin\left(\dfrac{\text{pos}}{10000^{2i/C}}\right), & \text{if } i \text{ is even,} \\ \cos\left(\dfrac{\text{pos}}{10000^{2i/C}}\right), & \text{if } i \text{ is odd.} \end{cases} \tag{3}$$

Although traditional position encoding methods are simple and efficient, they have certain inherent limitations. First, the traditional position encoding is predefined and does not change with the characteristics of a task or dataset, which can lead to inefficiencies and inaccuracies in certain situations. Second, these fixed encoding schemes do not consider the specific requirements of particular tasks and cannot adapt to the varying importance of position information in different tasks. In addition, when handling particularly long sequences, these traditional methods may encounter performance bottlenecks because they are typically designed for sequences of a specific length, and their effectiveness may drastically reduce when exceeding this range.

To address these problems, this section proposes a learnable position-encoding method that can treat position encoding as part of the model parameters, enabling them to be adjusted and optimized during the model training process. This adaptive method enables position encoding to change dynamically according to the specific characteristics of the task and dataset, thereby capturing positional information in the sequence more effectively. Thus, learnable position encoding can not only automatically adjust to accommodate sequences of different lengths but can also better adapt to various complex task requirements, enhancing the flexibility and performance of the model. The process is as follows.

First, a learnable encoding vector is initialized for each position. This can be achieved through a random initialization, represented as $\mathbf{P} \in \mathbb{R}^{L\times C}$. These learnable position encodings are then integrated into the input of the model and are added to the semantic information tensors output by the semantic information extraction modules in the following equation.

$$\mathbf{X}_{pe} = \mathbf{X} + \mathbf{P}. \tag{4}$$

Subsequently, during the training process of the model, these position encodings are updated through a backpropagation mechanism. The position-encoding vectors are adjusted based on the loss function and learning rate of the model to better adapt to a specific task. During the training process, the update of the position encoding $\mathbf{P}$ is given by the following equation.

$$\mathbf{P}^{(t+1)} = \mathbf{P}^{(t)} - \eta \cdot \nabla \mathscr{L}\left(\mathbf{P}^{(t)}, \Theta\right), \tag{5}$$

where $\mathbf{P}^{(t)}$ represents position encoding at the tth iteration. η indicates the learning rate. $\nabla \mathscr{L}(\mathbf{P}^{(t)}, \Theta)$ indicates the gradient of the model loss $\mathscr{L}$ with respect to the position encoding $\mathbf{P}^{(t)}$ and other model parameters Θ. Through the aforementioned self-optimization process, learnable position encoding can more accurately reflect the relative or absolute positions of each element in a sequence, thereby enhancing the adaptability and performance of the overall model for specific tasks.

2.4. Multistream Transformer. The multiflow Transformer simultaneously processes multiple tensors, as described above. The main structure contains multiple Transformers in parallel, with each Transformer containing multiple layers. Each layer performs computations using a multihead attention mechanism [31, 33], as shown in Figure 2. We use analog tensors to transmit the data stream, and the channel characteristics will be presented in Section 3.

Once the semantic information tensor obtained in the previous section enters the Transformer model, the first step is to split it into multiple heads. This is achieved by dividing the feature dimensions of the input tensor into smaller subvectors. If the input is a tensor with a feature dimension C, then the feature dimension of the tensor input to each head is C/N, where N is the number of heads. For each head, three separate linear transformations are performed using the weight matrices obtained during training. These are used to generate the Query (Q), Key (K), and Value (V), as illustrated in Equations (6a)–(6c).

$$\mathbf{Q}_h = \mathbf{X}_{pe}\mathbf{W}_h^Q, \tag{6a}$$

$$\mathbf{K}_h = \mathbf{X}_{pe}\mathbf{W}_h^K, \tag{6b}$$

$$\mathbf{V}_h = \mathbf{X}_{pe}\mathbf{W}_h^V, \tag{6c}$$

where $\mathbf{X}_{pe}$ represents the output of the position-encoding module, which serves as the input for the Transformer module. $\mathbf{W}_h^Q, \mathbf{W}_h^K$, and $\mathbf{W}_h^V$ represent the linear transformation matrices for the query, key, and value of the hth head, where $h \in [1, N]$. For each head, scaled dot-product attention calculations are performed. This involves computing the dot product of the query and key, dividing it by a scaling factor (the square root of the dimension of the key), and then applying the SoftMax function, as per equation (7), to obtain the weights. These weights are then multiplied by the values to obtain the output for each head. For the hth head, the self-attention computation is given by equation (8).

$$\text{softmax}(\alpha_i) = \frac{e^{\alpha_i}}{\sum_{j=1}^{n} e^{\alpha_j}}, \tag{7}$$

$$\begin{aligned}\text{head}_h &= \text{Attention}(Q_h, K_h, V_h)\\ &= \text{softmax}\left(\frac{Q_h K_h^T}{\sqrt{d_k}}\right)\cdot V_h,\end{aligned} \tag{8}$$

where α_i in equation (7) is a one-dimensional vector $\boldsymbol{\alpha}$, $i \in [1, n]$. It represents the calculation of the SoftMax function. In equation (8), d_k represents the key dimensions. The output of the multihead attention mechanism is obtained by concatenating the outputs of each head and multiplying the concatenated result by a linear transformation matrix, as depicted in equation (9). This output is added to the input and the output of the multihead attention mechanism, which can be represented by equation (10).

$$\text{MultiHead}\left(\mathbf{X}_{pe}\right) = \text{Concat}\left(\text{head}_1, \ldots, \text{head}_N\right)\cdot \mathbf{W}^O, \tag{9}$$

$$\mathbf{X}_{Att} = \mathbf{X}_{pe} + \text{MultiHead}\left(\mathbf{X}_{pe}\right), \tag{10}$$

where $\mathbf{W}^O$ represents the weight matrix used in the final linear transformation. Upon obtaining the above results, layer normalization is performed to normalize the distribution of each sample in the feature dimension, ensuring that different features contribute more evenly to the model. The calculation of layer normalization is shown in the following equation.

$$\mathbf{X}_{\text{LN}} = \frac{\gamma\left(\mathbf{X}_{Att} - \mu\right)}{\sqrt{\sigma^2 + \varepsilon}} + \beta, \tag{11}$$

where μ and σ represent, respectively, the mean and standard deviation of the input tensor along the sequence dimension, while ε is a negligible value to prevent zero-division errors. The learnable scaling (γ) and shifting (β) factors, which are one-dimensional and proportional to the feature size of the incoming tensor, are dynamically refined through the training phase. These factors modulate the normalized layer output to align more closely with the original input distribution. Furthermore, the model harnesses the transformed output and passes it through dual linear transformations, comprising a densely connected layer followed by a normalization phase and then activated by a nonlinear function. This process infuses nonlinearity into the model's output, granting the ability to discern intricate patterns and associations within the input data. Such enhancements amplify the model's representational power for higher-order operations, as shown in the following equation.

$$\mathbf{X}_{\text{FNN}} = \text{ReLU}\left(\mathbf{X}_{Att}\mathbf{W}_1 + b_1\right)\mathbf{W}_2 + b_2). \tag{12}$$

Through the mapping of two linear transformations using matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ and bias terms b_1 and b_2, the output results can better adapt to the input distribution. Finally, the output of the feedforward neural network (FNN) serves as the input for the next Transformer layer. The output from the last layer represents the final semantic information processed in all stages.

2.5. FL Training. In traditional centralized learning models, the source data and the label data are both stored on a central server. Model training and updating are also performed on the central server. The advantage of this approach is that a large amount of data can be used for centralized training, thereby improving the performance of the model. However, this approach also has data privacy and security issues because all data need to be transmitted to the central server.

In FL, data privacy and security issues are better addressed. The basic idea of FL is to distribute model training to various clients instead of centralizing raw data to a central server, meaning that the source data, including labels and features, remain under the control of the clients. The only information exchanged between the clients and the central server is the trained model parameters or the learned gradients. This study employs FL for training the semantic

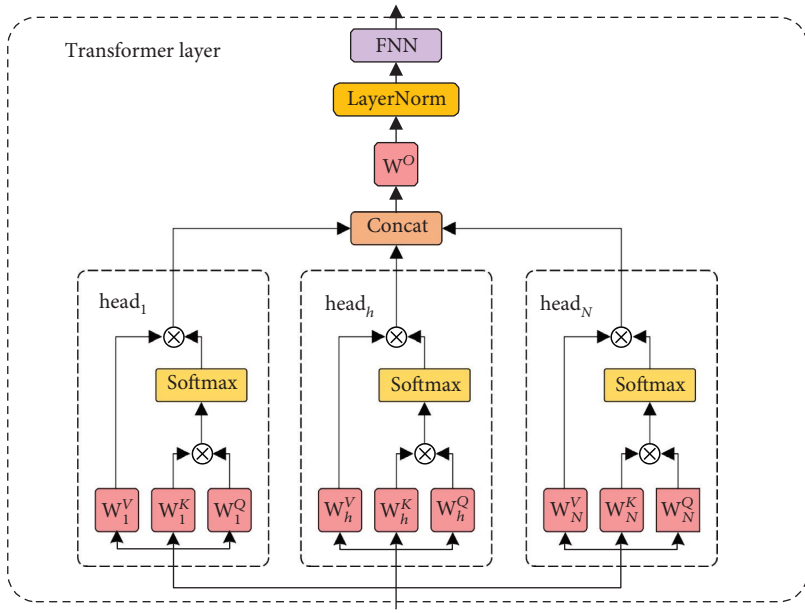


Figure 2: Workflow diagram of multihead self-attention.

encoder. This approach is premised on a collective model training regime where multiple client devices collaboratively enhance a centralized machine learning model using their respective local data sources as illustrated in Figure 3.

In the global iteration, the system adopts a universally recognized iteration mechanism known as federated averaging. During this phase, each participant carries out training on their model in isolation and subsequently uploads their model's parameters to a central processing server. The central server then integrates these individual contributions into a cohesive model by calculating the weighted mean of the parameters, which forms the basis for the updated global model. This updated model is then redistributed back to the clients [34].

The deployment of FL involves global model aggregation, which reduces the number of training epochs required by individual clients, thereby diminishing the computational demand on each client and ensuring the confidentiality of client data in a distributed computing landscape.

For the aggregation process, we propose a FL algorithm that incorporates local training accuracy and disparity compensation to enhance the generalization ability of the model and ensure effective fusion of data heterogeneity. The core of the adaptive compensation mechanism is to use the accuracy acc_k of each client model in the training dataset as a weight to adjust its influence on the global model parameter average. Accuracy is calculated using the following equation.

$$\mathrm{acc}_k = \frac{1}{|D_k|} \sum_{(x,y)\in D_k} \mathbb{I}\left(\hat{y}_k = y\right), \tag{13}$$

where $|D_k|$ is the total number of samples in dataset D_k. For each sample $(x, y) \in D_k$, $\hat{y}_k = \mathrm{argmax}(\mathscr{M}_k(x, \omega_k))$ is the predicted label obtained by applying the argmax function on the output of client k's model $\mathscr{M}_k(x, \omega_k)$. The indicator function $\mathbb{I}(\hat{y}_k = y)$ is 1 if the predicted label $\hat{y}_k$ matches the true label y and 0, otherwise. Subsequently, the weighted average for each client is determined by calculating the accuracy of the model. The weighted average formula is as follows.

$$w_k^{\mathrm{acc}} = \frac{\mathrm{acc}_k}{\sum_{k=1}^{K} \mathrm{acc}_k}, \tag{14}$$

where K indicates the total number of user models participating in FL, and w_k^{acc} indicates the local accuracy weight coefficient of the kth client.

Difference compensation is achieved by calculating the l_2 norm distance between each user's model parameter and the current global model parameter. This distance reflects the parameter differences between clients and can be expressed by the following equation.

$$d_k = \left\|\theta_k - \theta_g\right\|_2, \tag{15}$$

where θ_k represents the local model parameter of the kth client in the current iteration round, and θ_g is a global model parameter used by the client in the current iteration round. This difference d_k is then used to calculate the difference weight coefficient, which is used to adjust the contribution of each client model to the global model.

$$w_k^{\mathrm{diff}} = \frac{1}{1 + d_k}, \tag{16}$$

where w_k^{diff} denotes the difference in the weight coefficient. These two weights are integrated as the updated weight coefficient of the local model. The integration formula is as follows.

$$w_k = \lambda w_k^{\mathrm{acc}} + (1 - \lambda) w_k^{\mathrm{diff}}, \tag{17}$$

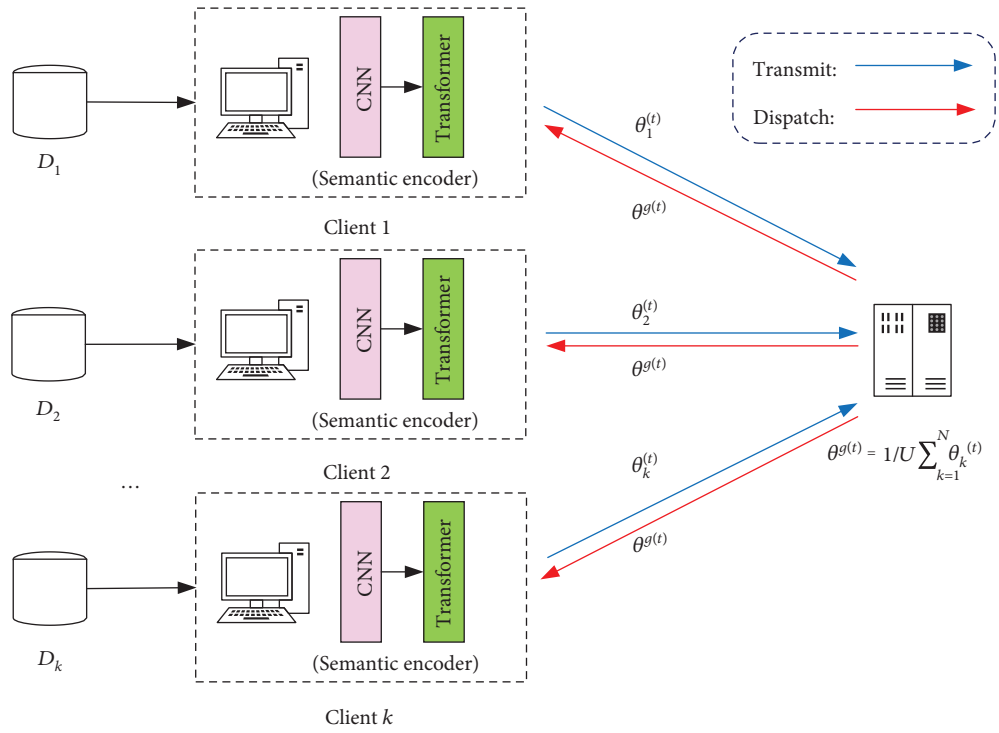


Figure 3: Federated training framework for the semantic encoder.

where λ is a hyperparameter used to balance the influence of adaptive weights and differential weights.

Finally, the integrated weight coefficients are used for weighted averaging, and the resulting global model is updated. The method is described by the following equation.

$$\widehat{\theta}_g = \sum_{k=1}^{K} w_k \theta_k. \tag{18}$$

Through the above method, the FL algorithm proposed in this study not only promotes a greater contribution of client models with better performance to the global model but also ensures that those models that are unique in the parameter space are reasonably representative, thereby enhancing the robustness of our approach. This method of integrating the local training accuracy and difference compensation provides an effective strategy for FL systems that deal with non-IID data problems. The key steps of our proposed adaptive weight-averaged FL are summarized in Algorithm 1, where $M(\cdot)$ represents the semantic communication models that correspond to different tasks and p_k represents the semantic information generated by different semantic communication models.

3. Simulation Results

In this section, the simulation results of three experiments are described. The first experiment compares the performance of the proposed adaptive FL algorithm and the classic FedAvg algorithm. The second experiment is a performance comparison of the learnable position encoding method, and the third experiment involves a performance comparison of our proposed approach against two single-modality semantic communication benchmarks from all modality data under different signal-to-noise ratios.

3.1. Simulation Settings

1. Simulation environment: the key hyperparameters, software, and hardware used for simulation are listed in Table 1.
2. Datasets: the datasets used for the experiments are tailored according to different modalities to ensure the accuracy and effectiveness of the experiments. For the text modality, we use the Stanford Sentiment Treebank (SST) and IMDb. The SST dataset contains movie reviews categorized into five sentiment levels, from the most negative to the most positive, containing approximately 11,855 sentences. The IMDb dataset contains movie reviews from the IMDb website, categorized as either positive or negative sentiments, with a total of 50,000 movie reviews, including 25,000 for training and 25,000 for testing.

   For the audio modality: the datasets used include YES–NO, SPEECHCOMMANDS, and LIBRISPEECH. The YES–NO dataset contains a series of "yes" and "no" spoken in Hebrew, with 61 audio files in the dataset. The SPEECHCOMMANDS dataset comprises recordings of simple commands with approximately 105,000 audio files. The LIBRISPEECH dataset features audio of English texts read from the LibriVox Audiobook project, containing approximately 360,000 audio files.

Initialization: θ_k, θ_g, local iteration round $\tau = 0$ with a total of a, global iteration round $t = 0$ with a total of g

Local training:

1. **Input:** Select mini-batch data from D_k
2. **When** the local model does not converge **or** $\tau < a$ **do:**
3. Through the semantic communication model: $M_{\theta_k^\tau}(\cdot) \longrightarrow p_k$
4. Calculate the loss ✓ by the semantic information generated by the model
5. Update the parameters of the model by SGD: $\theta_k^{\tau+1} \longleftarrow \theta_k^\tau \longleftarrow \eta \nabla_{\theta_k^\tau} \mathscr{L}$
6. Update the local iteration round: $\tau \longleftarrow \tau + 1$
7. **Return:** Updated local model $M_{\theta_k^\tau}(\cdot)$

Global Training:

1. **When** the global model does not converge **or** $t < g$ **do:**
2. Send the updated local model to the server
3. The server aggregated the parameters of all uploaded models: $\theta_1^t, \theta_2^t, \ldots, \theta_k^t$
4. The server calculates the weight coefficient w_k of each client by (14) to (17)
5. Update the global model parameters by (18): $\widehat{\theta}_g = \sum_{k=1}^{K} w_k \theta_k$
6. Send updated parameters to local models: $\theta_1^{t+1}, \theta_2^{t+1}, \ldots, \theta_k^{t+1} \longleftarrow \theta_g^t$
7. Update the global iteration round: $t \longleftarrow t + 1$
8. **Return:** Updated global model $M_{\theta_k^t}(\cdot)$

Algorithm 1: Adaptive FL training.

For the image modality: the datasets used include MNIST, CIFAR-10, and ImageNet-Dogs [35]. The MNIST dataset consisted of grayscale images of handwritten digits from 0 to 9, with 70,000 images: 60,000 for training and 10,000 for testing. Each image is preprocessed to a size of 32×32 pixels with one channel. The CIFAR-10 dataset contains color images of 10 different objects (cats, dogs, cars, etc.), with 60,000 images, 50,000 for training and 10,000 for testing. Each image has 32×32 pixels with three channels. The ImageNet-Dogs dataset includes color images of 120 different breeds of dogs, totaling 20,579 images, with 10,222 for training and 10,357 for testing. Each image is preprocessed to 224×224 pixels using three channels. For the 32×32 pixels images in the MNIST and CIFAR-10 datasets, we used bilinear interpolation to resize them to 224×224 pixels. This ensured that all input images had the same dimensions, making them suitable for input into the same model.

3. Model settings: for the semantic communication models of text and audio modalities, three one-dimensional convolutional layers are set up with identical internal parameters across all layers. The size of the convolution kernel is 4×4, the stride is 2, and the padding is set to one. The output channels for each layer are 8, 16, and 32, respectively. For image data, due to its inherently high dimensionality and rich spatial feature complexity, we adopt a comparatively larger and deeper CNN architecture compared to the text modality. Images typically presented as multi-channel pixel grids contain vast amounts of local and global visual information, requiring a more extensive network to effectively extract hierarchical features, such as edges, textures, and object parts. Our image CNN architecture consequently consists of a greater number of convolutional layers and filters, followed by pooling layers for spatial downsampling. Therefore, the semantic communication model is configured with five layers of two-dimensional convolutional layers, again with identical internal parameters across all layers. The size of the convolution kernel is 4×4, the stride is 2, and the padding is set to one. The number of output channels for each layer is 32, 64, 128, 256, and 512, respectively. In the multistream Transformer component, each stream contains two Transformer layers, with each layer having four heads. As for the settings of the FL part, the batch size is set to 64 while the number of global epochs is 100. Three user clusters participate in the FL, each containing 10 users, and the dataset is divided into three parts, one for each cluster. In the subsequent simulations, when referring to the proposed method, we consistently employed the proposed FL algorithm.
4. Benchmarks and performance metrics: we use both the Additive White Gaussian Noise (AWGN) channel and the Rayleigh channel to compare the performance of the different algorithms. In the experiment comparing the performance of position-encoding methods, the compared methods include the learnable position-encoding method proposed in this work, the traditional position-encoding method, and a scenario in which no position encoding is used. In the experiment comparing the performance of single-modality semantic communication models under different signal-to-noise ratios for text and audio modalities, the comparison models are LSTM [35], CNN [14], and Transformer [23]. For the image modality, the other two models compared are DeepJSCC [25] and a variational autoencoder (VAE)-based [36] semantic communication model.

Table 1: Hyperparameters, software, and hardware.

Batch size	64
Epochs	100
Learning rate	0.001
Software	Pycharm
CPU processor	12th Gen Intel (R) Core (TM) i7-12700H
GPU	RTX 3060 GPU
RAM	16 GB
Python version	3.7
Pytorch version	1.12.0
Numpy version	1.21.0
Seed	1234

We use image classification tasks to evaluate the performance of the proposed FL method. We conduct experiments on differences in data distribution. For large differences in data distribution, we let the users use different datasets, that is, MNIST, CIFAR-10, and ImageNet-Dogs. For small differences in data distribution, we let the users use the same CIFAR-10 dataset but with different parts of it.

3.2. Performance of Adaptive FL Training Algorithm. As presented in Figure 4, when client data are imbalanced, the classification accuracies of the method proposed in this paper, classic FedAvg, the FedSem algorithm [29], and the FedMargin algorithm [30] are compared, showing how they change with the number of full iterations during the training of the semantic communication model for image classification tasks. In this experiment, three user clusters participating in FL are established, each with 10 users. Each user cluster is assigned a different dataset, i.e., CIFAR-10, MNIST, and ImageNet-Dogs. The purpose is to ensure that the three clusters have an unbalanced data distribution.

For FedSem, which updates per batch, we define global epoch as the completion of one full pass through the total training data across all participating clients, equivalent to the number of batches processed multiplied by batch size and divided by total data size. This ensures a consistent measurement unit for global training progress across all methods. Although FedSem primarily focuses on robustness to noise and semantic efficiency rather than explicit non-IID handling, in our experimental setup, its application of information bottleneck principles for semantic feature extraction and the cooperative training mechanism allows it to achieve competitive performance, sometimes even outperforming basic FedAvg in certain scenarios, by learning more discriminative and compact representations.

It can be observed from the figure that the proposed aggregation algorithm shows an effective performance improvement compared with FedAvg when facing imbalanced datasets. Meanwhile, the FedSem in [29] shows a similar accuracy with the proposed algorithm when converged but has a slower convergence speed. Besides, the FedMargin method in [30] converges faster with a bit lower level of accuracy. It is important to note that these observations are based on a consistent set of hyperparameter settings applied across all compared algorithms to ensure a fair comparison under the same experimental conditions. While exhaustive hyperparameter optimization for each baseline could potentially improve their performances, our study focuses on demonstrating the effectiveness of our proposed method under a standardized configuration relevant to typical FL deployments.

In the case of non-IID data partition, relying solely on accuracy for performance evaluation may have limitations. Therefore, we also employed precision, recall, and F1-score as evaluation metrics. The performance of different methods across these metrics at a global epoch of 100 is presented in Table 2. As observed, the proposed method is slightly inferior to FedMargin in terms of recall but outperforms all other algorithms in other aspects.

As shown in Figure 5, we compare the classification accuracy of our proposed method with that of traditional FedAvg, the FedSem algorithm [29], and the FedMargin algorithm [30] when training the semantic communication model for image classification tasks. This model, originally designed for robust semantic feature extraction and used in reconstruction tasks, leverages its encoder to produce compact feature representations. These features are then fed into a newly added classification layer instead of a decoder to perform image classification. This comparison is made with balanced user data and examines how accuracy changes with the number of full iterations. In this experiment, the CIFAR-10 dataset is divided into three parts, and the three users participating in FL each use one part as their local dataset. As shown in the figure, the proposed method and other methods exhibit similar performance when the user dataset is relatively balanced.

3.3. Performance of Learnable Position Encoding. As shown in Figure 6, this section incorporates different position-encoding methods into DeepSC-CT, including the proposed learnable position-encoding method, the traditional position-encoding method, and no position encoding. Performance is evaluated by comparing the MSE of the model with the number of iterations. Among them, (a) and (b) are experiments based on text-modal semantic communication models, (c) and (d) are experiments based on audio-modal semantic communication models, and (e) and (f) are experiments based on image-modal semantic communication models. From the experimental results, the proposed learnable position-encoding method demonstrates good performance in different modality semantic communication models. In most cases, the proposed method shows enhanced convergence behavior in terms of both convergence speed and converged state. This experiment proves that the proposed position-encoding method can adapt to different modality datasets and exhibits good flexibility.

3.4. Performance of CNN-Transformer Model. From Figures 7, 8, and 9, the experimental analysis in this study presents a comprehensive assessment of the semantic communication models for three different modalities: texts, audio, and images. For text and audio modalities, the performance of the proposed model is compared against the LSTM [35], CNN [14], and Transformer [23] models whereas for the

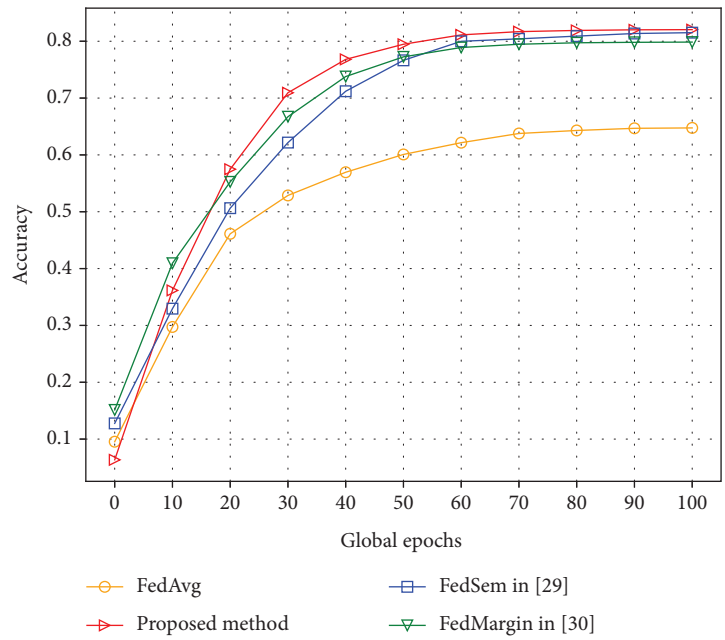


Figure 4: Performance comparison when the difference in data distribution is large.

image modality, the performance of the proposed method is compared against DeepJSCC [25] and VAE [36]. For text, MSE is used to measure the numerical difference between the predicted and actual text representations. For audio, MSE effectively quantifies the reconstruction quality by measuring the difference between the original and reconstructed audio signals, which are represented numerically. Peak signal-to-noise ratio (PSNR) is a widely used metric for assessing the quality of reconstructed images, which reflects the visual quality and the fidelity of the reconstructed image compared with the original one.

Figure 7 shows the text modality, where it is evident that the proposed model outperforms the traditional LSTM, CNN, and Transformer models in text restoration tasks. This advantage is primarily owing to the integration of CNN's local feature extraction capability and the Transformer module's global contextual awareness. In high SNR environments, the model utilizes semantic information in text data more effectively, thereby enhancing the accuracy of the restoration tasks. Even under low SNR conditions, where noise interference is present, the proposed model maintains good performance, demonstrating its robustness. Taking the IMDb dataset with the AWGN channel as an example, the MSE of the proposed method is 33.3% lower than CNN, 18.5% lower than Transformer, and 5.8% lower than LSTM when SNR is only 2 dB.

Figure 8 shows the audio modality, where the proposed model exhibits exceptional performance over the traditional LSTM, CNN, and Transformer models. The combination of audio data temporal characteristics with the ability of the CNN to extract audio features, along with the capability of the Transformer module to process global sequence characteristics, contributed to the stability and robustness of the model under various SNR values. Compared to LSTM, CNN, and Transformer, the proposed model is more effective in handling complex audio restoration tasks, particularly in Gaussian and Rayleigh channel environments. Taking the LIBRISPEECH dataset with the Rayleigh channel as an example, the MSE of the proposed method is 19.2% lower than CNN, 12.5% lower than Transformer, and 8.7% lower than LSTM when SNR is only 2 dB.

Figure 9 shows the image modality. The proposed model performs better in image restoration tasks than the DeepJSCC and VAE models. The CNN encoder preserves the spatial information while extracting image features, and the introduction of a Transformer module enhances the global perception of the model. In particular, in Rayleigh channels, the proposed model shows an improved performance owing to the multi-layer convolutional operations of the CNN encoder, which can learn more distinctive features. The proposed model performs better in low-resolution image restoration tasks, as local and global information become more closely related in low-resolution images. Taking the MNIST dataset with the CIFAR-10 channel as an example, the PSNR of the proposed method is 7.9% higher than DeepJSCC and 9.7% higher than VAE when SNR is only 2 dB.

Table 3 presents the overall MSE and an example of the text restoration. In this experiment, the proposed model performed a text restoration task on a passage from the IMDb test set and showed the original and restored sentences from that passage. The bold highlights incorrect words. The models used for comparison in the experiment are the LTSM and CNN. Table 3 shows that the proposed model demonstrates a higher degree of sentence restoration compared with the other two models and shows a lower MSE loss for the entire passage. This experiment proves that the proposed model performed better for text restoration.

Table 2: Comparison of other metrics for different methods.

Method	Precision	Recall	F1-score
FedAvg	0.67	0.60	0.63
FedSem [29]	0.80	0.82	0.81
FedMargin [30]	0.75	0.90	0.82
Proposed method	0.80	0.86	0.83

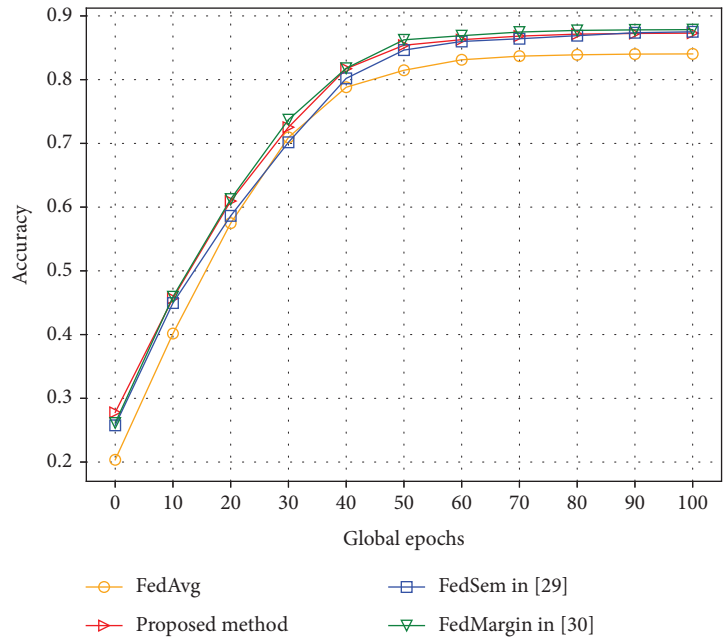


Figure 5: Performance comparison when the difference in data distribution is small.

Figure 10 shows the visual effects of the audio restoration. In this experiment, the proposed model is applied to perform an audio restoration task on a sentence from the LIBRISPEECH test set, and a spectrogram is displayed. The methods used for comparison are the LTSM, CNN, the traditional FedAvg algorithm, and the FL-based audio semantic communication method proposed in [28]. From the spectrogram, it is evident that the spectrum restored by the proposed model exhibits a higher similarity to the spectrum of the original audio with a lower MSE loss value. This experiment demonstrates that the proposed model performed better in restoring the audio.

As depicted in Figure 11, the visual output of the image restoration process is presented. The undertaken experiments utilized our novel model on a range of datasets varying in resolution and dimension. Results from these tests suggest that our approach has an enhanced ability to restore the originality of images, meticulously conserving the intricate details and structural integrity, thus offering a refined quality of image reconstruction when compared to prior methods such as DeepJSCC, VAE, and the traditional FedAvg algorithm. These findings highlight the model's robustness and adaptability across various datasets and image scales.

Table 4 presents the inference time, floating-point operations (FLOPs), and model size for the proposed method, DeepJSCC, VAE, LSTM, CNN, and Transformer on the ImageNet-Dogs dataset with a batch size of 1. The table also includes the Encoder Output Size, which represents the dimensionality of the compressed latent representation generated by the encoder before it is transmitted or passed to the decoder.

A smaller encoder output size implies a higher degree of compression of the input data into a more compact representation. This is generally desirable in scenarios like FL or semantic communication, where bandwidth and communication efficiency are critical, as it reduces the amount of data needing to be transmitted. However, excessively small output sizes risk losing crucial information, potentially leading to lower accuracy or reconstruction quality. Conversely, a larger output size retains more information, often leading to better performance, but at the cost of higher communication overhead and potentially more complex decoders. The optimal size is a trade-off that balances information preservation with communication efficiency for a given task and dataset.

Despite our proposed model having a larger model size compared with DeepJSCC, VAE, Transformer, and CNN, its FLOP count is notably lower than DeepJSCC and VAE and is also comparable to CNN and LSTM, suggesting that the model can perform efficiently with relatively limited computational resources when compared to other complex models. Regarding inference time, our proposed model's execution speed is slightly longer than DeepJSCC, CNN,

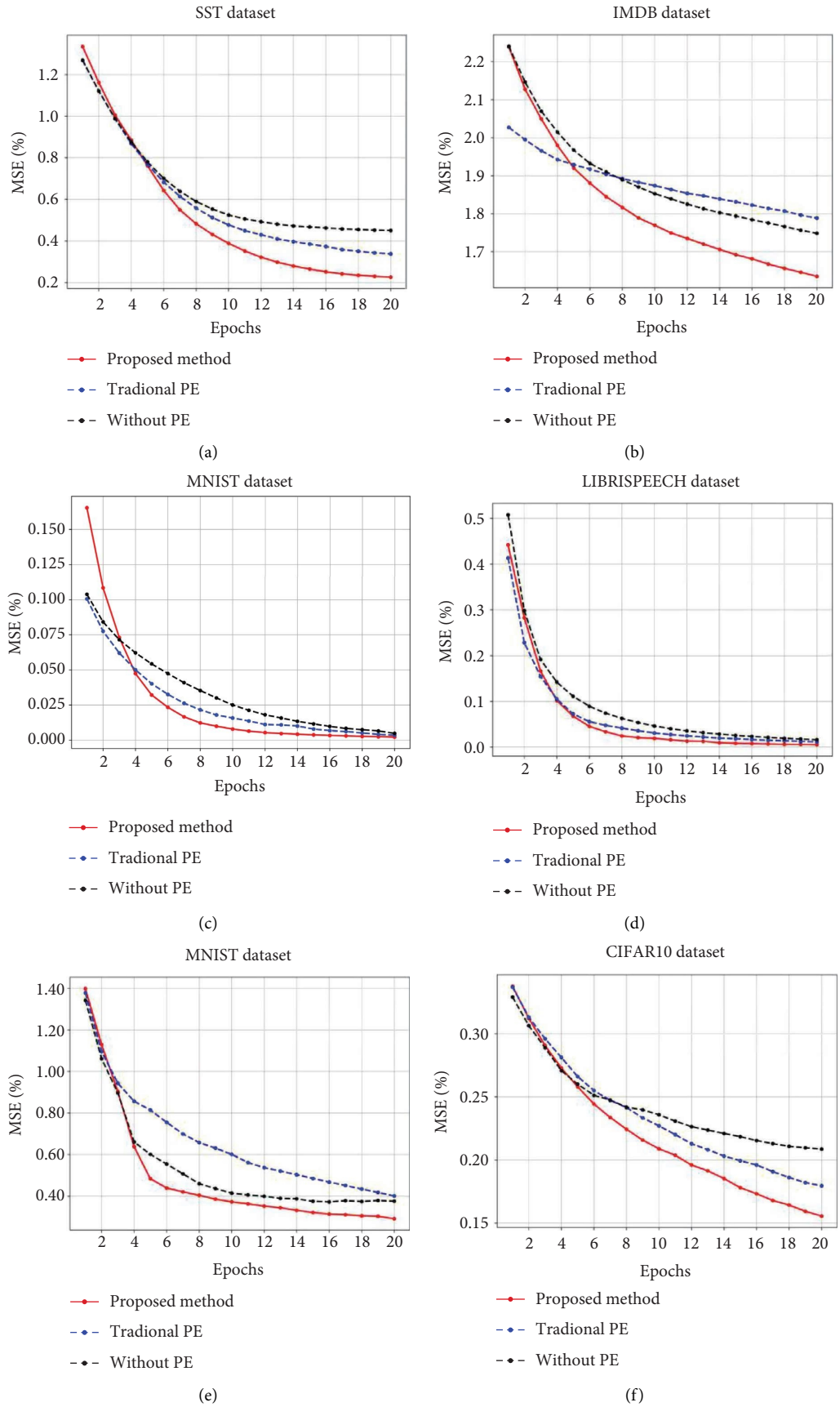
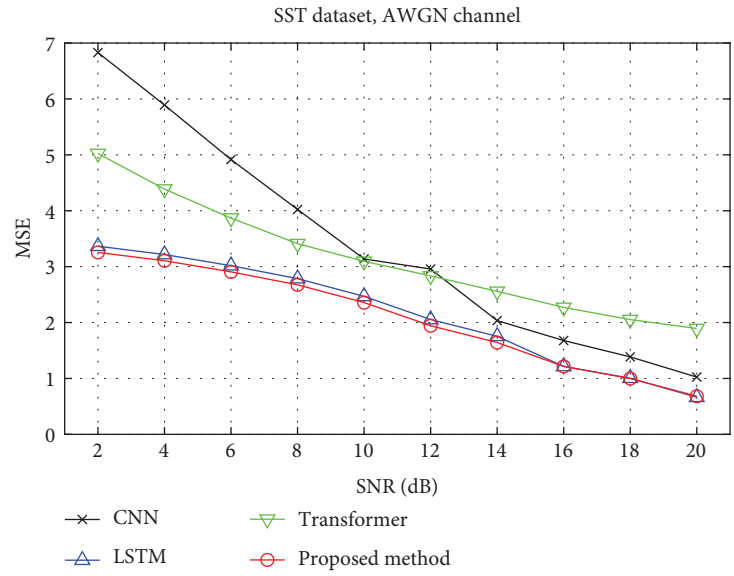
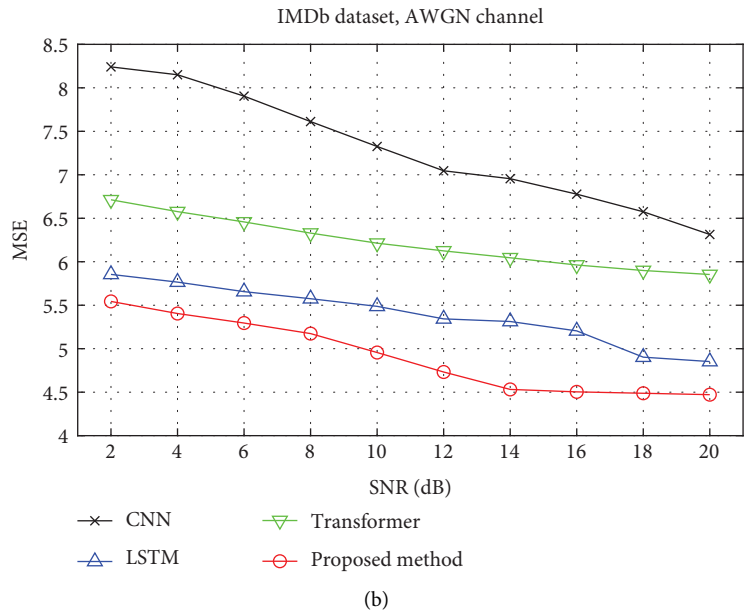


Figure 6: Performance comparison chart of position encoding. (a) MSE under SST dataset. (b) MSE under IMDb dataset. (c) MSE under YES–NO dataset. (d) MSE under LIBRISPEECH dataset. (e) MSE under MNIST dataset. (f) MSE under CIFAR-10 dataset.



(a)



(b)

Figure 7: Continued.

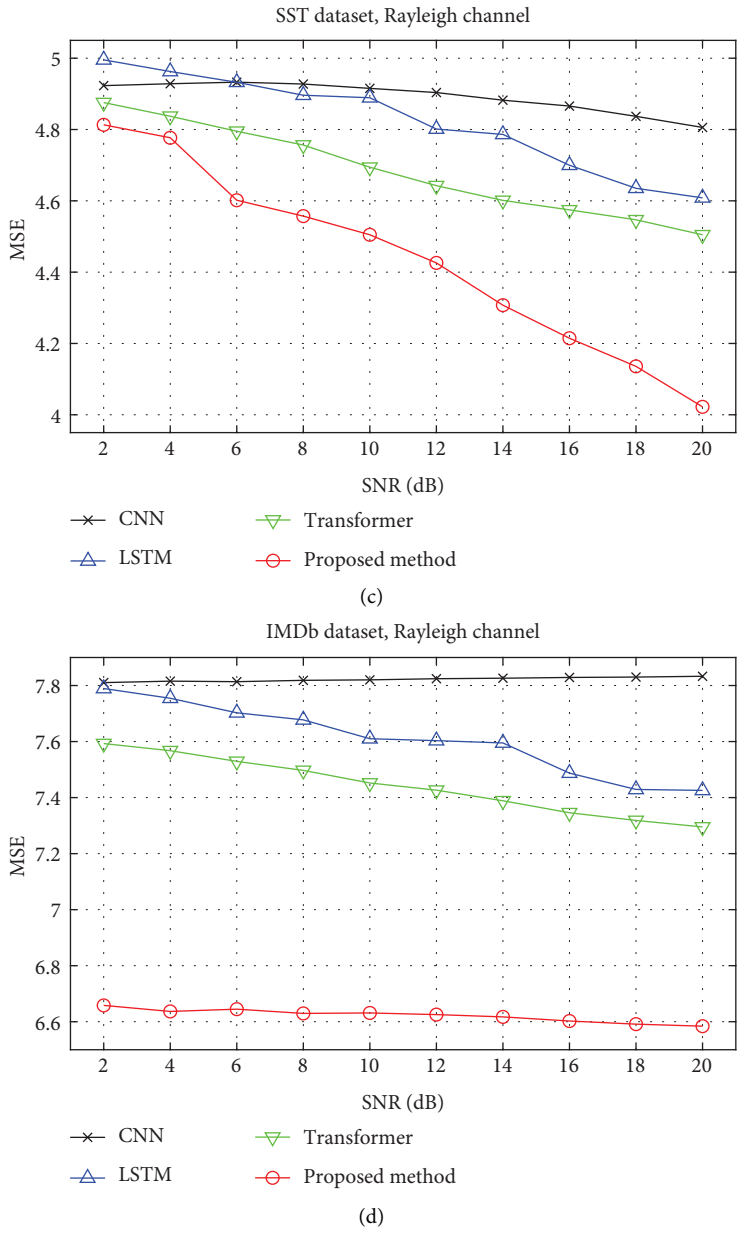


Figure 7: MSE performance versus the SNR. (a-b) Over the AWGN channel. (c-d) Over the Rayleigh channel.

Transformer, and LSTM. While not the fastest, its performance remains competitive and falls within an acceptable range. The VAE's smaller model size suggests that its layers might use smaller numbers of filters or less dense connections, but the specific operations (e.g., complex activation functions) or the sheer depth of its generative path can still result in a high FLOP count. In contrast, our proposed method, while having a larger model size, uses operations that are individually less FLOP-intensive, resulting in a lower overall FLOP count.

3.5. Limitations and Future Works. However, there are some limitations of the proposed method regarding its application in a production environment. First, in asynchronous FL, different clients may train and upload models at different speeds, causing synchronization issues that affect the convergence speed and performance of the model, which is an inherent limitation of FL. Besides, CNNs and Transformers are typically large and complex, posing challenges for training and inference on resource-constrained devices. Model compression and optimization techniques (e.g., pruning and quantization) may need further research to fit these constraints. Furthermore, semantic communication systems may require real-time data processing and transmission, while the training and updating process of FL can introduce delays, affecting the system's real-time performance.

While semantic communication and model compression aim to reduce the amount of data transmitted, the computational cost and time required for the encoding and

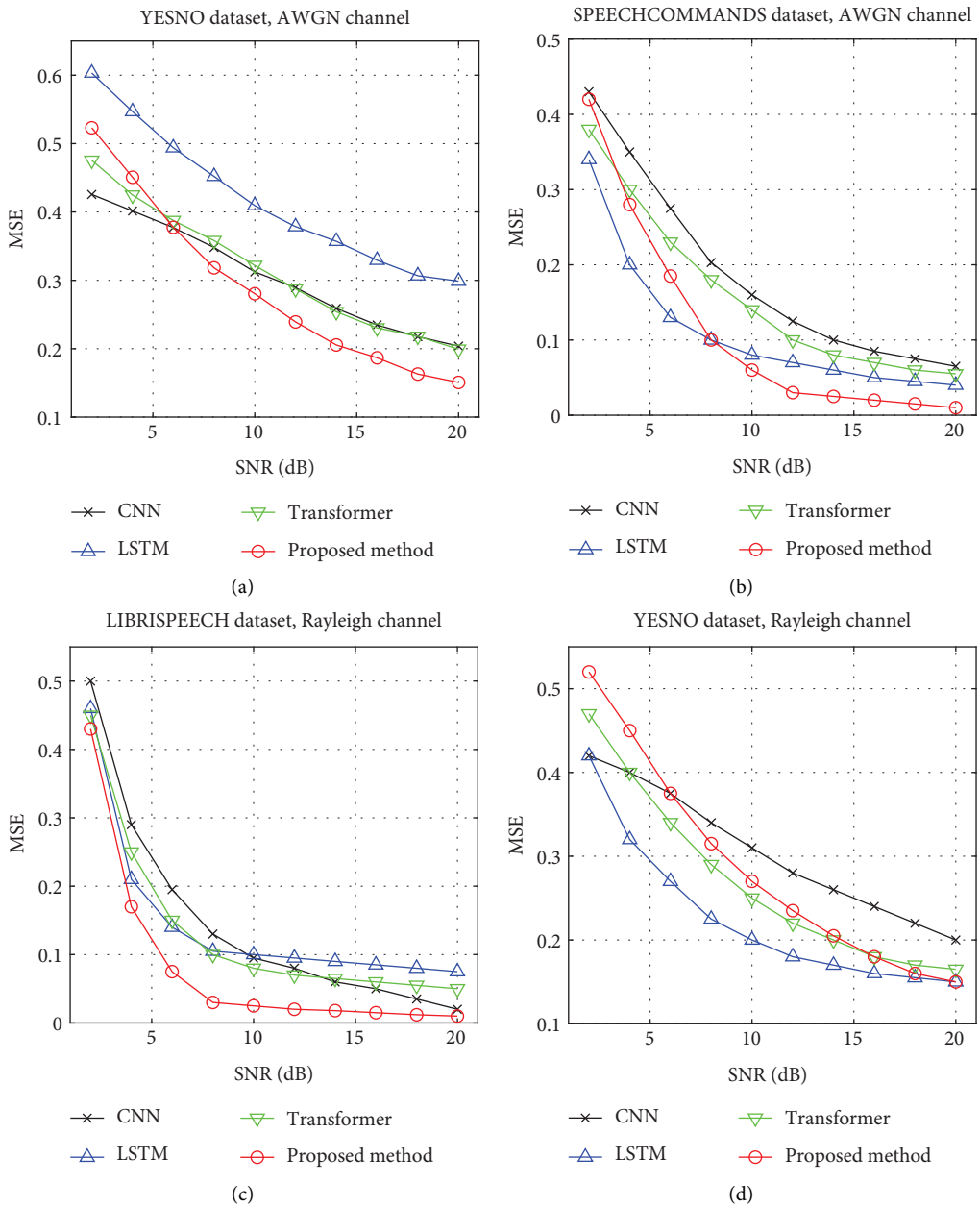


Figure 8: Continued.

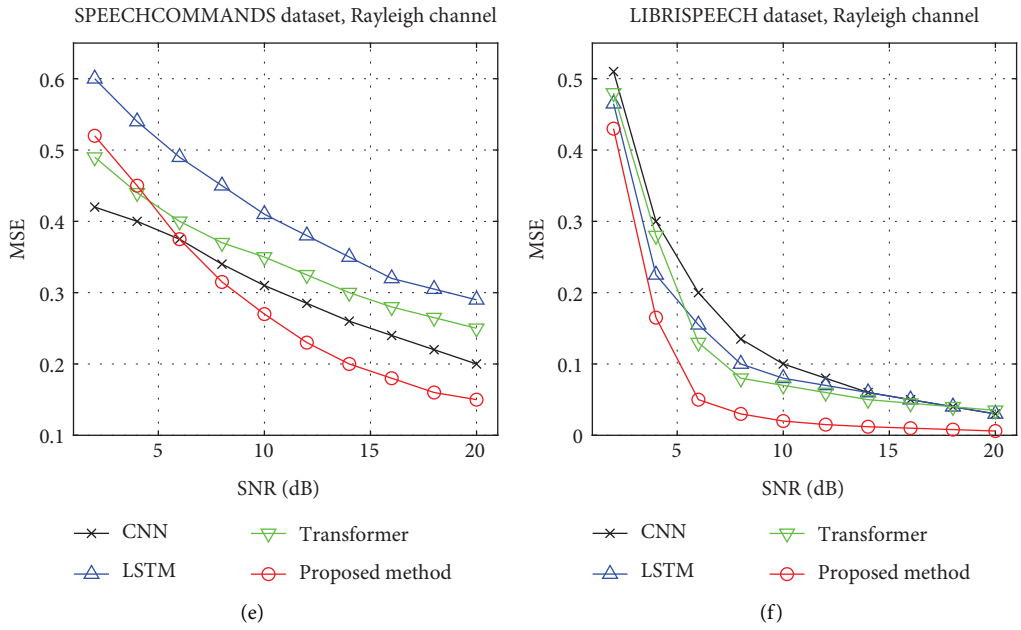


Figure 8: MSE performance versus the SNR. (a–c) Over the AWGN channel. (d–f) Over the Rayleigh channel.

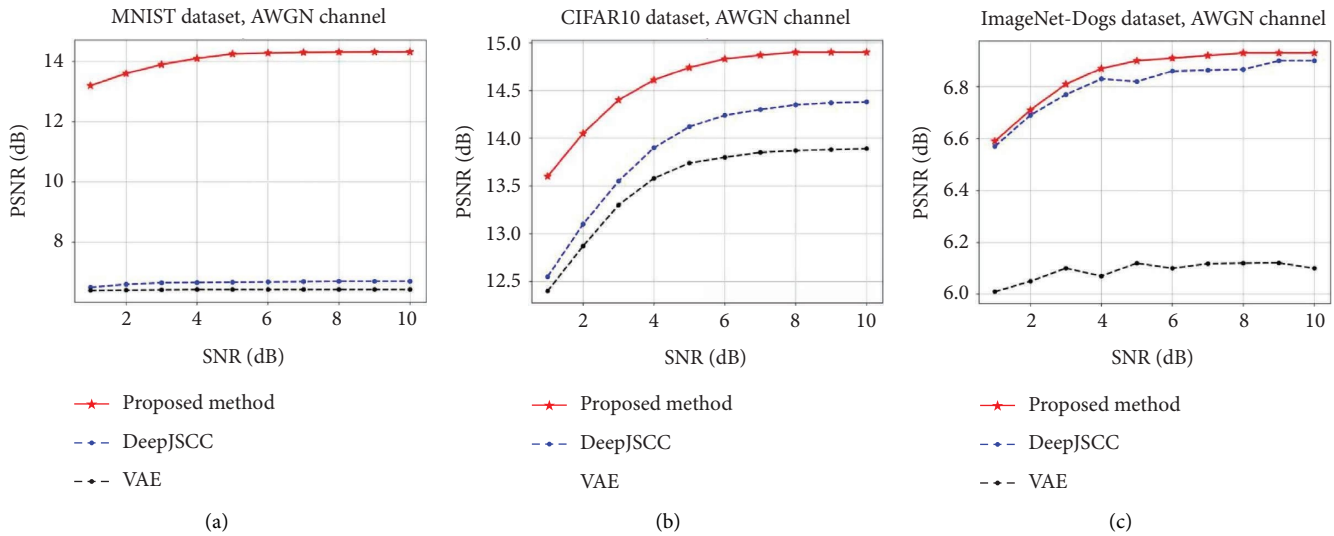


Figure 9: Continued.

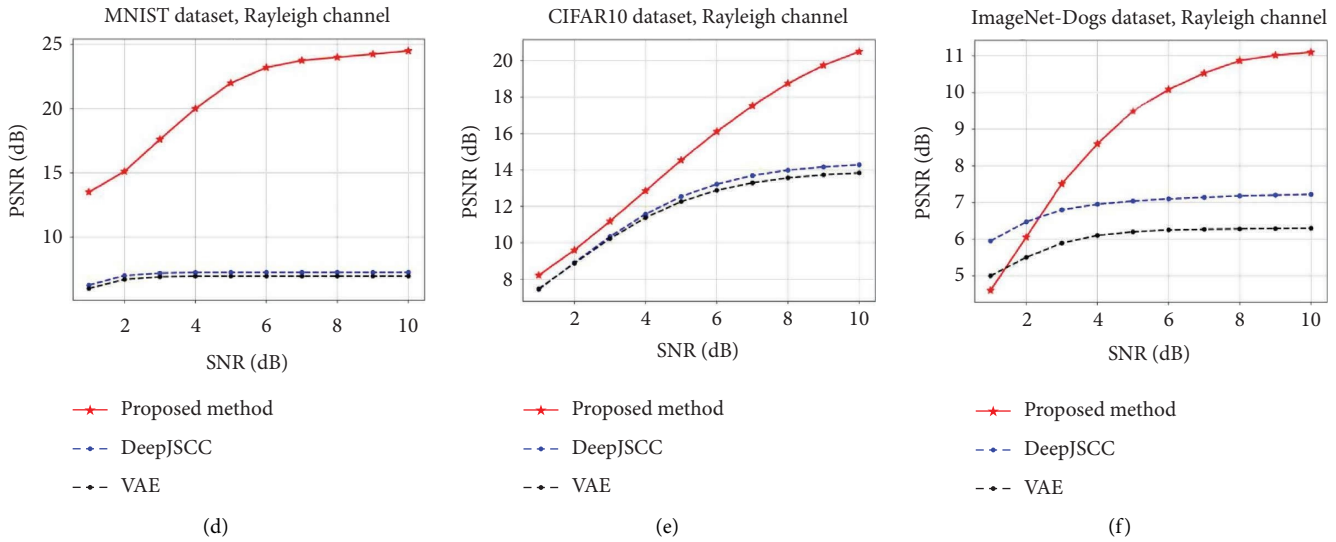


Figure 9: PSNR performance versus the SNR. (a–c) Over the AWGN channel. (d–f) Over the Rayleigh channel.

Table 3: Original text and texts recovered by different models.

Original	However, the film veers away from normal types and presents us with characters that are best described as “extremes”	MSE
Proposed method	However, the film veers away from normal types and presents us and characters that are best described as “extremes”	0.00958
LSTM	However, to the veers in from normal types are presents us are characters that are best described as “extremes”	0.02754
CNN	However, the film veers away from normal types and it us with characters <pad> ill best described as “extremes”	0.04702

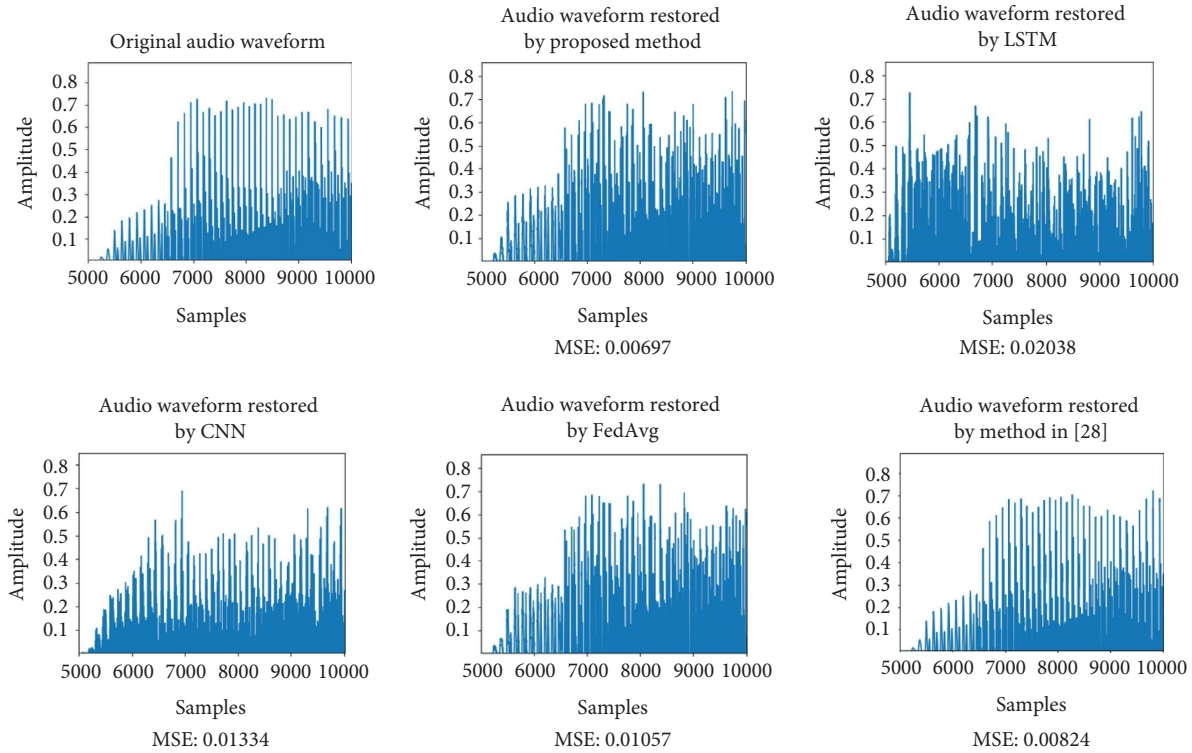


Figure 10: Original audio waveform and audio waveform recovered by different models.

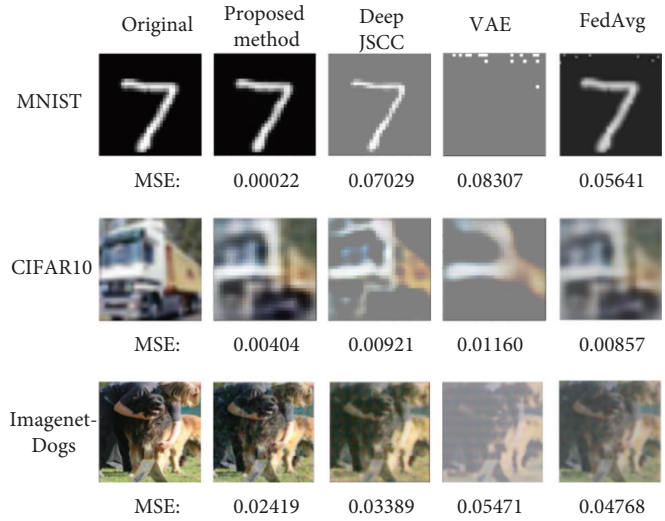


Figure 11: Original image and image recovered by different models.

Table 4: Inference speed and complexity comparison.

Method	Inference time (ms)	FLOPs (G)	Model size (M)	Encoder output size
Proposed method	23.3	1.96	33.63	(1, 1, 512)
DeepJSCC	21.1	2.16	5.58	(1, 512, 1)
VAE	49.2	5.05	10.07	(1, 512)
CNN	8.7	1.68	28.81	(1, 1, 512)
Transformer	4.5	0.19	1.72	(1, 1, 512)
LSTM	12.9	1.74	35.31	(1, 512)

compression processes themselves on resource-constrained edge devices have not been thoroughly evaluated in this work. The compression overhead (e.g., the time spent on semantic feature extraction by the encoder or applying model compression techniques such as pruning and quantization) needs to be carefully balanced against the transmission savings. In scenarios where the compression time outweighs the transmission time saved, the overall latency might increase, which poses a significant challenge for real-time applications.

For future studies, we plan to conduct research on model compression and optimization techniques to fit the computational and storage capabilities of resource-constrained devices. Crucially, this research will involve a comprehensive analysis of the trade-off between the computational overhead of compression, encoding, and the communication savings, aiming to develop methods where the overall latency is minimized. Meanwhile, low-latency model updates and efficient real-time data processing algorithms will be explored.

4. Conclusions

This paper introduced DeepSC-CT, a novel architecture integrating CNN and Transformer, which demonstrated significant advancements in semantic communication across diverse modalities, including text, audio, and images. The main works, including a highly effective adaptive position-encoding mechanism and an innovative adaptive FL algorithm, proved crucial. The former markedly improved positional information capture and system robustness, particularly under low signal-to-noise conditions, while outperforming traditional models and reducing communication overhead. The latter, by weighting client contributions based on local accuracy, conclusively surpassed the classic FedAvg algorithm in both robustness and communication efficiency, leading to enhanced generalization. Collectively, these contributions establish DeepSC-CT as a resource-efficient framework for robust multimodal semantic communication in privacy-preserving federated environments, effectively addressing the challenges of data heterogeneity and long-distance dependency capture.

Data Availability Statement

The data used to support the findings of this study are available from the corresponding author upon reasonable request.

Disclosure

This manuscript extends work preliminarily in a conference paper titled Federated Learning for Image Semantic Communication System Based on CNN and Transformer, presented at 2023 International Conference on Ubiquitous Communication.

Conflicts of Interest

The authors declare no conflicts of interest.

Funding

This work is supported by the National Natural Science Foundation of China No. 62371428, the Fundamental Research Funds for the Central Universities No. CUC23GG13, and the Royal Society International Exchanges 2023 Cost Share No. IEC\NSFC\233318.

References

[1] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y. J. A. Zhang, "The Roadmap to 6G: AI Empowered Wireless Networks," *IEEE Communications Magazine* 57, no. 8 (2019): 84–90, https://doi.org/10.1109/mcom.2019.1900271.

[2] P. Wang, Y. Li, L. Song, and B. Vucetic, "Multi-Gigabit Millimeter Wave Wireless Communications for 5G: from Fixed Access to Cellular Networks," *IEEE Communications Magazine* 53, no. 1 (2015): 168–178, https://doi.org/10.1109/mcom.2015.7010531.

[3] C. Han and Y. Chen, "Propagation Modeling for Wireless Communications in the Terahertz Band," *IEEE Communications Magazine* 56, no. 6 (2018): 96–101, https://doi.org/10.1109/mcom.2018.1700898.

[4] L. Lu, G. Y. Li, A. L. Swindlehurst, A. Ashikhmin, and R. Zhang, "An Overview of Massive MIMO: Benefits and Challenges," *IEEE Journal of Selected Topics in Signal Processing* 8, no. 5 (2014): 742–758, https://doi.org/10.1109/jstsp.2014.2317671.

[5] C. Huang, A. Zappone, G. C. Alexandropoulos, M. Debbah, and C. Yuen, "Reconfigurable Intelligent Surfaces for Energy Efficiency in Wireless Communication," *IEEE Transactions on Wireless Communications* 18, no. 8 (2019): 4157–4170, https://doi.org/10.1109/twc.2019.2922609.

[6] K. Sayood, *Introduction to Data Compression* (Morgan Kaufmann, 2017).

[7] A. Goldsmith, *Wireless Communications* (Cambridge University Press, 2005).

[8] M. Kountouris and N. Pappas, "Semantics-Empowered Communication for Networked Intelligent Systems," *IEEE Communications Magazine* 59, no. 6 (2021): 96–102, https://doi.org/10.1109/mcom.001.2000604.

[9] W. Weaver, "Recent Contributions to the Mathematical Theory of Communication," *Etc: A Review of General Semantics* 10, no. 4 (1953): 261–281.

[10] S. Li, T. Yao, S. Li, and L. Yan, "Semantic-Enhanced Multimodal Fusion Network for Fake News Detection," *International Journal of Intelligent Systems* 37, no. 12 (2022): 12235–12251, https://doi.org/10.1002/int.23084.

[11] H. Xie and Z. Qin, "A Lite Distributed Semantic Communication System for Internet of Things," *IEEE Journal on Selected Areas in Communications* 39, no. 1 (2021): 142–153, https://doi.org/10.1109/jsac.2020.3036968.

[12] Z. Weng and Z. Qin, "Semantic Communication Systems for Speech Transmission," *IEEE Journal on Selected Areas in Communications* 39, no. 8 (2021): 2434–2444, https://doi.org/10.1109/jsac.2021.3087240.

[13] E. Erdemir, T. Y. Tung, P. L. Dragotti, and D. Gündüz, "Generative Joint Source-Channel Coding for Semantic Image Transmission," *IEEE Journal on Selected Areas in Communications* 41, no. 8 (2023): 2645–2657, https://doi.org/10.1109/jsac.2023.3288243.

[14] C. H. Lee, J. W. Lin, P. H. Chen, and Y. C. Chang, "Deep Learning-Constructed Joint Transmission-Recognition for Internet of Things," *IEEE Access* 7 (2019): 76547–76561, https://doi.org/10.1109/access.2019.2920929.

[15] J. Wu, C. Wu, Y. Lin, et al., "Semantic Segmentation-Based Semantic Communication System for Image Transmission," *Digital Communications and Networks* 10, no. 3 (2024): 519–527, https://doi.org/10.1016/j.dcan.2023.02.006.

[16] X. Zhang, B. Zhang, W. Yu, and X. Kang, "Federated Deep Learning With Prototype Matching for Object Extraction From Very-High-Resolution Remote Sensing Images," *IEEE Transactions on Geoscience and Remote Sensing* 61 (2023): 1–16, https://doi.org/10.1109/tgrs.2023.3244136.

[17] H. Li, Z. Cai, J. Wang, et al., "FedTP: Federated Learning by Transformer Personalization," *IEEE Transactions on Neural Networks and Learning Systems* 35, no. 10 (2024): 13426–13440, https://doi.org/10.1109/tnnls.2023.3269062.

[18] L. Floridi, "Outline of a Theory of Strongly Semantic Information," *Minds and Machines* 14, no. 2 (2004): 197–221, https://doi.org/10.1023/b:mind.0000021684.50925.c9.

[19] K. Popper, *Conjectures and Refutations: The Growth of Scientific Knowledge* (Routledge, 2014).

[20] L. A. Zadeh, "Fuzzy Sets," *Information and Control* 8, no. 3 (1965): 338–353, https://doi.org/10.1016/s0019-9958(65)90241-x.

[21] L. A. Zadeh, "Probability Measures of Fuzzy Events," *Journal of Mathematical Analysis and Applications* 23, no. 2 (1968): 421–427, https://doi.org/10.1016/0022-247x(68)90078-4.

[22] H. Xie, Z. Qin, X. Tao, and K. B. Letaief, "Task-Oriented Multi-User Semantic Communications," *IEEE Journal on Selected Areas in Communications* 40, no. 9 (2022): 2584–2597, https://doi.org/10.1109/jsac.2022.3191326.

[23] H. Xie, Z. Qin, G. Y. Li, and B. H. Juang, "Deep Learning Enabled Semantic Communication Systems," *IEEE Transactions on Signal Processing* 69 (2021): 2663–2675, https://doi.org/10.1109/tsp.2021.3071210.

[24] N. Farsad, M. Rao, and A. Goldsmith, "Deep Learning for Joint Source-Channel Coding of Text," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing* (April 2018), 2326–2330, https://doi.org/10.1109/icassp.2018.8461983.

[25] E. Bourtsoulatze, D. Burth Kurka, and D. Gunduz, "Deep Joint Source-Channel Coding for Wireless Image Transmission," *IEEE Transactions on Cognitive Communications and Networking* 5, no. 3 (2019): 567–579, https://doi.org/10.1109/tccn.2019.2919300.

[26] D. B. Kurka and D. Gunduz, "DeepJSCC-f: Deep Joint Source-Channel Coding of Images With Feedback," *IEEE Journal on Selected Areas in Information Theory* 1, no. 1 (2020): 178–193, https://doi.org/10.1109/jsait.2020.2987203.

[27] Z. Yang, M. Chen, K. K. Wong, H. V. Poor, and S. Cui, "Federated Learning for 6G: Applications, Challenges, and Opportunities," *Engineering* 8 (2022): 33–41, https://doi.org/10.1016/j.eng.2021.12.002.

[28] H. Tong, Z. Yang, S. Wang, Y. Hu, W. Saad, and C. Yin, "Federated Learning Based Audio Semantic Communication over Wireless Networks," in *2021 IEEE Global Communication Conference* (December 2021), 1–6, https://doi.org/10.1109/globecom46510.2021.9685654.

[29] H. Wei, W. Ni, W. Xu, F. Wang, D. Niyato, and P. Zhang, "Federated Semantic Learning Driven by Information Bottleneck for Task-Oriented Communications," *IEEE Communications Letters* 27, no. 10 (2023): 2652–2656, https://doi.org/10.1109/lcomm.2023.3307096.

[30] U. Michieli, M. Toldo, and M. Ozay, "Federated Learning via Attentive Margin of Semantic Feature Representations," *IEEE*

Internet of Things Journal 10, no. 2 (2023): 1517–1535, https://doi.org/10.1109/jiot.2022.3209865.

[31] Y. Chen, X. Dai, D. Chen, et al., "Mobile-Former: Bridging MobileNet and Transformer," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (June 2022), 5260–5269, https://doi.org/10.1109/cvpr52688.2022.00520.

[32] K. Wu, H. Peng, M. Chen, J. Fu, and H. Chao, *Rethinking and Improving Relative Position Encoding for Vision Transformer* (2021).

[33] K. Yang, S. Wang, J. Dai, K. Tan, K. Niu, and P. Zhang, "WITT: A Wireless Image Transmission Transformer for Semantic Communications," in *2023 IEEE International Conference on Acoustics, Speech and Signal Processing* (May 2023), 1–5, https://doi.org/10.1109/icassp49357.2023.10094735.

[34] Z. Deng, S. Li, Y. Cai, and B. Su, "Federated Learning for Image Semantic Communication System Based on CNN and Transformer," in *2023 International Conference on Ubiquitous Communication* (July 2023).

[35] Y. Yu, X. Si, C. Hu, and J. Zhang, "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures," *Neural Computation* 31, no. 7 (2019): 1235–1270, https://doi.org/10.1162/neco_a_01199.

[36] Y. Bo, Y. Duan, S. Shao, and M. Tao, "Joint Coding-Modulation for Digital Semantic Communications via Variational Autoencoder," *IEEE Transactions on Communications* 72, no. 9 (2024): 5626–5640, https://doi.org/10.1109/tcomm.2024.3386577.