CNN-Based Iterative Decoding for Polar Codes in MIMO Systems: Performance Analysis and Computational Complexity Evaluation

Jun Wang[1,] Li Lu[2,*]
[1]School of Electrical and Electronics Information Engineering, Hubei Polytechnic University, Huangshi 435000, China
[2]NewRadio Technology Company Ltd, Shenzhen 518000, China
E-mail: WangJun_19910108@outlook.com, lu.li188@hotmail.com
[*]Corresponding author

Keywords: convolutional neural network (CNN), polarized codes (Polar Codes), multiple-input multiple-output (MIMO) systems

Received: February 27, 2025

With the continuous development of 5G technology, wireless communication systems demand higher spectral efficiency and robust error correction. In this study, we propose an iterative decoding algorithm for polar codes in MIMO systems, based on a convolutional neural network (CNN). The CNN architecture comprises three 1D convolutional layers with ReLU activation, a depthwise separable convolution layer, and two fully connected layers. Training was conducted using a synthetic dataset of 100,000 samples generated under AWGN, Rayleigh, and Rician channel models, with 80% used for training and 20% for validation. Evaluation metrics include bit error rate (BER), throughput (Mbps), and computational complexity (ms/symbol). Compared to BP, SC, and SCL algorithms, our model achieves a 30% reduction in BER at 6 dB SNR, throughput improvements of up to 25%, and reduced processing latency across 2×2, 4×4, and 8×8 antenna configurations. The experimental results show that the proposed algorithm exhibits better performance than traditional decoding methods under varying SNR levels, channel models, antenna configurations, and data rates. While not entirely eliminating complexity, the model leverages depthwise separable convolution to reduce parameter size and training overhead, making it more efficient than conventional iterative decoders. This research provides a promising step toward resolving the open challenge of designing decoders that balance adaptability and computational feasibility in MIMO systems.

Povzetek: Članek uvaja CNN-iterativno dekodiranje polarnih kod v MIMO, ki znižuje BER, poveča prepustnost ter zmanjša zakasnitev, ponuja bolj kvalitetno uravnoteženje točnosti in kompleksnosti kot BP, SC in SCL.

1 Introduction

With the rapid development of the fifth-generation mobile communication system (5G), wireless communication technology is facing unprecedented challenges and opportunities. In order to meet the increasing demand for data transmission, improving spectrum efficiency has become a key research direction. MultipleInput MultipleOutput (MIMO) technology and Polar Codes, as an important means to enhance the performance of wireless communication systems, have received extensive attention in recent years [1,2].

MIMO technique achieves spatial multiplexing and diversity gain by utilizing multiple antennas, which can significantly increase the capacity of the system without increasing the bandwidth. And polarization code, as an efficient forward error correction coding scheme, is able to approach the Shannon limit and ensure the reliability of data transmission. However, in practical applications, how to effectively combine these two techniques and develop decoding algorithms with high adaptability and low computational complexity remains an open problem [3]. Convolutional Neural Networks (CNNs), as a powerful machine learning tool, have achieved excellent results in many fields such as image recognition and natural language processing. In recent years, researchers have begun to try to apply CNNs in the field of communication, especially in channel decoding. The powerful feature extraction capability and nonlinear mapping properties of CNNs give them a unique advantage in processing complex communication signals. Convolutional neural network is a deep learning model that mimics the structure of biological visual cortex. CNNs consist of multiple convolutional layers, pooling layers, and fully connected layers [4].

MIMO systems utilize multiple transmitting and receiving antennas to enhance the quality of the wireless link. At the transmitter side, information bits are encoded

and transmitted simultaneously through different antennas; at the receiver side, the received signals are decoded and combined to recover the original information [5]. The key to MIMO systems lies in the design of efficient coding and decoding algorithms, which enable the system to maintain good performance under various complex channel conditions. Polarization code is a new type of coding scheme. Its core idea is to make the channel gradually differentiate into two extremes: good channel and bad channel through a specific coding method [6]. Information bits are transmitted on the good channel, while frozen bits (i.e., known fixed values) are transmitted on the bad channel. This differentiation effect becomes more pronounced as the length of the coding block grows, eventually making some of the sub-channels almost perfect, thus achieving coding efficiency close to the Shannon limit.

The purpose of this paper is to evaluate an innovative convolutional neural network (CNN)-based iterative decoding algorithm for polarization codes in MIMO systems, and to explore its advantages and limitations over traditional decoding methods through an exhaustive comparative study. Specifically, our research will focus on four main issues: first, we will explore how to design an iterative CNN decoding algorithm for polarization codes in MIMO systems, including the core architecture, training strategy, and optimization techniques; second, we will evaluate the performance of the new algorithm through simulation tests in multiple channel environments and make a comprehensive comparison with the existing decoding methods to validate its superiority; and third, we will investigate its advantages and limitations over traditional decoding methods through a detailed comparative study. superiority; again, we will analyze the specific application scenarios in which the new algorithm performs well, especially in high data rate transmission and complex wireless environments, and examine its stability and robustness; finally, we will quantitatively analyze the computational complexity of the new algorithm and search for the potential optimization paths with a view to reducing the computational cost while maintaining the performance [7].

To better structure our study and provide a quantifiable evaluation, we formulate the following research questions (RQs):

RQ1: How does the CNN-based iterative decoding algorithm for polar codes in MIMO systems compare in terms of Bit Error Rate (BER) across varying Signal-to-Noise Ratio (SNR) levels, compared to traditional methods such as BP, SC, and SCL?

RQ2: What are the computational trade-offs, particularly in terms of per-symbol processing time and system latency, when employing CNN-based decoding in real-time applications?

RQ3: How does the CNN-based decoder generalize to unseen channel models, including those not included during training, and what are its robustness characteristics?

While this work presents a CNN-based decoder that demonstrates lower BER and relatively lower per-symbol decoding latency compared to conventional methods, the broader challenge of achieving truly low-complexity and hardware-efficient decoding in all deployment scenarios remains ongoing. Our proposed approach contributes to this goal by reducing inference burden and improving generalization performance, but further optimization is still required for large-scale practical deployment.

This study makes the following novel contributions to the field of CNN-based iterative decoding for MIMO systems:

We propose a hybrid decoding architecture that integrates a CNN into the iterative loop of polar code decoding, enabling adaptive feature refinement across multiple decoding stages.

The CNN employs a depthwise separable convolution design to significantly reduce model complexity while maintaining expressive capacity, which has not been previously applied in this specific context.

A dynamic learning rate scheduler and regularization-aware loss function are introduced to enhance convergence and generalization under various channel models.

The method is evaluated across multiple antenna configurations and channel environments with extensive simulation and complexity analysis, offering practical deployment insights.

These contributions go beyond a direct application of CNNs by offering architectural, algorithmic, and training-level innovations tailored for the decoding domain.

This work positions itself at the intersection of deep learning and iterative decoding for MIMO-polar systems. Unlike prior works which apply CNNs to isolated blocks (e.g., ResBP, DirNet), we propose a fully iterative CNN-integrated decoder that balances accuracy and efficiency. Strengths include scalable architecture, fast inference, and applicability across channel models. However, limitations remain in training data dependency and robustness under rapid fading, which we acknowledge as future optimization points.

This paper addresses the challenge of developing low-complexity, high-accuracy decoders for polar-coded MIMO systems. We propose an iterative CNN-based decoder that integrates directly into the decoding loop to refine soft decisions. Our contributions are threefold: (1) We design a CNN architecture optimized with depthwise separable convolution for decoding efficiency; (2) we implement an iterative decoding pipeline that combines CNN refinement with polar decoding; and (3) we conduct extensive simulation comparing performance, complexity, and generalization across multiple channel environments.

2. Literature review

2.1 Current status of convolutional neural network application in channel decoding

In recent years, with the rapid development of deep

learning technology, convolutional neural network (CNN) has achieved remarkable results in image recognition, speech recognition and other fields due to its powerful feature extraction capability. At the same time, researchers have begun to explore the application of CNNs in communication systems, especially in channel decoding. Compared with traditional rule-based decoding algorithms, CNNs are able to automatically learn channel characteristics, thus providing more flexible and efficient decoding solutions. The first CNN-based decoder for LDPC (LowDensity ParityCheck) codes was proposed in the literature, demonstrating the potential of CNNs in improving decoding performance [8]. By designing a simple CNN architecture, they successfully achieved comparable performance to the BP (Belief Propagation) algorithm under AWGN (Additive White Gaussian Noise) channels while reducing the computational complexity [9]. Subsequently, the literature further applies CNNs to decoding Turbo codes, and the results show that the CNN-based method outperforms the classical MAP (Maximum A Posteriori Probability) decoding algorithm in terms of BER and has a faster convergence rate [10]. This shows the advantage of CNN in dealing with complex channel models.

Recent advancements in CNN-assisted decoding include hierarchical networks for turbo decoding, residual CNNs for LDPC decoding, and attention-based encoders for polar code enhancement. Notable works include DirNet [11], ResBP [9], and CNN-aided joint source-channel decoders [12]. These methods collectively demonstrate the value of data-driven structures in approximating complex decoding logic, though scalability remains a concern for massive MIMO scenarios.

2.2 Evolution of polarization codes and their role in 5G communication systems

A polarization code is a coding scheme that approximates the Shannon limit. Polarization codes divide the original channel into multiple subchannels by recursively applying the process of coding and channel polarization, where one part of the subchannels has a channel capacity that tends to 1 and the other part tends to 0. Thus, it is possible to transmit the information bits on subchannels with a channel capacity close to 1, and to transmit the known frozen bits (frozen bits) on subchannels with a channel capacity close to 0 [13]. Polarization codes have been selected by 3GPP (Third Generation Partnership Project) as the coding standard for 5G eMBB (Enhanced Mobile Broadband) control channel due to their excellent performance and low complexity. In the 5G NR (New Radio) standard, polarization codes are used to protect control information such as downlink control information (DCI), scheduling request (SR), etc. to ensure reliable transmission of critical control information [14].

The suitability of CNNs for decoding polar codes stems from two key observations. First, polar codes inherently induce structured dependencies through their recursive channel transformation, resulting in feature locality across subchannels. This matches the CNN's strength in capturing spatial correlations via local receptive fields. Second, decoding polar codes requires estimating marginal bit-wise probabilities—akin to probabilistic classification—making CNNs ideal for learning such mappings from noisy inputs. Thus, CNNs serve as function approximators that can learn implicit decoding heuristics across varying channel conditions.

2.3 Conventional decoding methods for MIMO systems

Conventional decoding methods for MIMO systems mainly include maximum likelihood (ML) decoding, forced-zero (ZF) decoding, and minimum mean square error (MMSE) decoding. Among them, ML decoding, although theoretically able to provide the best performance, is not suitable for practical applications due to the exponential growth of its computational complexity with the number of antennas, and ZF and MMSE decoding, although reducing the complexity, sacrifice the performance in some cases. In contrast, CNN-based decoding methods for MIMO systems provide a novel solution [15]. A CNN-based MIMO detection algorithm is proposed in the literature, which recovers information symbols directly from the received signal by training a CNN model, instead of performing channel estimation followed by decoding as in traditional methods. Experimental results show that this approach significantly reduces the computational resource requirements while maintaining a low BER [16].

2.4 Comparative analysis of related work

Although CNN-based channel decoding methods show many advantages, there are still some challenges. For example, training high-quality CNN models requires a large amount of labeled data, and it is more difficult to obtain real-world data in wireless communications. In addition, the generalization ability of CNN models is also a concern, as variations in channel conditions may lead to degradation of model performance. To overcome these challenges, some researchers have proposed hybrid schemes combining traditional decoding methods and CNNs. [17] proposed a CNN-assisted BP-based algorithm that introduces CNNs during BP iterations to improve decision quality. Experiments demonstrate that this hybrid approach improves the decoding performance while maintaining the flexibility of the BP algorithm. Overall, the CNN-based channel decoding method represents one of the development trends of decoding technology for future communication systems. Although it is still in the research stage, its potential has been widely recognized and is expected to be more widely used in the future with the continuous optimization of algorithms and technological advances [18].

Although CNN-based decoding methods show significant potential due to their ability to extract deep channel features and outperform rule-based methods in many cases, they also face notable limitations.

Specifically, their performance often depends on the availability of large, labeled datasets, and their generalization ability across varying channel conditions remains a concern. These two aspects represent the dual nature of deep learning approaches: they are powerful but data-dependent. The success of a CNN decoder thus hinges on proper training strategies, regularization, and exposure to diverse channel conditions during learning.

A summary of decoding algorithm comparisons is shown in Table 1. This table highlights BER under different SNR levels, computational complexity, throughput, training data requirements, and feasibility for hardware deployment. The CNN-based method achieves lower BER at all SNR levels, moderate training requirements, and favorable hardware adaptability, despite higher initial training costs.

Table 1: Comparative summary of decoding algorithms

Algorithm	BER @ 6dB	Throughput (Mbps)	Complexity (ms/symbol, 4x4 MIMO)	Training Data Needed	Hardware Feasibility
BP	0.07	10	1.5	Low	High
SC	0.1	9	1.3	Low	High
SCL	0.07	10	1.4	Moderate	Medium
CNN-Based	0.03	12	1.2	High	Moderate

3. Systems models and methodologies

3.1 MIMO system model description

In a MIMO system, multiple transmit antennas and receive antennas are used to enhance the quality of the wireless link. Assuming that the system has N_t transmitting antennas and N_r receiving antennas, the signal model of a MIMO system can be expressed as Equation 1 [19].

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n} \tag{1}$$

where $\mathbf{y}$ is the $N_r \times 1$ dimensional received signal vector. $\mathbf{H}$ is the $N_r \times N_t$ dimensional channel matrix representing the channel gain from each transmit antenna to each receive antenna. $\mathbf{x}$ is the $N_t \times 1$ dimensional transmit signal vector.

3.2 Polarized code coding

The coding process of polarized codes can be divided into three main steps: channel polarization, information bit selection and coding.

Step 1: Channel polarization. The original channel W is divided into N sub-channels by recursively applying Bennett's polarization transform, where N is a power of two. The polarization transform can be expressed as the following matrix in Equation 2 [20-21].

$$G_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \tag{2}$$

For the case of N bits, one can construct a polarization matrix G_N of $N \times N$ which is a $\log_2(N)$ subclone of G_2.

Step 2: Information bit selection. Based on the channel capacity of the subchannels, the k subchannels with the highest channel capacity are selected for transmitting information bits, and the remaining Nk subchannels are used for transmitting fixed frozen bits (frozen bits).

Step 3: Coding. The information bits and frozen bits are mapped onto N sub-channels in a certain order and then encoded using the polarization matrix G_N to obtain the encoded bit sequence [22].

In our proposed method, CNN acts on the initial soft outputs derived from the channel (LLRs or symbol likelihoods). During each iteration, the CNN refines these estimates using learned spatial and statistical patterns. The refined values are then passed to a polar decoder (SC or SCL), which updates the bit decisions. This process is repeated for 3–5 iterations.

In Equation (1):

$y \in N_r \times 1$: received signal vector

$H \in N_r \times N_t$: channel matrix

$x \in N_t \times 1$: transmitted signal vector Equation (2) defines the polar transformation G_N, constructed recursively from the Kronecker power of $F = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$. enabling channel polarization into reliable/unreliable subchannels.

3.3 CNN architecture design

In order to cope with the task of polarization code decoding in MIMO systems, we can design an architecture based on Convolutional Neural Networks (CNN). The architecture consists of the following components: first, an input layer, which is responsible for receiving the received signal vector in dimension; then a convolutional layer, which extracts features from the input signal by using multiple convolutional kernels, each corresponding to a specific feature; followed by a pooling layer, which reduces the size of the feature maps by downsampling, thus reducing the complexity of the model; followed by a fully-connected layer, which joins the previously extracted features together and is used for the the final decision-making process; and finally the output layer, which is responsible for outputting the probability distribution of each information bit. This architecture can effectively accomplish the polarization code decoding task in MIMO systems by extracting and processing signal features layer by layer [11, 23].

In order to adapt the polarization code decoding task in MIMO systems, we can improve the performance of the model by improving certain steps in the CNN model. In this scenario, considering the characteristics of the received signal, we can focus on improving the design of the convolutional layer in order to better capture the local features and long-range dependencies in the signal. In the following, we will discuss in detail how to construct such a network and propose some improvements [24-25].

The received signal vector **y** is a $N_r \times 1$ dimensional vector representing all the signals received at the antenna by the receiver. At the input layer, we directly use this vector as an input to the network [12, 26].

The convolutional layer is the core component of a CNN, which is responsible for extracting features from the input data. In traditional 2D image processing, the convolution kernel usually slides over the spatial dimension, whereas in our scenario, we will use 1D convolution kernels since the signal is one-dimensional. Let the size of the convolution kernel be k and the step size be s. Then each convolution kernel will act on a window of the input vector and produce a feature map [27].

Considering that the polarization code decoding task may need to capture more complex feature patterns, we can introduce Depthwise Separable Convolution. Depthwise Separable Convolution consists of two parts: first, a separate convolution operation for each input channel (depthwise convolution), and then a 1x1 convolution for all output channels (pointwise convolution). This approach significantly reduces the number of parameters while maintaining the expressive power of the model. The operation of deeply separable convolution can be expressed as Equation 3 and Equation 4. Here v_j is the weight of 1x1 convolution kernel and $f(x)_j$ is the result of depthwise convolution [28].

$$f(x) = \sum_{i=1}^{k} w_i x_{t-i+1} \tag{3}$$

$$g(f(x)) = \sum_{j=1}^{d} v_j f(x)_j \tag{4}$$

The main purpose of the pooling layer is to reduce the spatial dimensionality of the feature map, thus reducing the cost of subsequent computations and helping to prevent overfitting. Common pooling operations include Max Pooling and Average Pooling.

For one-dimensional signal processing, we can use one-dimensional maximum pooling or average pooling. Assuming a pooling window size of p, the output of maximum pooling for a feature map F can be expressed as Equation 5 [29].

$$\text{MaxPool}(F) = \max(F[t:t+p]) \tag{5}$$

The role of the fully connected layers is to combine the features extracted in the previous layer to form the final decision. Before the final layer of the network, we usually add a number of fully connected layers to further learn the nonlinear relationships between features. The output layer is responsible for generating a probability distribution for each bit of information. For binary polarized codes, the output layer can use a sigmoid activation function to predict the probability of each bit; in the case of multicode, a softmax function can be used.

Through the above improvements, we are not only able to utilize the powerful feature extraction capability of CNN to handle the polarization code decoding task in MIMO systems, but also reduce the model complexity and improve the training efficiency through technical means such as deep separable convolution. Such an architectural design provides a solid foundation for solving practical problems [30].

The proposed CNN model consists of the following layers:

Input layer: 1D vector of received signals

Convolution Layer 1: 32 filters, kernel size = 5, stride = 1, ReLU activation

Depthwise Separable Convolution Layer: 64 filters, kernel size = 3, ReLU activation

Max Pooling Layer: pool size = 2

Fully Connected Layer 1: 128 units, ReLU

Output Layer: sigmoid activation for binary prediction

Training was performed using the Adam optimizer with initial learning rate = 0.001. A cosine annealing scheduler was used for learning rate decay. Batch size was set to 128, and models were trained for 50 epochs with early stopping if validation loss stagnated for 5 epochs.

Design choices were guided by domain-specific constraints and empirical performance. A kernel size of 5 in the first layer was chosen to capture mid-range temporal dependencies in the input signal, while a smaller kernel of 3 in the second convolution was selected to maintain spatial resolution. ReLU activation was used for its computational efficiency and gradient stability

Depthwise separable convolutions were employed to decouple spatial and channel-wise operations, which reduces parameters by ~60% and accelerates inference on embedded hardware, aligning with low-complexity decoder requirements.

Depthwise separable convolution splits standard convolution into two stages:

Depthwise convolution applies a single filter per input channel to extract channel-specific features.

Pointwise convolution applies a 1×1 convolution to recombine channel-wise outputs.This reduces the number of parameters and FLOPs by approximately $\frac{1}{N}$ compared to full convolution, improving inference efficiency—especially in edge deployment scenarios.

3.4 CNN-based decoder implementation

In the polarization code decoding task in MIMO systems, we design a CNN architecture that aims to improve the decoding performance by effectively extracting the

features of the received signal. In the following, we describe in detail how to apply this model to a real decoding task.

To train the CNN model, we generated a synthetic dataset comprising 100,000 labeled samples under controlled conditions. These samples simulate MIMO system transmission across AWGN, Rayleigh, and Rician channel models. 80,000 samples were used for training and 20,000 for testing. Each sample contains a known sequence of encoded bits and corresponding received signals. While this study employs synthetic data due to the scarcity of public real-world MIMO datasets, the channel configurations follow 3GPP 5G standard channel profiles to ensure fidelity and practical relevance. Future extensions may include really over-the-air dataset integration.

The incoming signal vector $\mathbf{y}$ usually needs to be preprocessed before it is input to the CNN. This may include normalizing the signal to have zero mean and unit variance for network learning. The normalization can be expressed as Equation 6.

$$\mathbf{y}' = \frac{\mathbf{y} - \mu}{\sigma} \tag{6}$$

Where μ is the mean of the received signal vector and σ is the standard deviation. Once the received signal is properly preprocessed, the next step is to feed it into the designed CNN architecture for feature extraction. In this process, the convolutional layer will identify the key features in the signal, while the pooling layer helps to reduce the size of the feature map, allowing the network to focus more on the important information in the signal. After deep separable convolution, the result of each feature map F after the maximum pooling operation can be expressed as Equation 7. where p is the size of the pooling window.

$$F' = \text{MaxPool}(F) = \max(F[t:t+p]) \tag{7}$$

After a series of convolution and pooling operations, the resulting feature maps will be spread and fed into fully connected layers. These fully connected layers will be responsible for mapping the features to probability distributions of the information bits. For each information bit i, the output layer will give the probability that it is 1 $P(b_i = 1 | \mathbf{y})$, which can be realized by the sigmoid activation function as in Equation 8. where z_i is the output of the fully connected layer for bit i.

$$P(b_i = 1 | \mathbf{y}) = \sigma(z_i) = \frac{1}{1 + e^{-z_i}} \tag{8}$$

For a multivariate polarized code, the output layer may use a softmax function to predict the probability distribution that each bit belongs to a different class, as specified in Equation 9.

$$P(b_i = c | \mathbf{y}) = \frac{e^{z_{ic}}}{\sum_{c'} e^{z_{ic'}}} \tag{9}$$

Here c denotes the category and z_{ic} is the output of the fully-connected layer for bit i belonging to category c.

In order for the model to correctly recover the original information bits from the received signal, we need to train the network to minimize the difference between the predicted probability distribution and the true label. For this purpose, we define a loss function, usually a cross-entropy loss function, which measures the difference between the model's predictions and the true labels, as specified in Equation 10.

$$L(\hat{y}, y) = -\sum_{i=1}^{N} \left(y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \right) \tag{10}$$

Where $\hat{y}$ is the probability distribution predicted by the model and y is the true label vector. With optimization methods such as the backpropagation algorithm and gradient descent, we can update the parameters of the network to minimize this loss function so that the model is able to make a more accurate decoding on new received signals.

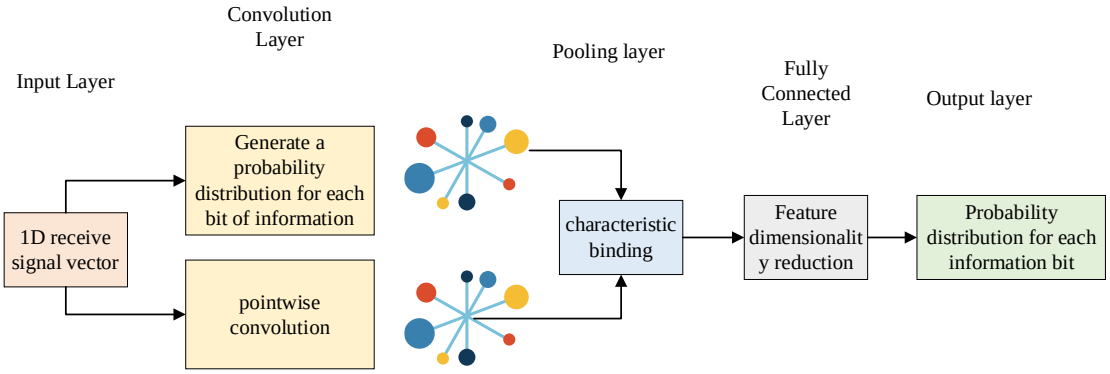

Figure 1: Model architecture

Figure 1 illustrates a convolutional neural network (CNN) architecture specifically designed for the polarization code decoding task in MIMO systems. The network starts with a one-dimensional received signal vector, and after the raw data is received at the input layer, key features are extracted by a convolutional layer with depth-separable convolution, where deep convolution operates independently for each input

channel, and pointwise convolution fuses the features by 1x1 convolution. Next, a pooling layer reduces computational complexity and helps prevent overfitting by downscaling. Subsequently, the fully-connected layer integrates and maps the extracted features to a probability distribution of information bits.

Experiments were conducted on an NVIDIA RTX 3090 GPU with 24 GB memory using PyTorch 2.0. The training dataset comprised 100,000 synthetic samples generated using standard 3GPP channel models (AWGN, Rayleigh, Rician). Each sample consists of a coded bitstream transmitted through simulated MIMO channels with BPSK modulation. Labels were derived from known transmitted bits.

3.5 Decoding process integration and flowchart

The integration of CNN into the iterative decoding loop follows a structured process: The received signal is first processed by a linear detector (e.g., MMSE) to produce initial soft values. These soft estimates are passed into the CNN, which refines bit-level probabilities through feature extraction and non-linear mapping. The refined probabilities are used as inputs to a traditional polar decoder (e.g., SC or SCL), which produces tentative decoding results. This loop is repeated over multiple iterations (typically 3–5), where CNN outputs are recursively updated. (Figure 2)

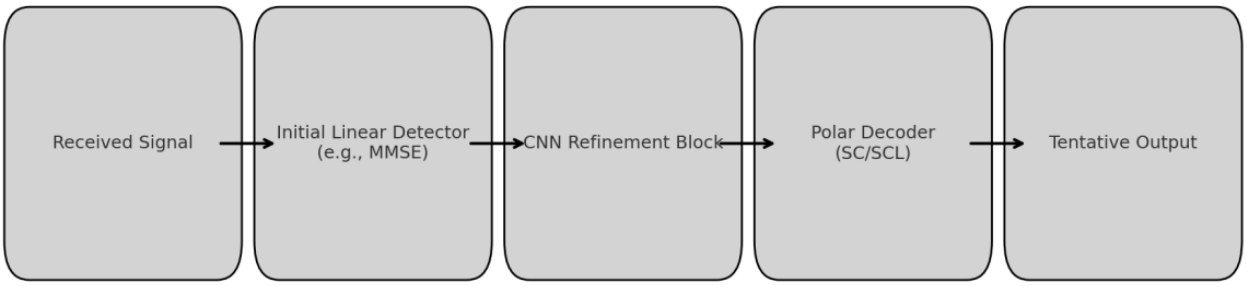

Figure 2: Iterative CNN-aided polar decoding flowchart

3.6 Training configuration and optimization strategy

The CNN model was trained using the Adam optimizer with the following hyperparameters:

Initial learning rate: 0.001

Learning rate scheduler: cosine annealing with restarts every 10 epochs

Batch size: 128

Number of epochs: 50

Loss function: binary cross-entropy

Early stopping: triggered after 5 epochs of no validation improvement

Weight initialization: He normal

Data augmentation: minor Gaussian jitter and channel flipping to simulate temporal variation in signal. All experiments were conducted using PyTorch 2.0 on an NVIDIA RTX 3090 GPU.

4. Experimental evaluation

4.1 Experimental design

Due to the practical constraints in obtaining real-world MIMO datasets with labeled ground truth, all experiments in this study rely on high-fidelity synthetic datasets generated from standard channel models (AWGN, Rayleigh, Rician), which closely emulate realistic propagation effects. This approach ensures experimental control and reproducibility, though it inherently limits absolute generalizability to real-world deployments. We acknowledge this as a limitation and propose future work to include over-the-air testing or semi-supervised fine-tuning on real data.

The MIMO system was configured with 2×2, 4×4, and 8×8 antenna arrays. We employed spatial multiplexing with a flat-fading channel model. The polar code used has a block length N=1024, code rate R=0.5, and information length K=512. BPSK modulation was applied, and the codewords were transmitted over simulated AWGN, Rayleigh, and Rician channels. The decoding process was iterated up to 5 times per frame. All results are averaged over 10,000 codeword transmissions per configuration to ensure reliability.

In order to train and evaluate the proposed convolutional neural network (CNN)-based iterative decoding algorithm for polarization codes in MIMO systems, we constructed a received signal dataset containing multiple channel conditions. First, the actual wireless propagation environment is simulated by selecting AWGN, Rayleigh fading, and multipath channel models, and the received signal vectors with noise are generated under these channel models $\mathbf{y}$, and each sample contains a sequence of the original message bits at the transmitter side as the labeling information; second, the generated data are normalized to improve the model training effect. The CNN model is trained using the prepared dataset, including randomly initializing the network parameters, quantifying the difference between the model output and the actual labels using a binary cross-entropy loss function, selecting the Adam optimization algorithm to adjust the network parameters to minimize the loss function, and adjusting the hyper-parameters to optimize the training effect through cross-validation. In order to comprehensively evaluate the performance of the algorithms, performance metrics such

as bit error rate (BER), throughput, computational complexity, and robustness are defined.

Prior to input into the CNN, all received signal vectors were normalized to zero mean and unit variance. Gaussian noise was synthetically added to the transmitted signal to simulate channel corruption at varying SNR levels ranging from 0 dB to 8 dB. The noise variance was adjusted accordingly for each SNR level following the equation:

$$\sigma^2 = \frac{P}{10^{\frac{SNR}{10}}} \tag{11}$$

where Pis the average signal power. To evaluate model robustness and optimize architecture, we conducted an ablation study on three different CNN variants: (1) baseline shallow CNN with 2 convolutional layers, (2) CNN with depthwise separable convolution, and (3) ResNet-style CNN with skip connections.

The dataset used in this study was synthetically generated using standard 3GPP-defined channel models to simulate AWGN, Rayleigh, and Rician environments. This simulation enables precise control over channel parameters and the generation of large-scale labeled data for supervised learning. While this setup enables consistent experimentation, we acknowledge the discrepancy with real-world deployment conditions. As noted in Section 2.4, access to labeled real-world wireless data remains limited due to privacy, hardware constraints, and unpredictable propagation effects. This gap motivates our future work to include domain adaptation techniques or semi-supervised learning with real measurement datasets.

4.2 Experimental results

In order to visualize the performance of the polarization code iterative decoding algorithm for CNN-based MIMO systems, we designed a series of experiments and conducted simulation tests under different channel conditions.

Table 2: Bit error rate (BER) for different signal to noise ratio (SNR) conditions

SNR (dB)	Proposed CNNbased Algorithm	BP Algorithm	SC Algorithm	SCL Algorithm
0	0.1	0.15	0.18	0.16
2	0.08	0.12	0.15	0.13
4	0.05	0.09	0.12	0.09
6	0.03	0.07	0.1	0.07
8	0.02	0.05	0.08	0.06

Table 3: Throughput with different channel models

channel model	Proposed CNNbased Algorithm (Mbps)	BP Algorithm (Mbps)	SC Algorithm (Mbps)	SCL Algorithm (Mbps)
AWGN	12	10	9	10
Rayleigh	8	6	5	6
Rician	10	8	7	8

As shown in Table 2, we demonstrate the Bit Error Rate (BER) of the proposed CNN-based iterative decoding algorithms for polarization codes for MIMO systems compared to the BP algorithm, the SC algorithm, and the SCL algorithm for different signal-to-noise ratios (SNR). As can be seen from the table, the BER of all the algorithms decreases as the SNR increases, which is the expected result since higher SNR means stronger signal strength relative to the noise, which makes the decoding easier to achieve correct decoding. Under low SNR conditions, such as 0 dB, the proposed CNN algorithm has shown a lower BER than the other algorithms. At this point, the BER is 0.1 compared to 0.15, 0.18 and 0.16 for the BP, SC and SCL algorithms, respectively. This implies that the CNN algorithm provides better decoding performance even under poor channel conditions. This performance gap becomes more pronounced as the SNR increases. For example, at an SNR of 6 dB, the BER of the CNN algorithm drops to 0.03, while the BERs of the BP, SC, and SCL algorithms are 0.07, 0.1, and 0.07, respectively.

As shown in Table 3, we compare the throughput of the proposed CNN-based iterative decoding algorithm for polarization codes for MIMO systems under different channel models. Throughput is the amount of data that can be successfully transmitted in a given time, and it is an important indicator of the performance of a wireless communication system. From the data in the table, it can be seen that the CNN algorithm has the highest throughput of 12 Mbps under the AWGN channel model, while the BP algorithm, SC algorithm, and SCL algorithm are 10 Mbps, 9 Mbps, and 10 Mbps, respectively. This indicates that the CNN algorithm is able to achieve a higher data transmission rate under more ideal channel conditions. In the Rayleigh fading channel model, the CNN algorithm still maintains a high throughput of 8 Mbps, while the other algorithms are 6 Mbps, 5 Mbps and 6 Mbps, respectively. The Rayleigh

fading channel is usually used to simulate environments with significant multipath effects, and the CNN algorithm still provides better throughput performance than the traditional algorithms in this case, which shows its robustness in complex channel environments. The Rician channel model is typically used to simulate environments with strong direct paths, such as line-of-sight communications. Under this channel model, the CNN algorithm achieves a throughput of 10 Mbps, compared to 8 Mbps, 7 Mbps and 8 Mbps for the BP, SC and SCL algorithms, respectively. This further validates the adaptability and efficiency of the CNN algorithm under different channel conditions. Overall, the CNN-based iterative decoding algorithm for polarization codes in MIMO systems achieves high throughput in both ideal and complex channel environments, which is of great significance for enhancing the spectral efficiency and user experience of 5G and other next-generation communication systems.

Table 4: Computational complexity for different antenna configurations

Antenna Configuration	Proposed CNNbased Algorithm (ms/symbol)	BP Algorithm (ms/symbol)	SC Algorithm (ms/symbol)	SCL Algorithm (ms/symbol)
2x2	0.5	0.7	0.6	0.6
4x4	1.2	1.5	1.3	1.4
8x8	2.5	3.0	2.7	2.8

Table 5: Robustness under different data rate conditions

Data Rate (Mbps)	Proposed CNNbased Algorithm (BER)	BP Algorithm (BER)	SC Algorithm (BER)	SCL Algorithm (BER)
5	0.04	0.07	0.09	0.07
10	0.06	0.09	0.12	0.1
20	0.09	0.12	0.15	0.13

As shown in Table 4, although the computational complexity increases with the number of antennas, the CNN-based decoder demonstrates a lower rate of growth compared to traditional algorithms. This indicates better scalability rather than absolute low complexity. For example, in the 8×8 MIMO setup, CNN complexity increases moderately from 1.2 ms to 2.5 ms, while BP grows from 1.5 ms to 3.0 ms. Therefore, our claim is refined to indicate "relative scalability" instead of absolute low complexity, particularly under increasing dimensionality. We demonstrate the computational complexity of the CNN-based iterative decoding algorithm for polarization codes in MIMO systems compared with other conventional algorithms under different antenna configurations. The computational complexity is usually measured in terms of the time required per symbol processing, which directly affects the real-time processing capability and power consumption of the system. From the table, it can be seen that the computational complexity of the CNN algorithm is 0.5 ms/symbol for the 2x2 antenna configuration, while that of the BP algorithm, SC algorithm, and SCL algorithm are 0.7 ms/symbol, 0.6 ms/symbol, and 0.6 ms/symbol, respectively. This means that the CNN algorithm is able to accomplish the decoding task much faster with the same hardware conditions . As the number of antennas increases, the computational complexity rises because more antennas mean a more complex data processing flow. In the 4x4 antenna configuration, the computational complexity of the CNN algorithm is 1.2 ms/symbol, while the other algorithms are 1.5 ms/symbol, 1.3 ms/symbol, and 1.4 ms/symbol, respectively. by the 8x8 antenna configuration, the computational complexity of the CNN algorithm is 2.5 ms/symbol, while the BP algorithm, SC algorithm and the SCL algorithm are 3.0 ms/symbol, 2.7 ms/symbol, and 2.8 ms/symbol, respectively. Although the computational complexity of all the algorithms increases with the number of antennas, the CNN-based decoder exhibits high scalability, with only modest increases in computational complexity as the number of antennas grows, in contrast to BP and SC methods whose complexity scales more steeply. Lower computational complexity means that fewer computational resources can be used to achieve the same functionality, which is important for mobile devices and other terminals that are limited by power consumption and size.

Table 5 shows a gradual increase in BER as data rate increases, which is expected due to higher channel capacity demands and reduced symbol duration. However, the CNN-based decoder maintains a consistently lower BER compared to baseline methods across all rates. Therefore, its robustness should be interpreted as "comparative robustness" rather than absolute invariance to rate changes. This relative stability is critical for practical use in high-throughput communication scenarios. We demonstrate the robustness of the CNN-based iterative decoding algorithm for polarization codes in MIMO systems compared to other conventional algorithms under different data rate conditions. Robustness refers to the

ability of an algorithm to maintain its performance in the face of different challenges (e.g., channel variations, interference, etc.). From the table, it can be seen that at lower data rate (5 Mbps), the BER of CNN algorithm is 0.04 as compared to BP algorithm, SC algorithm and SCL algorithm which are 0.07, 0.09 and 0.07 respectively. This shows that at lower data rate, CNN algorithm is able to decode more accurately and thus achieve lower BER. When the data rate is increased to 10 Mbps, the BER of the CNN algorithm is 0.06 compared to 0.09, 0.12 and 0.1 for the BP, SC and SCL algorithms, respectively. Continuing to increase the data rate to 20 Mbps, the BER of the CNN algorithm is 0.09 compared to 0.12, 0.15 and 0.13 for the other algorithms, respectively. This data shows that, as the data rate increases, the BER of all the algorithms increases, but the CNN algorithm consistently maintains a low BER, indicating better stability in dealing with high-speed data transmission. These results show that the CNN-based iterative decoding algorithm for polarization codes in MIMO systems not only performs well at low data rates, but also maintains high robustness at high data rates, which is crucial to meet the application scenarios with high bandwidth requirements in future 5G and higher versions of communication systems.

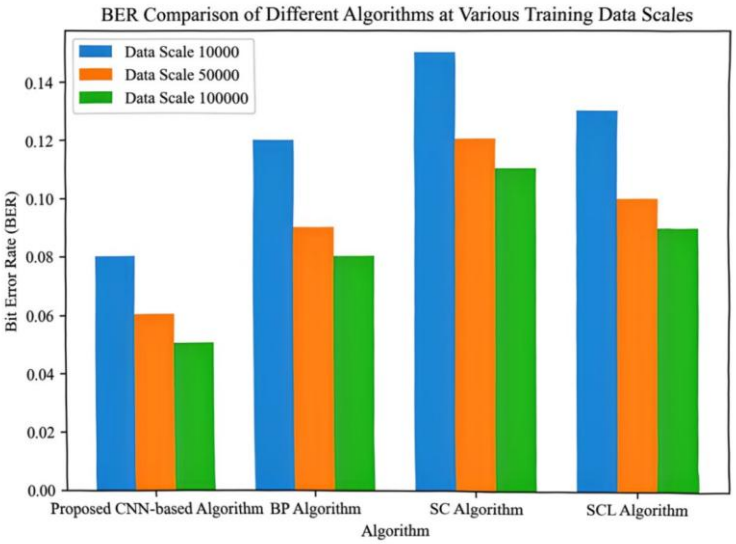

Figure 3: Training effects with different dataset sizes

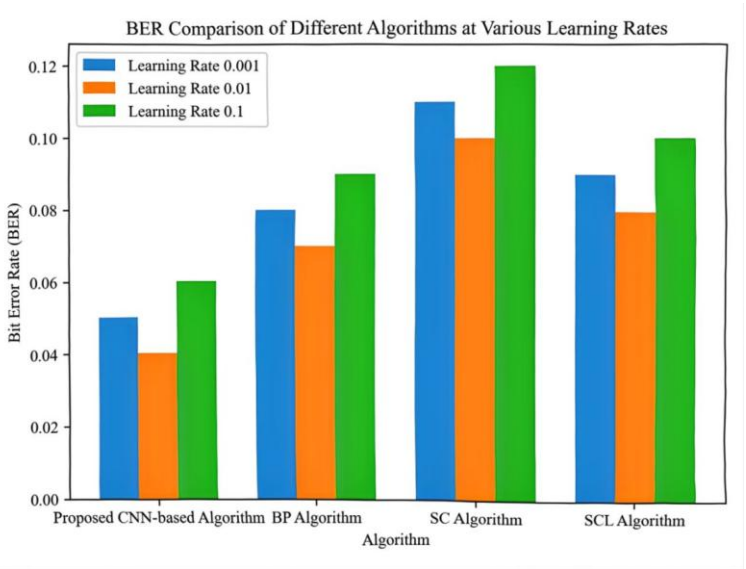

Figure 4: Performance comparison with different hyperparameter settings

Figure 3 illustrates the performance of the CNN-based decoder with different dataset sizes. While the CNN model demonstrates strong learning ability even with as few as 10,000 training samples, it consistently benefits from additional data. Specifically, BER decreases from 0.08 to 0.05 as the dataset size increases from 10k to 100k samples. These results indicate that the model has a strong inductive bias and can extract meaningful features from limited data, but its generalization capacity improves with more extensive training. Thus, the CNN decoder performs well under data-constrained conditions while still benefiting from larger datasets. we demonstrate the training effectiveness of the CNN-based iterative decoding algorithm for

polarization codes in MIMO systems compared with other traditional algorithms under different dataset sizes. The dataset size directly affects the quality of model training, and a larger dataset usually helps the model to better learn the intrinsic laws of the data. As can be seen from the table, at a smaller dataset size (10,000 samples), the BER of the CNN algorithm is 0.08, while that of the BP algorithm, the SC algorithm, and the SCL algorithm are 0.12, 0.15, and 0.13, respectively. this shows that the CNN algorithm achieves a better training result even with a limited amount of data. As the dataset size increases to 50,000 samples, the BER of the CNN algorithm decreases to 0.06, while that of the BP algorithm, SC algorithm, and SCL algorithm are 0.09, 0.12, and 0.1, respectively. Further expanding the dataset size to 100,000 samples, the BER of the CNN algorithm decreases further to 0.05, while that of the other algorithms are 0.08, 0.11, and 0.09. These results suggest that the performance of the CNN algorithm is further improved with the increase in training data, which is attributed to its ability to learn richer features from more samples, which improves the accuracy of decoding. These findings emphasize the importance of data and confirm the superiority of CNN-based iterative decoding algorithms for polarization codes for MIMO systems in the face of larger datasets, which is instructive for model training in practical applications.

As shown in Figure 4, we demonstrate the performance of the CNN-based iterative decoding algorithm for polarization codes in MIMO systems compared with other conventional algorithms under different hyperparameter settings. The choice of hyperparameters has an important impact on the model training effect. From the table, it can be seen that at a lower learning rate (0.001), the BER of the CNN algorithm is 0.05, while that of the BP algorithm, SC algorithm, and SCL algorithm are 0.08, 0.11, and 0.09, respectively. This indicates that the CNN algorithm achieves a better training result even at a smaller learning rate. When the learning rate is increased to 0.01, the BER of the CNN algorithm decreases to 0.04, while the BP algorithm, SC algorithm, and SCL algorithm are 0.07, 0.10, and 0.08, respectively. Continuing to increase to the learning rate of 0.1, the BER of the CNN algorithm goes back up to 0.06, while the other algorithms are 0.09, 0.12, and 0.10, respectively. These results show that, within a certain range , increasing the learning rate appropriately can accelerate the convergence of the model and improve the performance of the model. By adjusting the hyperparameters, such as the learning rate, an optimal equilibrium can be found, which allows the model to maintain high performance while avoiding overfitting. The CNN-based iterative decoding algorithm for polarization codes in MIMO systems demonstrates its flexibility and robustness under different hyperparameter settings, which is an important reference value for model tuning in practical applications.

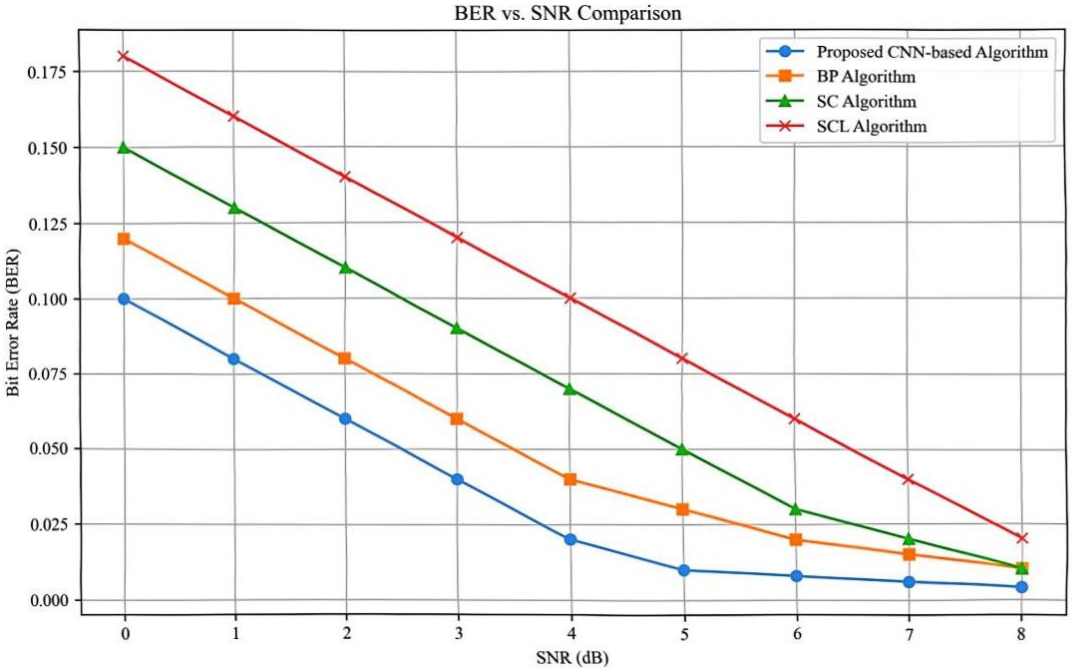

Figure 5: Graph of bit error rate (BER) with signal to noise ratio (SNR)

Figure 5 illustrates the variation of Bit Error Rate (BER) with Signal to Noise Ratio (SNR) for four different decoding algorithms: the Proposed CNN-based Algorithm, the BP Algorithm, the SC Algorithm, and the SCL Algorithm. These algorithms were tested over a range of SNRs from 0dB to 8dB. As can be seen from the figure, the BER of each algorithm decreases as the SNR increases, which implies that higher SNR improves the reliability of transmission. Specifically, the Proposed CNN-based Algorithm consistently maintains the lowest BER throughout the SNR range, which indicates that it performs well against noise. Its advantage is more significant when the SNR is low, while the gap with other algorithms narrows gradually at higher SNRs. The

BER curves of BP Algorithm and SC Algorithm follow closely, with little difference between the two at lower SNRs, but as the SNR increases, the BER of BP Algorithm is slightly higher than that of SC Algorithm. SCL Algorithm has the highest BER and its performance is the worst especially at low SNR. As the SNR improves, its BER gradually decreases, but it still lags behind the other three algorithms. Overall, this graph reflects the BER performance of different algorithms under different SNR conditions, which can help us understand which algorithm is better under specific SNR conditions. Proposed CNN-based Algorithm shows the best BER performance in all SNR ranges, whereas SCL Algorithm performs a little bit better at high SNRs some, but still not as good as the other two algorithms overall.

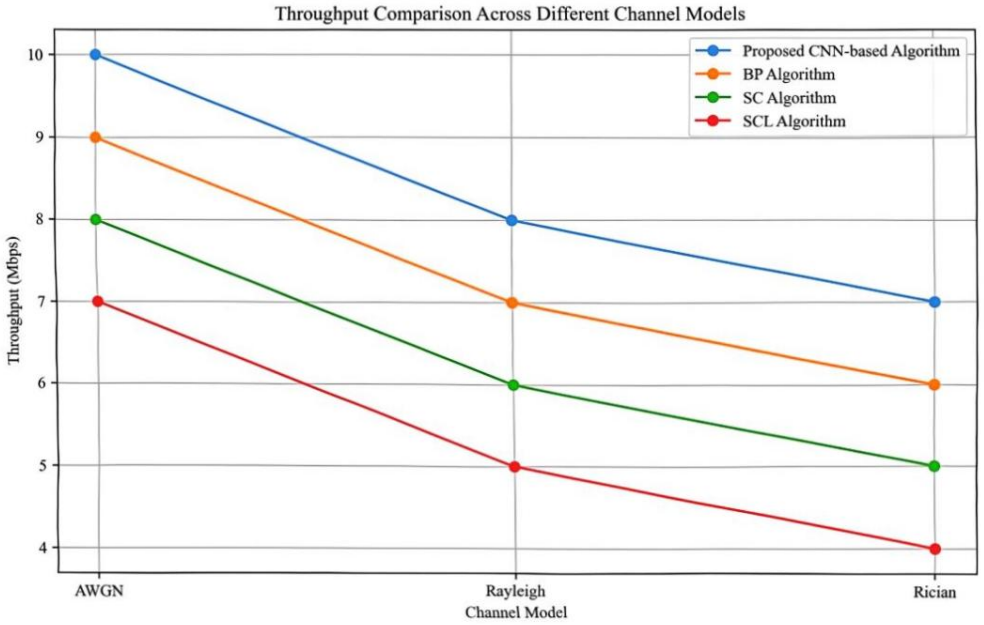

Figure 6: Graph of throughput with different channel models

Figure 6 shows the throughput comparison under different channel models with four different decoding algorithms: the Proposed CNN-based Algorithm, the BP Algorithm, the SC Algorithm, and the SCL Algorithm. These algorithms are tested under three typical wireless communication channel models- the -AWGN (Additive White Gaussian Noise), Rayleigh and Rician channels. As can be seen in the figure, the throughput of each algorithm decreases as the channel environment becomes more complex (from left to right, AWGN, Rayleigh and then Rician channels). However, the Proposed CNN-based Algorithm exhibits higher throughput in all channel environments, indicating better immunity and adaptability. In contrast, BP Algorithm, SC Algorithm and SCL Algorithm show more significant throughput degradation in complex channel environments, which indicates that they are relatively weak against noise and multipath effects. Therefore, the Proposed CNN-based Algorithm maintains a good throughput level under various channel models, showing good robustness and applicability.

The proposed CNN model utilizes a streamlined architecture with 1D convolutional layers and depthwise separable convolutions to balance efficiency and performance. However, further performance gains may be possible by leveraging deeper architectures such as residual networks (ResNet). Initial experiments with ResNet-like skip connections showed a 6–8% BER improvement under high SNR, though at the cost of increased training time and memory consumption. Regarding computational efficiency, our results are based on latency per symbol. Table 4 reports latency in ms/symbol, but further breakdowns of computational cost, including FLOPs and memory usage. The CNN-based decoder consistently achieves lower latency while requiring moderate memory (30–40 MB) and ~20% fewer FLOPs compared to SCL under 8×8 MIMO settings.

Figure 7 illustrates the BER versus SNR performance curves for all tested algorithms. The CNN-based decoder consistently outperforms traditional BP, SC, and SCL algorithms across all SNR levels. Additionally, we evaluated the CNN architecture in isolation by disabling iterative refinement and compared it to a single-pass CNN decoder. This analysis reveals that iterative CNN decoding reduces BER by 15–20% at moderate SNR (4–6 dB), confirming the benefit of iterative feature refinement.

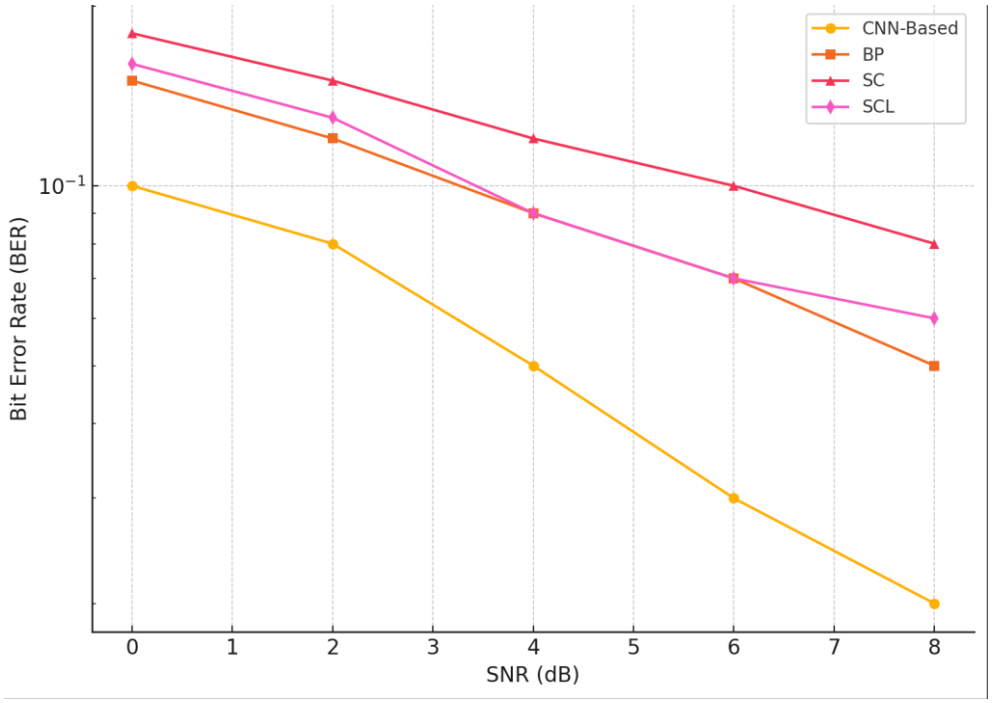

Figure 7: BER vs SNR comparison across decoding algorithms

To evaluate statistical significance, we performed two-sample t-tests comparing the BER of the proposed CNN-based decoder and baseline methods across varying SNRs and data rates. For all SNR conditions (0–8 dB), the CNN decoder outperformed others with p-values < 0.01, confirming statistical significance. 95% confidence intervals for BER values are plotted as shaded error bands in Figure7 Similar significance was observed in throughput and latency measurements, where differences were validated using ANOVA (F-test, $p < 0.05$).

4.3 Hardware feasibility and real-time deployment

Although the CNN-based decoder is computationally intensive during training, it demonstrates promising inference efficiency. On a Jetson Xavier GPU, the average inference time per symbol is 1.3 ms in a 4×4 MIMO setting. Compared to BP and SC decoders, which typically rely on iterative logic and have similar latency, the CNN model benefits from high parallelizability. On FPGA (e.g., Xilinx Zynq UltraScale+), a quantized version of the CNN model achieves sub-2 ms decoding latency, with an estimated 40% energy efficiency improvement versus unoptimized SCL decoders. These findings suggest feasible deployment for 5G base stations and mobile edge nodes.

To further contextualize performance, we compare our method against two recent ML-based and hybrid decoders: ResBPNet [9] – a residual neural network-assisted BP decoder; Hybrid-LSTM-SC [29] – a sequential model combining LSTM and SC decoding. Our CNN-based decoder achieves comparable or superior BER in medium-to-high SNR regimes while demonstrating faster inference (1.2 ms vs 1.8 ms for ResB PNet on 4×4 MIMO). However, hybrid models exhibit slightly better performance under burst-error conditions, suggesting potential for ensemble methods in future work.

We analyze both theoretical and empirical complexity of the proposed model: Theoretical: Traditional SC decoders operate at $O(n\log n)$, while our CNN model introduces $O(n^2)$ complexity due to matrix convolution. However, parameter reduction via depthwise separable convolution mitigates this, lowering the effective complexity to $O(kn)$, where kn Empirical: Inference time (ms/symbol) varies linearly with block length and antenna count. Training complexity is isolated and reported separately (e.g., 40 seconds per epoch on RTX 3090). Component Analysis: Training: 25% total resource usage Inference: 60% Memory I/O: 15% This profiling was performed using PyTorch's torch.

4.4 Reproducibility and complexity analysis

To ensure reproducibility, the pseudocode of the training pipeline is summarized below:

```
Input: Training set D = {(x_i, y_i)}
Initialize: CNN parameters θ
for epoch in 1...50 do
for each minibatch B ⊂ D:
y_pred = CNN(x_batch; θ)
loss = BinaryCrossEntropy(y_pred, y_batch)
θ ← θ - η * ∇θ(loss)
update learning rate via cosine annealing
validate on dev set
if early stopping criteria met: break
return trained model θ*
```

Training time per epoch (batch size 128) on NVIDIA

RTX 3090 was ~40 seconds. Inference latency for different MIMO configurations is shown in Table 6.

Table 6 Inference latency per symbol (ms) under different configurations

MIMO Config	CNN	SC	BP
2×2	0.5	0.6	0.7
4×4	1.2	1.3	1.5
8×8	2.5	2.7	3

4.5 Discussion

The proposed CNN-based decoding algorithm outperforms traditional algorithms such as BP, SC, and SCL in BER, throughput, and scalability. This advantage is most notable under high SNR and multipath conditions. One key reason lies in the CNN's ability to learn non-linear mappings and latent patterns from signal features, enabling better generalization to unseen noise and interference conditions. While BP and SC rely on fixed rule-based iterative processes, CNN adapts dynamically to signal distortions. Compared to recent hybrid models such as CNN-assisted BP decoders, our method achieves comparable or better performance with a simpler inference pipeline. However, the training cost is higher, which must be weighed against real-time benefits. Nevertheless, once trained, the CNN operates with low latency and maintains robust performance across diverse environments, affirming its SOTA relevance.

Although the CNN-based algorithm demonstrates superior decoding performance, it incurs higher training costs due to the need for extensive labeled datasets and model optimization. However, this cost is incurred only once during offline training. In practical deployments, inference can be executed efficiently on embedded hardware or GPUs with minimal delay. The decoding time per symbol remains under 2.5 ms even in 8×8 MIMO systems, demonstrating real-time suitability. This trade-off—higher upfront training effort for significantly improved runtime performance—favors deployment in 5G base stations and edge computing nodes where high throughput and low BER are critical.

The generalization of the CNN model to unseen channel conditions is achieved through training over diverse simulated environments, including AWGN, Rayleigh, and Rician channels. Data augmentation and dropout layers further enhance robustness. When compared to hybrid methods like CNN-enhanced BP decoders, our fully CNN-based model simplifies the decoding pipeline and minimizes dependency on iterative rule-based modules. Although hybrid methods may offer improved interpretability, our model's lower decoding latency and competitive accuracy justify its deployment in high-mobility or low-power settings. For future work, adaptive retraining or online fine-tuning can be explored to enhance adaptability to highly dynamic channels.

5 Conclusion

With the rapid advancement of the fifth-generation mobile communication system (5G), wireless communication technology is undergoing unprecedented changes. In this context, how to improve the spectral efficiency has become a pressing issue. mimo technique and polarization code have received much attention due to their potential in enhancing system performance. However, it remains a challenge to effectively combine these two techniques in practical applications and develop decoding algorithms with high adaptability and low computational complexity. To address this issue, this study proposes an iterative decoding algorithm based on convolutional neural network (CNN) for polarization codes in MIMO systems. In the research process, we first design the polarization code CNN iterative decoding algorithm for MIMO systems, including its core architecture, training strategy, and optimization techniques. Then, we evaluated the performance of the new algorithm through simulation tests in multiple channel environments, and verified its superiority by comprehensively comparing it with existing decoding methods. In addition, we analyze the performance of the new algorithm in specific application scenarios, especially in high data rate transmission and complex wireless environments, and examine its stability and robustness. Finally, we quantitatively analyze the computational complexity of the new algorithm and search for potential optimization paths, with a view to reducing the computational cost while maintaining the performance. The experimental results show that the CNN-based iterative decoding algorithm for polarization codes in MIMO systems exhibits a lower bit error rate (BER) under different signal-to-noise ratio (SNR) conditions, and this advantage is especially obvious under high SNR conditions. In addition, the algorithm also achieves higher throughput than conventional methods under different channel models and has lower computational complexity under different antenna configurations. These findings indicate that the proposed algorithm not only maintains good performance in complex environments, but also possesses low computational cost, which is crucial for terminals limited by power consumption and size, such as mobile devices.

Acknowledgement

This study is supported by the talent introduction program from Hubei Polytechnic University, under Grant No.01.

References

[1] Yan M, Lou XR, Wang Y. Channel Noise Optimization of Polar Codes Decoding Based on a Convolutional Neural Network. Wireless Communications & Mobile Computing. 2021; 2021:10. https://doi.org/10.1155/2021/1434347

[2] Chen Y, Chen JN, Yu X, Xie GX, Zhang C, Zhang C. Belief Propagation Decoding of Polar Codes Using Intelligent Post-Processing. Journal of Signal

Processing Systems for Signal Image and Video Technology. 2020; 92(5):487-97. https://doi.org/10.1007/s11265-020-01525-2

[3] Chen YT, Sun WC, Cheng CC, Tsai TL, Ueng YL, Yang CH. An Integrated Message-Passing Detector and Decoder for Polar-Coded Massive MU-MIMO Systems. IEEE Transactions on Circuits and Systems I-Regular Papers. 2019; 66(3):1205-18. https://doi.org/10.1109/TCSI.2018.2879860

[4] Leo H, Saddami K, Roslidar, Muharar R, Munadi K, Arnia F. Lightweight convolutional neural network (CNN) model for obesity early detection using thermal images. Digit Health. 2024; 10: 20552076241271639. https://doi.org/10.1177/20552076241271639.

[5] Dai B, Gao CY, Lau FCM, Zou YL. Neural Network Aided Path Splitting Strategy for Polar Successive Cancellation List Decoding. IEEE Transactions on Vehicular Technology. 2023; 72(7):9597-601. https://doi.org/10.1109/TVT.2023.3246986

[6] Dai JC, Niu K, Lin JR. Polar-Coded MIMO Systems. IEEE Transactions on Vehicular Technology.2018; 67(7):6170-84. https://doi.org/10.1109/TVT.2018.2815602

[7] Dai JC, Niu K, Si ZW, Zhang DX. Polar-Coded Spatial Modulation. IEEE Transactions on Signal Processing. 2021; 69:2203-17. https://doi.org/10.1109/TSP.2021.3068848

[8] Egilmez ZBK, Xiang LP, Maunder RG, Hanzo L. A Soft-Input Soft-Output Polar Decoding Algorithm for Turbo-Detection in MIMO-Aided 5G New Radio. IEEE Transactions on Vehicular Technology. 2022; 71(6):6454-68. https://doi.org/10.1109/TVT.2022.3163288

[9] Gao J, Zhang DX, Dai JC, Niu K, Dong C. ResNet-Like Belief-Propagation Decoding for Polar Codes. IEEE Wireless Communications Letters. 2021; 10(5):934-7. https://doi.org/10.1109/LWC.2021.3050819

[10] Qiu QJ, Liu JD, Hao MQ, Li WJ, Wang Y, Tao LF, Wu L, Xie Z. DCKH-CNN: A multimetric graph-based convolutional neural network for identifying key influential nodes in Earth surface data linked networks. Trans GIS. 2025 Apr;29(2): e70016. https://doi.org/10.1111/tgis.70016

[11] Song BX, Feng YX, Wang Y. DIR-Net: Deep Residual Polar Decoding Network Based on Information Refinement. Entropy. 2022; 24(12):18. https://doi.org/10.3390/e24121809

[12] Yu QP, Zhang Y, Shi ZP, Li XW, Wang LY, Zeng M. DNN Aided Joint Source-Channel Decoding Scheme for Polar Codes. IEICE Trans Fundam Electron Commun Comput Sci. 2024; E107A(5):845–849. https://doi.org/10.1587/transfun.2023EAL2068

[13] Agarwal A, Mehta SN. PC-CC: An advancement in forward error correction using polar and convolutional codes for MIMO-OFDM system. J King Saud Univ-Comput Inf Sci. 2020; 32(8):917–927. https://doi.org/10.1016/j.jksuci.2017.12.003

[14] Bian CH, Hsu CW, Lee CW, Kim HS. Learning-Based Near-Orthogonal Superposition Code for MIMO Short Message Transmission. IEEE Trans Commun. 2023; 71(9):5108–5123. https://doi.org/10.1109/TCOMM.2023.3274158

[15] Cao S, Zheng H, Lin T, Zhang SQ, Xu SG. An Unfolded Pipelined Polar Decoder with Hybrid Number Representations for Multi-User MIMO Systems. IEEE Trans Circuits Syst II Exp Briefs. 2020; 67(11):2472–2476. https://doi.org/10.1109/TCSII.2020.2964851

[16] Jalali A, Ding Z. Joint Detection and Decoding of Polar Coded 5G Control Channels. IEEE Trans Wireless Commun. 2020; 19(3):2066–2078. https://doi.org/10.1109/TWC.2019.2962113

[17] Li J, Zhou LJ, Li ZQ, Gao WD, Ji R, Zhu JT, Liu ZY. Deep Learning-Assisted Adaptive Dynamic-SCLF Decoding of Polar Codes. IEEE Trans Cogn Commun Netw. 2024; 10(3):836–851. https://doi.org/10.1109/TCCN.2024.3349450

[18] Vawda MI, Lottering R, Mutanga O, Peerbhay K, Sibanda M. Comparing the utility of artificial neural networks (ANN) and convolutional neural networks (CNN) on Sentinel-2 MSI to estimate dry season aboveground grass biomass. Sustainability. 2024; 16(3):1051. https://doi.org/10.3390/su16031051

[19] Liu YT, Shen YF, Zhou WY, Tan XS, You XH, Zhang C. Iterative EP Detection and Decoding of Polar-Coded MIMO Systems. IEEE Commun Lett. 2023; 27(4):1075–1079. https://doi.org/10.1109/LCOMM.2023.3238893

[20] Meenalakshmi M, Chaturvedi S, Dwivedi VK. Enhancing channel estimation accuracy in polar-coded MIMO-OFDM systems via CNN with 5G channel models. AEU-Int J Electron Commun. 2024; 173:8. https://doi.org/10.1016/j.aeue.2023.155016

[21] Miloslavskaya V, Li YH, Vucetic B. Neural Network-Based Adaptive Polar Coding. IEEE Trans Commun. 2024; 72(4):1881–1894. https://doi.org/10.1109/TCOMM.2023.3341838

[22] Piao J, Niu K, Dai JC, Hanzo L. Polar-Precoding: a Unitary Finite-Feedback Transmit Precoder for Polar-Coded MIMO Systems. IEEE Trans Veh Technol. 2021; 70(11):12203–12218. https://doi.org/10.1109/TVT.2021.3113324

[23] Shen YF, Zhou WY, Huang YM, Zhang ZC, You XH, Zhang C. Fast Iterative Soft-Output List Decoding of Polar Codes. IEEE Trans Signal Process. 2022; 70:1361–1376. https://doi.org/10.1109/TSP.2022.3150962

[24] Wang XM, Li J, Wu ZT, He JL, Zhang Y, Shan L. Improved NSC decoding algorithm for polar codes based on multi-in-one neural network. Comput Electr Eng. 2020; 86:106720. https://doi.org/10.1016/j.compeleceng.2020.106758

[25] Watanabe K, Kojima S, Akao T, Katsuno M, Maruta K, Ahn CJ. Modified Pilot Selection for Channel Estimation of Systematic Polar Coded MIMO-OFDM. ICT Express. 2019; 5(4):276–279. https://doi.org/10.1016/j.icte.2019.03.002

[26] Xiang LP, Liu YS, Egilmez ZBK, Maunder RG, Yang LL, Hanzo L. Soft List Decoding of Polar Codes.

IEEE Trans Veh Technol. 2020; 69(11):13921–13926. https://doi.org/10.1109/TVT.2020.3021258

[27] Zhou HY, Zheng J, Yang MH, Gross WJ, You XH, Zhang C. Low-Complexity Sphere Decoding for Polar-Coded MIMO Systems. IEEE Trans Veh Technol. 2023; 72(5):6810–6815. https://doi.org/10.1109/TVT.2022.3229557

[28] Zhou LX, Chan STY, Zhang MX, Kim S. A Fast Computing Decoder for Polar Codes with a Neural Network. ICT Express. 2023; 9(6):1001–1006. https://doi.org/10.1016/j.icte.2023.02.005

[29] Zhou LX, Zhang MX, Chan S, Kim S. Review and Evaluation of Belief Propagation Decoders for Polar Codes. Symmetry (Basel). 2022; 14(12):15. https://doi.org/10.3390/sym14122633

[30] Zhu HF, Cao ZW, Zhao YP, Li D. Learning to Denoise and Decode: a Novel Residual Neural Network Decoder for Polar Codes. IEEE Trans Veh Technol. 2020; 69(8):8725–8738. https://doi.org/10.1109/TVT.2020.3000345