electronics

MDPI

Article

DMCNet-Pro: A Model-Driven Multi-Pilot Convolution Neural Network for MIMO-OFDM Receivers

Pengyuan Li [1,2], Tianlin Zhu [2], Yutong Xin [3,4], Gang Yuan [2], Xiong Yu [2], Zejian Lu [4,*], Zili Liu [4] and Qing Yan [3]

[1] Polytechnic Institute, Zhejiang University, Hangzhou 310058, China; pengyuan_li@163.com
[2] Beijing Institute of Tracking and Telecommunications Technology, Beijing 100094, China; 13911966061@139.com (T.Z.); asqasq2022@163.com (G.Y.); 19800232763@163.com (X.Y.)
[3] School of Artificial Intelligence, Beijing University of Posts and Telecommunications, Beijing 100876, China; xinyutong@bupt.edu.cn (Y.X.); qingyan@bupt.edu.cn (Q.Y.)
[4] China Academic of Electronics and Information Technology, Beijing 100041, China; liuzili1@cetc.com.cn
* Correspondence: lzj_bupt@163.com

Abstract: Nowadays, wireless communication technology is evolving towards high data rates, a low latency, and a high throughput to meet increasingly complex business demands. Key technologies in this direction include multiple-input multiple-output (MIMO) and orthogonal frequency division multiplexing (OFDM). This research is based on our previous work DMCNet. In this article, we focus on studying the deep learning (DL) application of neural networks to solve the reception of single-antenna OFDM signals. Specifically, in multi-antenna scenarios, the channel model is more complex compared to single-antenna cases. By leveraging the characteristics of DL, such as automatic learning of parameters using deep neural networks, we treat the reception process of MIMO-OFDM signals as a black box and utilize neural networks to accomplish the signal reception task. Moreover, we propose a data-driven multi-pilot convolution neural network for MIMO-OFDM receivers (DMCNet). By incorporating complex convolution and complex fully connected structures, we design a receiver network to recover the transmitted signals from the received signals. We validate the accuracy and robustness of DMCNet under different channel conditions, comparing the bit error rates with different schemes. Additionally, we discuss the factors influencing various channel effects. At the same time, we also propose a model-driven scheme, DMCNet-pro, which has a higher accuracy and fewer parameters in some scenarios. The experimental results demonstrate that the DL-based reception scheme exhibits promising feasibility in terms of accuracy and interference resistance when compared to traditional approaches.

Keywords: MIMO; OFDM; multiple pilots; neural networks; complex convolution

Citation: Li, P.; Zhu, T.; Xin, Y.; Yuan, G.; Yu, X.; Lu, Z.; Liu, Z.; Yan, Q. DMCNet-Pro: A Model-Driven Multi-Pilot Convolution Neural Network for MIMO-OFDM Receivers. *Electronics* **2024**, *13*, 330. https://doi.org/10.3390/electronics13020330

Academic Editor: David A. Sánchez-Hernández

Received: 20 November 2023
Revised: 7 January 2024
Accepted: 8 January 2024
Published: 12 January 2024

1. Introduction

Orthogonal frequency division multiplexing (OFDM) and multiple-input multiple-output (MIMO) are key technologies used in current 5G and even future 6G wireless communications [1]. OFDM enables parallel transmission of signals by using overlapping orthogonal sub-channels to transmit data. The data streams are modulated onto various sub-channels. Its main advantages are the high spectrum efficiency and the ability to cope with adverse channel conditions without complex equalization filters. MIMO technology utilizes multiple antenna arrays at both the transmitting and receiving ends of the system. At the transmitting end, the serial data streams are parallelized through space-time transformations and are sent out through different transmitting antennas. At the receiving end, multiple antennas are used to receive the signals, and the received signal is processed through decoding modules. MIMO technology fully utilizes spatial and temporal diversity and multiplexing gains, significantly increasing the channel capacity and spectrum efficiency without increasing the occupied bandwidth [2].

However, the communication environment is becoming increasingly complex, and the transmission of signals is subject to various unfavorable factors, making it increasingly difficult to accurately receive signals. Traditional multi-antenna, multi-carrier signal receiver schemes often require channel estimation, symbol detection, and other steps. In multi-antenna scenarios, researchers have proposed optimal signal detection algorithms, also known as Maximum Likelihood (ML) detection algorithms [3]. These methods exhaustively search through all possible transmitted signals to find the optimal solution, providing the best performance in terms of solution accuracy, but they come with a high computational complexity. Commonly used linear signal detection algorithms include the Zero Forcing (ZF) detection algorithm [4] and the Minimum Mean Square Error (MMSE) detection algorithm [5]. The ZF algorithm is a linear precoding technique used in wireless communication. It aims to eliminate interference by designing a matrix that cancels out signals from unwanted users, improving system performance, especially in multi-user scenarios. The ZF algorithm has a simple solution but suffers from significant performance degradation when the noise power is high. The MMSE detection algorithm takes noise into account and performs better than the ZF algorithm at lower signal-to-noise ratios. However, in increasingly complex communication environments, these traditional methods exhibit poor robustness, susceptibility to interference, and limited accuracy. Therefore, in the field of wireless signal reception, there is a need for new technologies to complement and optimize traditional methods.

Fortunately, with the continuous development of computer hardware, the training capability of deep neural networks has become stronger. In recent years, the development of deep learning theory has been rapid, and related research has been applied to the field of wireless communications. Compared to traditional approaches, deep learning algorithms learn knowledge automatically from existing data and experimental results, surpassing the limitations of traditional mathematical approaches and exhibiting superior performance and adaptability in certain scenarios. Reference [6] first proposed the design of a single-antenna OFDM receiver using a five-layer fully connected structure. The fully connected network does not require explicit channel estimation and signal detection, treating the entire receiver as a black box. This breaks the modular structure of traditional approaches, and the experimental results demonstrate that the neural-network-based approach exhibits stronger robustness. Model-driven approaches combine deep learning with expert knowledge in the field of wireless communications. The receiver is divided into two sub-networks: channel estimation (CE) and signal detection (SD). A typical representative is ComNet [7] proposed by the team from Southeast University. Subsequently, SwitchNet [8], also based on a model-driven approach, was proposed to address performance issues in ComNet. Additionally, DCCN [8], based on a complex convolutional structure, includes a receiver, but its code is not open-source, making it difficult to reproduce the simulation links and training data. On the other hand, some individual achievements in researching deep learning MIMO receivers have been published in recent years. These studies focus on the design of single-carrier MIMO receivers. In 2018, Shi Jin et al. proposed OAMP-Net [9], which uses a multi-layer iterative neural network approach to design MIMO receivers.

In current practical applications, MIMO and OFDM are often combined, known as MIMO-OFDM. Compared to pure MIMO or OFDM techniques, the signal transmission process of MIMO-OFDM is more complex. Factors such as the number of antennas and the number of subcarriers and the specific characteristics of OFDM, such as cyclic prefix and peak clipping, increase the complexity of the communication process. Therefore, this paper aims to expand the aforementioned research by designing a small-scale MIMO-OFDM receiver that incorporates the principles of deep learning. The link of the DL-based MIMO-OFDM receiver is shown in Figure 1. Specifically, in this research, we start by studying the reception scheme for OFDM signals and extending it to MIMO-OFDM receiver design. Deep learning methods are employed to conduct research in this field, treating the entire receiver as a black box and utilizing deep neural networks to perform tasks such as channel estimation and signal detection. Furthermore, a deep-learning-based receiver

called DMCNet is proposed [10], which combines complex convolution and complex, fully connected structures. This receiver network aims to recover the transmitted signals from the received signals and validate their accuracy and robustness under different channel conditions. Finally, based on the data-driven model DMCNet, a model-driven approach is proposed. Experimental results demonstrate that the model-driven approach, when combined with relevant algorithms from traditional communications, achieves a higher accuracy and requires fewer parameters in certain scenarios.

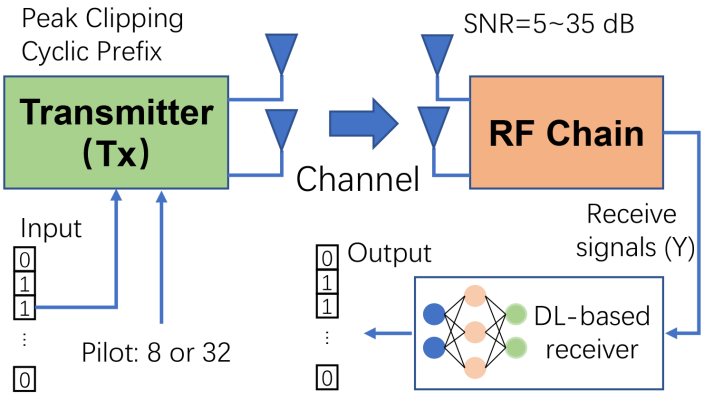

Figure 1. The link of a DL-based MIMO-OFDM receiver.

Motivated by the discussion above, we design a model-driven multi-pilot convolution neural network for MIMO-OFDM receivers (DMCNet-pro). Comparing to previous work, a signal denoising module is added to denoise the pilot frequency and data, respectively, in DMCNet-pro. The network we propose shows great performance regarding accuracy and robustness. Additionally, fewer parameters are required for this network in some scenarios.

The rest of this paper is organized as follows. Section 2 introduces the dataset and the channel model. Section 3 introduces the structure of DMCNet. Section 4 introduces the structure of DMCNet-pro. Section 5 introduces the experimental design and results. Section 6 introduces the conclusion of the experiment. We only consider a simplified single user scenario for a clear mathematical analysis, establishing fundamental principles before tackling complexities like multi-user interference in realistic settings.

2. Dataset and Channel Model

This paper focuses on the design of a 2×2 MIMO-OFDM wireless communication receiver. The 2×2 MIMO system strikes a balance between performance and simplicity with two transmit and two receive antennas. It effectively improves system capacity, reliability, and signal coverage, making it widely adopted across wireless communication standards. The required dataset for the experiments includes a 2×2 MIMO channel model and simulation programs for transmission and reception using the channel model. The wireless channel model follows the Wireless World Initiative New Radio (WINNER II) [11], and the channel scenario represents a typical urban street environment. In the simulation programs, the communication frequency was set to 1.85 GHz. The sampled channel data consist of 340,000 data samples, with each channel dataset containing 256 real numbers representing the channel matrix of a 2×2 wireless channel. Each element in the matrix consists of 32 complex numbers. The generated data are divided into training, testing, and validation datasets, consisting of 300,000, 20,000, and 20,000 samples, respectively.

Multi-carrier modulation is implemented using OFDM with 256 subcarriers. The transmitter includes QAM of the transmitted bits X, insertion of pilot symbols, IFFT, addition of a cyclic prefix (CP) with a length of 32, and optional peak clipping. The settings for cyclic prefix and peak clipping can be adjusted through flags in the simulation program.

Two transmit antennas generate random bit sequences of length 512. The combined bit stream of the two sequences is denoted as X, with a length of 1024. The random bit

signals are input into the simulation link, and the receiver obtains the signal y after passing through the channel. Since pilot symbols are added at the transmitter, the received signal y consists of two time slots: one for pilot symbols and one for data. Taking eight pilot symbols as an example, the transmitter first reads the known pilot sequence from a pre-generated pilot file, which contains 32 random bits ($8 \times 2 \times 2 = 32$) in total. The pilot data remain unchanged throughout the training, testing, and validation processes. Then, a random channel sample is selected from the channel samples. The signal after passing through the channel is a result of the data sequence and pilot sequence combined, containing 2048 floating point numbers.

3. The Structure of DMCNet

3.1. The Data Flow of DMCNet

The designed deep-neural-network-based receiver in this paper, referred to as DMCNet, is shown in Figure 2. DMCNet consists of three main modules: signal denoising, pilot processing, and signal detection. In the network models for 32 pilot symbols and 8 pilot symbols, the dimensions of the output at each layer are different. Therefore, only the structure of the model is provided here, while the detailed dimensions of each layer and the specifics of each module will be discussed in the following sections.

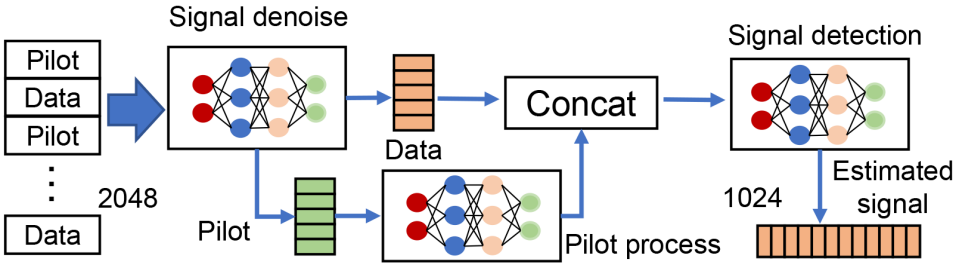

Figure 2. The structure of DMCNet.

Although DMCNet is divided into three modules, the main idea of DMCNet is to use a data-driven network to construct an end-to-end training model while enriching the model's computation by incorporating convolutional modules. As is well known, in the field of computer vision, images are often stored and processed as real numbers. However, in the field of signal transmission, input signals often need to be mapped to the complex plane to perform operations that record both the magnitude and phase information. In this paper, the received signal is treated as a complex number, and complex neural network modules are used for information processing. We primarily employ complex convolution and complex fully connected structures. Subsequently, there have been some open-source implementations of this paper. In this paper, we primarily employ complex convolution and complex fully connected structures. Compared to traditional neural networks, complex neural networks have a more expressive power. The main difference between complex networks and real networks lies in the multiplication operation, as shown in the following equation:

$$(a + bi) \times (c + di) = (ac - bd) + (ad + bc)i. \tag{1}$$

For a complex fully connected network with one input neuron and one output neuron, assuming the input is $a + bi$ and the weight is $c + di$, according to the equation above, the complex fully connected network can be implemented using a real fully connected layer with an input size of 2 and an output size of 2. In this case, the four weights of the fully connected layer are $c, -c, d, -d$. Therefore, the complex fully connected layer can be implemented using a real fully connected layer with double the size. For the complex convolutional module, it can be implemented using higher-dimensional real convolutional layers. Let the input complex matrix be $\mathbf{W} = \mathbf{A} + i\mathbf{B}$ and the complex vector be $\mathbf{h} = \mathbf{x} + i\mathbf{y}$, where $\mathbf{A}$ and $\mathbf{B}$ are real matrices and $\mathbf{x}$ and $\mathbf{y}$ are real vectors. The convolution result of $\mathbf{W}$ and $\mathbf{h}$ can be represented as:

$$\mathbf{W}^{*}\mathbf{h} = (\mathbf{A}^{*}\mathbf{x} - \mathbf{B}^{*}\mathbf{y}) + i(\mathbf{B}^{*}\mathbf{x} + \mathbf{A}^{*}\mathbf{y}). \tag{2}$$

As we can see, the real and imaginary parts of the convolution result can be represented in matrix form as follows:

$$\begin{bmatrix} \Re(\mathbf{W}^{*}\mathbf{h}) \\ \Im(\mathbf{W}^{*}\mathbf{h}) \end{bmatrix} = \begin{bmatrix} \mathbf{A} & -\mathbf{B} \\ \mathbf{B} & \mathbf{A} \end{bmatrix} * \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}. \tag{3}$$

Further information on the definition of activation functions and batch normalization (BN) layers in complex neural networks can be found in [12]. In this section, the experimental setups utilize the architecture of complex neural networks. Next, we will introduce the various data-processing modules in DMCNet separately.

3.2. Signal Denoising Module

The transmitted signal is subject to interference after passing through the channel, and the received signal contains varying levels of noise depending on the signal-to-noise ratio (SNR). A lower SNR corresponds to higher noise in the received signal, which significantly affects the overall signal detection performance. To mitigate the impact of channel noise, this paper introduces a denoising module in the first layer of DMCNet. The received signal is divided into data and pilot components, and each component undergoes denoising using their respective denoising networks. Two approaches were explored in the experiments. The first approach utilizes a two-layer fully connected network, where the input signal is initially expanded to twice its original size in the first layer and then compressed back to the original length in the second layer. The second approach employs a residual structure for the two-layer fully connected network, which incorporates a shortcut connection that subtracts the output of the fully connected layer from the original signal. This approach aligns better with the definition of the noise model. The structure of the denoising module is shown in Figure 3:

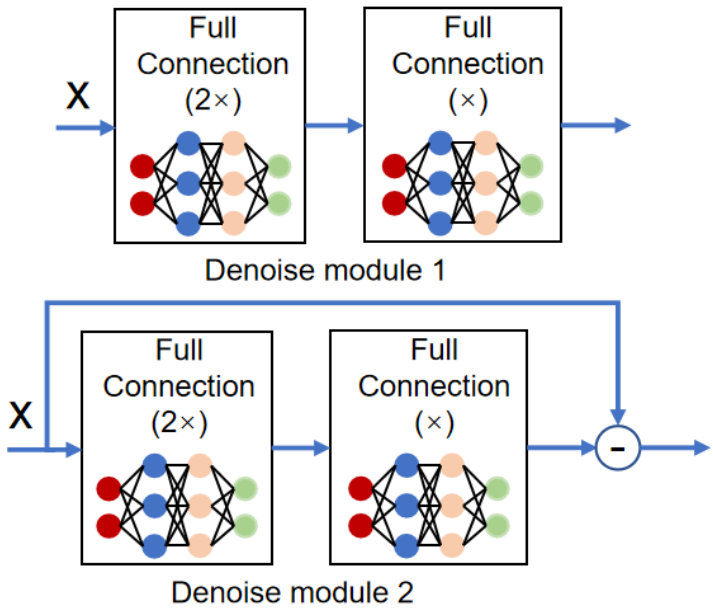

Figure 3. Denoising module structure.

3.3. Pilot Processing Network

In [6], it is mentioned that pilot signals and data signals are processed using the same fully connected network. However, as explained in Section 2, the pilot signals and data signals are not symmetrical regarding their positions during communication, even though they are transmitted through the same channel. The pilot signals are typically shorter

in length compared to the data signals, and the original sequence of the pilot signals is known, allowing for channel estimation. To handle the pilot signals more precisely, this paper introduces a separate pilot processing and analysis step. The pilot signals are first processed and analyzed independently before being merged and processed together with the data signals. To accomplish this, a pilot processing module is designed; the diagram of the pilot processing module is shown in Figure 4 and the pilot processing module's details are shown in Table 1.

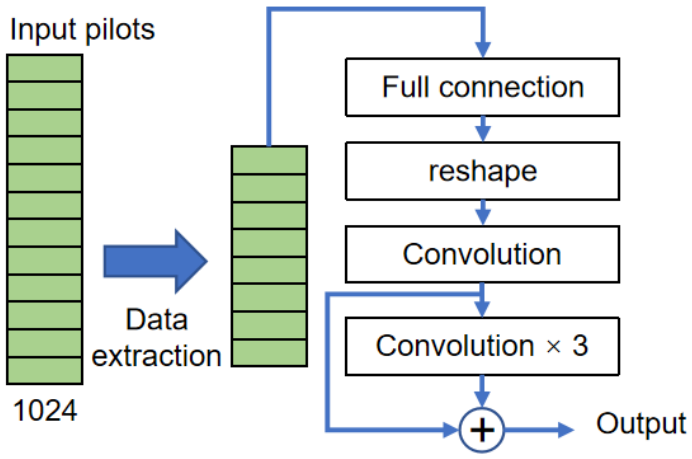

Figure 4. Pilot processing structure.

Table 1. Pilot processing module details.

	Module Type	Output Shape (8 Pilots)	Output Shape (32 Pilots)
Input pilot	–	1024	1024
Data extraction	–	64	256
Full connection layer	full connection	1024	2048
Reshape	reshape	[1, 256, 4]	[1, 256, 8]
Convolution layer 0	1 × 9	[1, 256, 8]	[1, 256, 16]
Convolution layer 1	1 × 9	[1, 256, 16]	1, 256, 32]
Convolution layer 2	1 × 9	[1, 256, 32]	[1, 256, 32]
Convolution layer 3	1 × 9	[1, 256, 8]	[1, 256, 16]
Convolution layer 4	1 × 9	[1, 256, 8]	[1, 256, 16]

After denoising, the received signal with a length of 2048 is divided into data and pilot components, each with a length of 1024. For an 8-pilot link data scenario, this means that pilot data are inserted every 32 subcarriers, and the remaining subcarriers have null pilot data, which does not contain useful information and are outputted through the communication link. In the approach described in [6], regardless of the number of pilot signals, all the data are inputted to the input layer of the neural network and the output result is obtained through a fully connected network. However, in the context of this paper's research scenario, there are

more antennas and the number of OFDM subcarriers has doubled, resulting in the output data from the link increasing from 256 to 2048. Therefore, in this paper, some of the unused pilot positions are discarded. Since the data are shifted by half a pilot position, the model first extracts a set of pilot output data every 16 subcarriers. In the case of eight pilot signals, a total of (256/16) $\times$ 4 = 64 floating point numbers are extracted, which represent the known pilot data through the channel link. The extracted pilot output data are initially processed through a residual module for information extraction. They are first dimensionally expanded using a fully connected network to the size of 1024 and then reshaped into a 1-row, 256-column dataset, with the third dimension of the tensor representing the number of channels. In the case of eight pilot signals, the reshaped tensor has a shape of (1, 256, 4). The dataset is then passed through a complex convolution with a size of 1 $\times$ 9 to expand the channels, resulting in an output size of (1, 256, 16).

A residual network structure is used here, where the pilot data, after being dimensionally expanded and reshaped, are further processed through a residual module for purification. The residual module is similar to the one described in Section 3 and consists of three layers of complex convolution. The output of the third layer is added to the original input of the first layer. The output of the residual module serves as the pilot result.

3.4. Signal Detection Network

The signal detection module's details are shown in Table 2. For the data part, a tensor of length 1024 is also extracted. These 1024 data points represent the information of the data part after passing through the channel, so no further extraction is needed. Similarly, these data are reshaped into a (1, 256, 4) tensor and concatenated with the implicit channel estimation results of the pilot part in the third dimension. This combined tensor serves as the input to the signal detection network. For eight pilots, the dimensions of the merged data are (1, 256, 12). In this data with 12 channels, the first 8 channels contain the results of the pilot part through the pilot module and the last 4 channels contain the original data part after reshaping. The merged tensor of size (1, 256, 12) is then expanded in the channels through two consecutive convolutions, resulting in a tensor of size (1, 256, 100). It is then processed through three Res-Block residual refinement modules. Each Res-Block consists of three cascaded convolutional layers, and the output after the first convolution is directly connected to the final output in a shortcut manner. Finally, the intermediate results are dimensionally reduced to a (1, 256, 4) tensor through convolution, flattened into 1024 floating point data, and the final estimation result is obtained through a bit decision function. It is worth noting that due to the relatively large number of channels in the data of the Res-Block part, a channel attention (CA) module is added to each Res-Block to weight the multi-channel data and improve system performance.

Table 2. Signal detection module details.

	Module Type	**Output Shape (8 Pilots)**	**Output Shape (32 Pilots)**
Merge data	–	[1, 256, 12]	[1, 256, 20]
Convolution layer 0	1 $\times$ 9	[1, 256, 50]	[1, 256, 128]
Convolution layer 1	1 $\times$ 9	[1, 256, 100]	[1, 256, 256]
Res-Block 0	Res-Block	[1, 256, 100]	[1, 256, 256]
Res-Block 1	Res-Block	[1, 256, 100]	[1, 256, 256]
Res-Block 2	Res-Block	[1, 256, 100]	[1, 256, 256]

Table 2. *Cont.*

	Module Type	Output Shape (8 Pilots)	Output Shape (32 Pilots)
Convolution layer 2	1×9	[1, 256, 4]	[1, 256, 4]
Sigmoid	–	[1, 256, 4]	[1, 256, 4]
Bit decision	Threshold function	[1, 1024]	[1, 1024]

4. The Structure of DMCNet-Pro

Model-Driven Receiver Network Structure

In Section 3, a data-driven scheme is adopted to treat the whole receiver as a black box for end-to-end model training. In some previous work, such as [13], a model-driven scheme was used to solve the problem of a single-antenna OFDM signal receiver. Based on the research results in Section 3, a model-driven receiver design is also used in this paper. DMCNet, designed in Section 3, includes several network components to process different input data and perform end-to-end model training. This section attempts to split and improve it, and design a model-driven deep learning receiver scheme.

The model-driven network model is shown in Figure 5, which is named DMCNet-pro in this paper. The model-driven network DMCNet-pro includes signal denoising, a channel estimation network, and a signal detection network. In Section 3, it was shown that the model is also divided into several modules; however, in the training process, as a whole, an end-to-end training method is adopted. The final output result of the receiver and the transmitted signal are used to calculate the loss function, and then the optimizer uses this loss function to adjust the parameters of the entire network to reduce the loss of the next iteration, and the process is repeated to guide the specified training cycle. In the model-driven scheme, the loss function is calculated and trained in stages by using some known information.

In DMCNet, a signal denoising network is added to initially remove the noise of the received signal passing through the channel. In DMCNet-pro, a signal denoising module is also added to denoise the pilot frequency and data, respectively. The difference is that the denoising network needs to be trained separately at this time. The input signal of the signal denoising module is the received signal, the label is the result after the same input passes through the noiseless channel, and the loss function is the mean square error loss between the actual received signal y and the result $\dot{y}$ after the same signal passes through the noiseless channel, which can be expressed as:

$$loss = \frac{1}{n}\sum_{i=1}^{n}(y_i - \dot{y}_i)^2 \tag{4}$$

where n represents the length of signal y. This loss function is passed to the Adam optimizer, which adjusts the model parameters.

The second step is required to train the channel estimation network. As mentioned above, the pilot frequency can actually be understood as a known sequence, which experiences the same channel propagation path as the actual data sequence to be sent; that is, it undergoes the same channel operation, and then obtains a string of output sequences. Through the output sequence and the known pilot sequence, the distribution of the channel can be estimated and the data can be received. Therefore, the received data and the known transmit pilot sequence can be used for channel estimation, and the least-squares (LS) channel estimation algorithm is used for initialization, that is:

$$\widehat{H}_{LS} = \frac{y_P}{x_P} \tag{5}$$

where y_p represents the pilot part of the received signal and x_p represents the pilot sequence known at both ends of the transceiver. The initialized channel estimate $\widehat{H}_{LS}$ serves as the input to the channel estimate network. After initialization by the LS algorithm, the size of $\widehat{H}_{LS}$ is a vector with a length of 256, and the input channel estimation network is further purified. Since precise estimation of the output channel matrix is required, a convolution is added after Res-Block, and the output dimension of the convolution is [1, 256, 1]. The output $\widehat{H}$ is computed with the actual sample matrix H for the mean square error value, which is used as input to the model optimizer to adjust the network parameters.

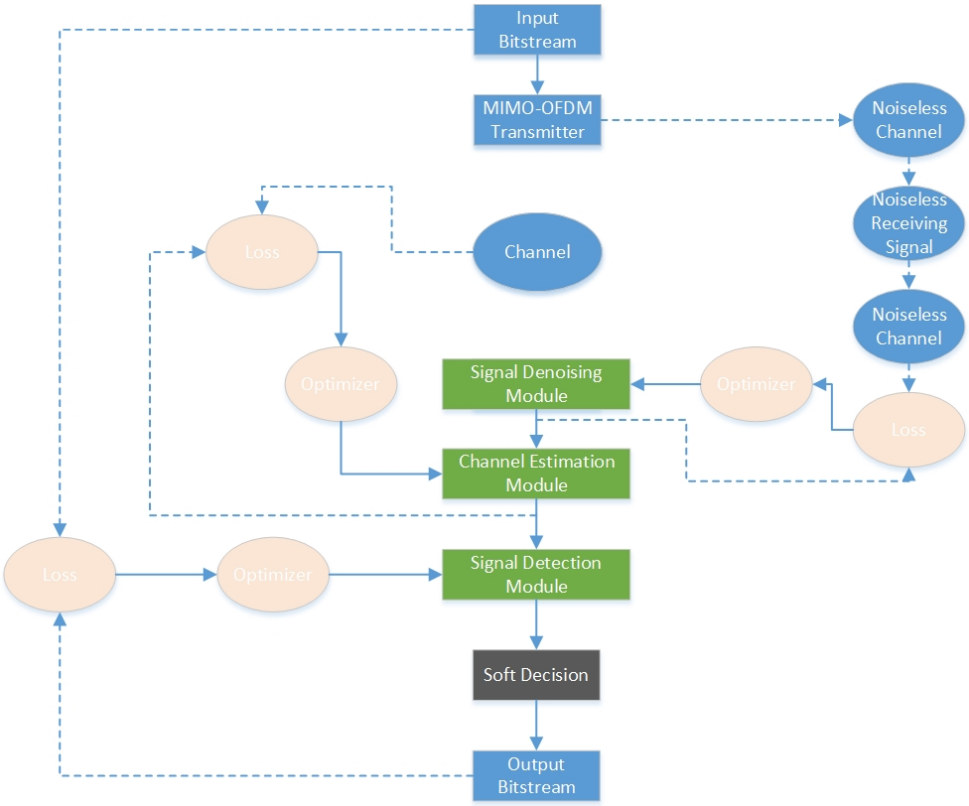

Figure 5. Model-driven MIMO-OFDM receiver.

The third step is required to train the network of the signal detection module. The structure of channel estimation and signal detection model is shown in Figure 6. First, the signal is initialized using the Zero Forcing (ZF) method. The ZF method can improve signal quality and system performance, particularly in multi-user scenarios.

$$\widehat{x}_{ZF} = \frac{y_D}{\widehat{H}} \tag{6}$$

where y_D is the data part of the received signal, $\widehat{H}$ is the output result of the channel estimation module, and the output $\widehat{x}_{ZF}$ is the input of the signal detection network. Comparing to DMCNet, the input of the signal detection network is the result of superposition of data and the pilot processing results. Here, $\widehat{H}$ and $\widehat{x}_{ZF}$ are also input to the signal detection network at the same time and superimposed. Similar to the signal monitoring network in DMCNet, the signal detection network structure in DMCNET-Pro is:

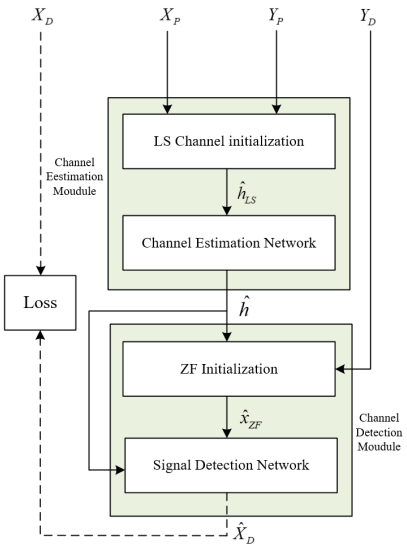

Figure 6. Channel estimation and signal detection model.

As we can see, the structure of the model-driven network is largely the same as that of DMCNet, but in this network, there are fewer output channels for each layer of convolution. The structure of signal detection model is shown in Figure 7. At eight pilots, $\hat{H}$ and $\hat{x}_{ZF}$ synthesize a feature map of size [1, 256, 5], which is first enhanced by convolution and then further purified by several refine blocks, while with the 32-pilot Res-Block, the number of convolutional output channels is 100, 100, and 50, respectively. Compared with DMCNet, it has fewer channels and fewer parameters, but the experimental results show that its performance is similar.

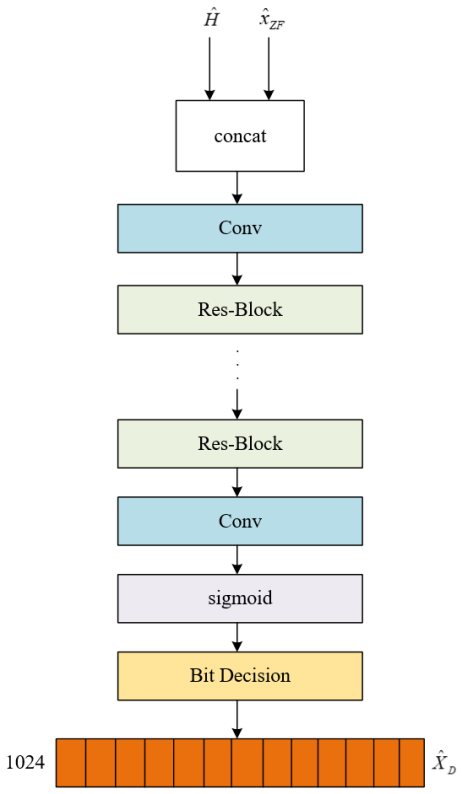

Figure 7. Signal detection model.

5. Experimental Design and Results

5.1. Experimental Design

The experiments in this section were conducted on a laboratory server platform. The server used had two NVIDIA GeForce RTX 2080 Ti GPUs, 16 GB of RAM, and 64 GB of GPU memory. The neural network was implemented using the Keras deep learning framework. The channel dataset used during the training process was collected from a wireless channel model following the Wireless World Research Forum's New Radio [11] initiative (WINNER II). The experimental process included two scenarios: 8 pilots and 32 pilots. In the eight-pilot scenario, there were 8 known complex training sequences for each of the two signals, totaling 16 known complex values. Additionally, 32 random bits (0 s and 1 s) were generated using random numbers. Similarly, for the 32-pilot scenario, 128 random bits were generated. The generated random bits were written to files named "Pilot8" and "Pilot32", respectively, and the pilot sequences remained unchanged throughout the subsequent experiments.

The channel model was the same as described in Section 2. Two antennas at the transmitter generated random data of 512 bits each, which were then transmitted through an Additive White Gaussian Noise (AWGN) simulated channel. The output of the AWGN channel serves as the input to the network. During the training process, the batch size was set to 128. Batches of 128 samples were generated using a Python generator for training, and the number of steps per epoch was set to 1000, meaning that 1000 batches of samples were input per epoch for training. The total number of epochs was set to 200. The Adam optimizer was used to optimize the model parameters. Since the final output is a binary sequence, binary cross-entropy was used to compute the model loss. It can be mathematically represented by the following equation:

$$\text{Loss} = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot \log(p(y_i)) + (1 - y_i) \cdot \log(1 - p(y_i)). \tag{7}$$

In this equation, y represents the binary label (0 or 1) and $P(y)$ denotes the probability of the output label. The learning rate follows a step-wise decay approach, where it is reduced by 0.75 times every 30 epochs.

For the model-driven receiver network DMCNet-pro, model training is carried out in a phased way, which is divided into four stages: training the denoising module (100 epochs); fixing the denoising module parameters and training the channel estimation module (100 epochs); training the parameters of the fixed-signal denoising module and the cardio estimation module to the signal detection module (100 epochs); and end-to-end training with a small learning rate (0.0001), fine-tuning the model, and training (200 epochs). The other settings for the experiment are the same as for DMCNet.

5.2. Performance and Parameter Count of Fully Connected Networks

In [6], the authors employed a fully connected network (DNN) as the receiver for single-antenna OFDM signals. For the MIMO-OFDM system studied in this paper, the experiments were also conducted using a fully connected network. The number of fully connected layers used in the experiments was 3, with a gradually increasing number of neurons in each layer. The output data were fixed as the 16 bits from the 128th to the 143rd positions. The last layer of the fully connected network utilized a sigmoid activation function to ensure the output fell within the [0, 1] range. Finally, a bit decision function was employed to convert the neural network output into binary (0 or 1) bits:

$$output = \begin{cases} 0 & x <= 0.5 \\ 1 & \text{else} \end{cases}. \tag{8}$$

The bit error ratio (BER) performance of the fully connected network under 8 pilot tones and 32 pilot tones is shown in Figures 8 and 9.

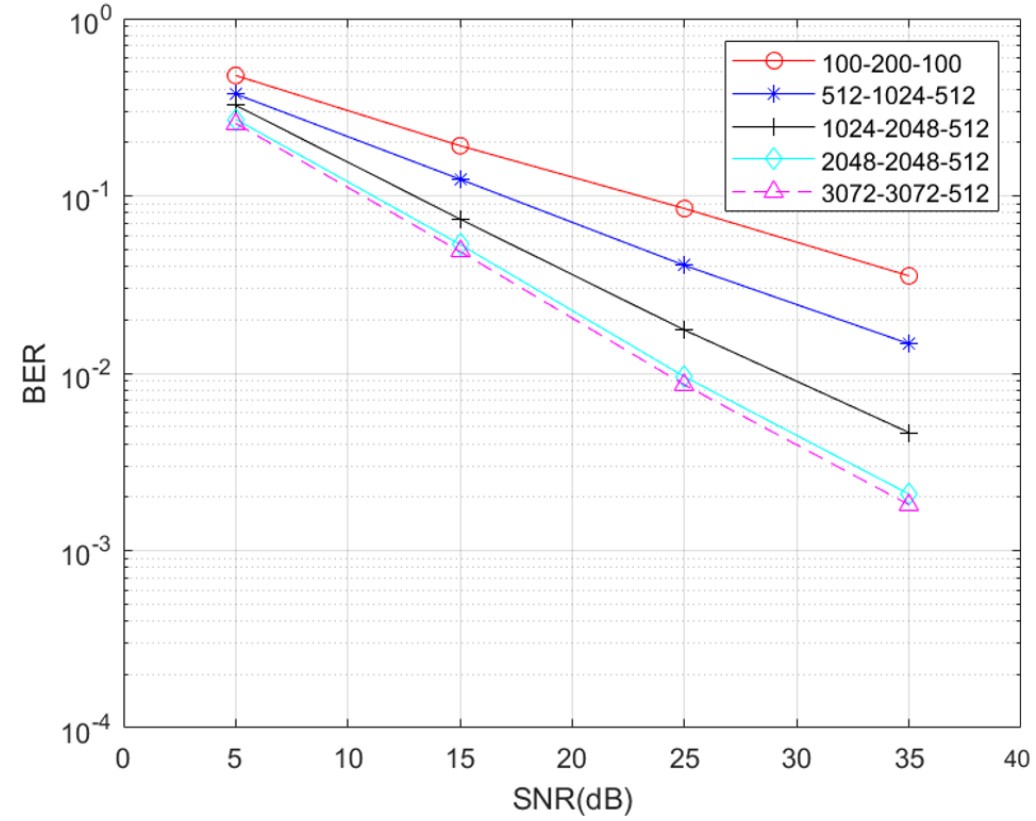

Figure 8. BER of MIMO-OFDM fully connected receiver under eight pilot tones.

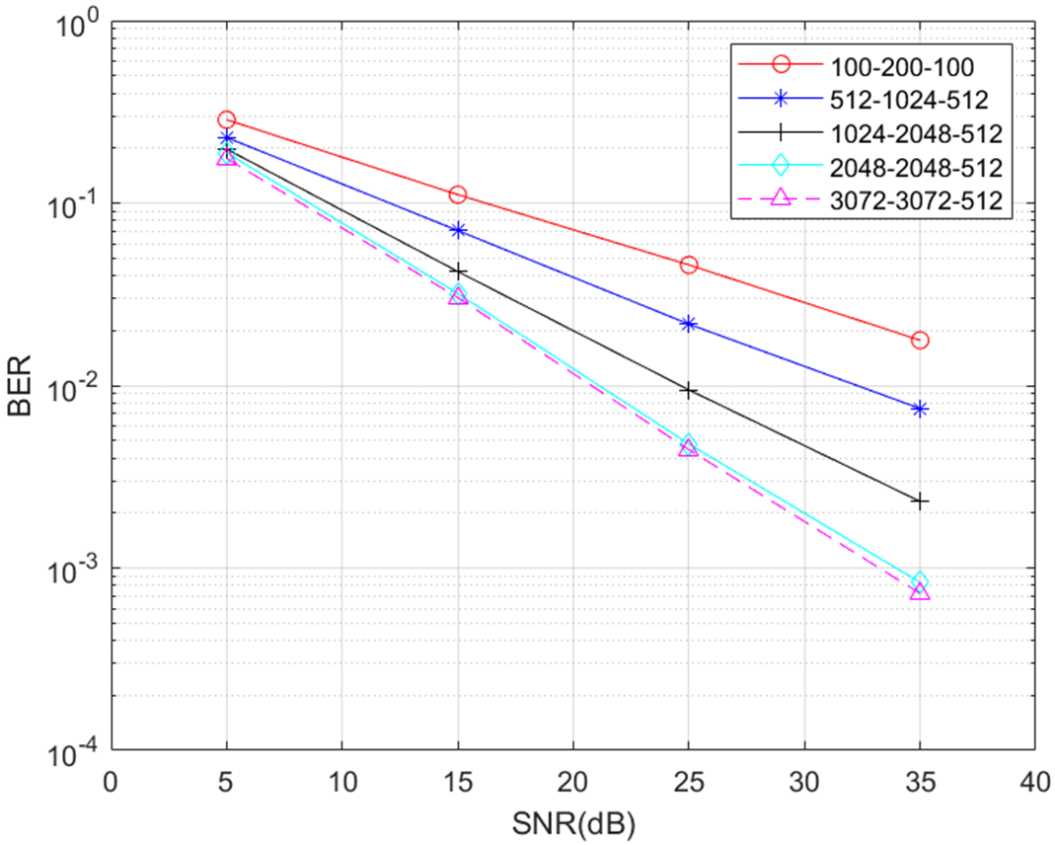

Figure 9. BER of MIMO-OFDM fully connected receiver under 32 pilot tones.

It can be observed that in the MIMO-OFDM scenario, the receiver performance of the fully connected network is significantly worse compared to the single-antenna OFDM scenario. By increasing the number of neurons in the fully connected layers, the BER performance can be gradually improved. However, once the number of neurons in the fully connected layers reaches a certain value, the network's bit error rate reaches a bottleneck, and further increasing the network size does not yield significant benefits.

In [6], the authors employed a scheme where each fully connected network estimates only 16 input bits, resulting in a total of 512 input bits. Therefore, 32 parallel fully connected networks are required for estimation, and their output results are subsequently concatenated. However, for the MIMO-OFDM receiver, the number of input bits is higher, necessitating a larger number of sub-networks. To investigate the impact of the output bit number on the receiver's BER performance, we conducted experiments with output lengths of 8 bits, 16 bits, 32 bits, 64 bits, and 128 bits under eight pilot symbols. The predicted data corresponded to the input starting from the 128th data point, which, respectively, covered the ranges of 128–135 bits, 128–142 bits, 128–159 bits, 128–171 bits, and 128–255 bits. The results are shown in Figure 10:

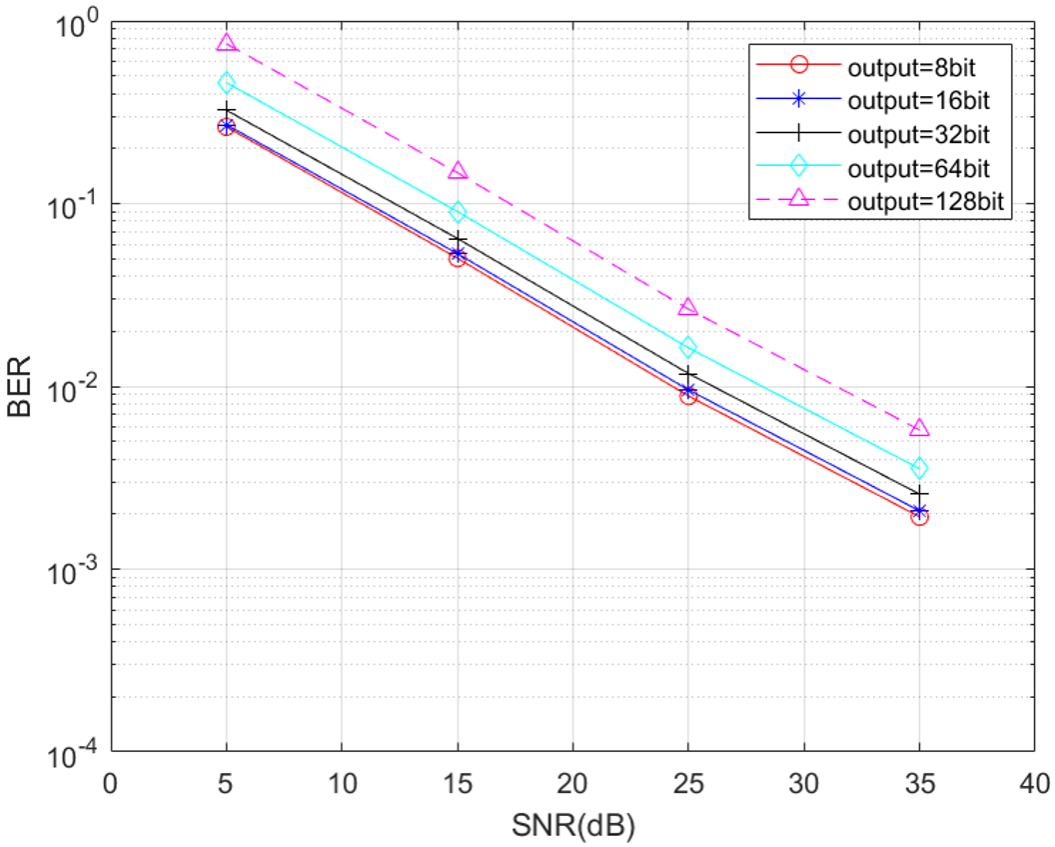

Figure 10. Relationship between prediction length and BER of an MIMO-OFDM fully connected receiver.

Based on the experimental results, it can be concluded that the performance of a single fully connected network deteriorates as the number of predicted bits increases. As the prediction length decreases, the BER gradually decreases, but at a slower rate. When the output length is 16 bits, further reducing the output length has little impact on performance as it reaches saturation. Additionally, a smaller number of predicted bits per network requires a larger number of sub-networks. In this study, a prediction length of 16 bits was used, which means a total of 64 sub-networks are needed to concatenate the final output.

Furthermore, the parameter count is a significant concern in the fully connected solution. In the case of MIMO-OFDM receivers with a large number of input bits, using a

fully connected approach requires multiple large sub-networks to combine the outputs. The parameter count for each parameter configuration in the graph is already substantial, and employing a fully connected approach with a high parameter count yields a mediocre performance. Therefore, the fully connected signal reception solution suitable for single-antenna scenarios is not applicable to MIMO-OFDM scenarios. The following fully connected experiments employ a 2048-2048-512 network configuration.

5.3. Results and Analysis of the DMCNet

This study compares the results between the 8-pilot and 32-pilot scenarios. As shown in Figure 11, it can be observed that in the eight-pilot scenario, with fewer pilots, the traditional approach struggles to utilize pilot data for channel information recovery and signal detection. Therefore, the BER performance saturates and is the worst when the SNR exceeds 15. On the other hand, in the deep-learning-based approach, both fully connected and convolutional networks show continuous BER reductions as the SNR increases. The conclusion drawn from this is that the neural-network-based approach exhibits stronger robustness regardless of the number of pilots, while the traditional approach heavily relies on pilot data for accurate channel estimation. The superiority of neural networks in this aspect mainly stems from their ability to approximate any function, and the characteristics of wireless channels can be seen as complex functions that can be learned from training data and network models.

In the case of 32 pilots, the traditional LMMSE-MMSE receiver can leverage more pilots to recover channel information, and the performance of the neural network algorithm is comparable to the traditional method.

Comparing the fully connected and convolutional receiver schemes, it can be observed that the convolutional receiver outperforms the fully connected receiver. In the single-antenna OFDM scenario, where the input–output signal length is short, the fully connected network can approximate the characteristics of the channel. However, in the multi-antenna scenario, with longer input signal lengths and where the channel state is represented as a multidimensional matrix, the operation of the input signal through the channel becomes more complex and the limitations of the fully connected network operation are insufficient to fit the channel model. The introduction of convolutional modules and residual modules in DMCNet enriches the network's operation and allows for a better fit to the channel model, resulting in a lower BER compared to the fully connected approach.

Furthermore, DMCNet does not require input data segmentation, and its output length is 1024 bits, allowing for the reception of the entire input signal through a single network. In terms of the parameter count, in the eight-pilot scenario, DMCNet has a parameter count of 4,202,496, while the fully connected network has a parameter count of 7,348,224 $\times$ 64. The parameter count of DMCNet is only 9% of the fully connected network, making it more suitable for practical deployment and usage.

Based on the experimental results, it can be concluded that the proposed DMCNet is suitable for receiving 2 $\times$ 2 MIMO-OFDM signals and outperforms fully connected networks in terms of performance with fewer parameters. It exhibits a stronger robustness in scenarios with fewer pilots, being able to fit results with limited data. Its performance in scenarios with more pilots is comparable to traditional approaches.

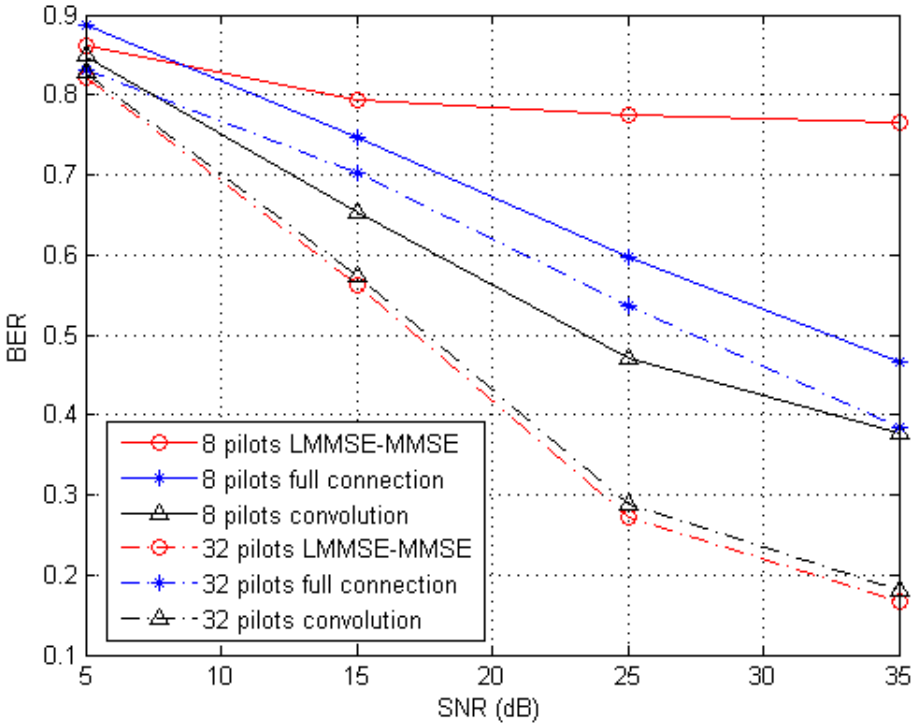

Figure 11. Accruacy comparison of different signal detection algorithms.

5.4. Results and Analysis of DMCNet-Pro

It can be seen that at 32 pilots, the accuracy of the model-driven scheme is slightly improved compared with the data-driven method and the traditional method, but at 8 pilots, the accuracy of the model-driven scheme is very close to that of DMCNet. The main reason for this is that when there is a large number of pilot sequences, the receiver can use traditional methods to perform channel estimation and signal detection comparatively. Therefore, LS initialization and ZF initialization provide guidance to the neural network, and the neural network can gradually adjust parameters in the right direction to reduce errors. However, in eight-pilot mode, due to the small pilot frequency, it is difficult for traditional methods to perform signal estimation and signal detection, and the accuracy of LS initialization and ZF initialization is not high, so the gain brought by changing to a model-driven model is not large.

During the experiment, it is found that in the model-driven receiver, the number of parameters of the model can be greatly reduced and the accuracy is close to that of the data-driven model. This is reflected in the number of output channels of Res-Block three-layer convolution. For this reason, an experiment was designed to explore the influence of the number of output channels of Res-Block on the performance. The BER of each group is shown in Figure 12, where 512-512-256* indicates the BER of the data-driven model DMCNet at 32 pilot frequency. It can be seen from the results that in the model-driven network, when the number of Res-Block output channels reaches 200-200-100, the accuracy is already better than that in the data-driven network with more parameters. When the number of parameters continues to increase, the gain in accuracy is not obvious. Therefore, the set of parameters of 200-200-100 is selected at 32 pilot frequency to set the number of channels in each layer of Res-Block. By comparing the parameter number and FLOPs of the 32-pilot model-driven network DMCNet, it can be seen that the model-driven scheme can achieve a better or similar receiving accuracy with less computation, which indicates that in the data-driven scheme, a large number of parameters and operations is needed to fit the signal transmission process. By making full use of the known information, part of the network parameters can effectively be replaced, and the method used for sub-module

training also makes the training of each module's parameters more efficient. The same experiment was also performed at eight pilots. Due to the low accuracy of traditional algorithms for channel estimation and signal detection at eight pilots, the model-driven scheme does not bring significant accuracy or parameter number gains at eight pilots.

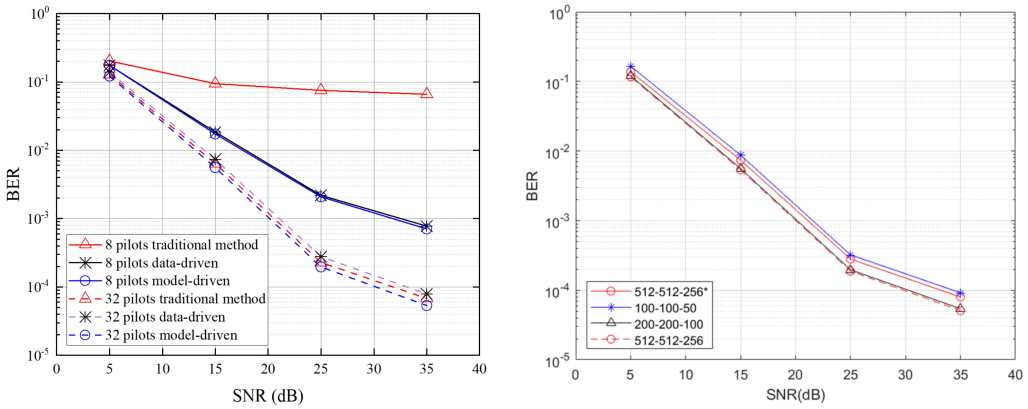

Figure 12. Comparison of data-driven and model-driven results (**left**) and model-driven Res-Block channel numbers (**right**).

6. Conclusions

This paper discusses the problem of signal receivers based on deep learning in an MIMO-OFDM scenario. Compared with a single antenna, the output signal length in the multi-antenna scenario is longer and the receiver design is more complex. The proposed receiver based on deep learning, DMCNet, combines complex convolution and a complex full connection structure, which has a better performance and fewer parameters than previous fully connected networks.

Compared with traditional receiver algorithms, the receiver scheme based on deep learning is more robust, and the accuracy is less affected when nonlinear noise (such as cyclic prefix and peak clipping) is introduced, which also indicates that the deep learning receiver has a certain ability to fit the channel changes. In addition, the influence of some key modules on the accuracy of the neural network is compared. Based on the data-driven model DMCNet, this work expands the model-driven scheme to DMCNET-Pro. Combined with the relevant algorithms in traditional communication, the model-driven scheme has a higher accuracy and fewer parameters in some scenarios.

In summary, the MIMO-OFDM receiver based on deep learning has more advantages than traditional methods in some aspects, and has certain research and application prospects.

Author Contributions: Conceptualization, T.Z.; Methodology, P.L. and Y.X.; Formal analysis, X.Y. and Z.L. (Zili Liu); Investigation, Q.Y.; Writing—original draft, G.Y.; Writing—review & editing, Z.L. (Zejian Lu). All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by National Key Technologies R&D Program (2022YFB4300400), China Scholarship Council (202106475002), Belt and Road Initiative Innovative Talent Exchange Foreign Expert Project (DL2023107003L), and in partly by National Natural Science Foundation of China (61801035).

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Qin, Z.; Ye, H.; Li, G.Y.; Juang, B.H.F. Deep learning in physical layer communications. *IEEE Wirel. Commun.* **2019**, *26*, 93–99. [CrossRef]
2. Cao, B.; Yang, Y.; Ran, P.; He, D.; He, G. ACCsiNet: Asymmetric convolution-based autoencoder framework for massive MIMO CSI feedback. *IEEE Commun. Lett.* **2021**, *25*, 3873–3877. [CrossRef]
3. Liu, H.; Li, G.; Cheng, X.; Li, D. Performance analysis framework of ML MIMO receiver over correlated Rayleigh fading channel. In Proceedings of the 2006 IEEE International Conference on Communications, Guilin, China, 25–28 June 2006; Volume 9, pp. 4137–4142.
4. Wang, C.; Au, E.K.; Murch, R.D.; Mow, W.H.; Cheng, R.S.; Lau, V. On the performance of the MIMO zero-forcing receiver in the presence of channel estimation error. *IEEE Trans. Wirel. Commun.* **2007**, *6*, 805–810. [CrossRef]
5. Eraslan, E.; Daneshrad, B.; Lou, C.Y. Performance indicator for MIMO MMSE receivers in the presence of channel estimation error. *IEEE Wirel. Commun. Lett.* **2013**, *2*, 211–214. [CrossRef]
6. Ye, H.; Li, G.Y.; Juang, B.H. Power of deep learning for channel estimation and signal detection in OFDM systems. *IEEE Wirel. Commun. Lett.* **2017**, *7*, 114–117. [CrossRef]
7. Jiang, P.; Wang, T.; Han, B.; Gao, X.; Zhang, J.; Wen, C.K.; Jin, S.; Li, G.Y. Artificial intelligence-aided OFDM receiver: Design and experimental results. *arXiv* **2018**, arXiv:1812.06638.
8. Zhao, Z.; Vuran, M.C.; Guo, F.; Scott, S.D. Deep-waveform: A learned OFDM receiver based on deep complex-valued convolutional networks. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 2407–2420. [CrossRef]
9. He, H.; Wen, C.K.; Jin, S.; Li, G.Y. A model-driven deep learning network for MIMO detection. In Proceedings of the 2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Anaheim, CA, USA, 26–29 November 2018; pp. 584–588.
10. Xin, Y.; Peng, J.; Lu, Z.; Lee, Y.; Yang, Y. Dmcnet: Data-driven multi-Pilot convolution neural network for MIMO-OFDM receiver. In Proceedings of the 2023 IEEE International Conference on Network Intelligence and Digital Content (IC-NIDC), Beijing, China, 3–5 November 2023; pp. 1–5.
11. Winner, I. *WINNER II Channel Models*; IST-4-027756 WINNER II, D. 1. 1. 2 V1. 2; MathWorks: Natick, MA, USA, 2007.
12. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
13. Gao, X.; Jin, S.; Wen, C.K.; Li, G.Y. ComNet: Combination of Deep Learning and Expert Knowledge in OFDM Receivers. *IEEE Commun. Lett.* **2018**, *22*, 2627–2630. [CrossRef]