BidDeepSC-1.58b: 1.58-bit Bidirectional Slimmable Semantic Communication System

Tung Son Do, Thanh Phung Truong, Quang Tuan Do, Dongwook Won, Wonjong Noh and Sungrae Cho

***Abstract*—The large model sizes and high computational demands of the current semantic communication systems pose challenges for deployment in environments with limited resources. Furthermore, existing semantic communication frameworks predominantly operate unidirectionally, significantly increaseing model redundancy and resource requirements in scenarios requiring simultaneous bidirectional information exchange. To address these challenges, we propose a BidDeepSC-1.58b, integrating three complementary innovations: a unified encoder-decoder architecture enabling bidirectional communication with a halved parameter count, 1.58-bit quantization for extreme model compression, and a slimmable neural network with an intelligent semantic fidelity predictor that dynamically selects optimal width configurations. The experimental results demonstrate that the proposed BidDeepSC-1.58b achieves 88.1% model size reduction (3.79 MB footprint) while outperforming baseline models by over 30% in semantic similarity under challenging channel conditions. Weight-tying enables bidirectional communication with minimal performance trade-offs, showing only 2.5% degradation at -6 dB SNR that diminishes to 0.6% at 18 dB SNR, a pattern consistent across diverse proposed and baseline architectures. The framework reduces inference latency by up to 88% and enables dynamic width adaptation that achieves 39.2% computational savings while maintaining semantic performance thresholds. This system enables the efficient deployment of semantic communication in resource-constrained environments and facilitates bidirectional semantic information exchange across a broad range of applications.**

***Index Terms*—Bidirectional transmission, deep learning, semantic communication, sixth generation (6G), slimmable architecture.**

I. Introduction

The fast growth of technology, including augmented reality, virtual reality, robots, automation, the Internet of Things (IoT), artificial intelligence, and machine learning, has increased in data consumption. Complicated new applications facilitating two-way information exchange between devices such as self-driving vehicles and mission-oriented drones, are demanding higher bandwidths, dense connections, and ultra-reliable, low-latency communications. Conventional communication systems face significant challenges in managing extensive connectivity over constrained wireless infrastructures [1], [2]. Although fifth-generation (5G) technology has progressed, it may be unsuitable for sophisticated, automated, and intelligent networks. In response to these problems such as rising data needs, high-performance computing, communication connection bottlenecks, and the existing constraints of wireless technology, semantic communication has emerged as a promising solution.

Semantic communication represents a paradigm shift, focusing on transmitting meaning rather than just bits and marking the transition to semantic and effectiveness levels. Unlike traditional Shannon-based approaches that treat all bits as equally important and prioritize raw transmissions, semantic communication emphasizes conveying the most relevant information, significantly reducing data volume while preserving essential semantic content. This approach could revolutionize data transmission in next-generation networks. The pioneering work by Xie *et al.* [3] demonstrated the effectively integration of deep learning techniques for semantic information extraction, representation, and transmission, establishing methods to evaluate the semantic characteristics of messages. This foundational research integrated deep learning methods into semantic communication, establishing a basis for advancements in intelligent communication architectures.

Although semantic communication offers significant advantages, it faces several critical challenges. First, most current semantic communication systems are designed for unidirectional transmission, limiting their practical application in real-world scenarios requiring a bidirectional information exchange. Lightweight and efficient architectures that can handle two-way semantic information exchanges while maintaining high performance are in demand. Second, the current deep learning-enabled semantic communication systems face significant resource constraints due to their substantial model sizes and computational complexity. This problem poses challenges for deployment in resource-constrained environments, such as edge devices or IoT nodes [3], [4]. These resource limitations significantly affect scenarios in which computational resources, memory, and energy are constrained, such as in edge-computing applications and IoT devices. Last, another considerable challenge is the limited adaptability of semantic communication systems to dynamic operational conditions. The current systems employ fixed model architectures that cannot efficiently adjust their computational resources and semantic processing capabilities based on real-time network conditions. This rigid architecture is problematic in scenarios requiring dynamic resource allocation. For instance, under favorable channel conditions, systems unnecessarily consume the maximum computational power to achieve satisfactory performance, leading to resource inefficiency. Conversely, during poor channel conditions, these systems must employ the maximum computational resources to maintain acceptable

Tung Son Do, Thanh Phung Truong, Quang Tuan Do, Dongwook Won, and Sungrae Cho are with the School of Computer Science and Engineering, Chung-Ang University, Seoul 00007, Republic of Korea (e-mail: tsdo, tptruong, dqtuan, dwwon@uclab.re.kr; srcho@cau.ac.kr)

Wonjong Noh is with the School of Software, Hallym University, Chuncheon 24252, Republic of Korea. (email: wonjong.noh@hallym.ac.kr)

performance levels.

A. Related Work

1) Directional Semantic Communication: In terms of unidirectional semantic communication systems, Xie *et al.* [3] established DeepSC, a foundation with a Transformer-based architecture that concurrently optimizes semantic and channel coding, enabling the robustly recovery of sentence meaning in noisy channels. Xie *et al.* [4] later addressed IoT device constraints via Lite-DeepSC, a streamlined version of the original model. Moreover, Zhou *et al.* [5] developed an innovative system based on the Universal Transformer architecture to address semantic noise. Additionally, Dai *et al.* [6] advanced the field by introducing a novel nonlinear transform combined with deep joint source-channel coding. The method concurrently learns latent source representations and an entropy model, yielding an efficient transmission adaptable to diverse source distributions. In addition, Yoo *et al.* [7] proposed a hybrid convolutional neural network and Vision Transformer (ViT) semantic communication system combining the strengths of both architectures. Yi *et al.* [8] introduced a system that incorporates a shared knowledge base. This method enhances explainability and efficiency by transmitting only information not present in the knowledge base, while maintaining semantic performance. Futher, Hu *et al.* [9] proposed a framework based on a vector quantized-variational auto-encoder and an adversarial training to combat the semantic noise and reduce overhead transmission. Xie *et al.* [10] proposed a unified Transformer-based framework for multimodal tasks, including VQA, and introduced a layerwise Transformer for enhanced multimodal data fusion. Additionally, Zhang *et al.* [11] designed a Transformer-based unified-decoder framework with lightweight task heads to serve multiple tasks across text, image, and speech modalities. Moreover, Do *et al.* [12] proposed a 3M-DeepSC that reconstructs both text and image for large language model (LLM) communication. Their system incorporates a Mamba-based architecture to process multimodal data efficiently, replacing traditional Transformer-based designs. They also introduced a novel multimodal semantic similarity metric and a two-stage training algorithm to optimize bit-based and semantic performance.

Regarding bidirectional semantic communication systems, Yu *et al.* [13] proposed a two-way semantic communication (TW-SC) system that exploits weight reciprocity in the transceiver. This approach eliminates the need for bidirectional information feedback during training and applies a conditional generative adversarial network for channel modeling. However, TW-SC must store transmitter and receiver weights at each endpoint, doubling the storage requirements compared with its unidirectional counterparts. Despite the significant potential of bidirectional semantic communication systems, research in this area has remained limited.

2) Slimmable Neural Networks: Yu *et al.* [14] first introduced slimmable neural networks to enable a single neural network to operate at various widths, facilitating an instant and adaptive accuracy–efficiency trade-offs at runtime. Their initial work employed switchable batch normalization to train networks that could execute at predefined width settings. Building on this foundation, Yu *et al.* [15] developed universally slimmable networks (US-Nets), extending the concept to support execution at arbitrary widths. These networks introduced several innovative techniques, including post-statistics batch normalization and enhanced training methods, such as the sandwich rule and in-place distillation. The authors also proposed AutoSlim [16], a one-shot approach for automatically determining optimal channel numbers in neural networks, achieving improved accuracy–efficiency tradeoffs without manual intervention. Further advancing this field, Li *et al.* [17] introduced Dynamic Slimmable Networks (DS-Net) in 2021, implementing dynamic width adjustment during test time based on input complexity while maintaining hardware efficiency via a novel dynamic slicing scheme. The network employed a double-headed gate to predict adaptive width configurations, substantially reducing computation with minimal influence on accuracy. These progressive developments in slimmable networks have significantly enhanced the flexibility, efficiency, and performance of deep learning models. This evolution has enabled adaptive inference across diverse computational budgets using a single trained model, a significant advancement in efficient neural network design.

3) Quantized Neural Networks: Extreme quantization techniques have pushed the limits of model compression, focusing on binarization and low-bit representations. In early work on binarized convolutional neural networks, Rastegari *et al.* [18] and Bulat *et al.* [19] demonstrated the potential of using binary weights and converting arithmetic operations to efficient XNOR and bitcount operations. Yang *et al.* [20] formulated quantization as a differentiable nonlinear mapping function comprising learnable sigmoid functions and proposed a continuous relaxation training method with increasing temperature parameters to bridge the gap between soft and hard quantization. Subsequently, Maly *et al.* [21] introduced a novel pre-processing step enabling memory-less scalar quantization while preserving network performance, providing rigorous theoretical guarantees for error decay with well-behaved training data. Xu *et al.* [22] proposed an elastic quantization network supporting multiple configurations via weight distribution regularization and progressive group guidance techniques. More recently, extreme quantization has been successfully applied to LLMs, with Wang *et al.* [23] pioneered BitNet for 1-bit LLMs, significantly reducing the memory and energy footprint while maintaining competitive performance. Building on this work, Ma *et al.* [24] introduced BitNet 1.58b using ternary weights $-1, 0, 1$, theoretically requiring only $\log_2(3) \approx 1.58$ bits per value. The success of extreme quantization has expanded to ViT, with Yuan *et al.* [25] proposing ViT 1.58b, significantly reducing the computational complexity and memory consumption while maintaining competitive accuracy with full-precision counterparts.

B. Motivations, Contributions and Organizations

Recent advances in semantic communication have motivated several critical research directions:

- **Efficient Bidirectional Communication:** Yu *et al.* [13] pioneered bidirectional semantic communication with

TABLE I: Key notation

Notation	Definition
I_j, $\hat{I}_j$	Input and reconstructed sentences of user j
L, $\hat{L}$	Word lengths for the input and reconstructed sentence
θ_{enc}^{sem}, θ_{dec}^{sem}	Parameters of the Unified Slimmable Semantic block in the encoding and decoding states
θ_{enc}^{chan}, θ_{dec}^{chan}	Parameters of the Unified Slimmable Channel block in the encoding and decoding states
SF_j, $\widehat{SF}_j$	Extracted and reconstructed semantic features
X_j	Encoded features before the transmission of user j
Z	Representation of additive white Gaussian noise
Y_j	Received features at the receiver side
θ_{BL}, $\tilde{\theta}_{BL}$	Weight of μ-BitLinear in full precision and ternary values
μ_{BL}	Average value of θ_{BL}
ϵ	Small constant to prevent division by zero
b	Number of bits for activation quantization
x, $\tilde{x}$	Original full-precision and quantized input values
y	Output after matrix multiplication of $\tilde{x}$ and $\tilde{\theta}_{BL}$
$y_{dequant}$	Dequantized output back to full precision
β	Dequantized scaling factor
θ_Q, θ_K, θ_V, θ_O	Query, key, value and output projections parameters
B_c, B_r	Block size for columns and rows
T_c, T_r	Number of blocks for columns and rows
N	Sequence length inside FlashAttention layer
M	On-chip SRAM memory size
θ_{MHFA}, θ_{FFN}	Parameters of Bit Multi-Head Attention, Feed-forward network in Semantic block
θ_{SBL1}, θ_{SBL2}	Parameters of two μ-BitLinear layers inside FFN layer
θ_{head}, θ_{sem}	Two shared parameters for Unified Semantic block
VS	Vocabulary size of the Embedding layer
θ_{EB}, θ_{LN}	Parameters of the Embedding and Linear layer of Semantic Block
θ_{CBL1}, θ_{CBL2}	Parameters of two μ-BitLinear layers in Unified Channel Block
α, $\mathcal{A}$	Width multiplier and width multiplier set
d, d'	Original and slimmable μ-BitLinear dimenstions
d_{AH}, d'_{AH}	Original and slimmable attention head dimension of Bit Multi-head Flash Attention layer
h, h'	Original and adjusted of head number, respectively, after applying the α width multiplier
D_{chan}, D'_{chan}	Original and adjusted numbers of channel symbols, respectively, after applying the α width multiplier
D_{sem}, D'_{sem}	Original and adjusted sizes of the semantic features, respectively, after applying the α width multiplier
IC, IC'	Original and adjusted immediate channel sizes, respectively, after applying the α width multiplier
θ	Parameter of the overall BidDeepSC framework
G_ω, θ_ω	Semantic Fidelity Predictor with its parameters
EF	Embedding feature of the input text
μ_t, σ_t	Mean and standard deviation of embedding feature EF
SS_{th}	The threshold sentence similarity of Slim-Optimizer
γ, $\mathcal{G}$	The SNR value and the SNR set values
θ_{NFCB1}, θ_{NFCB2}	Two parameters of NFCB blocks inside SFP

their TW-SC model; however, this system must store both encoder and decoder models at each communication endpoint. This redundancy increases memory requirements and computational overhead. Consequently, streamlined and resource-efficient bidirectional semantic communication architectures are needed that to address real-time, two-way semantic information without duplicating model components.

- **Resource-Efficient Edge Deployment:** Previous studies have proposed several methods such as pruning and neural network quantization to reduce resource consumption [4], [26], [27]. However, more aggressive compression techniques could further optimize model efficiency. Extreme quantization techniques that dramatically reduce model size and computational demands while maintaining competitive performance could enable the widespread deployment of semantic communication systems in highly resource-constrained edge-computing scenarios.
- **Adaptive Network Architecture:** The fixed nature of the current semantic communication architectures necessitates developing flexible systems to adjust their computational resources dynamically based on channel conditions. This need motivates the design of innovative neural network architectures that can intelligently scale their complexity and processing capabilities according to real-time network conditions and device constraints, optimizing performance and resource utilization across operational scenarios.

This research addresses the following three crucial research questions:

- Question 1: How can we design a scheme that enables bidirectional exchange for semantic communication without separately storing the encoder and decoder?
- Question 2: What compression techniques can be applied to semantic communication models to deploy them on resource-constrained edge devices while maintaining comparable performance?
- Question 3: How can semantic communication be flexible enough to adjust the computational load dynamically based on varying channel conditions and device capabilities?

This work integrates the following three principal components into a unified framework tailored for semantic communication to address these questions: bidirectional weight-tying, low-bit quantization, and slimmable neural networks. This holistic integration permits the proposed framework to ensure significant memory savings, dynamic scalability, and highly efficient bidirectional transmission, that is, a new paradigm for resource-efficient and adaptive semantic communication. This research addresses these critical questions via the following crucial contributions:

- **Unified Encoder-Decoder via Weight-Tying:** We propose the BidDeepSC system, consolidating encoder and decoder weights to enable efficient bidirectional semantic communication. This unified approach significantly reduces memory requirements and computational overhead by eliminating the need for separate encoder-decoder models at each endpoint, directly addressing Question 1.
- **1.58-bit Precision Quantization:** We implement 1.58-bit precision quantization in BidDeepSC, dramatically reducing the model size and computational demand while sustaining performance. This approach enables deployment on resource-constrained edge devices and broadens the applicability of semantic communication systems, addressing Question 2.
- **Dynamic Adaptation with Slimmable Networks:** We integrate slimmable neural networks and a slim optimizer to adapt model complexity intelligently based on communication conditions and resource constraints. This adaptive strategy optimizes performance and resource utilization across diverse scenarios, addressing Question 3.
- We develop a novel training algorithm combining weight-tying and slimmable progressive mechanisms. The algo-

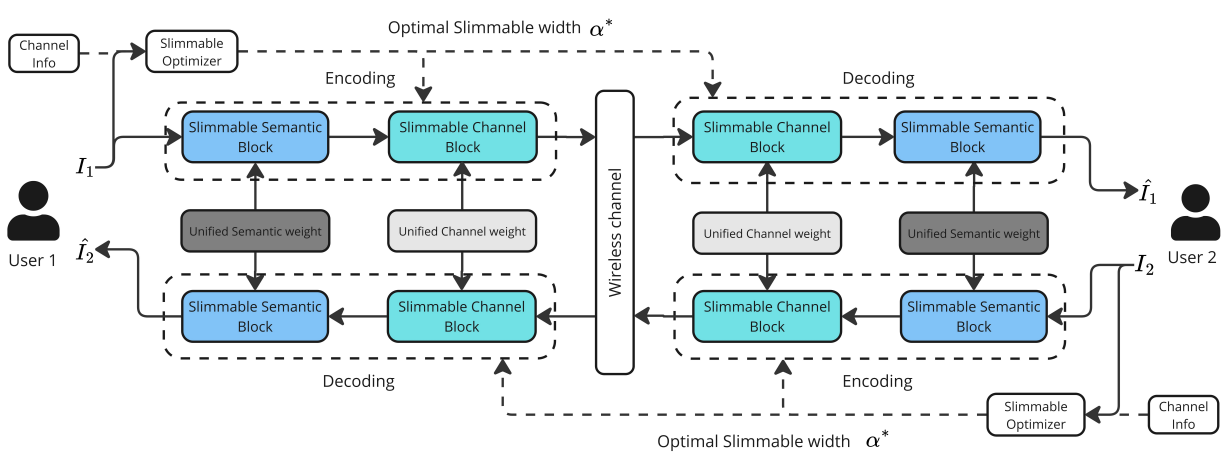


Fig. 1: Bidirectional deep-learning-enabled semantic communication System (BidDeepSC) system framework.

rithm ensures aligned performance across width configurations while enabling efficient bidirectional communication via parameter sharing.

- We validate the proposed framework via comprehensive experiments across diverse channel environments and demonstrate the effectiveness of weight-tying and 1.58-bit quantization on the proposed model and baseline systems using ablation studies.

The remainder of this paper is organized as follows. Section II presents the proposed framework for BidDeepSC. Next, Section III details the proposed semantic communication approach and algorithms employed in the study. Then, Section IV presents the comprehensive simulation results. Finally, Section V concludes the paper. For quick reference, Table I comprehensively lists the primary notation applied throughout this work.

II. Bidirectional Deep-learning-enabled Semantic Communication Framework

This section outlines the framework of the proposed BidDeepSC, which consists of unified encoder/decoder blocks that allow a two-way semantic data exchange without cost compared to unidirectional DeepSC models. The framework specifically targets resource-constrained IoT and edge devices operating within strict hardware limitations: memory footprints below 16MB and computational budgets under 20 MFLOPS. Unlike existing bidirectional approaches that require storing separate encoder-decoder pairs at each endpoint, effectively doubling memory requirements, our unified architecture addresses the fundamental challenge of enabling efficient two-way communication within the strict resource limitations of edge computing environments

A. System Model

As depicted in Fig. 1, the proposed BidDeepSC involves two users at opposite ends of the communication channel. The process is a flow of encoding and decoding operating for any user j transmitting to user k, where $j, k \in 1, 2$. The system accepts inputs $I_j \in \mathbb{R}^L$ where j and L represent the index of the user and the word length of the input, respectively. In Fig. 1, each side contains a Slim-Optimizer, a unified semantic weight and a unified channel weight which would load into corresponding Slimmable Semantic Blocks (SBs) and Slimmable Channel Blocks (CBs) in both encoding and decoding passes. At the beginning of transmission of input I_j, the input enters into a Slim-Optimizer with the channel information to dete the optimal slimmable width α^*. Then, the Slimmable Block extracts for extracting the D_{sem} semantic features from the input I_j, expressed as follows:

$$SF_j = \mathbf{SSB}(I_j|\theta_{enc}^{sem}), SF_j \in \mathbb{R}^{L\times D_{sem}} \tag{1}$$

where $SF_j \in \mathbb{R}^{L\times D_{sem}}$ represents the semantic features, and $\mathbf{SSB}$ and θ_{enc}^{sem} denote the Slimmable SB and its unified weight in the encoding state, respectively. Afterward, SF_j is processed Slimmable CB to dynamic feature selection to satisfy the number of channel symbols D_{chan}, expressed as follows:

$$X_j = \mathbf{SCB}(SF_j|\theta_{enc}^{chan}), X_j \in \mathbb{R}^{1\times(L\times D_{chan})} \tag{2}$$

where X_j denotes the encoded features of user j, and $\mathbf{SCB}$ and θ_{enc}^{chan} represent Slimmable CB and its unified weight in the encoding state, respectively. Then, X_j is transmitted to the receiver and the received features at the receiver are represented as follows:

$$Y_j = hX_j + Z, Y_j \in \mathbb{R}^{1\times(L\times D_{chan})} \tag{3}$$

where $h \in \mathbb{C}$ represents the channel gain coefficient, and $Z \sim \mathcal{CN}(0, \sigma^2)$ denotes the additive white Gaussian noise (AWGN). The received features are processed via the Slimmable CB to reconstruct the semantic features, expressed as follows:

$$\widehat{SF}_j = \mathbf{SCB}(Y_j|\theta_{dec}^{chan}), \widehat{SF}_j \in \mathbb{R}^{L\times D_{sem}} \tag{4}$$

where $\widehat{SF}_j$ indicates the reconstructed semantic feature, and $\mathbf{SCB}$ and θ_{dec}^{chan} represent the Slimmable CB and its unified weight in the decoding state, respectively. In addition, $\widehat{SF}_j$

processed Slimmable SB to reconstruct the text, expressed as follows:

$$\hat{I}_j = \underset{v}{\text{argmax}}(\mathbf{SSB}(\widehat{SF}_j(v)|\theta_{dec}^{sem}), v \in \mathbb{R}^{1,VS}), \hat{I}_j \in \mathbb{R}^L \quad (5)$$

where $\hat{I}_j$ denotes the reconstructed text signal, and **SSB** and θ_{dec}^{sem} represent the Slimmable SB and its unified parameter in the decoding state, respectively.

B. Performance Metrics

The bilingual evaluation understudy (BLEU) metric was applied to assess the text quality. This metric assesses the similarity between the original text I_j of length L and the reconstructed text $\hat{I}_j$ of length $\hat{L}$ by comparing their n-grams. The BLEU score is calculated as follows:

$$\log \text{BLEU} = \min\left(1 - \frac{L}{\hat{L}}, 0\right) + \sum_{n=1}^{N} u_n \log p_n \quad (6)$$

where u_n represents the weights of the n-grams, and p_n indicates the n-gram score, expressed as follows:

$$p_n = \frac{\sum_k \min(C_k(\hat{I}_j), C_k(I_j))}{\sum_k C_k(\hat{I}_j)} \quad (7)$$

where C_k indicates the frequency counter function for the k^{th} element in the n^{th} gram. the BLEU score range is a scalar between 0 and 1, evaluating the similarity between the reconstructed and source text, with 1 indicating perfect similarity between the reconstructed and source texts.

Moreover, we adopted the semantic similarity metric [28] to evaluate the meaning of the reconstructed versus the original text, calculated as follows:

$$\text{SemSim}(\hat{I}_j, I_j) = \frac{\boldsymbol{B}(I_j) \cdot \boldsymbol{B}(\hat{I}_j)}{||\boldsymbol{B}(I)|| \times ||\boldsymbol{B}(\hat{I}_j)||} \quad (8)$$

where $\boldsymbol{B}$ represents the bidirectional encoder representations from Transformers (BERT), which is a huge pre-trained language model used to extract semantic information. The semantic similarity SS ranges from 0 to 1, with higher values indicating a higher semantic correlation between the original and reconstructed texts.

III. Proposed System: BidDeepSC-1.58b

A. Unified Semantic Encoder/Decoder Architectures

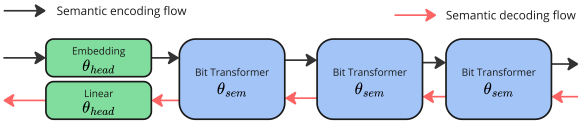


Fig. 2: Semantic Block in the encoding and decoding state.

1) Encoding and Decoding in Semantic Blocks: The SB architecture efficiently manage both encoding and decoding operations using shared parameters. The SB operation, illustrated in Fig. 2, is expressed as follows:

$$\begin{cases} \text{BitTransformer}^3(\text{Embedding}(I_j|\theta_{head})|\theta_{sem}) & \text{for semantic encoding} \\ \text{Linear}(\text{BitTransformer}^3(\widehat{SF}_j|\theta_{sem})|\theta_{head}) & \text{for semantic decoding} \end{cases}$$

Only two required weights are required in the Semantic block: θ_{head} for the Embedding layer in encoding and the Linear layer in decoding and θ_{sem} for Bit Transformer Blocks. In encoding, the Embedding layer in the encoding state converts discrete inputs to a dense vector, formulated as follows:

$$y_{EB} = \text{Embedding}(x_{EB}|\theta_{EB}) = \theta_{EB}[x_{EB}, :], \quad (9)$$

where $x_{EB} \in \mathbb{R}^L$ denotes the input tensor with L tokens; $\theta_{EB} \in \mathbb{R}^{\text{VS}\times D_{sem}}$ represents the lookup weight of the embedding, VS indicates for the embedding vocabulary size, and $y_{EB} \in \mathbb{R}^{L\times D_{sem}}$ denotes the output. The Linear layer in decoding performs a linear transformation on its input, described as follows:

$$y_{LN} = \text{Linear}(x_{LN}|\theta_{LN}) = \theta_{LN} \cdot x, \quad (10)$$

where $x_{LN} \in \mathbb{R}^{L\times D_{sem}}$ denotes the input vector, $\theta_{LN} \in \mathbb{R}^{\text{VS}\times D_{sem}}$ represents the Linear weight, and $y_{LN} \in \mathbb{R}^{L\times \text{VS}}$ indicates the output vector. From another perspective, the Embedding layer in encoding transforms $x_{EB} \in \mathbb{R}^L$ into $y_{EB} \in \mathbb{R}^{L\times D_{sem}}$ whereas the Linear layer transforms $x_{LN} \in \mathbb{R}^{L\times D_{sem}}$ into the output $y_{LN} \in \mathbb{R}^{L\times \text{VS}}$. Moreover, their weight shapes are the same $\mathbb{R}^{\text{VS}\times D_{sem}}$; therefore, $\theta_{LN} \equiv \theta_{EB}$. Thus, a unified semantic weight θ_{head} can be employed by setting $\theta_{head} = \theta_{EB}$ for the embedding layer in encoding and the linear layer in decoding.

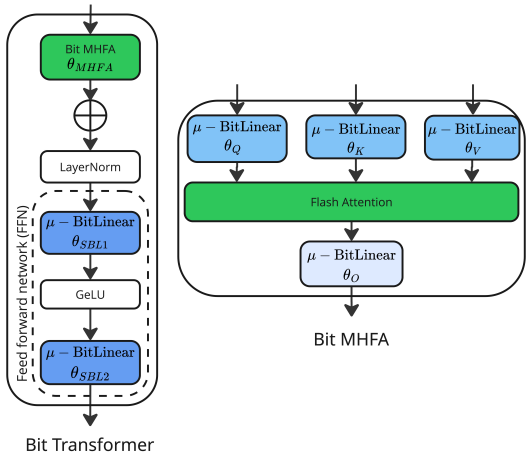


Fig. 3: Architecture of the Bit Transformer and Bit Multi-Head Attention block.

The SB uses three identical Bit Transformer blocks for encoding and decoding. This configuration allows for weight sharing across all blocks, reducing storage requirements. We assigned the unified weight θ_{sem} to the average weight value of all SBs, express as: $\theta_{sem} = \mathbb{E}(SBs)$. Moreover, this architecture can be conceptualized as a single SB sequentially executed thrice. Beyond memory optimization, this approach offers marginal performance benefits. By loading the shared weights into the fast-access memory (e.g., SRAM of GPU) once and performing three sequential computations, the system eliminates the need for repeated weight transfers. This method achieves greater computational efficiency than naive swapping of distinct weights for each layer. Thus, by setting unified

weights θ_{head} and θ_{sem}, a shared SB can be achieved enabling the construction of a bidirectional communication model.

The Bit Transformer Block, as illustrated in Fig. 3 comprises a Bit Multi-Head Attention, LayerNorm (LN) activation and a Feed-Forward network (FFN). The following equations describe the flow:

$$\begin{aligned} \text{SB}(x) &= \text{FFN}(\text{LN}(x + O_{MHFA}(x))|\theta_{FFN}) \\ &= \text{FFN}\left(\text{LN}(x + \mathcal{B}(H(x)|\theta_O))|\theta_{FFN}\right), \end{aligned} \tag{11}$$

where SB represents for Semantic block; FFN and θ_{FFN} represent the feed-forward network and its parameters, respectively. The FFN includes two μ-BitLinear layers and a Gaussian Error Linear Unit (GELU) activation function, expressed as follows:

$$\text{FFN}(x) = \text{BL}(\text{GELU}(\text{BL}(x|\theta_{SBL1}))|\theta_{SBL2}) \tag{12}$$

where BL represents the μ-BitLinear layer; θ_{SBL1}, and θ_{SBL2} denote the parameters of the respective μ-BitLinear layers in the FFN layer.

The Bit Multi-Head Flash Attention (Bit MHFA), illustrated in Fig. 3 comprises a set of three linear projections, a Flash Attention and an output linear projection. By letting $Q(x) \triangleq \mathcal{B}(x|\theta_Q)$, $K(x) \triangleq \mathcal{B}(x|\theta_K)$, and $V(x) \triangleq \mathcal{B}(x|\theta_V)$ denote the Query, Key, and Value projections with the corresponding parameters $\theta_Q \in \mathbb{R}^{L\times d}$, $\theta_K \in \mathbb{R}^{L\times d}$, and $\theta_V \in \mathbb{R}^{L\times d}$, $H(x) \triangleq \mathcal{F}(Q(x), K(x), V(x)|\theta_F)$ denotes the Flash Attention operation. The output of the Bit Multi-Head Flash Attention block can be calculated as

$$O_{MHFA}(x|\theta_{MHFA}) = \mathcal{B}(H(x)|\theta_O), \tag{13}$$

where $x \in \mathbb{R}^{L\times D}$ represent the input tensor, θ_{MHFA} denotes the block parameter combined from subblocks parameters, $\mathcal{B}(.)$ denotes the μ-BitLinear layer, $\mathcal{F}(.)$ signifies the Flash Attention layer with the corresponding parameter θ_F, and $\theta_O \in \mathbb{R}^{L\times d}$ indicates the parameter of the output μ-BitLinear layer.

Flash Attention [29] is employed in the proposed system instead of traditional Self-Attention owing to its improved efficiency and scalability. Flash Attention achieves linear memory complexity by computing attention in smaller blocks, that can be stored in the SRAM, allowing the memory to handle longer sequences more effectively. Moreover, Flash Attention offers better computational efficiency by optimizing operations and memory access, making it suitable for tasks with extended context and for deployment in IoT and resource-constrained environments.

By letting $B_c \triangleq \left\lfloor \frac{M}{4d_{AH}} \right\rfloor$ and $B_r \triangleq \min\left(\left\lfloor \frac{M}{4d_{AH}} \right\rfloor, d_{AH}\right)$ denote the block sizes for the columns and rows, respectively, where M is the size of the on-chip SRAM, d_{AH} denotes the dimension of the attention heads, and $T_r \triangleq \left\lceil \frac{N}{B_r} \right\rceil$ and $T_c \triangleq \left\lceil \frac{N}{B_c} \right\rceil$ denote the number of blocks for rows and columns, respectively, with N as the sequence length, the Flash Attention layer can be calculated as follows:

$$\begin{aligned} H(x) \triangleq& \mathcal{F}(Q(x), K(x), V(x)) \\ &= \sum_{i=1}^{T_r} \sum_{j=1}^{T_c} \text{softmax}\left(\frac{Q_i (K_j)^T}{\sqrt{d}}\right) V_j, \end{aligned} \tag{14}$$

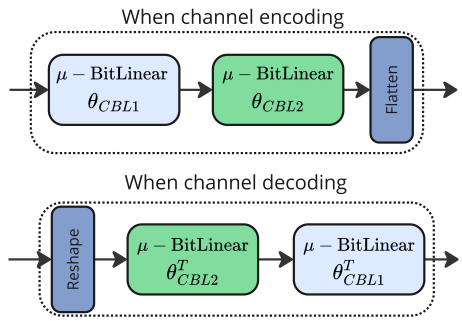


Fig. 4: Channel Block in encoding and decoding states.

where $Q_i \in \mathbb{R}^{B_r\times d_{AH}}$, $K_j \in \mathbb{R}^{B_c\times d_{AH}}$, and $V_j \in \mathbb{R}^{B_c\times d_{AH}}$ represent the chunked tensors of Q, K, and V, respectively. Here, the chunked tensors can be expressed as follow:

$$\begin{aligned} &[Q_1; Q_2; \ldots; Q_{T_r}] = Q, \\ &[K_1; K_2; \ldots; K_{T_c}] = K, \\ &[V_1; V_2; \ldots; V_{T_c}] = V. \end{aligned} \tag{15}$$

B. Unified Channel Encoder/Decoder Architectures

The Channel block (CB), as illustrated in Fig. 4, comprises two μ-BitLinear layers that function bidirectionally for both encoding and decoding processes. Only two weights θ_{CBL1} and θ_{CBL2} are applied. The operation of the CB can be formally expressed as follows:

$$\begin{cases} \text{Flatten}(\text{BL}(\text{BL}(SF_j|\theta_{CBL1})|\theta_{CBL2})) & \text{for channel encoding} \\ \text{BL}(\text{BL}(\text{Reshape}(Y_j)|\theta^T_{CBL2})|\theta^T_{CBL1}) & \text{for channel decoding} \end{cases}$$

where the parameters $\theta_{\text{CBL1}} \in \mathbb{R}^{D_{sem}\times IC}$ and $\theta_{\text{CBL2}} \in \mathbb{R}^{IC\times D_{chan}}$ represent the weights of the first and second μ-BitLinear layers in the channel encoding, respectively. In addition, IC denotes the intermediate channel size of the CB layer. The Flatten operation transforms the output tensor into a one-dimensional vector, whereas the Reshape operation restores the tensor to its original dimensions. The channel encoding process:

- Input $SF_j \in \mathbb{R}^{L\times D_{sem}}$ is transformed into $\mathbb{R}^{L\times IC}$ by the first μ-BitLinear module with the weight shape $\mathbb{R}^{IC\times D_{sem}}$.
- The output is transformed into $\mathbb{R}^{L\times D_{chan}}$ by the second μ-BitLinear module with the weight shape $\mathbb{R}^{D_{chan}\times IC}$.
- Finally, the Flatten operation converts the output to a one-dimensional vector for transmission.

The channel decoding process reverses these steps as follows:

- Input $Y_j \in \mathbb{R}^{1\times(L\times D_{chan})}$ is reshaped to $\mathbb{R}^{L\times D_{chan}}$.
- The first μ-BitLinear layer with the weight shape $\mathbb{R}^{D_{chan}\times IC}$ transforms it into $\mathbb{R}^{L\times IC}$.
- The second μ-BitLinear layer with the weight shape $\mathbb{R}^{IC\times D_{sem}}$ produces the final output $\mathbb{R}^{L\times D_{sem}}$.

The weight shape of the second μ-BitLinear in decoding is the transposed counterpart of the first μ-BitLinear in encoding and vice versa. The two μ-BitLinear in decoding employ the transposed corresponding weights θ^T_{CB2} and θ^T_{CB1} of the channel encoding parameters, respectively, to use the shared

parameters. This approach ensures symmetrical and efficient bidirectional transformations, optimizing the parameter utilization.

C. μ-BitLinear for 1.58 bit quantization

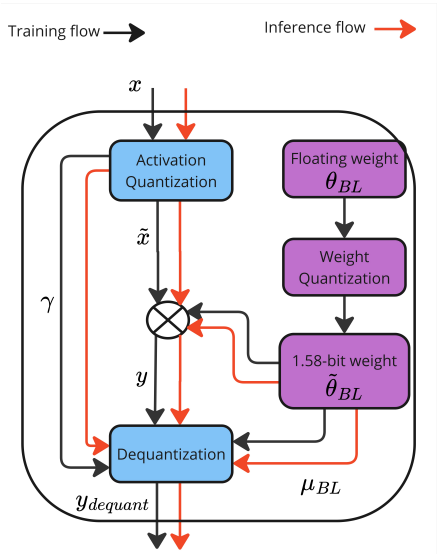


Fig. 5: μ-BitLinear layer architecture.

1) μ-BitLinear Layer: For 1.58 quantization, we propose the μ-BitLinear module which derived from BitLinear approach [24], illustrated in Fig. 5. The μ-BitLinear module applies a novel modification of the inference mode.

By letting $\tilde{x} \triangleq AQ(x)$ be the output of the activation quantization function, denoted as $AQ(.)$, with the input x, letting $\tilde{\theta}_{BL} \triangleq WQ(\theta_{BL})$ be the output of the weight quantization function, denoted as $WQ(.)$, with the original floating weight θ_{BL}, and letting $\mu_{BL} \triangleq \mathbb{E}(|\theta_{BL}|)$ be the absolute mean value of the weight θ_{BL}, the operation of the proposed μ-BitLinear module can be expressed as follows:

$$y_{dequant} = y \times \frac{\gamma \times \mu_{BL}}{127}, \tag{16}$$

where $y \triangleq \tilde{\theta}_{BL} \cdot \tilde{x}$ and $\gamma \triangleq \max(|x|)$ represent the auxiliary parameters, and 128 is the maximum constraint value of 8-bit precision $2^{8-1} = 128$.

The WQ function constrains the floating weight to the ternary value set $\{-1, 0, 1\}$. The process is defined as follows:

$$\tilde{\theta}_{BL} = RoundClip\left(\frac{\theta_{BL}}{\mu_{BL} + \epsilon}, -1, 1\right) \tag{17}$$

where ϵ is a value to avoid division by zero; and $RoundClip(x, a, b) \triangleq Clip(Round(x), a, b)$ combines the nearest integer rounding and clipping to constrain the output within a specific range. The $Clip$ function is defined as $Clip(x, a, b) = max(a, min(b, x))$, and the Round function defined as:

$$\text{Round}(x) = \begin{cases} \lfloor x \rfloor & \text{if } x - \lfloor x \rfloor < 0.5 \\ \lceil x \rceil & \text{if } x - \lfloor x \rfloor \geq 0.5 \end{cases} \tag{18}$$

The activation quantization AQ is proposed to normalize and quantize the input to 8-bit precision. The activation quantization sets the distribution of input to standard distribution and maps the values to the 8-bit integer interval $[-127, 127]$. By setting $s \triangleq Q_b/\gamma$ as the scaling factor for x input, the operation of AQ function is expressed as follows:

$$\tilde{x} = \text{Clip}\left(x \times \frac{128}{\gamma}, -128 + \epsilon_{act}, 128 - \epsilon_{act}\right) \tag{19}$$

where ϵ_{act} denotes a small floating number that prevents clipping overflow.

2) Inference mode: The original μ-BitLinear design continuously quantizes the floating-point weights to ternary values during training, with the gradients updating the original weights. However, this process becomes redundant during inference when weights remain static. To address this inefficiency, we propose a novel inference mode for the μ-BitLinear. In inference mode:

- Upon activation, all μ-BitLinear layers in the model undergo a one-time weight quantization to obtain $\tilde{\theta}_{BL}$.
- Subsequent forward passes use the pre-quantized 1.58-bit weights $\tilde{\theta}_{BL}$ directly, eliminating the need for repeated weight quantization.
- The original floating-point weights θ_{BL} are removed, conserving memory.

This approach offers two primary advantages: 1) reducing the computational overhead by eliminating redundant quantization operations during inference while improving the computational efficiency and 2) conserving memory by removing θ_{BL} during evaluation, significantly reducing the memory footprint. Consequently, the evaluation mode optimizes the computational efficiency and memory usage of the model during inference.

D. Slimmable Mechanism

We employed the slimmable mechanism [14], [15] to enable the dynamic adjustment of model complexity based on varying communication conditions and device capabilities. This adaptive approach allows the system to optimize its performance and resource utilization across a wide range of scenarios. In the proposed BidDeepSC framework, we applied a slimmable mechanism for both SB and CB. The slimmable mechanism was implemented using a set of width multipliers $\mathcal{A} \triangleq \{\alpha_1, \alpha_2, ..., \alpha_n\}$ on the network. These multipliers control the proportion of channels or neurons employed in each layer. For a given layer with output dimension d, the slimmable version is:

$$d' = \lfloor \alpha \cdot d \rfloor \tag{20}$$

where d' denotes the adjusted output dimension. As a basis of the whole system, the slimmable μ-BitLinear layer defined as:

$$\text{Slimmable } \mu\text{-BitLinear(x)} \triangleq \tilde{W}[:, : d'] \cdot \text{FA}(x)[:, : d'] \tag{21}$$

where $\tilde{W}[:, : d']$ represents the first d' columns of the quantized weight matrix, and $\text{Quant}(\text{LN}(x))[:, : d']$ denotes the first d' elements of the quantized and normalized input.

1) Slimmable Semantic Blocks: In the SB, all μ-BitLinears blocks are adopted slimmable to be become Slimmable μ-BitLinear(x). Thus, the dimension of the semantic features adjusts as:

$$D'_{sem} = \lfloor \alpha \cdot D_{sem} \rfloor \tag{22}$$

where D'_{sem} represents the adjusted semantic features. Moreover, in the Bit MHFA component, the number of attention heads and the dimension of each head are runtime adjusted according to the width multiplier, expressed as follows:

$$\begin{aligned} h' &= \max(1, \lfloor \alpha \cdot h \rfloor) \\ d'_{AH} &= \lfloor \alpha \cdot d_{AH} \rfloor \end{aligned} \tag{23}$$

where h denotes the original number of attention heads of Bit MHFA, respectively, and h' and d'_{AH} are the adjusted number of attention heads, and the dimension of each head, respectively.

2) Slimmable Channel Block: In terms of CB, both two μ-BitLinear layers are adopted to the slimmable mechanism. Thus, the intermediate size IC and channel symbol size D_{chan} inside CB are scaled according to the width multiplier, expressed as follows:

$$\begin{aligned} IC' &= \lfloor \alpha \cdot IC \rfloor \\ D'_{chan} &= \lfloor \alpha \cdot D_{chan} \rfloor. \end{aligned} \tag{24}$$

3) Runtime Adaptation: The slimmable architecture allows the BidDeepSC framework to adapt its complexity at runtime. This adaptation can be triggered based on various factors:

- Channel conditions: In poor channel conditions, a wider model ($\alpha = 1.0$) may be used to enhance robustness, whereas a slimmer model ($\alpha = 0.5$ or 0.75) can be employed to conserve resources in favorable conditions.
- Device capabilities: Resource-constrained devices can operate with slimmer models, whereas more powerful devices can use the full model capacity.
- Energy considerations: The width multiplier can be adjusted to balance performance and energy consumption based on the battery levels or power constraints.

Based on these factors, the system operator can create a dynamic runtime adjustment plan suitable for real-world scenarios.

E. Slim-Optimizer

1) Semantic Fidelity Predictor: Building upon the flexible slimmable architecture, we introduce the Semantic Fidelity Predictor (SFP), a lightweight prediction neural network that enables adaptive transmission by estimating the semantic fidelity of reconstructed text under various configurations, which is illustrated in Fig. 6. Unlike traditional approaches that require transmission and reconstruction for quality evaluation, SFP predicts semantic similarity before transmission, facilitating resource-efficient communication. The SFP received four inputs: the current channel signal-to-noise ratio (SNR), width multiplier $\alpha \in \mathcal{A}$, and the statistical features of text input μ_t, σ_t. Let G_ω denote for the SFP with paramter θ_ω, then the predicted semantic similarity $\widehat{SS}$ is defined as:

$$\widehat{SS} = G_\omega(\mu_t, \sigma_t, \gamma, \alpha; \theta_\omega) \tag{25}$$

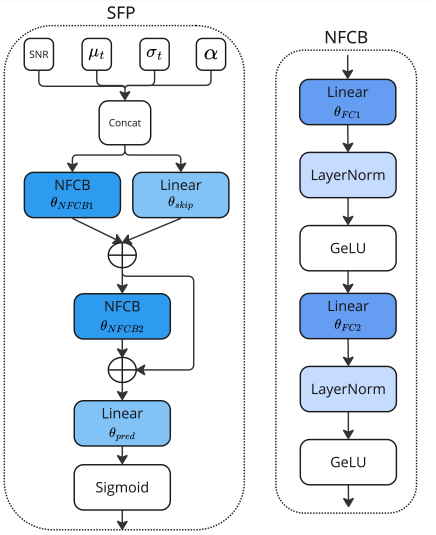


Fig. 6: Architecture of the Semantic Fidelity Predictor (SFP) and its core block Normalized Full-Connected Block (NFCB)

where μ_t and σ_t is the mean and standard deviation of embedding feature EF $\triangleq Embedding(I_j, \theta_{head}) \in \mathbb{R}^{L \times D_{sem}}$, defined as:

$$\begin{aligned} \mu_t &= \frac{1}{L}\sum_{l=1}^{L} \mathrm{EF}[l,:] \\ \sigma_t &= \sqrt{\frac{1}{L}\sum_{l=1}^{L}(\mathrm{EF}[l,:] - \mu_t)^2}. \end{aligned} \tag{26}$$

The operations of SFP G_ω are expressed as:

$$\begin{aligned} x &= \mathrm{Concat}(\gamma, \mu_t, \sigma_t, \alpha) \\ x_{\text{merged}} &= \mathrm{NFCB}(x|\theta_{\text{NFCB1}}) \oplus \mathrm{Linear}(x|\theta_{\text{skip}}) \\ x_{\text{pred}} &= \mathrm{Linear}(\mathrm{NFCB}(x_{\text{merged}}|\theta_{\text{NFCB2}})|\theta_{\text{pred}}) \\ G_\omega(\mu_t, \sigma_t, \gamma, \alpha; \theta_\omega) &= \mathrm{Sigmoid}(x_{\text{pred}}) \end{aligned} \tag{27}$$

where θ_{NFCB1}, θ_{NFCB2} denotes for parameters of the first and second Neural Feedback Control Block (NFCB), respectively; θ_{skip}, θ_{pred} represents for parameters of the skip connection linear layer and the prediction linear layer, respectively. Here, the operation of NFCB as a core block of the SFP, illustrated in Fig. 6, is expressed as follows:

$$\begin{aligned} x_{\text{inter}} &= \mathrm{GeLU}(\mathrm{LayerNorm}(\mathrm{Linear}(x|\theta_{\text{FC1}}))) \\ \mathrm{NFCB}(x; \theta_{\text{NFCB}}) &= \mathrm{GeLU}(\mathrm{LayerNorm}(\mathrm{Linear}(x_{\text{inter}}|\theta_{\text{FC2}}))) \end{aligned} \tag{28}$$

where θ_{FC1} and θ_{FC2} are the parameters of the first and second linear layers, respectively.

2) Slim-Optimizer: The Slim-Optimizer implements an efficient search strategy to determine optimal width multiplier for each transmission. This algorithm addresses a fundamental challenge in semantic communication systems: dynamically adjusting the computational complexity based on both content characteristics and channel conditions while maintaining semantic fidelity, which is presented in Algorithm 1. The algorithm employs an ascending-order search through the available

width multipliers, evaluating the predicted semantic similarity against a predefined threshold. This threshold-based approach enables early termination once a satisfactory configuration is identified, prioritizing resource efficiency while ensuring semantic fidelity requirements are met.

Algorithm 1 Slim-Optimizer

Require: SFP G_ω, statistical features μ_t, σ_t, channel SNR γ, width multipliers $\mathcal{A}$, target sentence similarity threshold SS_{th}

Ensure: Optimal width multiplier α^*

Sort $\mathcal{A}$ in ascending order: $\mathcal{A}_{asc}$
// Test each discrete width multiplier
for α in $\mathcal{A}_{asc}$ **do**
  Predict the semantic similarity: $\widehat{\text{SS}} = G_\omega(\mu_t, \sigma_t, \gamma, \alpha)$
  if $\widehat{\text{SS}} \geq \text{SS}_{th}$ **then**
    Return: α as the optimal width α^*
  end if
end for
Return: $\max(\mathcal{A})$ as the optimal width α^*

F. Training the Bidirectional Slimmable Model

Algorithm 2 Weight-Tying Procedure

Require: Embedding weight θ_{EB}, Sender Bit Transformer block layers $\text{SenderBTs}_{l=1}^{3}$, Receiver Bit Transformer layers $\text{ReceiverBTs}_{l=1}^{3}$, Sender CB layer SenderCB, Receiver CB layer ReceiverCB

Ensure: θ_{head}, θ_{sem}, θ_{CBL1}, θ_{CBL2}

Initialize empty dictionary SumParams, AvgParams
for each parameter name pn in SenderBTs_1 **do**
  $\text{SumParams}[pn] \leftarrow 0$
  for $l := 1$ to 3 **do**
    $SumParams[pn] \leftarrow SumParams[pn] + SenderBTs[l][pn] + ReceiverBTs[l][pn]$
  end for
  $\text{AvgParams}[pn] \leftarrow \text{SumParams}[pn]/(2 \times 3)$
end for
$\theta_{head} \leftarrow \theta_{EB}$
$\theta_{sem} \leftarrow \text{AvgParams}$
$\theta_{CBL1} \leftarrow \text{SenderCB}[CB1]$
$\theta_{CBL2} \leftarrow \text{SenderCB}[CB2]$
Return θ_{head}, θ_{sem}, θ_{CBL1}, θ_{CBL2}

The training algorithm for the BidDeepSC framework incorporates two critical aspects: weight-tying for bidirectional communication and width distillation for slimmable networks.

1) Weight-tying Mechanism: We employed a weight-tying mechanism between the sender and receiver ends to implement bidirectional communication without the need to store separate encoder and decoder models. This approach ensures that the encoder and decoder share the same parameters, significantly reducing memory requirements and computational complexity. weight-tying is implemented as follows:

- **Semantic Blocks (SBs):** For the Bit Transformer layers within the SBs at both the sender and receiver sides, the weights are shared by averaging the parameters across all layers and updating each layer with the averaged parameters. Moreover, a single weight matrix θ_{head} is used for both the Embedding layer on the sender side and the μ-BitLinear layer on the receiver side, ensuring weight consistency across the encoding and decoding processes.
- **Channel Blocks (CBs):** One CB is on each side. Due to their symmetric, the weight of the CB on the sender side is shared with the counterpart in the receiver side.

Algorithm 2 summarizes the weight-tying procedure used in our model.

2) Slimmable Progressive Training: The slimmable mechanism enables the model to operate at multiple widths during inference. However, training a slimmable network presents challenges in maintaining consistent performance across all model widths. To address this problem, Yu *et al.* [15] employed slimmable distillation technique. Nevertheless, this technique exposes drawbacks in semantic communication systems, that are sensitive to overfitting. The largest width is trained first; hence, the narrower widths may override the trained representations of the larger one in an attempt to determine better representations in their slimmed sections. Narrower widths tend to optimize for their current width, potentially ruining the learned representations of wider counterparts. Thus, the model is prone to overfitting and misalignment, potentially leading to narrower widths outperforming larger ones.

From this motivation, we propose slimmable progressive training of the model at multiple widths simultaneously. The critical principle is to train the network at its narrowest width then subsequently train larger-width configurations. By building on efficient representations learned by narrower models, wider variants could achieve better alignment and stability across widths.

3) Loss Function: The training of the BidDeepSC-1.58b framework employs a dual-objective optimization approach targeting both the core semantic communication system and the Semantic Fidelity Predictor (SFP). For the primary BidDeepSC model, cross-entropy loss is adopted to measure the difference between the input sequence $I \in \mathbb{R}^L$ and the reconstructed output $\hat{I} \in \mathbb{R}^{L \times VS}$, formulated as follows:

$$\mathcal{L}(I, \hat{I}) = -\sum_{l=1}^{L} q(w_l)p(w_l) + (1 - q(w_l))log(1 - p(w_l)) \quad (29)$$

where $q(w_l)$ denotes the real probability that the l-th word, w_l, appears in input sentence I, and $p(w_l)$ is the predicted probability that the l-th word, w_l, appears in sentence $\hat{I}$.

Moreover, SFP is specifically trained to minimize the mean squared error between predicted and actual semantic similarity scores, enabling accurate runtime estimation of transmission quality:

$$\mathcal{L}_\omega(\mu_t, \sigma_t, \gamma, \alpha) = (\text{SemSim}(I_j, \hat{I}_j) - \widehat{\text{SS}})^2 \quad (30)$$

where $\widehat{\text{SS}}$ denotes the predicted similarity from the SFP model. This secondary optimization objective facilitates the dynamic width selection mechanism that adaptively balances semantic fidelity and computational efficiency.

4) BidDeepSC Training algorithm: Algorithm 3 represents the overall training process of the proposed system including weight-tying and slimmable progressive mechanisms.

Algorithm 3 Training Algorithm

Require: Training datasets including input tokens X; number of epochs NE; Training signal-to-noise ratio (SNR) TrainingSNR; Learning rate η; Width multipliers $\mathcal{A}$

Ensure: Optimized parameters θ^*_{head}, θ^*_{sem}, θ^*_{CBL1} and θ^*_{CBL2}

$\mathcal{A}_{desc} = Sort(\mathcal{A}, mode = asc)$
Initialize θ_{head}, θ_{sem}, θ_{CBL1} and θ_{CBL2} parameters
Initialize the overall BidDeepSC parameter θ
Load θ_{head}, θ_{sem}, θ_{CBL1} and θ_{CBL2} parameters into BidDeepSC
Set the TrainingSNR for BidDeepSC
for $epoch \leftarrow 1$ to NE **do**
  for mini-batch x in X **do**
    Set the cumulative gradient to 0
    for α in $\mathcal{A}_{desc}$ **do**
      Compute the predicted $\hat{I} \leftarrow BidDeepSC(I, \alpha)$
      Compute the loss $\mathcal{L}_\alpha \leftarrow \mathcal{L}(\hat{I}, I)$
      Update the parameters $\theta \leftarrow \theta - \eta \nabla \mathcal{L}_\alpha$
    end for
  end for
  Weight tying θ_{head}, θ_{sem}, θ_{CBL1} and θ_{CBL2} using Algorithm 2
  Load θ_{head}, θ_{sem}, θ_{CBL1} and θ_{CBL2} parameters into BidDeepSC
end for
Return: Optimized θ^*_{head}, θ^*_{sem}, θ^*_{CBL1} and θ^*_{CBL2}

5) SFP Training algorithm: The training process follows Algorithm 4, illustrating the generation of training samples and model optimization.

Algorithm 4 SFP Training

Require: BidDeepSC model with parameters θ, training dataset X, validation dataset X_{val}, SNR values $\mathcal{G}, \gamma_{max}]$, width multipliers $\mathcal{A}$

Ensure: Optimized SFP parameters θ^*_ω

Initialize SFP parameters θ_ω, training dataset $\mathcal{D} = \{\}$
// Generate training samples
for mini-batch x in X **do**
  for SNR γ in $\mathcal{G}$ **do**
    for width multiplier α in $\mathcal{A}$ **do**
      Extract embedding features EF = Embedding$(x|\theta_{head})$
      Compute statistical features μ_t, σ_t from EF
      Computed predicted $\hat{x} \leftarrow$ BidDeepSC$(x, \alpha|\theta^*)$
      Compute semantic similarity SS $\leftarrow$ SemSim$(x, \hat{s})$
      Add $(\mu_t, \sigma_t, \gamma, \alpha, \text{SS})$ to $\mathcal{D}$
    end for
  end for
end for
// Train SFP
for epoch $e = 1$ to E **do**
  for batch $(\mu_t, \sigma_t, \gamma, \alpha, \text{SS})$ in $\mathcal{D}$ **do**
    Compute $\widehat{\text{SS}} = G_\omega(\mu_t, \sigma_t, \gamma, \alpha)$
    Compute loss $\mathcal{L}_\omega = (\text{SS} - \widehat{\text{SS}})^2$
    Update $\theta_\omega \leftarrow \theta_\omega - \eta \nabla \mathcal{L}_\omega$
  end for
  Evaluate on validation set X_{val}
end for
Return: Optimized θ^*_ω

IV. Simulation Results

A. Dataset

A comprehensive dataset amalgamated from five distinct sources is employed To validate the generalization across contexts and demonstrate the capabilities of the proposed BidDeepSC-1.58b system. This diverse collection spans a broad spectrum, ranging from common-sense knowledge to specialized expert data, ensuring a robust evaluation of the performance of the proposed system across domains. The following list provides the component datasets:

- CovidQA [30]: Dataset on scientific queries regarding the coronavirus disease 2019 (COVID-19),
- OpenBookQA [31]: Subject comprehension evaluation dataset,
- SQuADv2 [32]: Dataset on reading comprehension based on Wikipedia,
- FinQA [33]: Financial analysis and numerical reasoning dataset, and
- CommonsenseQA [34]: General knowledge challenge dataset.

Table II presents the number of training and testing data.

TABLE II: Summary of datasets used for simulation

Dataset	Training set	Testing set
CommensenseQA	9741	1140
FinQA	12,502	2294
OpenbookQA	9914	1000
SQuAD v2	260,638	23,731
CovidQA	3230	808
Total	296,025	28,973

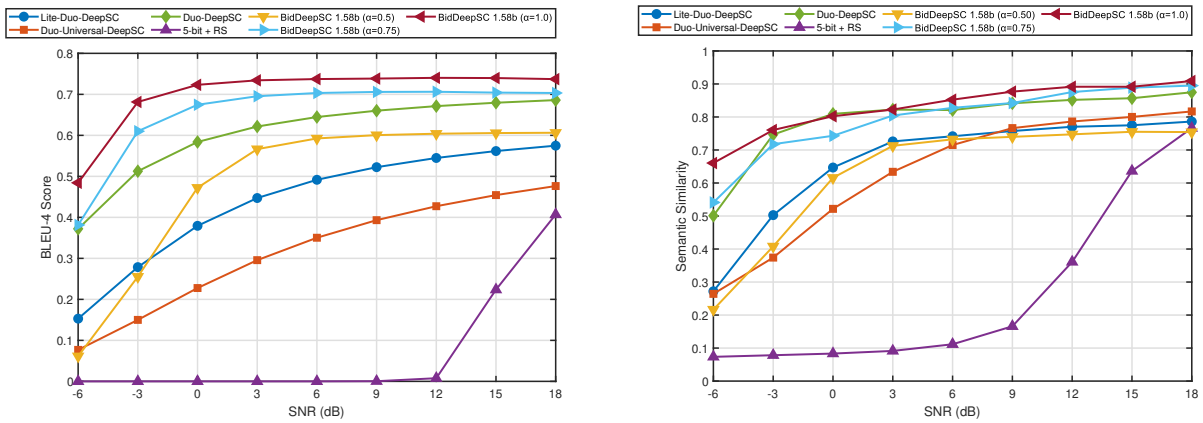
B. Simulation Setup

This experiment was conducted using a system equipped with an Intel Core i7-14700 with 2.1 GHz and an Nvidia GeForce RTX 4070Ti Super with 16 GB VRAM. Table III provides the other simulation setups

TABLE III: Simulation setups

Name	Value
Vocabulary size VS	30 522
Model hidden Size D_{sem}	128
Number of channel symbol D_{chan}	16
Immediate channel size IC	64
Learning rate η	1.00E-04
Tokenizer	bert-base-uncased
BERT model $\boldsymbol{B}$	all-MiniLM-L6-v2
Batch size	256
Training epochs	20
Set of width multipliers $\mathcal{A}$	{0.5, 0.75, 1.0}
Training SNR	12
Optimizer Threshold Sentence Similarity SS_{th}	0.80

For fair comparison, we implement bidirectional versions



(a) BLEU-4 (b) Semantic Similarity

Fig. 7: Performance of the proposed BidDeepSC-1.58b, Lite-DeepSC, DeepSC and Universal-DeepSC framworks with respect to the BLEU-4 and semantic similarity.

of existing unidirectional semantic communication systems. These 'Duo-' variants require storing complete encoder-decoder pairs at each communication endpoint. Specifically:

- Duo-DeepSC: A bidirectional implementation of the foundational Transformer-based DeepSC architecture [28], utilizing separate encoder-decoder pairs for each transmission direction.
- Lite-Duo-DeepSC: A bidirectional implementation derived from Lite-DeepSC [4], incorporating quantization (8-bit) and pruning techniques (0.6 sparsity ratio) to reduce computational requirements
- Duo-Universal-DeepSC: A bidirectional variant of Universal-Transformer [5]. Universal-Transformer-based semantic communication system incorporating an adaptive circulation mechanism to enable the flexible transmission of sentences with yielding semantic information.

This traditional approach ensures bidirectional capability but doubles the storage requirements compared to unidirectional systems, as each node must store parameters for both transmission and reception. In contrast, our proposed BidDeepSC-1.58b uses weight-tying to share parameters between encoder and decoder, eliminating this storage redundancy while maintaining bidirectional functionality. While the Duo- baseline models require double storage due to separate encoder-decoder pairs, they do not incur additional computational overhead during operation, as only one direction (either encoding or decoding) is active at any given time.

Moreover, we adopted the conventional 5-bit + RS method which is a conventional method that employs fixed-length 5-bit coding as the source coding and Reed-Solomon (RS) coding as the channel coding.

C. Performance Comparison

Fig. 7 illustrates the performance of BidDeepSC-1.58b compared to baselines across various SNR levels. Performance improves and then saturates with increasing SINR levels for all models.

First, we can see that the proposed approach full-width configuration (α=1.0) consistently provides higher performance than the reduced-width variants (α=0.75, 0.50), although the α=0.75 configuration maintained competitive performance.

Second, the proposed models with (α=1.0, 0.75) consistently outperformed all baselines in both BLEU-4 score and semantic similarity across almost entire SNR range. These performance gains become critical for resource-limited devices operating in challenging wireless environments, where existing approaches fail to maintain semantic integrity due to their computational overhead preventing real-time processing. At a -6 dB SNR, BidDeepSC-1.58b (α=1.0) achieved a BLEU-4 score of 0.4843 and a semantic similarity of 0.6605, surpassing the next-best model, Duo-DeepSC, by 30.15% on BLEU-4 score (0.3721) and 31.86% on semantic similarity (0.5009). At an 18 dB SNR, the proposed BidDeepSC-1.58b framework (α=1.0) demonstrated superior performance, achieving a BLEU-4 score of 0.7372 and semantic similarity of 0.9089, outperforming the benchmark Duo-DeepSC baseline by 7.48% and 3.90% in these metrics (0.6859 and 0.8748, respectively).

Lastly, the conventional 5-bit + RS approach demonstrated significantly inferior performance, particularly at lower SNR levels, achieving semantic similarity below 0.1 at 6 dB SNR and remaining considerably behind all deep learning-based approaches throughout the testing range. This performance gap highlights the fundamental advantage of semantic communication frameworks over traditional bit-oriented transmission systems in maintaining meaning across challenging channel conditions.

D. Model Size and Computation Complexity

Table IV provides a detailed comparison of the model size, parameter count, and computational demands across proposed system and baselines. The proposed BidDeepSC-1.58b framework demonstrates exceptional efficiency in terms of storage requirements. With a model size of only 3.7935 MB, it achieves a 79.9% reduction compared to Lite-Duo-DeepSC

TABLE IV: Comparison of model size, parameter count and computational complexity for BidDeepSC-1.58b and baseline semantic communication systems

Model	Size (MB)	Stored Parameters	MFLOPS
Lite-Duo-DeepSC	18.8783	21,443,476	27.14
Duo-DeepSC	107.5767	28,200 596	50.88
Duo-Universal-DeepSC	139.1680	28,800,728	47.32
BidDeepSC-1.58b ($\alpha = 1.0$)	3.7935	4,170,624	17.50
BidDeepSC-1.58b ($\alpha = 0.75$)	3.7935	4,170,624	14.12
BidDeepSC-1.58b ($\alpha = 0.5$)	3.7935	4,170,624	7.82

(18.8783 MB), a 96.5% reduction compared to Duo-DeepSC (107.5767 MB), and a 97.3% reduction compared to Duo-Universal-DeepSC (139.1680 MB). This substantial decrease in storage requirements enables deployment in severely resource-constrained environments, particularly in IoT devices with limited memory capacity. This sub-4MB footprint enables deployment on commercial IoT platforms like ESP32 or STM32 series, which existing approaches cannot support due to their memory requirements exceeding available flash storage. The proposed BidDeepSC-1.58b framework also requires only 4,170,624 parameters, representing an 80.5% reduction compared to Lite-Duo-DeepSC (21,443,476 parameters) and approximately 85.5% reduction compared to both Duo-DeepSC (28,200,596 parameters) and Duo-Universal-DeepSC (28,800,728 parameters). The significant parameter reduction is primarily attributable to the unified encoder-decoder architecture and extreme 1.58-bit quantization employed in the proposed system. The proposed BidDeepSC-1.58b framework provides lower computational complexity in terms of mega floating-point operations per second (MFLOPS). At full width (α = 1.0), BidDeepSC-1.58b requires only 17.50 MFLOPS, which is 35.5% lower than Lite-Duo-DeepSC (27.14 MFLOPS), 65.6% lower than Duo-DeepSC (50.88 MFLOPS), and 63.0% lower than Duo-Universal-DeepSC (47.32 MFLOPS). The slimmable architecture enables adaptive computational scaling, with requirements decreasing to 14.12 MFLOPS at α = 0.75 and 7.82 MFLOPS at α = 0.5. This decrease represents a computational reduction of up to 84.6% compared to traditional Duo-DeepSC implementations.

TABLE V: End-to-end inference latency comparison (in ms) of BidDeepSC-1.58b and baseline semantic communication systems

Model	Mean ± Std	Min	Max	Median	P95
Duo-DeepSC	2.91 ± 1.14	2.02	12.33	2.28	4.90
Lite-Duo-DeepSC	2.50 ± 0.88	2.03	10.26	2.21	3.99
Duo-Universal-DeepSC	39.04 ± 6.69	31.04	167.55	37.68	47.28
5-bit + RS	1.12 ± 0.46	0.26	2.01	1.15	1.52
BidDeepSC-1.58b ($\alpha = 1.0$)	0.47 ± 0.17	0.39	1.93	0.43	0.84
BidDeepSC-1.58b ($\alpha = 0.75$)	0.39 ± 0.14	0.33	1.79	0.36	0.73
BidDeepSC-1.58b ($\alpha = 0.5$)	0.31 ± 0.12	0.26	1.58	0.28	0.62

Table V compares the end-to-end inference latency comparison of BidDeepSC-1.58b at three slimmable widths and baselines. The proposed BidDeepSC-1.58b demonstrates notable inference performance, achieving a mean processing time of only 0.47 ms at full width (α=1.0). This result represents an 83.8% reduction compared to Duo-DeepSC (2.91 ms) and an 81.2% reduction compared to Lite-Duo-DeepSC (2.50 ms). The improvement becomes substantially more pronounced when compared to Duo-Universal-DeepSC, yielding a 98.8% latency reduction. The conventional 5-bit + RS approach, although faster than other baselines, still has 58% higher latency and three times greater variability (±0.46 ms vs. ±0.17 ms) compared to the full-width BidDeepSC-1.58b.

The slimmable architecture of BidDeepSC offers significant computational efficiency gains while maintaining a fixed 3.79MB model size across all configurations. Reducing width from $\alpha = 1.0$ to $\alpha = 0.5$ decreases inference latency by 34% (0.47ms to 0.31ms) for 1.58-bit quantization, with similar 30% improvements observed in FP16 implementations (2.60ms to 1.83ms). This computational advantage comes with a semantic performance trade-off, decreasing similarity by 17% (from 0.9089 to 0.75424) at an 18dB SNR.

The performance-complexity balance displays strong SNR-dependency, with narrowing performance gaps at higher SNR levels ($\geq$9 dB), making $\alpha = 0.75$ configurations optimal for balancing computation complexity and semantic fidelity in favorable channel conditions. This characteristic enables adaptive resource allocation based on transmission environments, preserving computational resources when channel conditions permit while maintaining semantic quality thresholds.

E. Slim-Optimizer Efficiency

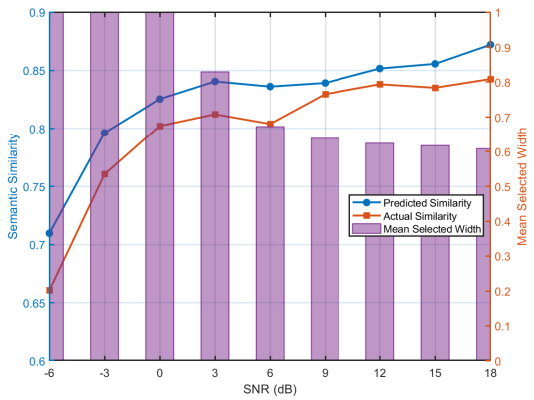


Fig. 8: Semantic similarity prediction accuracy and dynamic width allocation of Slim-Optimizer across SNR levels

The Slim-Optimizer demonstrates efficient width selection across varying channel conditions. The experimental results in Table 8 reveal that the predicted accuracy lies within 7.45% of measured actual values across all SNR levels. At low SNRs (-6 to 0 dB), the optimizer consistently selects maximum width (α=1.0), ensuring semantic fidelity in challenging conditions. As channel conditions improve (3 to 18 dB), the average selected width decreases systematically from 0.829 to 0.608, achieving 39.2% computational savings while maintaining semantic similarity above the semantic similarity threshold (0.80). These results validate the capacity of Slim-Optimizer, providing the required performance using minimal computational complexity across diverse operational scenarios by optimally controlling of the slim width.

F. Ablation Study on Weight-tying

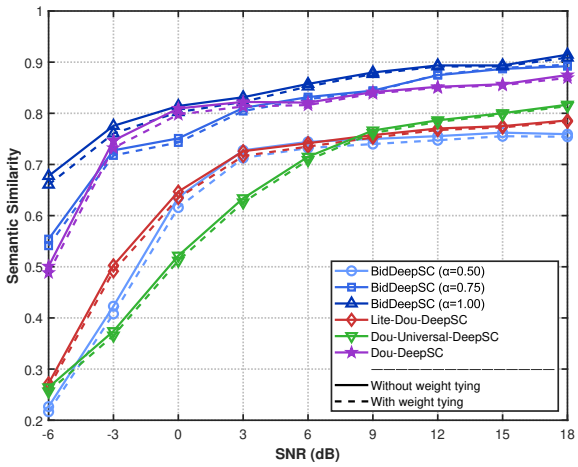


Fig. 9: Ablation study on weight-tying of proposed model and baseline models across SNR levels.

Fig. 9 presents an ablation study of BidDeepSC-1.58b between models with and without weight-tying across different widths and SNR levels in terms of semantic similarity. For all width multipliers α, the weight sharing provides slightly lower performance, consistently under-performing with $\alpha = 0.5$, whereas it performs competitively at $\alpha = 0.75$ and 1.0. At low SNRs (-6 to 0dB), models without weight-tying generally provide better performance than their weight-tied counterparts. For instance, at a -6 dB SNR, the full-width ($\alpha = 1.0$) model without weight sharing achieves a semantic similarity of 0.6775, compared to 0.6605 for the weight-tied model. This 2.5% performance gap represents the fundamental trade-off of parameter sharing: while separate encoder-decoder pairs could specialize for their respective tasks, the double memory requirement (15.91MB vs 7.95MB) exceeds typical IoT device constraints, making weight-tying necessary despite minor performance degradation. This pattern holds across baseline architectures, with performance gaps of 3.1% (Lite-Dou-DeepSC), 3.0% (Dou-Universal-DeepSC), and 2.5% (Dou-DeepSC) at -6 dB SNR. However, as the SNR increases, the performance gap significantly narrows. From a 9 dB SNR onwards, the performance difference becomes minimal, especially for larger-width multipliers. At an 18 dB SNR, the full-width ($\alpha = 1.0$) model with weight sharing (0.9089) slightly underperforms its non-weight-tied counterpart (0.9146), with a marginal difference of only 0.0057. All baseline models similarly converge to negligible differences: 0.4% (Lite-Dou-DeepSC), 0.3% (Dou-Universal-DeepSC), and 0.6% (Dou-DeepSC) at 18 dB SNR. These findings suggest that, although weight sharing may slightly compromise performance in very noisy conditions, it offers comparable performance in better channel conditions, especially for larger model widths. This outcome makes the weight-tying technique valuable for enabling bidirectional communication capabilities with minimal performance trade-offs, particularly in moderate-to-high SNR scenarios.

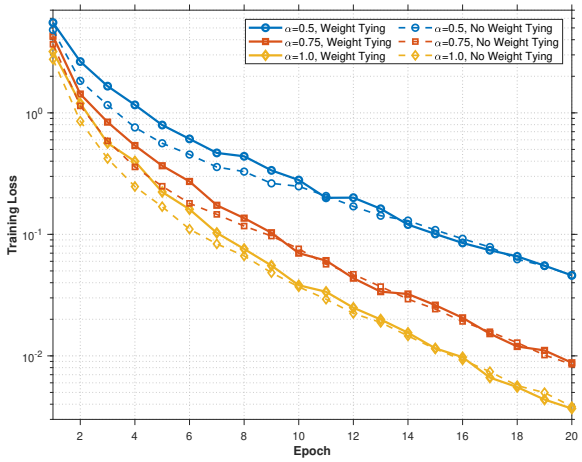


Fig. 10: Ablation study on the training loss of BidDeepSC-1.58b with and without weight-tying.

Fig. 10 illustrates the convergence behavior of BidDeepSC-1.58b with and without weight-tying across different width multipliers. For all width configurations, models without weight-tying initially demonstrate faster convergence in the first few epochs. For instance, at epoch 5, the full-width ($\alpha = 1.0$) model without weight-tying achieves a loss of 0.1689 compared to 0.2232 for its weight-tied counterpart. However, as training progresses beyond epoch 10, the convergence rates become remarkably similar, with both approaches achieving comparable final loss values. By epoch 20, the full-width models reach nearly the same loss values (0.0036 with weight-tying versus 0.0039 without weight-tying), demonstrating that weight-tying ultimately achieves equivalent optimality despite the slower initial convergence. The impact of the width multiplier on convergence is significant for both training approaches, with smaller width models ($\alpha = 0.5$) consistently exhibiting higher loss values throughout training compared to larger-width configurations. This convergence analysis confirms that, although weight-tying might slightly affects the initial training dynamics, it performs comparably final performance with no significant effect on ultimate convergence, making it a valuable technique to enable bidirectional capabilities with minimal performance trade-offs.

The experimental analysis reveals a favorable tradeoff profile for weight-tying in bidirectional semantic communication systems. Weight-tying achieves precise parameter count reduction by 50% (from 8,341,248 to 4,170,624) and corresponding model size reduction (from 15.91MB to 7.95MB in FP16). weight-tying has minimal impact on performance in favorable channel conditions. Semantic similarity differences become negligible ($\approx$0.005) at a 9dB SNR and above though a modest gap exists at low SNR (-6dB to 0dB). These findings demonstrate that weight-tying provides substantial storage complexity gains with minimal performance impact, particularly in moderate-to-favorable channel conditions.

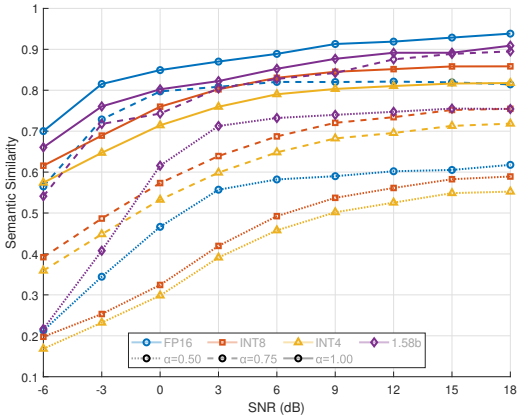


Fig. 11: Ablation study on 1.58-bit quantization across SNR levels and model widths, where INT4, INT8 and FP16 denote for 4-bit integer quantization, 8-bit integer quantization and 16-bit floating point quantization, respectively.

G. Ablation Study on Quantization Levels

We conducted a comprehensive analysis to evaluate the influence of various quantization techniques on model performance and efficiency. Fig. 11 illustrates the performance of the proposed 1.58-bit quantized model across different signal-to-noise ratio (SNR) levels and model widths. The 1.58-bit quantization approach demonstrates remarkable resilience across challenging channel conditions while maintaining competitive performance compared to higher-precision alternatives. The results reveal that the 1.58-bit quantized model achieves semantic similarity ranging from 0.6605 at a -6 dB SNR to 0.9089 at an 18 dB SNR under a width of $\alpha = 1.0$. Notably, at higher SNR levels (12 to 18 dB), the 1.58-bit model outperforms its INT4 and INT8 counterparts, demonstrating the effectiveness of ternary weight representation. This superior performance can be attributed to the optimization of 1.58b quantization, enabling more effective gradient propagation during training and reducing quantization noise compared to uniform quantization schemes in INT4 and INT8. Similarly, under width $\alpha = 0.75$, the 1.58-bit model performs competitively, particularly at higher SNR levels where it achieves up to 0.8950 semantic similarity.

TABLE VI: Ablation study related to the model size and stored parameters of the BidDeepSC variants

Weight-tying	Quantization	Size (MB)	Stored Parameters
No	FP32	31.8193	8,341,248
No	FP16	15.9097	8,341,248
Yes	FP16	7.9548	4,170,624
Yes	INT8	5.9823	4,170,624
Yes	INT4	4.8356	4,170,624
Yes	1.58-bit	3.7935	4,170,624

Table VI presents the model size reduction achieved via quantization techniques in combination with weight-tying. The baseline model without weight-tying using FP32 precision requires 31.82 MB of storage for 8,341,248 parameters. Implementing weight-tying alone reduces the parameter count by 50% to 4,170,624, with a corresponding reduction in model size to 7.95 MB when using FP16 precision. Progressive quantization further reduces the model size, with INT8, INT4, and 1.58-bit quantization achieving model sizes of 5.98 MB, 4.84 MB, and 3.79 MB, respectively. Each quantization method presents different trade-offs in size versus performance. INT8 requires 5.98 MB, which is 57.7% larger than 1.58-bit, while maintaining similar semantic similarity according to Figure 11. INT4 achieves better compression at 4.84 MB but still exceeds 1.58-bit by 27.3%, and Figure 11 shows it underperforms at high SNR levels. These comparisons demonstrate that 1.58-bit quantization provides the best combination of minimal storage (3.79 MB) and strong semantic performance. Notably, the proposed 1.58-bit quantization approach provides an 88.1% reduction in model size compared to the FP32 baseline model while maintaining robust performance across various SNR conditions. This superior compression permits deployment in resource-constrained environments without substantial performance degradation.

TABLE VII: Ablation study on end-to-end inference latency (ms) for BidDeepSC by different quantization levels and slimmable width multipliers.

Quantization	α	Mean $\pm$ Std	Min	Max	Median	P95
FP16	1.0	2.60 $\pm$ 0.84	2.12	10.51	2.34	4.60
	0.5	1.83 $\pm$ 0.67	1.53	8.47	1.72	3.45
INT8	1.0	2.32 $\pm$ 0.79	1.92	9.85	2.12	4.15
	0.75	1.89 $\pm$ 0.68	1.63	8.72	1.78	3.62
	0.5	1.52 $\pm$ 0.61	1.31	7.83	1.43	3.07
INT4	1.0	1.98 $\pm$ 0.74	1.68	9.25	1.84	3.75
	0.75	1.67 $\pm$ 0.63	1.43	8.36	1.59	3.38
	0.5	1.32 $\pm$ 0.58	1.17	7.09	1.23	2.83
1.58-bit	1.0	0.47 $\pm$ 0.17	0.39	1.93	0.43	0.84
	0.75	0.39 $\pm$ 0.14	0.33	1.79	0.36	0.73
	0.5	0.31 $\pm$ 0.12	0.26	1.58	0.28	0.62

Table VII demonstrates the considerable influence of quantization and model width on the BidDeepSC's end-to-end inference latency. Moving from FP16 to 1.58-bit quantization dramatically reduces mean latency by 82% (from 2.60 to 0.47 ms) at full width ($\alpha = 1.0$). This latency reduction stems from the ternary weight representation enabling hardware-efficient operations: matrix multiplications reduce to simple additions and subtractions, eliminating costly floating-point arithmetic required by FP16 (2.60ms) and reducing memory bandwidth by 90% compared to INT8 (2.32ms). Moreover, width reduction provides additional efficiency gains. The 1.58-bit model with $\alpha = 0.5$ achieves an outstanding 0.31ms mean latency, which represents an 88% reduction from the FP16 full-width model. Notably, this configuration also has the lowest variability ($\pm$0.12ms) and minimum latency (0.26ms). That is, the combined effect of extreme quantization and reduced width provides remarkable computational overhead reduction without sacrificing communication performance, as demonstrated in our earlier semantic similarity results. This efficiency is further evidenced by the dramatic reduction in P95 latency values from 4.60ms (FP16, $\alpha = 1.0$) to 0.62ms (1.58-bit, $\alpha = 0.5$), indicating enhanced consistency for time-sensitive applications. These findings confirm that BidDeepSC's slimmable architecture with 1.58-bit quantization achieves an optimal balance between computational complex-

ity and communication performance, making it ideally suited for resource-constrained edge devices operating across varying channel conditions.

Remark 1. *For image transmission, convolutional feature extractors could replace text embeddings while maintaining the unified encoder-decoder structure. Radar and sensor signals align naturally with our temporal architecture but require extending ternary weights to quaternary {-1, -j, +j, +1} for complex-valued representations. Future work will address these multimodal challenges through modality-specific quantization schemes that preserve semantic features under extreme compression, cross-modal weight-sharing mechanisms that prevent catastrophic interference between diverse data distributions, and dynamic width allocation strategies for real-time multimodal streams. Additionally, comprehensive real-world latency measurements on embedded platforms such as ARM Cortex-M and RISC-V microcontrollers will validate deployment feasibility and guide hardware-specific optimizations for ultra-low-power multimodal IoT applications, ultimately enabling efficient bidirectional semantic communication across text, image, and signal modalities in resource-constrained environments.*

V. Conclusion

This paper presentes BidDeepSC-1.58b, a bidirectional semantic communication framework addressing critical computational complexity and performance challenges through three synergistic innovations: weight-tying, 1.58 quantization, and slimmable architecture. The weight-tying mechanism between encoder and decoder components significantly reduces the parameter count while sustaining performance at moderate-to-high SNR levels. Implementating of 1.58-bit quantization further optimizes storage requirements while achieving competitive or superior performance compared to its higher-precision counterparts. The slimmable architecture enables dynamic complexity adjustments based on channel conditions, optimizing the performance-efficiency trade-off in diverse operational scenarios. The validation via experiments demonstrates that the proposed framework consistently outperforms baseline systems in semantic fidelity metrics while substantially reducing computational demands and the memory footprint. These advances facilitate deployment of semantic communication system deployment in resource-constrained environments while maintaining robust performance across varying transmission conditions.

References

[1] G. Stanco, A. Navarro, F. Frattini, G. Ventre, and A. Botta, "A comprehensive survey on the security of low power wide area networks for the Internet of Things," *ICT Express*, vol. 10, no. 3, pp. 519–552, Jun. 2024.

[2] M. t. Ergen, "Edge computing in future wireless networks: A comprehensive evaluation and vision for 6G and beyond," *ICT Express*, Aug. 2024.

[3] H. Xie, Z. Qin, G. Y. Li, and B.-H. Juang, "Deep learning enabled semantic communication systems," *IEEE Trans. Signal Process.*, vol. 69, pp. 2663–2675, 2021.

[4] H. Xie and Z. Qin, "A lite distributed semantic communication system for Internet of Things," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 142–153, Jan. 2021.

[5] Q. Zhou, R. Li, Z. Zhao, C. Peng, and H. Zhang, "Semantic communication with adaptive universal transformer," *IEEE Wireless Commun. Lett.*, vol. 11, no. 3, pp. 453–457, Mar. 2022.

[6] J. t. Dai, "Nonlinear transform source-channel coding for semantic communications," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 8, pp. 2300–2316, Aug. 2022.

[7] H. Yoo, L. Dai, S. Kim, and C.-B. Chae, "On the role of ViT and CNN in semantic communications: Analysis and prototype validation," *IEEE Access*, vol. 11, pp. 71 528–71 541, 2023.

[8] P. Yi, Y. Cao, X. Kang, and Y.-C. Liang, "Deep learning-empowered semantic communication systems with a shared knowledge base," *IEEE Trans. Wireless Commun.*, vol. 23, no. 6, pp. 6174–6187, Jun. 2024.

[9] Q. Hu, G. Zhang, Z. Qin, Y. Cai, G. Yu, and G. Y. Li, "Robust semantic communications with masked VQ-VAE enabled codebook," *IEEE Trans. Wireless Commun.*, vol. 22, no. 12, pp. 8707–8722, Dec. 2023.

[10] H. Xie, Z. Qin, and G. Y. Li, "Task-oriented multi-user semantic communications for VQA," *IEEE Wireless Commun. Lett.*, vol. 11, no. 3, pp. 553–557, Mar. 2022.

[11] G. Zhang, Q. Hu, Z. Qin, Y. Cai, G. Yu, and X. Tao, "A unified multi-task semantic communication system for multimodal data," Jun. 2024.

[12] T. S. Do, T. P. Truong, T. Do, H. P. Van, and S. Cho, "Lightweight multiuser multimodal semantic communication system for multimodal large language model communication," Aug. 2024. [Online]. Available: http://dx.doi.org/10.22541/au.172479430.09168922/v1

[13] K. Yu, Q. He, and G. Wu, "Two-way semantic communications without feedback," *IEEE Trans. Veh. Technol.*, vol. 73, no. 6, pp. 9077–9082, Jun. 2024.

[14] J. Yu, L. Yang, N. Xu, J. Yang, and T. Huang, "Slimmable neural networks," Dec. 2018.

[15] J. Yu and T. Huang, "Universally slimmable networks and improved training techniques," Oct. 2019.

[16] J. Yu and T. S. Huang, "Network slimming by slimmable networks: Towards one-shot architecture search for channel numbers," 2019. [Online]. Available: http://arxiv.org/abs/1903.11728

[17] C. Li, G. Wang, B. Wang, X. Liang, Z. Li, and X. Chang, "Dynamic slimmable network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 8607–8617.

[18] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet classification using binary convolutional neural networks," in *Comput. Vis.–ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 525–542.

[19] A. Bulat and G. Tzimiropoulos, "Xnor-net++: Improved binary neural networks," 2019. [Online]. Available: http://arxiv.org/abs/1909.13863

[20] J. t. Yang, "Quantization networks," in *2019 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*. Long Beach, CA, USA: IEEE, Jun. 2019, pp. 7300–7308.

[21] J. Maly and R. Saab, "A simple approach for quantizing neural networks," *Appl. Comput. Harm. Anal.*, vol. 66, pp. 138–150, Sep. 2023.

[22] K. Xu, L. Han, Y. Tian, S. Yang, and X. Zhang, "EQ-Net: Elastic quantization neural networks," *2023 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2023.

[23] H. Wang, S. Ma, L. Dong, S. Huang, H. Wang, L. Ma, F. Yang, R. Wang, Y. Wu, and F. Wei, "Bitnet: Scaling 1-bit transformers for large language models," 2023. [Online]. Available: https://arxiv.org/abs/2310.11453

[24] S. Ma, H. Wang, L. Ma, L. Wang, W. Wang, S. Huang, L. Dong, R. Wang, J. Xue, and F. Wei, "The era of 1-bit llms: All large language models are in 1.58 bits," 2024. [Online]. Available: https://arxiv.org/abs/2402.17764

[25] Z. Yuan, R. Zhou, H. Wang, L. He, Y. Ye, and L. Sun, "Vit-1.58b: Mobile vision transformers in the 1-bit era," 2024. [Online]. Available: https://arxiv.org/abs/2406.18051

[26] W. t. Yang, "Semantic communication meets edge intelligence," *IEEE Wireless Commun.*, vol. 29, no. 5, pp. 28–35, Oct. 2022.

[27] X. Luo, H.-H. Chen, and Q. Guo, "Semantic communications: Overview, open issues, and future research directions," *IEEE Wireless Commun.*, vol. 29, no. 1, pp. 210–219, Feb. 2022.

[28] H. Xie, Z. Qin, G. Y. Li, and B.-H. Juang, "Deep learning based semantic communications: An initial investigation," in *GLOBECOM 2020 IEEE Global Commun. Conf.*, Dec. 2020, pp. 1–6.

[29] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré, "FlashAttention: Fast and memory-efficient exact attention with IO-awareness," Jun. 2022.

[30] T. Möller, A. Reina, R. Jayakumar, and M. Pietsch, "COVID-QA: A question answering dataset for COVID-19," in *Proc. 1st Workshop NLP COVID-19 ACL 2020*, K. t. Verspoor, Ed. Online: Association for Computational Linguistics, Jul. 2020.

[31] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, "Can a suit of armor conduct electricity? a new dataset for open book question answering," 2018. [Online]. Available: https://arxiv.org/abs/1809.02789

[32] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," 2016. [Online]. Available: https://arxiv.org/abs/1606.05250

[33] Z. Chen, W. Chen, C. Smiley, S. Shah, I. Borova, D. Langdon, R. Moussa, M. Beane, T.-H. Huang, B. Routledge, and W. Y. Wang, "Finqa: A dataset of numerical reasoning over financial data," 2022. [Online]. Available: https://arxiv.org/abs/2109.00122

[34] A. Talmor, J. Herzig, N. Lourie, and J. Berant, "Commonsenseqa: A question answering challenge targeting commonsense knowledge," 2019. [Online]. Available: https://arxiv.org/abs/1811.00937