Separate Source Channel Coding Is Still What You Need: An LLM-Based Rethinking

REN Tianqi[1], LI Rongpeng[1], ZHAO Mingmin[1], CHEN Xianfu[2], LIU Guangyi[3], YANG Yang[4], ZHAO Zhifeng[1,5], ZHANG Honggang[6]

(1. College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China;
2. Shenzhen CyberAray Network Technology Co., Ltd., Shenzhen 518000, China;
3. China Mobile Research Institute, Beijing 100053, China;
4. The Internet of Things Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511453, China;
5. Zhejiang Lab, Hangzhou 311121, China;
6. Faculty of Data Science, City University of Macau, Macao 999078, China)

DOI: 10.12142/ZTECOM.202501005

https://kns.cnki.net/kcms/detail/34.1294.TN.20250319.0931.004.html, published online March 20, 2025

Manuscript received: 2025-01-02

Abstract: Along with the proliferating research interest in semantic communication (SemCom), joint source channel coding (JSCC) has dominated the attention due to the widely assumed existence in efficiently delivering information semantics. Nevertheless, this paper challenges the conventional JSCC paradigm and advocates for adopting separate source channel coding (SSCC) to enjoy a more underlying degree of freedom for optimization. We demonstrate that SSCC, after leveraging the strengths of the Large Language Model (LLM) for source coding and Error Correction Code Transformer (ECCT) complemented for channel coding, offers superior performance over JSCC. Our proposed framework also effectively highlights the compatibility challenges between SemCom approaches and digital communication systems, particularly concerning the resource costs associated with the transmission of high-precision floating point numbers. Through comprehensive evaluations, we establish that assisted by LLM-based compression and ECCT-enhanced error correction, SSCC remains a viable and effective solution for modern communication systems. In other words, separate source channel coding is still what we need.

Keywords: separate source channel coding (SSCC); joint source channel coding (JSCC); end-to-end communication system; Large Language Model (LLM); lossless text compression; Error Correction Code Transformer (ECCT)

Citation (Format 1): REN T Q, LI R P, ZHAO M M, et al. Separate source channel coding is still what you need: an LLM-based rethinking [J]. ZTE Communications, 2025, 23(1): 30 - 44. DOI: 10.12142/ZTECOM.202501005

Citation (Format 2): T. Q. Ren, R. P. Li, M. M. Zhao, et al., "Separate source channel coding is still what you need: an LLM-based rethinking," *ZTE Communications*, vol. 23, no. 1, pp. 30 - 44, Mar. 2025. doi: 10.12142/ZTECOM.202501005.

1 Introduction

Semantic communication (SemCom) has garnered significant attention in recent years, with researchers exploring innovative approaches to enhance the efficiency and reliability of information transmission[1]. Generally, SemCom leverages deep learning-based joint source-channel coding (JSCC) methods to preserve global semantic information and local texture during the transmission process. DeepJSCC[2] pioneers these works by implementing JSCC with feedback and allowing for real-time adaptation to channel conditions. Along with its steady progress, JSCC has been substantially studied, mostly with the optimization objective shifting from bit error rates to the semantic relevance of the transmitted information in SemCom[3-14]. However, albeit the awfully exploded research interest, one critical question remains unsolved: why does the joint approach stand out, as separate source channel coding (SSCC), shall promise a greater degree of freedom from an optimization perspective?

As the terminology implies, SSCC encompasses two decoupled ingredients: source coding and channel coding. The former part lies in effectively compressing the context, and the effectiveness of underlying deep neural networks (DNN)-based predictors, such as recurrent neural networks (RNN)-

This work was supported in part by the National Key Research and Development Program of China under Grant No. 2024YFE0200600, the Zhejiang Provincial Natural Science Foundation of China under Grant No. LR23F010005, and the Huawei Cooperation Project under Grant No. TC20240829036.

based DeepZip[15], Long Short Term Memory (LSTM) - based[16 - 17] and hybrid DNN-based Dzip[18], have been validated widely in achieving satisfactory text compression. More prominently, Transformer-based[19] and Large Language Model (LLM)-based compression have emerged recently[20 - 24]. The latest research[25] unveils the equivalence between compression and prediction. In other words, in the general framework where statistical models predict symbols and encoders use predictive probabilities to perform compression, better predictive models lead directly to better compressors[25]. Hence, the astonishing capability of LLM implies the potential for an unprecedented source codec. On the other hand, the Error Correction Code (ECC) plays an indispensable role in channel coding. Although some advanced algebraic block codes like Bose - Chaudhuri - Hocquenghem (BCH) codes[26], Low-Density Parity-Check (LDPC) codes[27] and Polar codes[28] can somewhat ensure the reliability of transmission, the efficient decoding of ECC is an unresolved difficulty. Recently, DNNs have started to demonstrate their contribution to channel coding. For example, deep learning models are implemented to achieve belief propagation (BP) decoding[29 - 31], while a model-free Error Correction Code Transformer (ECCT) for algebraic block codes[32] contributes to the enhancement of decoding reliability.

In this paper, on top of an LLM-based arithmetic coding (LLM-AC) system, the proposed SSCC framework integrates fine-grained, semantics-aware probability modeling and encoding with ECCT-enhanced channel decoding, thus forming a closed-loop optimization framework. To the best of our knowledge, this work represents the first comprehensive integration of LLM-based compression and ECCT-complemented channel decoding for a holistic SemCom architecture. Through extensively showcasing the performance superiority over JSCC, we argue this performance improvement primarily arises after tackling the underlying incompatibility between conventional SemCom approaches[3 - 7, 11 - 14] and digital communication architectures[33]. Particularly, those approaches simply assume the deliverability of encoded semantic feature vectors while neglecting the energy costs associated with transmitting high-precision floating point numbers[33]. However, further quantization[9 - 10] and digital modulation can compromise the widely assumed existence of performance superiority in JSCC. Meanwhile, in contrast to the direct utilization of the astonishing semantic interpretation capability[34 - 36], the deployment of LLMs focuses on the compression and encoding of text to squeeze the largely untapped redundancy. Therefore, our work is also significantly different from existing integrations of generative AI (GAI) and SemCom[37 - 43]. Furthermore, the adoption of ECCT boosts the effectiveness of SSCC in specific cases. In summary, our comprehensive evaluation of LLM and ECCT-based SSCC demonstrates that separate source channel coding is still what we need.

The rest of this paper is organized as follows. Section 2 introduces the SSCC system model, while its key components are enumerated in Section 3. Section 4 provides numerical results demonstrating the performance superiority of the proposed SSCC system. Finally, Section 5 concludes this paper with discussions on future works. For convenience, we list the major notations of this paper in Table 1.

2 System Model

Our SSCC framework encompasses the following ingredients.

1) Source encoding

The input text sequence denoted as $\boldsymbol{s}$ undergoes a source encoder that converts characters into a compressed binary message $\boldsymbol{m} \in \{0,1\}^K$. During source encoding, arithmetic coding (AC) can be leveraged for effective compression here. For LLM-based processing, an intermediate result (i.e., a sequence of tokens $\boldsymbol{t}$) can be obtained during the transformation from $\boldsymbol{s}$ to $\boldsymbol{m}$.

Table 1. Major notations used in this paper

Notation	Definition
$\boldsymbol{s}$, $\hat{\boldsymbol{s}}$	The transmitted text sequence and the recovered text sequenceat the receiver side
$\boldsymbol{t}$, $\hat{\boldsymbol{t}}$	The transmitted token sequence and the recovered token sequence at the receiver side
C_s, C_e	The source code and the channel code (error correction code)
ρ, $\tilde{\rho}$	The source distribution and the predicted probability distribution via LLM
$\mathcal{D}$, D_i, τ	The dictionary of source coder, the i-th character in the dictionary, and the vocabulary of the dictionary
$\mathbb{I}_k$, l_k, u_k	The probability interval in step k of source coding and its corresponding lower and upper bounds
$\boldsymbol{m}$, $\hat{\boldsymbol{m}}$	The message encoded by the source coder and the received (and channel decoded) message
λ	The probability interval, determined by the codeword, in a decimal form
N, K	The codeword length and message length of error correction code $C_e(N,K)$
$\boldsymbol{G}$, $\boldsymbol{H}$	The generator matrix and the parity check matrix
$\boldsymbol{x}$, $\boldsymbol{x}_b$, $\boldsymbol{x}_s$	The transmitted codeword encoded by the channel coder and its binary and sign form
$\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{x}}_b$	The soft approximation of codeword and its binary form
$\mathcal{N}(\cdot,\cdot)$, σ_n	The Gaussian distribution and the standard deviation of noise
h	The channel fading coefficient
z, $\tilde{z}$, $\hat{z}$	The additive Gaussian noise, as well as its corresponding multiplicative noise and the prediction result by ECCT
$\boldsymbol{y}$, $\boldsymbol{y}_b$, $\tilde{\boldsymbol{y}}$	The noisy codeword, its binary form, and the result of pre-processing noisy codeword
$\mathrm{syn}(\cdot)$	The syndrome of codes defined in ECCT
$f(\cdot)$	The decoding function of ECCT
$\boldsymbol{W}$	The learnable embedding matrix for high-dimensional mapping
$\boldsymbol{g}(\cdot)$	The code-aware self-attention mask

ECCT: Error Correction Code Transformer LLM: Large Language Model

2) Channel encoding and modulation

The message $\boldsymbol{m}$ is then encoded via an LDPC code $C_e(N,K)$, which is selected for its excellent error-correction capabilities and compatibility with iterative decoding algorithms, as mentioned in Ref. [32]. The encoding process employs a generator matrix $\boldsymbol{G}$ to transform the message in $\boldsymbol{m}$ to a codeword $\boldsymbol{x}_b \in \{0,1\}^N$. The parity check matrix $\boldsymbol{H}$, which satisfies $\boldsymbol{G} \cdot \boldsymbol{H}^T = 0$ and $\boldsymbol{H} \cdot \boldsymbol{x}_b = 0$, is a key component of the LDPC decoding process. Afterwards, binary phase shift keying (BPSK) modulation maps the binary codeword $\boldsymbol{x}_b$ to a sequence of symbols $\boldsymbol{x}_s \in \{\pm 1\}^N$, suitable for transmission over the wireless channel. Notably, other error correction codes, such as Polar codes[28], can be applied as well.

3) Channel

The modulated signal $\boldsymbol{x}_s$ is transmitted over a noisy channel, modeled as an additive white Gaussian noise (AWGN) channel or a Rayleigh fading channel. The received signal $\boldsymbol{y} \in \mathbb{R}^N$ is corrupted by additive noise $\boldsymbol{z} \sim \mathcal{N}(0, \sigma_n^2)$, resulting in $\boldsymbol{y} = h\boldsymbol{x}_s + \boldsymbol{z}$, where h is the channel fading coefficient.

4) Demodulation and channel decoding

BPSK demodulation recovers a binary codeword $\hat{\boldsymbol{x}}_b \in \{0,1\}^N$ from $\hat{\boldsymbol{x}}$. Subsequently, the channel decoder reconstructs the message $\hat{\boldsymbol{m}} \in \{0,1\}^K$ from $\hat{\boldsymbol{x}}_b$. In contrast to conventional approaches that employ either hard-decision (e.g., the bit-flipping algorithm) or soft-decision (e.g., the sum-product algorithm) algorithms to decode LDPC codewords transmitted through the channel, some complementary decoding modules, such as ECCT, can be applied prior to demodulation to enhance the decoding performance. Notably, ECCT can provide an estimation of the transmitted codeword $\hat{\boldsymbol{x}}_b$, denoted as $\hat{\boldsymbol{x}}$, while subsequent demodulation and information bits extraction are then performed on the estimated codeword $\hat{\boldsymbol{x}}$.

5) Source decoding

The recovered message $\hat{\boldsymbol{m}}$ is ultimately decoded by the source decoder, which reconstructs the text sequence $\hat{\boldsymbol{s}}$ from the message, effectively reversing the encoding process. Similar to the encoder, the decoder can implement arithmetic decoding.

In comparison, JSCC typically employs an end-to-end DNN to implement source and channel codecs. Here, the terminology "end-to-end" implies the joint training of source and channel codes, as adopted in most works. Further details on JSCC can be found in Ref. [1] and the references therein. In the following section, we will address how to leverage the strength of LLM to enhance text compression and reconstruction, combined with the robustness of ECCT-complemented LDPC codes for error correction, as shown in Fig. 1.

3 Proposed SSCC Framework

In this section, we introduce LLM-based source coding and ECCT-complemented channel coding.

3.1 LLM-Based Source Coding

Given a source distribution ρ, lossless compression aims to encode a text sequence $\boldsymbol{s}$ sampled from ρ into a binary code $\boldsymbol{m} = C_s(\boldsymbol{s})$ of minimal possible length with no loss of original infor-

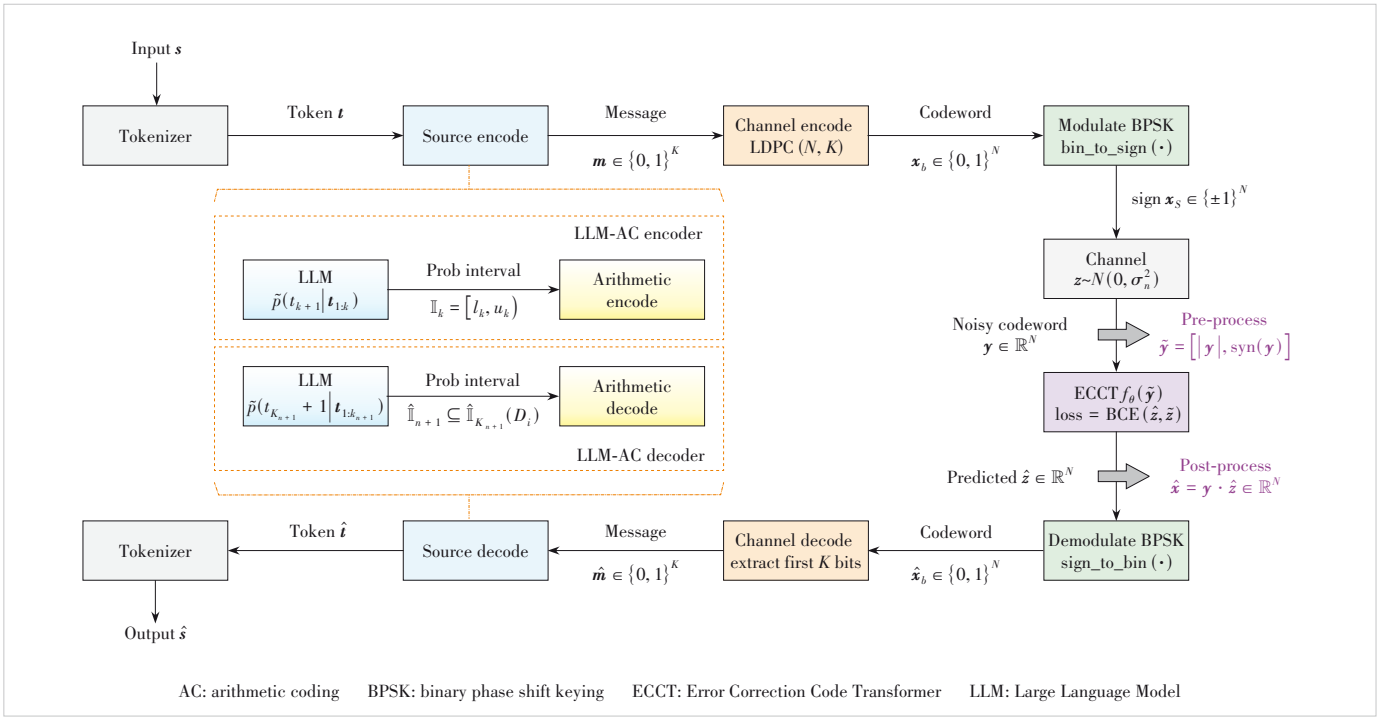


Figure 1. Framework of LLM-based and ECCT-complemented SSCC system

mation. According to Shannon's source coding theorem[44], the optimal expected bit length is $L_{\min} = \mathbb{E}_{\boldsymbol{s}\sim\rho}\left[-\log_2\rho(\boldsymbol{s})\right]$. To obtain such optimal length, arithmetic coding[45-46], a form of entropy encoding, is typically adopted, relying on a probabilistic model over ρ or its marginal distribution. Arithmetic coding implies that frequently used characters are stored with fewer bits while rarely occurring characters correspond to more bits, resulting in fewer bits used in total.

In particular, the input text sequence $\boldsymbol{s}$ undergoes tokenization by the LLM tokenizer, which converts characters into a sequence of tokens $\boldsymbol{t}$ for processing by the LLM. The LLM subsequently generates a compact representation of the text, effectively encoding the tokens into a compressed binary message $\boldsymbol{m} \in \{0,1\}^K$. Specially, considering a dictionary $\mathcal{D}$ of τ tokens, the input sequence $\boldsymbol{s}$ is first parsed into the token sequence $\boldsymbol{t}$. Given the first k tokens $\boldsymbol{t}_{1:k}$, the $(k+1)$-th token t_{k+1} can be inferred as a predicted probability distribution $\tilde{\rho}\left(t_{k+1}|\boldsymbol{t}_{1:k}\right)$. Here, $\tilde{\rho}\left(t_{k+1}|\boldsymbol{t}_{1:k}\right)$ indicates the LLM's estimation of the true distribution $\rho\left(t_{k+1}|\boldsymbol{t}_{1:k}\right)$. The incremental decoding nature in LLM enables it to accurately predict the probability distribution of the next token based on known ones, thereby providing a sub-optimal estimation of the true distribution[25]. As shown in Fig. 2, selecting the next character effectively narrows down the probabilistic interval where the sequence is located, which means the code $\boldsymbol{m}$ is determined once the interval is fixed. Starting with $\mathbb{I}_0 = [0,1)$, the previous interval determined by $\boldsymbol{t}_{1:k}$ in step k is defined as $\mathbb{I}_k = [l_k, u_k)$. Therefore, denoting $p\left(t_{k+1} = D_j\right) = \tilde{\rho}\left(t_{k+1} = D_j|\boldsymbol{t}_{1:k}\right)$,

$$\mathbb{I}_{k+1}(D_i) = \begin{bmatrix} l_k + (u_k - l_k) \times \sum_{j<i} p\left(t_{k+1} = D_j\right), \\ l_k + (u_k - l_k) \times \sum_{j\leqslant i} p\left(t_{k+1} = D_j\right) \end{bmatrix} \quad (1).$$

In practice, we consider finite precision arithmetic encoders, referring to Ref. [47], with pseudo-code provided in Appendix 1. Consequently, we can obtain a binary code $\boldsymbol{m} = C_s(\boldsymbol{s})$ of the shortest length, completely corresponding to the probability interval determined by the sequence. At the receiver side, if the receiver shares a consistent source distribution $\tilde{\rho}$ with the sender, given the received (and channel-decoded) bit sequence $\hat{m}$ corresponding to $C_s(\boldsymbol{s})$, we can decode $t_{K_{n+1}} = D_i \in \mathcal{D}$ by identifying D_i, such that

$$\hat{\mathbb{I}}_{n+1} = [l_{n+1}, u_{n+1}) = \begin{cases} \left[l_n, \frac{1}{2}(l_n + u_n)\right), \text{if } m_{n+1} = 0 \\ \left[\frac{1}{2}(l_n + u_n), u_n\right), \text{if } m_{n+1} = 1 \end{cases} \subseteq \hat{\mathbb{I}}_{K_{n+1}}(D_i) = [L, U) \quad (2),$$

where $L = l_{K_{n+1}} + \left(u_{K_{n+1}} - l_{K_{n+1}}\right) \times \sum_{j<i} p\left(t_{K_{n+1}+1} = D_j\right)$ and $U = l_{K_{n+1}} + \left(u_{K_{n+1}} - l_{K_{n+1}}\right) \times \sum_{j\leqslant i} p\left(t_{K_{n+1}+1} = D_j\right)$. For more details, please refer to Appendix 1.

Fig. 3 illustrates such LLM-based arithmetic encoding and decoding, where the LLM provides a probability interval according to the text sequence $\boldsymbol{s}$. Unlike the online setting, which trains the model on the data to be compressed, this paper assumes the availability of a well-trained LLM and employs it to compress different datasets, following the offline setting used in Ref. [24].

Remark 1: Ref. [25] figures out that the expected code length achieved by leveraging LLM as a compressor could be represented as the cross-entropy, that is,

$$H(\rho, \tilde{\rho}) \coloneqq \mathbb{E}_{\boldsymbol{s}\sim\rho}\left[\sum_{i=1}^{n} -\log_2 \tilde{\rho}\left(s_i|\boldsymbol{s}_{<i}\right)\right] \quad (3),$$

where ρ is the source distribution and $\tilde{\rho}$ is the estimation of ρ via a parametric probabilistic model. Hence, the compres-

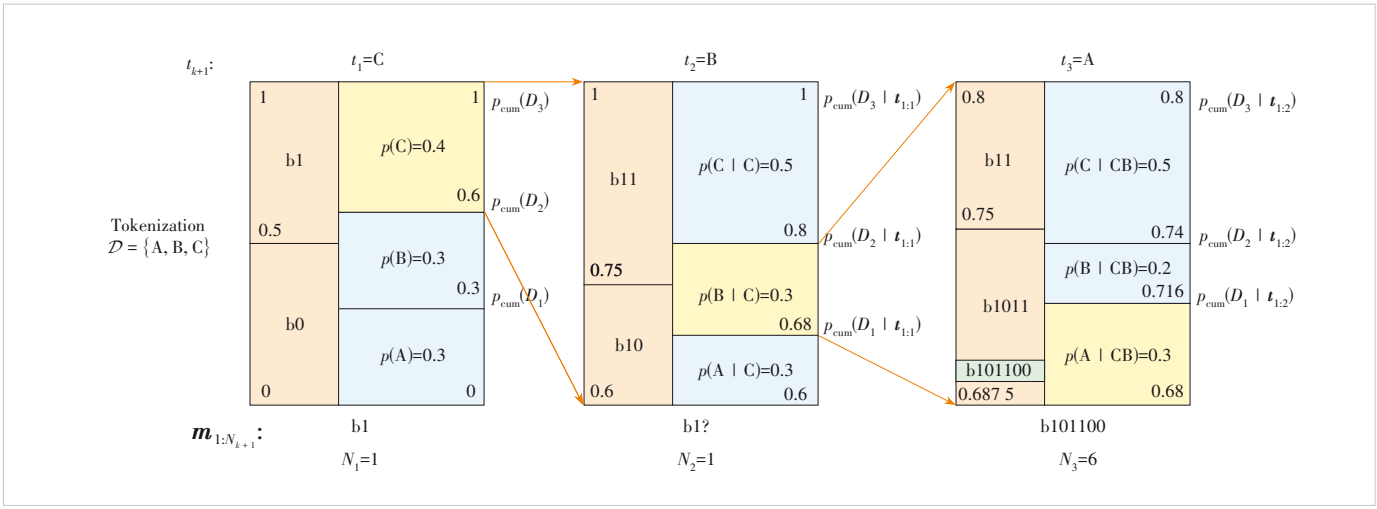


Figure 2. An example of arithmetic coding

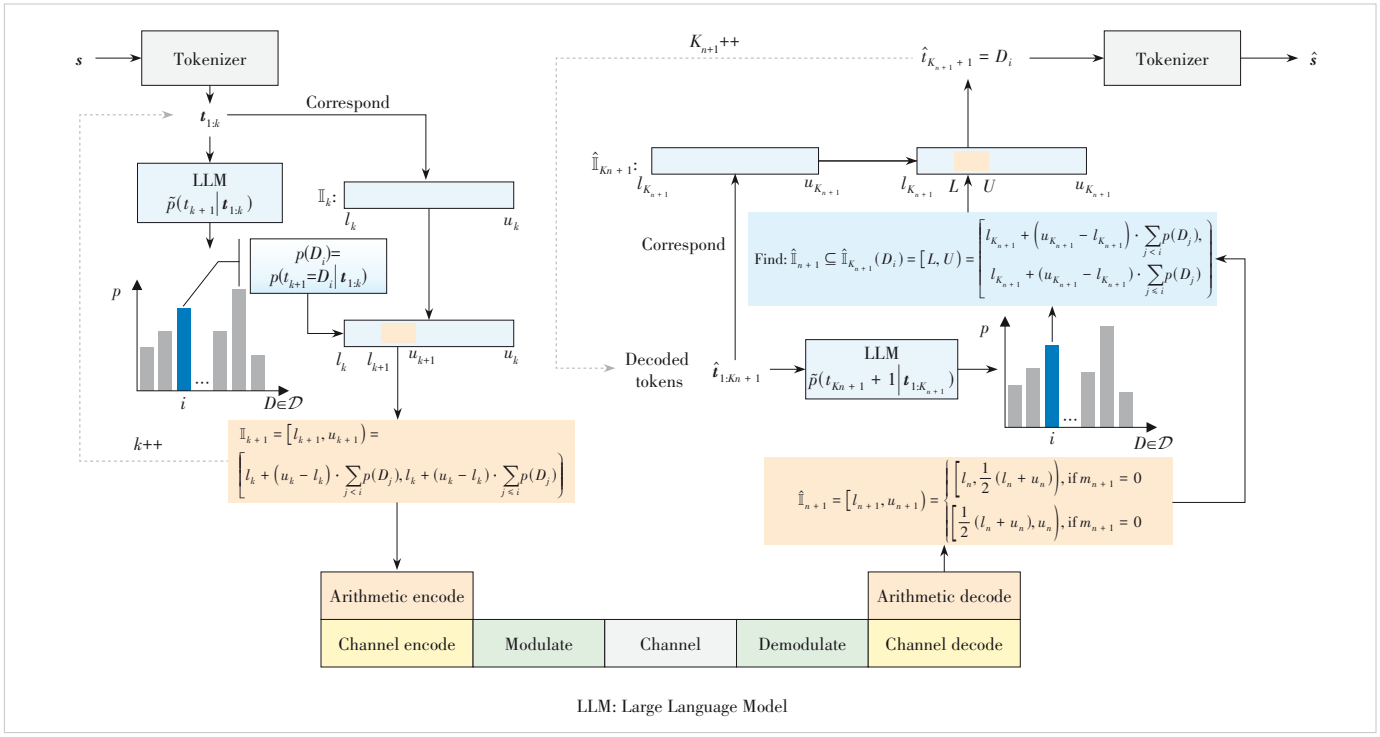


Figure 3. LLM-based arithmetic encoding and decoding

sion shares the same training objective as prediction. Therefore, it can be interpreted as the link between the model log-loss and the compression rate, providing theoretical support for the employment of LLM for source coding.

3.2 Error Correction Code Transformer

ECCT[48] belongs to the complementary Transformer-like module. It ensures the channel decoding reliability. Notably, ECCT involves specific preprocessing and post-processing steps to avoid overfitting effectively. Without the loss of generality, before preprocessing, the syndrome of codes is defined by

$$\mathrm{syn}(\boldsymbol{y}) \coloneqq \boldsymbol{H}\boldsymbol{y}_b = \boldsymbol{H}\mathrm{sign\_to\_bin}(\boldsymbol{y}) = \frac{1}{2}\boldsymbol{H}\left(1 - \mathrm{sign}(\boldsymbol{y})\right) \in \{0,1\}^{N-K} \tag{4}$$

This should be checked first upon receiving the signal since corruption could be detected immediately if $\mathrm{syn}(\boldsymbol{y})$ is a non-zero vector. In other words, an all-zero syndrome ensures that the received signal suffers no distortion. Note that the function $\mathrm{sign\_to\_bin}(\cdot)$ could be viewed as a hard decision on $\boldsymbol{y}$ and $\mathrm{sign}(\cdot)$ here denotes a sign function defined by

$$\mathrm{sign}(y)\begin{cases} 1, & y > 0 \\ 0, & y = 0 \\ -1, & y < 0 \end{cases} \tag{5}$$

Next, ECCT constructs a $2N - K$ dimensional input embedding by concatenating the element-wise magnitude and syndrome vectors, such that

$$\tilde{\boldsymbol{y}} \coloneqq \left[|\boldsymbol{y}|, \mathrm{syn}(\boldsymbol{y})\right] \in \mathbb{R}^{2N-K} \tag{6}$$

where $[\cdot,\cdot]$ denotes vector/matrix concatenation and $|\boldsymbol{y}|$ denotes the absolute value (magnitude) of $\boldsymbol{y}$.

The objective of the decoder is to predict the multiplicative noise $\tilde{\boldsymbol{z}}$ from $\boldsymbol{y}$, where $\boldsymbol{y} = h\boldsymbol{x}_s + \boldsymbol{z} = \boldsymbol{x}_s(h + \boldsymbol{x}_s\boldsymbol{z}) = \boldsymbol{x}_s\tilde{\boldsymbol{z}}$. Compared to traditional Transformer architectures[19], ECCT introduces two additional modules for positional reliability encoding and code aware self-attention, as shown in Fig. 4. Notably, ECCT processes the channel output $\boldsymbol{y}$ as input and generates a prediction $\hat{\boldsymbol{z}}$ of the multiplicative noise $\tilde{\boldsymbol{z}}$. The key differences between ECCT and traditional Transformer architectures are highlighted in the dashed-line boxes in Fig. 4. Implementation details are provided in Appendix 2.

Finally, the training process aims to minimize the binary cross entropy (BCE) loss between the predicted noise $\hat{\boldsymbol{z}}$ and the multiplicative noise $\tilde{\boldsymbol{z}}$, given by

$$\mathrm{loss} = \mathrm{BCELoss}(\hat{\boldsymbol{z}}, \tilde{\boldsymbol{z}}) = -\frac{1}{N}\sum_i \begin{pmatrix} \mathrm{bin}(\tilde{z}_i) \cdot \log\left(\sigma(\hat{z}_i)\right) + \\ \left(1 - \mathrm{bin}(\tilde{z}_i)\right) \cdot \log\left(1 - \sigma(\hat{z}_i)\right) \end{pmatrix} \tag{7}$$

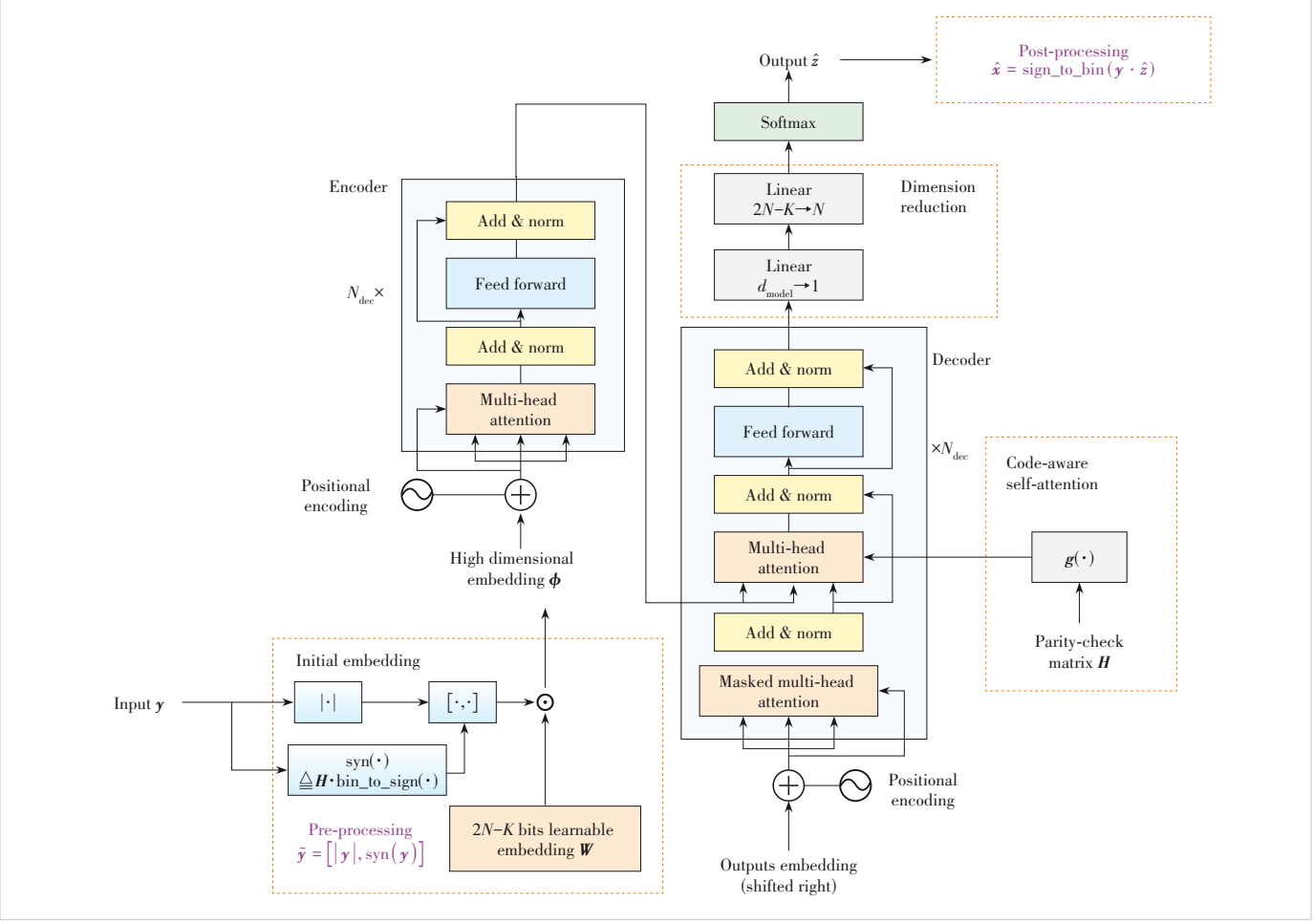


Figure 4. ECCT architecture

where $\sigma(\cdot)$ denotes the sigmoid activation function.

Remark 2: The estimation of multiplicative noise is represented as $\hat{z} = f(\tilde{y})$, while the post-processing step estimates x by $\hat{x} = \text{sign\_to\_bin}(y \cdot f(\tilde{y}))$. Given that, for correct estimation, $\text{sign}(\hat{z}) = \text{sign}(\tilde{z})$. Therefore,

$$\hat{x} = \text{sign\_to\_bin}(y \cdot f(\tilde{y})) = \text{sign\_to\_bin}(x_s \tilde{z} \cdot \hat{z}) = \text{sign\_to\_bin}(x_s) = x \quad (8).$$

In other words, ECCT contributes to noise-free channel coding.

4 Experiments

In this section, we compare the proposed method with traditional SSCC approaches and existing JSCC solutions under both AWGN and Rayleigh fading channels.

4.1 Simulation Settings

To facilitate comparison, we utilize a pre-processed dataset consisting of the standard proceedings of the European Parliament[49]. A segment of this dataset is selected as an example and fed as the source to a Generative Pre-Trained Transformer 2 (GPT2)[50] model for source coding. In this numerical experiment, we primarily choose the smallest GPT2-base model with 124 million parameters, while larger models (e. g., the 355-million-parameter GPT2-medium, the 774-million-parameter GPT2-large, and the1.5-billion-parameter GPT2-XL) are subsequently used for comparative analysis. Arithmetic coding based on the LLM is configured with a precision limit of 31 bits. For channel coding, we adopt an LDPC code with an information word length of 24 and a codeword length of 49, denoted as LDPC(49, 24), resulting in a code rate close to 1/2. Subsequently, ECCT is used for algebraic block code decoding, which is capable of training on diverse error correction codes. The hyperparameter settings for ECCT training are detailed in Table 2. For comparative analysis, we select Deep Learning-Based Semantic Communication (DeepSC)[12], Universal Transformer (UT)[14], and UT

with quantization* as benchmark JSCC algorithms. Considering the subsequent signal-to-noise ratio (SNR) performance comparison, both algorithms are trained using mixed precision (i.e., float16), which, as discussed later, has a minimal negative impact on SNR computation. Key parameters used for training DeepSC and UT are also listed in Table 2. Besides, the traditional approach employs Huffman coding for source coding. Furthermore, bilingual evaluation understudy (BLEU)[51] and semantic similarity measured by BERT[52] are used to measure performance, as these metrics are widely recognized in natural language processing.

Most existing SemCom works evaluate the performance with respect to the SNR $= 10\log_{10}\left(E_{\text{tb}}/N_0\right)$ dB, where E_{tb} denotes the energy associated with transmitting a single bit after source/channel coding and digital modulation, and N_0 represents the noise power spectral density. However, since different coding and modulation schemes across different communication methodologies result in varying numbers of bits transmitted over the physical channel, such a comparative metric of SNR ignores the differences in delivering different numbers of bits. Instead, referring to the total energy consumption E_{total} by sending $\text{Num}_{\text{unified}}$ bits through the physical channel in an LLM-based SSCC system, we propose a consistent definition of SNR in terms of an LLM-based SSCC reference baseline $\text{SNR}_{\text{unified}}$, as a function of the practically employed bits Num.

Table 2. Mainly used hyperparameters in the experiments

Model	Hyperparameter	Value
ECCT	Learning rate	10^{-4}
	Batch size	128
	Number of decoder layers	6
	Dimension of embedding	32
	Number of attention heads	8
DeepSC	Learning rate	10^{-4}
	Batch size	64
	Number of encoder/decoder layers	4
	Dimension of embedding	128
	Dimension of FFN	512
	Number of attention heads	8
UT	Learning rate	10^{-4}
	Batch size	64
	Number of encoder/decoder layers	3
	Dimension of embedding	128
	Dimension of FFN	1 024
	Number of attention heads	8

DeepSC: Deep Learning-Based Semantic Communication
ECCT: Error Correction Code Transformer
FFN: Feed Forward Network
LLM: Large Language Model
UT: Universal Transformer

Mathematically, this is expressed as:

$$\text{SNR} = 10\log_{10}\left(\frac{E_{\text{total}}}{N_0 \cdot \text{Num}}\right) = 10\log_{10}\left(\frac{E_{\text{total}}}{N_0 \cdot \text{Num}_{\text{unified}}} \times \frac{\text{Num}_{\text{unified}}}{\text{Num}}\right) = \text{SNR}_{\text{unified}} + 10\log_{10}\left(\frac{\text{Num}_{\text{unified}}}{\text{Num}}\right) \tag{9},$$

where $\text{SNR}_{\text{unified}}$ is used as an independent variable for aligning E_{total}, while for bit-oriented transmission (resp. float-based JSCC), Num denotes the number of bits (resp. float vectors) transmitted through the channel.

On the other hand, as mentioned in Section 1 and Ref. [33], deep learning-based JSCC systems extract the semantic feature of information to embed vectors in latent space, which is incompatible with digital communication systems. For JSCC methodologies like UT[14] and DeepSC[12], transmitting a float number certainly consumes far more energy than delivering a binary bit. In this case, if float16 is adopted, we can roughly assume it consumes an additional $10 \times \log_{10}(16) \approx 12.041$ dB. Hence, for the float-based JSCC methods, the unified evaluation metric is further modified to maintain a consistent energy consumption across different methodologies. In summary,

$$\text{SNR} = \begin{cases} \text{SNR}_{\text{unified}} + 10\log_{10}\left(\frac{\text{Num}_{\text{unified}}}{\text{Num}}\right) + 12.041, & \text{float based} \\ \text{SNR}_{\text{unified}} + 10\log_{10}\left(\frac{\text{Num}_{\text{unified}}}{\text{Num}}\right), & \text{otherwise} \end{cases} \tag{10}.$$

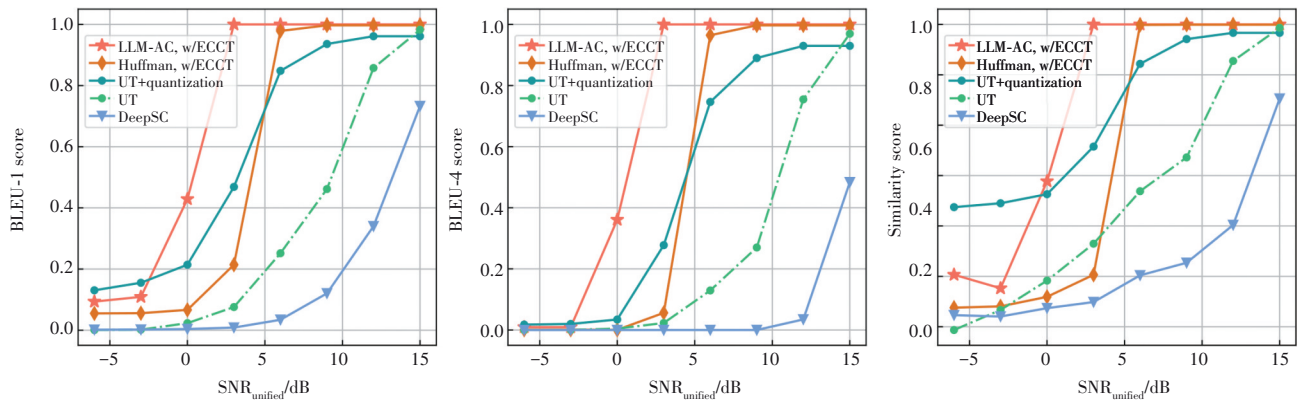
During evaluation, experiments are conducted for different schemes in terms of $\text{SNR}_{\text{unified}}$.

4.2 Numerical Results

In this section, we implement the GPT2-base model as a compressor and ECCT-complemented LDPC$(49, 24)$ as the error correction code, and compare it with DeepSC, UT, UT with quantization and the classical SSCC encompassing Huffman coding and ECCT (Fig. 5). The results demonstrate the superior performance of the proposed SSCC over the other three schemes. Similarly, we evaluate the performance under a Rayleigh fading channel in Fig. 6, where the results show that our system has a clear advantage in terms of the word-level BLEU score. However, in terms of semantic similarity, both the LLM-based and the traditional Huffman-based SSCC systems exhibit some disadvantages at lower SNRs, but still maintain a noticeable advantage at high SNRs.

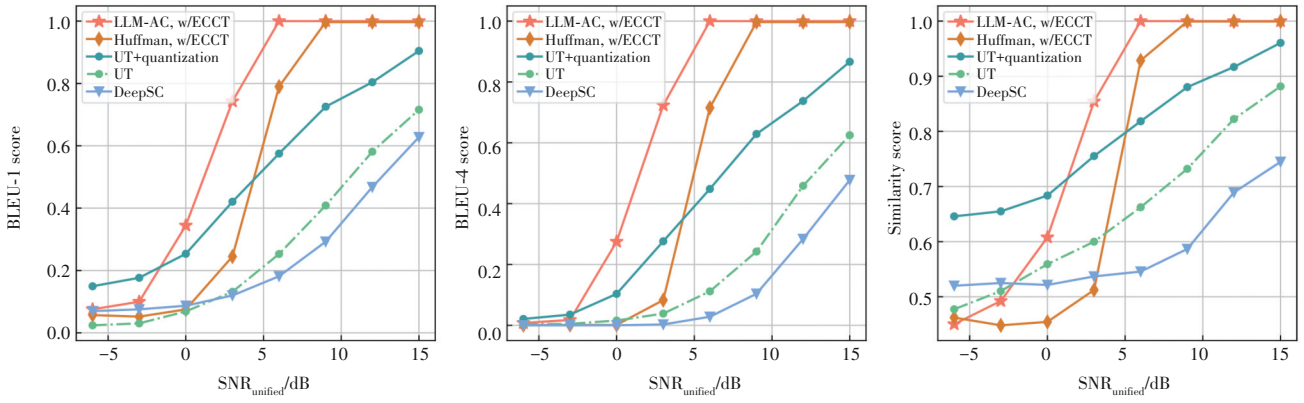
In addition to presenting our key experimental results

* Compared to DeepSC and UT that directly transmit the encodes floats, UT with quantization maps the encoding results to a fixed number (30) of bits for transmission.



AC: arithmetic coding
BLEU: bilingual evaluation understudy
DeepSC: Deep Learning-Based Semantic Communication
ECCT: Error Correction Code Transformer
LLM: Large Language Model
SNR: signal-to-noise ratio
UT: Universal Transformer

Figure 5. BLEU and similarity scores versus SNR$_{unified}$ are evaluated for the same number of transmitted symbols. The proposed LLM-based SSCC is compared with Huffman coding with LDPC(49, 24) in BPSK, DeepSC, UT, and UT with quantization under the AWGN channel

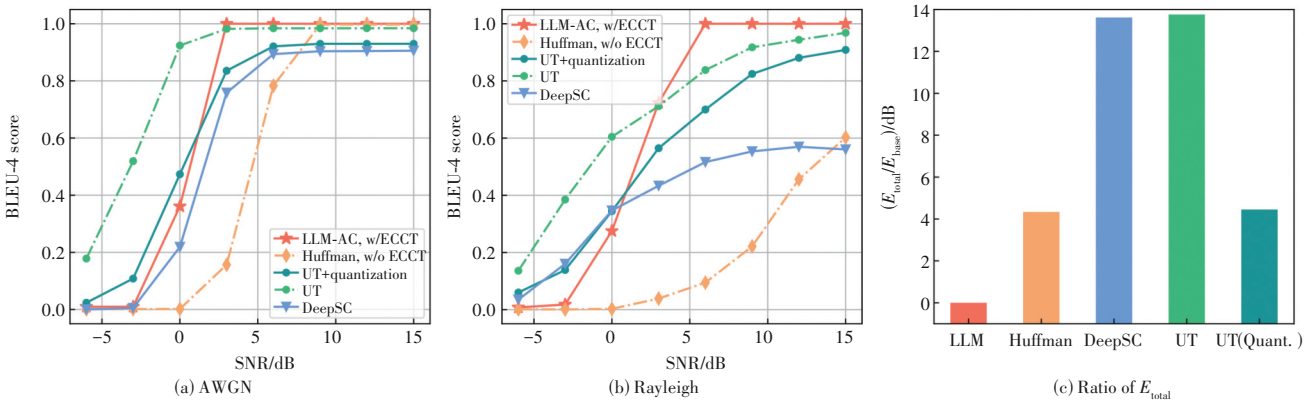


AC: arithmetic coding
BLEU: bilingual evaluation understudy
DeepSC: Deep Learning-Based Semantic Communication
ECCT: Error Correction Code Transformer
LLM: Large Language Model
SNR: signal-to-noise ratio
UT: Universal Transformer

Figure 6. BLEU and similarity scores versus SNR$_{unified}$ are evaluated for the same number of transmitted symbols. The proposed LLM-based SSCC is compared with Huffman coding with LDPC(49, 24) in BPSK; DeepSC, UT, and UT with quantization trained under the Rayleigh fading channel

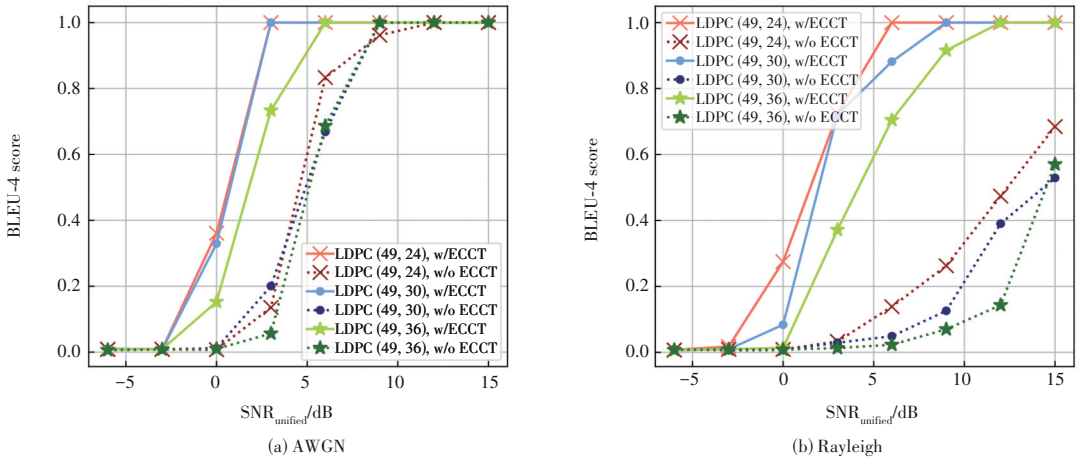
with SNR$_{unified}$ as the alignment metric, Fig. 7 provides performance comparisons using traditional SNR alignment, as well as the ratio of E_{total} used by different systems over our LLM-based solution. This illustrates the additional energy consumption of JSCC systems in achieving superior performance. Apparently, JSCC systems in SemCom achieve significant gains mainly due to the extra energy consumption.

Afterward, we validate the contributing effectiveness of ECCT[32] by comparing the performance of error correction codes with different coding rates under the same code length. Without loss of generality, the evaluation results based on LDPC under AWGN and Rayleigh fading channels are given in Fig. 8. Notably, while the work in Ref. [32] does not include Rayleigh channel results, inspired by the subsequent work on Denoising Diffusion Error Correction Codes (DDECC[53]), we extend ECCT to Rayleigh channels in a similar manner. It can be observed from Fig. 8 that compared to traditional LDPC decoding methods such as bit-flipping, ECCT provides consistent performance improvements. Furthermore, for error correction codes of the same length, lower coding rates demonstrate better recovery of noisy signals under the same SNR. More importantly, without ECCT, traditional algorithms struggle to decode noisy signals under Rayleigh channels effectively, and reducing the coding rate slightly improves the performance trivially. However, ECCT trained under the Rayleigh channel achieves as competitive



AC: arithmetic coding
AWGN: additive white Gaussian noise
BLEU: bilingual evaluation understudy
DeepSC: Deep Learning-Based Semantic Communication
ECCT: Error Correction Code Transformer
LLM: Large Language Model
SNR: signal-to-noise ratio
UT: Universal Transformer

Figure 7. BLEU-4 score versus SNR is evaluated for the same number of transmitted symbols. The proposed LLM-based SSCC is compared with Huffman coding with LDPC(49, 24) in BPSK (without ECCT), DeepSC, UT, and UT with quantization trained under (a) AWGN and (b) Rayleigh fading channels; (c) shows the ratio of E_{total} among different systems



AWGN: additive white Gaussian noise
BLEU: bilingual evaluation understudy
ECCT: Error Correction Code Transformer
LDPC: Low-Density Parity-Check
SNR: signal-to-noise ratio

Figure 8. BLEU-4 score versus $SNR_{unified}$ for the same number of transmitted symbols, with different code rates using LDPC(49, 24)/ LDPC(49, 30)/ LDPC(49, 36) in BPSK, compared with the situations removing ECCT, under (a) AWGN and (b) Rayleigh fading channels

performance as that under the AWGN channel.

Considering the scaling law and emergent abilities of LLMs, we evaluate the performance by combining different models from the GPT2 family with ECCT-complemented LDPC channel coding (i. e., a high-rate LDPC(121, 110) code). Both the end-to-end SSCC performance in Fig. 9 and the compression rate in Fig. 10 indicate a notable performance improvement after adopting a model larger than GPT2. However, the performance difference among GPT2-medium, GPT2-large, and GPT2-XL is marginal. We hypothesize that while increasing the model size beyond a certain threshold contributes significantly to system performance, variations within a specific range of model scales yield diminishing returns. Furthermore, inspired by Ref. [54], the performance comparison with Zlib and static Huffman coding in Fig. 10 demonstrates that LLM-based arithmetic coding significantly outperforms traditional methods. Moreover, a scaling law is observed in the compression performance, which somewhat corroborates the findings of Ref. [54].

The experimental results presented in Table 3 further investigate the influence of the token block size on performance. It can be observed that at higher SNR levels, the performance generally improves as the block size increases, indicating that larger block sizes facilitate enhanced semantic

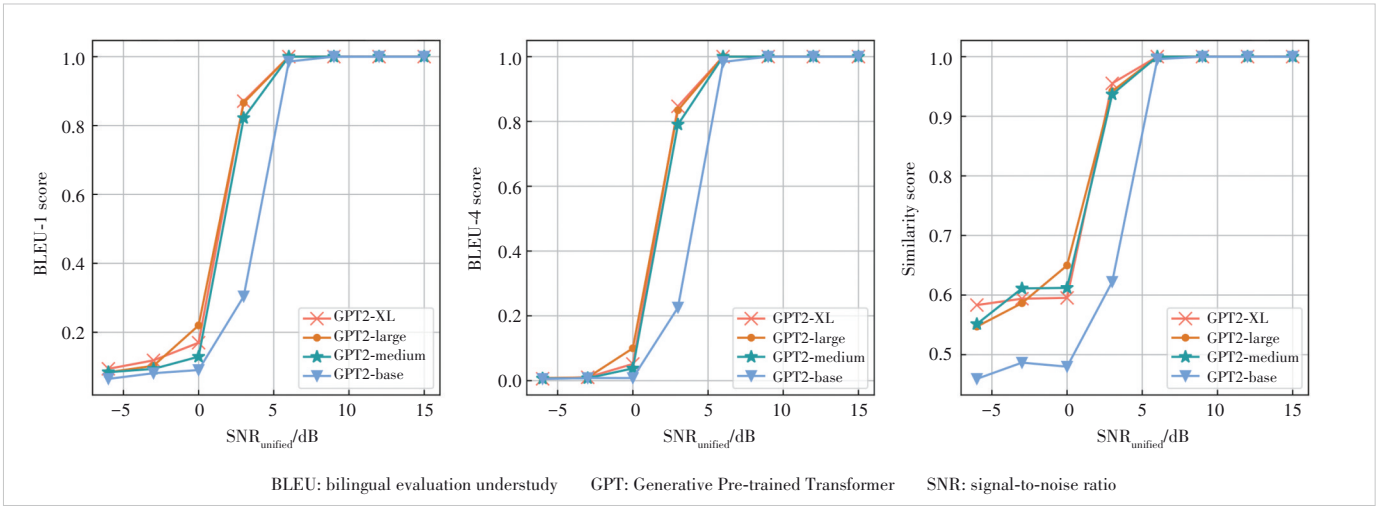


Figure 9. BLEU and similarity scores of models versus SNR$_{unified}$, with different parameter scales (GPT2, GPT2-medium, GPT2-large, GPT2-XL), using LDPC(121, 110) as the error correction code

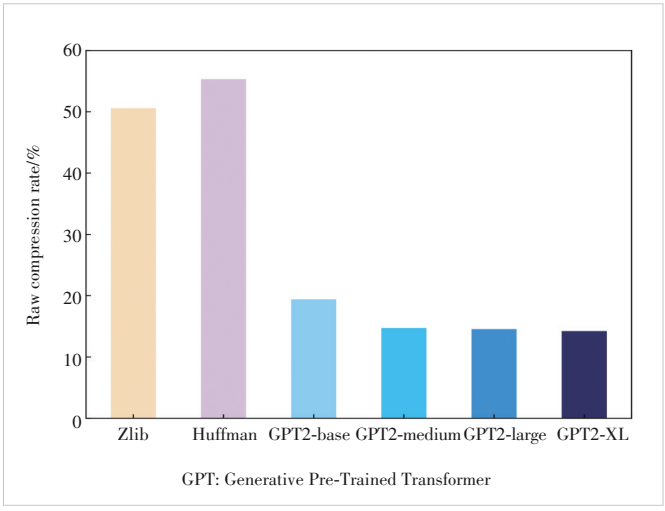


Figure 10. Compression rate comparison between traditional methods (Zlib and Huffman coding) and LLM-AC

Table 3. Influence of token block sizes on system performance during LLM-based arithmetic source encoding for SNR={−6, 0, 6}

Block size	Similarity			BLEU-1			BLEU-4		
	−6	0	6	−6	0	6	−6	0	6
16	**0.770 8**	0.915 7	0.999 3	**0.197 5**	**0.645 2**	0.987 7	**0.007 2**	**0.508 1**	0.983 0
32	0.712 3	**0.935 9**	0.998 4	0.172 5	0.584 2	0.978 7	0.005 5	0.466 6	0.969 4
64	0.700 1	0.893 8	0.999 9	0.116 0	0.580 1	0.996 9	0.001 8	0.427 0	0.992 2
128	0.758 7	0.857 3	**0.999 9**	0.183 1	0.434 4	**0.999 9**	0.003 8	0.252 9	**0.999 9**

BLEU: bilingual evaluation understudy SNR: signal-to-noise ratio
LLM: Large Language Model

preservation due to their ability to capture more contextual information. However, at lower SNR levels, the performance declines with an increase in the block size, suggesting that smaller blocks may be more resilient to avoid cumulative source decoding errors in these challenging scenarios.

5 Conclusions and Discussions

In this paper, we present a comprehensive analysis and evaluation of SSCC, with a comprehensive comparison to JSCC in the context of SemCom. Our proposed SSCC framework, which integrates LLMs for source coding and ECCT for enhanced channel coding, demonstrates significant performance improvements over JSCC in terms of recovery performance at both the word and semantic levels under both AWGN and Rayleigh fading channels. This highlights the potential effectiveness of SSCC in information transmission. In particular, through extensive experiments, we validate the strong compressive capability of LLMs to eliminate redundancy in text and the robustness of ECCT in enhancing decoding reliability under various channel conditions. In a word, separate source channel coding is still what we need.

Nevertheless, despite the validated performance superiority of SSCC, there remain several important issues worthy of further clarification and investigation.

1) The performance evaluation of text transmission sounds inspiring. The proposed SSCC framework is channel-agnostic, while given the well-known generality issues, the DNN-based JSCC faces a performance decline when the channel changes significantly. However, an extension to image transmission can be more challenging, and several issues like sequential tokenization require effective solutions. In this regard, potential solutions can incorporate patch division from Vision Transformer (ViT) to replace text tokenization, thereby segmenting images into semantic units for encoding. Consequently, the LLM-AC text predictor can be transformed into a probability modeler for image patches. Furthermore, the iterative decoding of ECCT can mitigate the error propagation issues in traditional JSCC, which is particularly crucial for multimedia transmission with high-

fidelity requirements. On the other hand, our experimental experience indicates the accuracy of channel coding is of vital importance for end-to-end performance. Hence, we only consider a relatively low, fixed code rate here. However, systematic tuning of the code rate is also a worthwhile direction for future research.

2) This paper only considers the classical JSCC design, while ignoring the latest quantization and digital modulation techniques that have emerged in the development of JSCC. For example, Refs. [9] and [10] show that utilizing a sparsity module to quantize the image embedding can yield significant performance gain. However, Refs. [9] and [10] have not compared their approaches with the remarkable capabilities of LLMs, and thus it remains unclear whether these amendments would enable JSCC to surpass LLM-based SSCC in a fair comparison. Nevertheless, given the inspiring results in this paper, there is no doubt that SSCC should be carefully improved rather than dismissed.

3) What we have to acknowledge is that integrating LLMs into the SSCC framework requires substantial computational resources for both encoding and decoding processes. However, we currently leverage pre-trained LLMs, which possess inherent generalization capabilities and can handle a broad range of natural language datasets. This contrasts with JSCC methods, which often rely on training with specific datasets to achieve superior performance. If a specific dataset is employed, we can explore the possibility of model distillation. By utilizing a Transformer model with significantly fewer parameters while retaining the LLM's tokenizer and performing self-supervised training on the target dataset, we can substantially reduce computational overhead while maintaining reasonable performance. We will further investigate model distillation in future work.

4) The discussions on JSCC are limited to the scenario to recover the semantics as accurately as possible. For SemCom[1], effectiveness-level or pragmatic communications may target at accomplishing different tasks under remotely controlled, noisy environments, rather than simple recovery of accurate semantics. In such cases, the underlying philosophy of JSCC may offer unique advantages.

5) Extensive works have been conducted to improve the performance of model-free decoders. For example, Ref. [55] proposes a systematic and double mask eliminating the difficulty of identifying the optimal parity-check matrix (PCM) from numerous candidatesfrom the same code. For performance enhancement on moderate code-length decoding, U-ECCT is proposed in Ref. [56] inspired by U-Net, while in Ref. [53], the Denoising Diffusion Probabilistic Model (DDPM)[57] is employed to model the transmission over channels as a diffusion process. Furthermore, a foundation model for channel codes is proposed in Ref. [58] for application to unseen codes. Therefore, these recent works are worthy to be evaluated in the SSCC framework.

References

[1] LU Z L, LI R P, LU K. Semantics-empowered communication: a tutorial-cum-survey [J]. IEEE communications surveys and tutorials, 2024, 26 (1): 41 – 79. DOI: 10.1109/COMST.2023.333334
[2] KURKA D B, GÜNDÜZ D. DeepJSCC-f: deep joint source-channel coding of images with feedback [J]. IEEE journal on selected areas in information theory, 2020, 1(1): 178 – 193. DOI: 10.1109/JSAIT.2020.2987203
[3] BAO Z C, LIANG H T, DONG C, et al. MDVSC: wireless model division video semantic communication for 6G [C]//Proc. IEEE Globecom Workshops (GC Wkshps). IEEE, 2023: 1572 – 1578. DOI: 10.1109/GCWkshps58843.2023.10464666
[4] JIA Y J, HUANG Z, LUO K, et al. Lightweight joint source-channel coding for semantic communications [J]. IEEE communications letters, 2023, 27(12): 3161 – 3165. DOI: 10.1109/LCOMM.2023.3329533
[5] LIU S C, GAO Z, CHEN G J, et al. Transformer-based joint source channel coding for textual semantic communication [C]//Proc. IEEE/CIC International Conference on Communications in China (ICCC). IEEE, 2023: 1 – 6. DOI: 10.1109/ICCC57788.2023.10233424
[6] LIU X Y, HUANG Z, ZHANG Y L, et al. CNN and attention-based joint source channel coding for semantic communications in WSNs [J]. Sensors, 2024, 24(3): 957. DOI: 10.3390/s24030957
[7] LU Z L, LI R P, LEI M, et al. Self-critical alternate learning based semantic broadcast communication [J]. IEEE transactions on communications, 2024: 1. DOI: 10.1109/tcomm.2024.3487513
[8] TONG W J, LIU F F, SUN Z F, et al. Image semantic communications: an extended rate-distortion theory based scheme [C]//Proc. IEEE Globecom Workshops (GC Wkshps). IEEE, 2022: 1723 – 1728. DOI: 10.1109/GCWkshps56602.2022.10008733
[9] TONG S Y, YU X X, LI R P, et al. Alternate learning based sparse semantic communications for visual transmission [C]//Proc. 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC). IEEE, 2023. DOI: 10.1109/pimrc56721.2023.10293971
[10] TONG S Y, YU X X, LI R P, et al. Alternate learning-based SNR-adaptive sparse semantic visual transmission [J]. IEEE transactions on wireless communications, 2025, 24: 1737 – 1752. DOI: 10.1109/TWC.2024.3512652
[11] WANG J, WANG S X, DAI J C, et al. Perceptual learned source-channel coding for high-fidelity image semantic transmission [C]//Proc. IEEE Global Communications Conference. IEEE, 2022: 3959 – 3964. DOI: 10.1109/GLOBECOM48099.2022.10001359
[12] XIE H Q, QIN Z J, LI G Y, et al. Deep learning enabled semantic communication systems [J]. IEEE transactions on signal processing, 2021, 69: 2663 – 2675. DOI: 10.1109/tsp.2021.3071210
[13] ZHANG W Y, BAI K Y, ZEADALLY S, et al. DeepMA: end-to-end deep multiple access for wireless image transmission in semantic communication [J]. IEEE transactions on cognitive communications and networking, 10(2): 387 – 402. DOI: 10.1109/tccn.2023.3326302
[14] ZHOU Q Y, LI R P, ZHAO Z F, et al. Semantic communication with adaptive universal transformer [J]. IEEE wireless communications letters, 2022, 11(3): 453 – 457. DOI: 10.1109/LWC.2021.3132067
[15] GOYAL M, TATWAWADI K, CHANDAK S, et al. DeepZip: lossless data compression using recurrent neural networks [C]//Proc. Data Compression Conference (DCC). IEEE, 2019. DOI: 10.1109/dcc.2019.00087
[16] BELLARD F. Lossless data compression with neural networks [EB/OL]. (2019-05-04)[2024-11-20]. https://bellard.org/nncp/nncp.pdf
[17] LIU Q, XU Y L, LI Z. DecMac: a deep context model for high efficiency arithmetic coding [C]//Proc. International Conference on Artificial Intelligence in Information and Communication (ICAIIC). IEEE, 2019. DOI: 10.1109/icaiic.2019.8668843
[18] GOYAL M, TATWAWADI K, CHANDAK S, et al. DZip: improved

general-purpose loss less compression based on novel neural network modeling [C]//Proc. Data Compression Conference (DCC). IEEE, 2021. DOI: 10.1109/dcc50243.2021.00023
[19] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need [C]//Proc. 31st International Conference on Neural Information Processing Systems. NIPS, 2017: 6000–6010
[20] HUANG C, XIE Y Q, JIANG Z Y, et al. Approximating human-like few-shot learning with GPT-based compression [EB/OL]. (2023-08-14) [2024-11-12]. https://arxiv.org/abs/2308.06942v1
[21] MITTU F, BU Y H, GUPTA A, et al. FineZip: pushing the limits of large language models for practical lossless text compression [EB/OL]. (2024-09-25)[2024-11-12]. https://arxiv.org/abs/2409.17141v1
[22] MAO Y, CUI Y F, KUO T W, et al. A fast transformer-based general-purpose lossless compressor [EB/OL]. (2022-03-30) [2024-11-12]. https://arxiv.org/abs/2203.16114v2
[23] NARASHIMAN S S, CHANDRACHOODAN N. AlphaZip: neural network-enhanced lossless text compression [EB/OL]. (2024-09-23) [2024-11-12]. https://arxiv.org/abs/2409.15046v1
[24] VALMEEKAM C S K, NARAYANAN K, KALATHIL D, et al. LLMZip: Lossless text compression using large language models [EB/OL]. (2023-06-06)[2024-11-12] https://arxiv.org/abs/2306.04050v2
[25] DELÉTANG G, RUOSS A, DUQUENNE P-A, et al. Language modeling is compression [EB/OL]. (2023-09-19)[2024-10-20]. https://arxiv.org/abs/2309.10668
[26] BOSE R C, RAY-CHAUDHURI D K. On a class of error correcting binary group codes [J]. Information and control, 1960, 3(1): 68–79. DOI: 10.1016/s0019-9958(60)90287-4
[27] GALLAGER R. Low-density parity-check codes [J]. IRE transactions on information theory, 1962, 8(1): 21–28. DOI: 10.1109/TIT.1962.1057683
[28] ARIKAN E. Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels [J]. IEEE transactions on information theory, 2009, 55(7): 3051–3073. DOI: 10.1109/TIT.2009.2021379
[29] NACHMANI E, BE′ERY Y, BURSHTEIN D. Learning to decode linear codes using deep learning [C]//Proc. 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton). IEEE, 2016: 341–346. DOI: 10.1109/ALLERTON.2016.7852251
[30] NACHMANI E, MARCIANO E, LUGOSCH L, et al. Deep learning methods for improved decoding of linear codes [J]. IEEE journal of selected topics in signal processing, 12(1): 119–131. DOI: 10.1109/jstsp.2017.2788405
[31] NACHMANI E, WOLF L. Hyper-graph-network decoders for block codes [C]//Proc. 33st International Conference on Neural Information Processing Systems. NIPS, 2019: 2329–2339
[32] CHOUKROUN Y, WOLF L. Error correction code transformer [C]//Proc. 36st International Conference on Neural Information Processing Systems. NIPS, 2022: 38695–38705
[33] HUANG J H, YUAN K, HUANG C, et al. D2-JSCC: digital deep joint source-channel coding for semantic communications [EB/OL]. (2024-03-12)[2024-11-20]. https://arxiv.org/abs/2403.07338v3
[34] JIANG P W, WEN C K, YI X P, et al. Semantic communications using foundation models: design approaches and open issues [J]. IEEE wireless communications, 2024, 31(3): 76–84. DOI: 10.1109/MWC.002.2300460
[35] LIANG C S, DU H Y, SUN Y, et al. Generative AI-driven semantic communication networks: architecture, technologies and applications [J]. IEEE transaction on cognitive communications and networking, 2024, early access. DOI: 10.1109/TCCN.2024.3435524
[36] JIANG F B, PENG Y B, DONG L, et al. Large AI model-based semantic communications [J]. IEEE wireless communications, 31(3): 68–75. DOI: 10.1109/mwc.001.2300346
[37] GRASSUCCI E, BARBAROSSA S, COMMINIELLO D. Generative semantic communication: diffusion models beyond bit recovery [EB/OL]. (2023-06-07)[2024-11-12]. https://arxiv.org/abs/2306.04321v1
[38] CHANG M K, HSU C T, YANG G C. GenSC: generative semantic communication systems using BART-like model [J]. IEEE communications letters, 2024, 28(10): 2298–2302. DOI: 10.1109/LCOMM.2024.3450309
[39] GUO S S, WANG Y H, LI S J, et al. Semantic importance-aware communications using pre-trained language models [J]. IEEE communications letters, 2023, 27(9): 2328–2332. DOI: 10.1109/LCOMM.2023.3293805
[40] XIE H Q, QIN Z J, TAO X M, et al. Toward intelligent communications: large model empowered semantic communications [J]. IEEE communications magazine, 2025, 63(1): 69–75. DOI: 10.1109/MCOM.001.2300807
[41] QIAO L, MASHHADI M B, GAO Z, et al. Latency-aware generative semantic communications with pre-trained diffusion models [EB/OL]. (2024-03-05)[2024-11-12]. https://arxiv.org/abs/2403.17256v2
[42] JIANG F B, DONG L, PENG Y B, et al. Large AI model empowered multimodal semantic communications [J]. IEEE communications magazine, 2025, 63(1): 76–82. DOI: 10.1109/mcom.001.2300575
[43] YANG W T, XIONG Z H, MAO S W, et al. Rethinking generative semantic communication for multi-user systems with large language models [EB/OL]. (2024-08-16) [2024-11-12]. https://arxiv. org/abs/2408.08765v3
[44] SHANNON C E. A mathematical theory of communication [J]. Bell system technical journal, 1948, 27(3): 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x
[45] RISSANEN J J. Generalized kraft inequality and arithmetic coding [J]. IBM journal of research and development, 1976, 20(3): 198–203. DOI: 10.1147/rd.203.0198
[46] PASCO R. Source coding algorithms for fast data compression (Ph.D. Thesis abstr.) [J]. IEEE transactions on information theory, 1977, 23(4): 548. DOI: 10.1109/TIT.1977.1055739
[47] HOWARD P G, VITTER J S. Arithmetic coding for data compression [J]. Proceedings of the IEEE, 1994, 82(6): 857–865. DOI: 10.1109/5.286189
[48] BENNATAN A, CHOUKROUN Y, KISILEV P. Deep learning for decoding of linear codes: a syndrome-based approach [C]//Proc. IEEE International Symposium on Information Theory (ISIT). IEEE, 2018: 1595–1599. DOI: 10.1109/ISIT.2018.8437530
[49] KOEHN P. Europarl: a parallel corpus for statistical machine translation [C]//Proc. Machine Translation Summit. International Association for Machine Translation, 2005: 79–86
[50] RADFORD A, WU J, CHILD R, et al. Language models are unsupervised multitask learners [EB/OL]. [2024-10-20]. https://cdn. openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf
[51] PAPINENI K, ROUKOS S, WARD T, et al. BLEU: a method for automatic evaluation of machine translation [C]//Proc. 40th Annual Meeting on Association for Computational Linguistics. USAACL, 2001. DOI: 10.3115/1073083.1073135
[52] DEVLIN J, CHANG M-W, LEE K, et al. BERT: pre-training of deep bidirectional transformers for language understanding [C]//Proc. 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), Association for Computational Linguistics, 2019: 4171–4186. DOI: 10.18653/v1/N19-1423
[53] CHOUKROUN Y, WOLF L. Denoising diffusion error correction codes [EB/OL]. (2022-09-16) [2024-11-12]. https://arxiv. org/abs/2209.13533v1
[54] HUANG Y Z, ZHANG J H, SHAN Z F, et al. Compression represents intelligence linearly [EB/OL]. (2024-04-15)[2024-11-12]. https://arxiv.org/abs/2404.09937v2
[55] PARK S J, KWAK H Y, KIM S H, et al. How to mask in error correction code transformer: systematic and double masking [EB/OL]. (2023-08-16)[2024-11-12]. https://arxiv.org/abs/2308.08128v2

[56] NGUYEN D T, KIM S. U-shaped error correction code transformers [J]. IEEE transactions on cognitive communications and networking, 2024: 1. DOI: 10.1109/tccn.2024.3482349
[57] HO J, JAIN A, ABBEEL P. Denoising diffusion probabilistic models [C]//Proc. 34th International Conference on Neural Information Processing Systems. NIPS, 2020: 6840 – 6851. DOI: 10.48550/arXiv.2006.11239
[58] CHOUKROUN Y, WOLF L. A foundation model for error correction codes [C]//12th International Conference on Learning Representations. ICLR, 2024. DOI: 10.48550/arXiv.2405.04050

Appendix 1: Pseudo code of finite precision arithmetic codec

Appendix A: Pseudo code for encoder

Algorithm 1 Finite-precision arithmetic encoding

Require: N_k: Current number of emitted bits m_{N_k}

Require: $p_{\text{cum}}\left(D_i|\boldsymbol{t}_{1:k}\right)$: Cumulative probability of token $t_{k+1}=D_i\in\mathcal{D}$ given first k tokens

Require: l_k, u_k: Current interval determined by the first k tokens

Require: ε_k: Number of scaling bits

Initialization:
$N_{k+1}\leftarrow N_k$
$l_{k+1}\leftarrow l_k+\left(u_k-l_k\right)p_{\text{cum}}\left(D_{i-1}|\boldsymbol{t}_{1:k}\right)$ // If $k=0$, use $p_{\text{cum}}\left(D_{i-1}\right)$
$h_{k+1}\leftarrow l_k+\left(u_k-l_k\right)p_{\text{cum}}\left(D_i|\boldsymbol{t}_{1:k}\right)$ // If $k=0$, use $p_{\text{cum}}\left(D_i\right)$
$\varepsilon_{k+1}\leftarrow\varepsilon_k$
Scaling:
while any of the scaling conditions is met **do**
  if $u_{k+1}<0.5$ **then**
    // Scaling 1
    $l_{k+1}, u_{k+1}\leftarrow 2l_{k+1}, 2u_{k+1}$
    $m_{N_{k+1}+1}\leftarrow 0$ // Emit one bit's 0
    $m_{N_{k+1}+2:N_{k+1}+1+\varepsilon_{k+1}}\leftarrow 1$ // Emit ε_{k+1} bits' 1
    $N_{k+1}\leftarrow N_{k+1}+1+\varepsilon_{k+1}$
    $\varepsilon_{k+1}\leftarrow 0$
  else if $l_{k+1}\geqslant 0.5$ **then**
    // Scaling 2
    $l_{k+1}, u_{k+1}\leftarrow 2\left(l_{k+1}-0.5\right), 2\left(u_{k+1}-0.5\right)$
    $m_{N_{k+1}+1}\leftarrow 1$ // Emit one bit's 1
    $m_{N_{k+1}+2:N_{k+1}+1+\varepsilon_{k+1}}\leftarrow 0$ // Emit ε_{k+1} bits' 0
    $N_{k+1}\leftarrow N_{k+1}+1+\varepsilon_{k+1}$
    $\varepsilon_{k+1}\leftarrow 0$
  else if $0.25\leqslant l_{k+1}<0.5\leqslant u_{k+1}<0.75$ **then**
    // Scaling 3
    $l_{k+1}, u_{k+1}\leftarrow 2\left(l_{k+1}-0.25\right), 2\left(u_{k+1}-0.25\right)$
    $\varepsilon_{k+1}\leftarrow\varepsilon_{k+1}+1$
  end if
end while
return $N_{k+1}, \boldsymbol{m}_{N_k+1:N_{k+1}}$ // Updated emitted bits

Appendix B: Pseudo code for decoder

Algorithm 2 Finite-precision arithmetic decoding

Require: K_n: Current number of decoded tokens

Require: $p_{\text{cum}}\left(D_i|\boldsymbol{t}_{1:K_n}\right)$: Cumulative probability of token $t_{K_{n+1}+1}=D_i\in\mathcal{D}$ given first K_n tokens

Require: l_n, u_n: Current interval determined by the first n bits

Require: l_{K_n}, u_{K_n}: Interval of sequence $t_{1:K_n}$ that has been decoded

Initialization:
$K_{n+1}\leftarrow K_n$
$l_{K_{n+1}}, h_{K_{n+1}}\leftarrow l_{K_n}, h_{K_n}$
if the $(n+1)$-th bit $m_{n+1}=0$ **then**
  $l_{n+1}, h_{n+1}\leftarrow l_n, \frac{1}{2}\left(l_n+h_n\right)$
else
  $l_{n+1}, h_{n+1}\leftarrow \frac{1}{2}\left(l_n+h_n\right), h_n$
end if
while Not End-of-Sentence symbol **do**
  Search:
  Find $D_i\in\mathcal{D}$ such that:
    $L=l_{K_{n+1}}+\left(u_{K_{n+1}}-l_{K_{n+1}}\right)p_{\text{cum}}\left(D_{i-1}|t_{1:K_n}\right)$ // If $K_n=0$, use $p_{\text{cum}}\left(D_{i-1}\right)$
    $U=l_{K_{n+1}}+\left(u_{K_{n+1}}-l_{K_{n+1}}\right)p_{\text{cum}}\left(D_i|t_{1:K_n}\right)$ // If $K_n=0$, use $p_{\text{cum}}\left(D_i\right)$
    $L\leqslant l_{n+1}<u_{n+1}<U$ // i. e. current interval of the n-th bit is included in the interval of D_i
  if D_i exists **then**
    Update:
    $K_{n+1}\leftarrow K_{n+1}+1$
    $t_{K_{n+1}}\leftarrow D_i$//Output D_i to the token sequence t
    $l_{K_{n+1}}, u_{K_{n+1}}\leftarrow L, U$
    Scaling: Similar to the Scaling in Algorithm 1
    Go to Search
  else
    return $K_{n+1}, t_{K_n+1:K_{n+1}}$
  end if
end while
return $K_{n+1}, t_{K_n+1:K_{n+1}}$ // Updated decoded tokens

Appendix 2: Two key modules of ECCT

Appendix C: Positional reliability encoding

For the channel output $\boldsymbol{y}$, the positional reliability encoding transforms each dimension of $\tilde{\boldsymbol{y}}$ into a high d dimensional embedding ϕ, which enriches the information of input embedding vectors and replaces $\tilde{\boldsymbol{y}}$ as the input of ECCT. The transformation is defined by

$$\phi_i = \begin{cases} |\boldsymbol{y}_i| \boldsymbol{W}_i, & \text{if } i \leqslant N \\ \text{bin\_to\_sign}\left(\text{syn}\left(\boldsymbol{y}_{i-N+1}\right)\right)\boldsymbol{W}_i, & \text{otherwise} \end{cases} \tag{11},$$

where $\{\boldsymbol{W}_i \in \mathbb{R}^d\}_{i=1}^{2N-K}$ denotes the learnable embedding matrix representing the bit's position-dependent one-hot encoding. The encoding method corresponds to the input reliability and is positional, since unreliable information of low magnitude would collapse to the origin, while the syndrome scales negatively. Hence, it is termed positional reliability encoding.

Appendix D: Code-aware self-attention

The code-aware attention mask mechanism aims to integrate code-specific sparse marks that incorporate the inherent structural characteristics of their respective PCM as the domain knowledge. Given a codeword defined by the generator matrix $\boldsymbol{G}$ and parity check matrix $\boldsymbol{H}$, the attention mask is defined by $g(\boldsymbol{H}):\{0,1\}^{(n-k)\times n} \rightarrow \{-\infty, 0\}^{(2n-k)\times(2n-k)}$, the construction of which is shown in Algorithm 3. Then, the code-aware self-attention mechanism could be represented as

$$A_H(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \text{Softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T + g(\boldsymbol{H})}{\sqrt{d}}\right)\boldsymbol{V} \tag{12},$$

where $\boldsymbol{Q}$, $\boldsymbol{K}^*$ and $\boldsymbol{V}$ denote the query, key and value in self-attention. During the implementation, the code-aware attention mask mechanism is used as an enhancement of the multi-head-attention module in the classical Transformer architecture.

Algorithm 3 Pseudo code of building the attention mask

Require: parity-check matrix $\boldsymbol{H}$ of error correction code $C_e(N,K)$

mask ← eye$(2N-K)$
for $i = 1, 2, \cdots, N-K$ **do**
  idx ← where$(\boldsymbol{H}[i] == 1)$
  for j in idx **do**
    mask$[N+i, j]$, mask$[j, N+i]$ ← 1
    for l in idx **do**
      mask$[j, l]$, mask$[l, j]$ ← 1
    end for
  end for
end for
mask ← $-\infty(\neg$mask$)$
return mask // Output attention mask $g(\boldsymbol{H})$

Biographies

REN Tianqi received his BE degree in electronic science and technology from Zhejiang University, China in 2024. He is currently pursuing his ME degree in electronic and information engineering with Zhejiang University. His research interests include application of large language models in communication scenarios and semantic communications.

LI Rongpeng (lirongpeng@zju.edu.cn) is currently an associate professor with the College of Information Science and Electronic Engineering, Zhejiang University, China. He was a research engineer with the Wireless Communication Laboratory, Huawei Technologies Co., Ltd. from August 2015 to September 2016. He was a visiting scholar with the Department of Computer Science and Technology, University of Cambridge, UK from February 2020 to August 2020. His research interest currently focuses on networked intelligence for communications evolving (NICE). He received the Wu Wenjun Artificial Intelligence Excellent Youth Award in 2021. He serves as an Editor for *China Communications*.

ZHAO Mingmin received his BEng and PhD degrees in information and communication engineering from Zhejiang University, China in 2012 and 2017, respectively. From December 2015 to August 2016, he was a visiting scholar with the Department of Electrical and Computer Engineering, Iowa State University, USA. From July 2017 to July 2018, he was a research engineer with Huawei Technologies Co., Ltd. He is currently a lecturer with the College of Information Science and Electronic Engineering, Zhejiang University. Since May 2019, he has been a visiting scholar with the Department of Electrical and Computer Engineering, National University of Singapore. His research interests include channel coding, algorithm design and analysis for advanced MIMO, cooperative communication, and machine learning for wireless communications.

CHEN Xianfu received his PhD degree (with Hons.) from Zhejiang University, China in 2012. In 2012, he joined the VTT Technical Research Centre of Finland, as a research scientist and as a senior scientist from 2013 to 2023. He is currently a chief research engineer with the Shenzhen CyberAray Network Technology Co., Ltd., China. His research interests include various aspects of wireless communications and networking, with emphasis on human-level and artificial intelligence for resource awareness in next-generation communication networks. Dr. CHEN was the recipient of the 2021 IEEE Communications Society Outstanding Paper Award and the 2021 IEEE Internet of Things Journal Best Paper Award. He is an editor of *IEEE Open Journal of the Communications Society*, an academic editor of *Wireless Communications and Mobile Computing*, and an associate editor of *China Communications*.

LIU Guangyi received his PhD degree from Beijing University of Posts and Telecommunications, China in 2006. He is currently the chief scientist of 6G in China Mobile Communication Corporation (CMCC), the founding member and

* It is worthwhile to point that $\boldsymbol{K}$ distinguishes from K, which represents the length of the error correction code in the previous text.

the co-chair of the 6G Alliance of Network AI, and the vice-chair of the THz Industry Alliance in China and the Wireless Technology Working Group of IMT-2030 (6G) Promotion Group supported by Ministry of Information and Industry Technology of China. He has been leading the 6G research and development with CMCC since 2018. He has led the Research and Development of 4G's evolution and 5G in CMCC from 2006 to 2020. He has acted as a Spectrum Working Group Chair and the Project Coordinator of LTE Evolution and 5G eMBB in the Global TD-LTE Initiative from 2013 to 2020 and led the industrialization and globalization of TD-LTE evolution and 5G eMBB.

YANG Yang is a professor with the IoT Thrust, the Director of the Research Center for the Digital World with Intelligent Things (DOIT), and the associate vice-president for Teaching and Learning with The Hong Kong University of Science and Technology (Guangzhou), China. He is also an adjunct professor with the Department of Broadband Communication at Peng Cheng Laboratory, the chief scientist of IoT with Terminus Group, and a senior consultant for Shenzhen Smart City Technology Development Group, China. His research interests include multi-tier computing networks, 5G/6G systems, AIoT technologies, intelligent services and applications, and advanced wireless testbeds. He has been the chair of the Steering Committee of the Asia-Pacific Conference on Communications (APCC) from 2019 to 2021. Currently, he is serving the IEEE Communications Society as the chair for the 5G Industry Community and chair for the Asia Region at Fog/Edge Industry Community. He is a fellow of IEEE.

ZHAO Zhifeng received his BE degree in computer science, ME degree in communication and information systems, and PhD degree in communication and information systems from the PLA University of Science and Technology, China in 1996, 1999, and 2002, respectively. From 2002 to 2004, he acted as a post-doctoral researcher with Zhejiang University, China, where his studies focused on multimedia next-generation networks (NGNs) and softswitch technology for energy efficiency. Currently, he is with the Zhejiang Lab as the Chief Engineering Officer. His research areas include software-defined networks (SDNs), wireless networks in 6G, computing networks, and collective intelligence. He is the Symposium Co-Chair of ChinaCom 2009 and 2010. He is the TPC Co-Chair of the 10th IEEE International Symposium on Communication and Information Technology (ISCIT 2010).

ZHANG Honggang is a professor with the Faculty of Data Science, City University of Macau, China. He was the founding Chief Managing Editor of *Intelligent Computing*, a Science Partner Journal, and a professor with the College of Information Science and Electronic Engineering, Zhejiang University, China. He was an Honorary Visiting Professor with the University of York, UK, and an International Chair Professor of Excellence with the Université Européenne de Bretagne and Supélec, France. His research interests include cognitive radio networks, semantic communications, green communications, machine learning, artificial intelligence, intelligent computing, and the Internet of Intelligence. He is a co-recipient of the 2021 IEEE Communications Society Outstanding Paper Award and the 2021 IEEE Internet of Things Journal Best Paper Award. He was the leading guest editor for the special issues on green communications of the *IEEE Communications Magazine*. He is the associate editor-in-chief of *China Communications*. He is a fellow of IEEE.