FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

DEGREE PROGRAMME IN WIRELESS COMMUNICATIONS ENGINEERING

MASTER'S THESIS

DEEP LEARNING SIC APPROACH FOR UPLINK MIMO-NOMA SYSTEM

Author	Mar Lwin Khin
Supervisor	Prof. Nandana Rajatheva
Second Examiner	Pekka Pirinen
Technical Advisor	Vismika Ranasinghe Mudiyanselage

August 2022

Khin M. (2022) Deep Learning SIC Approach for Uplink MIMO-NOMA System.
University of Oulu, Faculty of Information Technology and Electrical Engineering, Degree Programme in Wireless Communications Engineering. Master's thesis, 50p.

ABSTRACT

Deep learning-based successive interference cancellation (DL-SIC) for uplink multiple-input multiple-output – non-orthogonal multiple access (MIMO-NOMA) system tries to optimize the users' bit error rate (BER) and total mean square error (MSE) performance with higher order modulation schemes. The recent work of DL-SIC receiver design for users with a QPSK modulation scheme is investigated in this thesis to validate its performance as a potential alternative approach to traditional SIC receivers for NOMA users. Then, a DL-SIC receiver design for higher order modulation with less dependence on modulation order in the output layer is proposed, which enables us to decode the users with different modulation schemes. In our proposed design, we employ two deep neural networks (DNNs) for each SIC step. The system model is considered an M-antenna base station (BS) that serves two uplink users with a single antenna in the Rayleigh fading channel. The equivalent conventional minimum mean square error-based SIC (MMSE-SIC) and zero-forcing-based SIC (ZF-SIC) receivers are implemented as a baseline comparison.

The simulation results showed that the BER performance of the proposed DL-SIC receiver for both users with QPSK modulation results in a 10 dB gain between BER of 10^{-2} and 10^{-3} compared to the ZF-SIC receiver. Furthermore, the performance difference between the proposed scheme and ZF-SIC is significantly high when both users transmit with 16QAM. Overall, the proposed DL-SIC receiver performs better in all signal-to-noise ratio (SNR) regions than the equivalent ZF-SIC receivers and also aids in mitigating the SIC error propagation problem. In addition, it improves the processing latency due to the benefits of the parallelized computing architecture and decreases the complexity of traditional SIC receivers.

Keywords: deep learning, successive interference cancellation, uplink, multiple-inputs multiple-outputs, non-orthogonal multiple access, higher-order modulation, deep neural networks.

TABLE OF CONTENTS

ABSTRACT
TABLE OF CONTENTS
FOREWORD
LIST OF ABBREVIATIONS AND SYMBOLS
1 INTRODUCTION 8
1.1 Motivation and Objectives 9
1.2 Thesis Outlines 9
2 BACKGROUND AND LITERATURE REVIEW 11
2.1 Deep Learning as a Potential Approach to the Challenges of Physical Layer Communication 11
2.2 Deep Neural Network Basic 13
2.2.1 Traning a Neural Network 15
2.3 Deep Learning Libraries 17
2.4 Activation Functions 18
2.5 Loss Functions 19
2.6 Literature Review 20
2.6.1 Deep Learning Based Joint Channel Estimation and Detection 20
2.6.2 Deep Learning Approach MIMO-NOMA Systems 21
3 DEEP LEARNING-BASED SIC WITH QPSK MODULATION 23
3.1 System Model 23
3.2 Implementation of DNN for SIC Receiver 25
3.3 Test and Results 26
4 DEEP LEARNING-BASED SIC FOR HIGHER-ORDER MODULATION 29
4.1 Introduction 29
4.2 DL-SIC for Higher-Order Modulation Using One DNN at Each SIC Step 30
4.2.1 Implementation 30
4.2.2 Test and Results 31
4.3 DL-SIC for Higher-Order Modulation Using Two DNNs at Each SIC Step 33
4.3.1 Implementation 35
4.3.2 Test and Results 36
4.4 Comparisons and Discussion 41
5 FUTURE WORK 45
6 CONCLUSION 46
7 REFERENCES 48

FOREWORD

This thesis was completed as part of the Master's degree program in Wireless Communications Engineering at the University of Oulu in Finland.

First and foremost, I would like to thank Prof. Nandana Rajatheva, my mentor and supervisor, for the immense support, guidance, and inspiration throughout my Master's studies. Next, I would like to express my gratitude to Prof. Hla Myo Tun and Dr. Kimmo Mäkeläinen for the support and inspiration during my post-graduate studies for a telecommunication diploma at Yangon Technological University, Yangon, Myanmar. I thank Vismika Ranasinghe Mudiyanselage, my technical advisor, for the suggestions and guidance during my Master's thesis. I am also grateful to Adj. Prof. Pekka Pirinen for his insightful remarks and support of my thesis work. I am also thankful to all the teachers and mentors who have taught and given me guidance since my childhood.

Finally, I would like to express my sincere gratitude to my family for raising me and for their unconditional love and support throughout my life.

Oulu, 12th August 2022

Khin Mar Lwin

LIST OF ABBREVIATIONS AND SYMBOLS

Acronyms

5G	Fifth Generation
API	Application Programming Interface
AWGN	Additive White Gaussian Noise
BAIR	Berkeley Artificial Intelligence Research
BER	Bit Error Rate
CF	Cyclic Prefix
CNN	Convolutional Neural Networks
CPU	Central Processing Unit
CSI	Channel State Information
DL	Deep Learning
DL-SIC	Deep Learning-based Successive Interference Cancellation
DNN	Deep Neural Network
DOF	Degree of Freedom
DRL	Deep Reinforcement Learning
EP	Error Propagation
ELU	Exponential Linear Unit
FDE	Frequency Domain Equalization
FNN	Feed Forward Neural Network
FTN	Faster than Nyquist
GAN	Generative Adversarial Network
SGD	Stochastic Gradient Descent
GPU	Graphics Processing Units
LMMSE	Linear Minimum Mean Square Error
LS	Least Square
LSTM	Long Short-Term Memory
MIMO	Multiple-Input Multiple-Output
ML	Machine Learning
MLP	Multi-Layer Perceptron
MMSE	Minimum Mean Square Error
MSE	Mean Squared Error
mMTC	Massive Machine Type Communication
NLP	Natural Language Processing
NN	Neural Network
NOMA	Non-orthogonal Multiple Access
OFDMA	Orthogonal Frequency Division Multiple Access
OMA	Orthogonal Multiple Access
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase-Shift Keying
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RNN	Recurrent Neural Network
SELU	Scaled Exponential Linear Unit

SGD	Stochastic Gradient Descent
SIC	Successive Interference Cancellation
SIMO	Single-Input Multiple-Output
SINR	Signal-to-Interference-Plus-Noise Ratio
SNR	Signal-to-Noise Ratio
TPU	Tensor Processing Unit
ZF	Zero-Forcing

Symbols

n	Number of inputs in the input layer
m	Number of neurons in the hidden layer,
$\{e_1, e_2, \dots, e_m\}$	Set of m neurons
q	Number of neurons in the output layer
$\{o_1, o_2, \dots, o_q\}$	Set of q neurons
W	The weight matrix in the hidden layer
$\widehat{W}$	The weight matrix in the output layer
$\omega_m \in \mathbb{R}^{1\times n}$	The weight vector of m^{th} neuron for n inputs
$\widehat{\omega}_q \in \mathbb{R}^{1\times m}$	The weight vector of q^{th} neuron for m inputs
B	The bias vector for the m neurons at the hidden layer
$\hat{B}$	The bias vector for the q neurons at the output layer
b_m	The bias value at the m^{th} neuron in the hidden layer
$\hat{b}_q$	The bias value at the q^{th} neuron in the output layer
$\hat{Y}$	The output vector of the hidden layer
$\hat{y}_m$	The output of m^{th} neuron
$\hat{O}$	The output vector of the output layer
$\hat{o}_q$	The output of q^{th} neuron
$s(.)$	The nonlinear activation function
$\mathbb{R}^{m\times n}$	Real matrix with dimension $m \times n$
$\mathbb{R}^{q\times m}$	Real matrix with dimension $q \times n$
$\mathbb{R}^{1\times n}$	Real vector with dimension $1 \times n$
$\mathbb{R}^{1\times m}$	Real vector with dimension $1 \times m$
$\mathbb{R}^{m\times 1}$	Real vector with dimension $m \times 1$
$\mathbb{R}^{q\times 1}$	Real vector with dimension $q \times 1$
B_{jc}	Binary indicator ground truth such that $B_{jc} = 1$ if and only if the j^{th} sample is a member of the c^{th} class
C	Modulation order
G^D	Additive white gaussian noise at BS for data transmission
G^T	Additive white gaussian noise at BS for pilot transmission
H	Channel matrix
h_k	Channel vector for k^{th}user
$\hat{\mathrm{h}}_k$	Channel estimation vector for k^{th}user
J	Number of pilot symbols
K	Number of users
M	Number of receive antennas

N	Number of data symbols
P	Total peak power
P_r	Received signal power
Q	Number of layers of DNN
$SINR_k$	Signal-to-Interference-and-Noise Ratio for k^{th} user
$SINR_1$	Signal-to-Interference-and-Noise Ratio for the first user
t	Output vector of softmax function
t_c	An output probability value of c^{th} class
t_{jc}	Output probability value that the j^{th} input belongs to c^{th} class
V	Power allocation matrix
W_k	Weight matrix of MMSE estimation for k^{th} user
X^D	Data symbols matrix
X_k^D	Data symbols vector for k^{th}user
$\hat{X}_k^D$	Estimated data symbols vector for k^{th} user
X^L	Pilot symbols matrix
X_k^L	Pilot symbols vector for k^{th}user
Y^D	Received signal matrix during data transmission
Y^L	Received signal matrix during pilot transmission
Z_k	Weight matrix of ZF estimation for k^{th}user
$\mathbb{C}^{K\times J}$	Complex matrix with dimension $K \times J$
$\mathbb{C}^{K\times N}$	Complex matrix with dimension $K \times N$
$\mathbb{C}^{M\times 1}$	Complex vector with dimension M
$\mathbb{C}^{M\times J}$	Complex matrix with dimension $M \times J$
$\mathbb{C}^{M\times K}$	Complex matrix with dimension $M \times K$
$\mathbb{C}^{M\times N}$	Complex matrix with dimension $M \times N$
$\mathbb{C}^{1\times N}$	Complex vector with dimension N
$\mathbb{C}^{1\times J}$	Complex vector with dimension J
λ_k	Power allocation factor for k^{th} user
ρ	Received SNR
$f_k(.)$	Function of k^{th} DNN
$\sigma(.)$	Softmax function
$\mathcal{CN}(0, N_0 I)$	Circularly symmetric complex Gaussian distribution with zero mean and variance N_0

1. INTRODUCTION

Today, wireless networks and related services are critical components of the modern digitized world and significantly impact our daily lives. As the fifth generation (5G) era starts, consumers and businesses begin identifying processes and channels to increase efficiency and boost livelihood. Therefore, applications and use cases such as autonomous vehicle control, intelligent transportation system, smart agriculture, and factory cell automation promised by 5G are emerging. Due to those different bodies of wireless communications, the existing communication mechanism needs improvement in low latency, reliability, availability, and so forth [1].

The performance of non-orthogonal multiple access (NOMA) has been recognized as one of the promising technologies. Unlike orthogonal multiple access (OMA), NOMA simultaneously uses the same frequency resources for different users within the same cell. It can also improve cell-edge throughput and reduce transmission latency [2]. NOMA can be applied to many former wireless communication techniques, including multiple-input and multiple-output (MIMO). In recent years, the application of MIMO to NOMA has received considerable attention from many researchers. MIMO-NOMA stands out for its superior spectral efficiency performance. [3].

With the growing mobile connectivity, the diversity and complexity of wireless networks have increased [4], encouraging machine learning to introduce wireless architecture. Deep learning (DL) is part of machine learning (ML), and the deep learning approach to wireless communication has become of widespread interest. Deep learning can handle massive volumes of data. Mobile networks generate large amounts of various types of data at a rapid speed. Most mobile systems have limited labeled data because manual data annotation needs human interaction that is both costly and time-consuming. Traditional supervised learning is only effective when there is a sufficient amount of labeled data. In contrast, deep learning provides several approaches for using unlabeled data to learn in an unsupervised manner [5].

Deep learning could help operate in pure data-based methods, which means that sets of trained data can be used to optimize network architecture. Deep learning employs a deep neural network (DNN) to investigate data representation in each layer. Conventionally, the communication system is the connected structure of processing blocks such as modulation, and detection, where these blocks are analyzed and modeled mathematically. However, unknown factors could be challenging to analyze mathematically in a practical communication system. With the approach of DL, the networks/systems are optimized over a large training set of data, and a mathematically tractable model is not necessarily required [6].

Successive interference cancellation (SIC) is a decoding process of users' signals in multiuser systems. The process of SIC is to decode the users successively according to their signal strength. The SIC decodes the strongest user signal while treating the other as an interference. Before the following user's signal is decoded, the decoded signal of the first user is subtracted from the combined signal. Hence, the next decoded signal is decoded with the benefit of removing the previous signal [2]. However, the SIC method can cause error propagation (EP) and receiver complexity. As the number of users increases, both uplink and downlink systems suffer from a higher receiver complexity [7]. Implementing SIC with novel deep learning could help to reduce the problems of EP and receiver complexity.

1.1 Motivation and Objectives

Successive interference cancellation is a critical process in the MIMO-NOMA system. In practice, traditional SIC decoders are imperfect. When the error has arisen from decoding one user, the subsequent users will suffer error propagation, and their signals will likely be inaccurately decoded. Even if one user's signal is decoded correctly, the reconstruction of the signal can be incorrect because the channel estimation error may occur [7]. The errors raised by the imperfect SIC decoding can severely affect the system's performance.

As a fundamental requirement of a communications system, it must be capable of reliably sending and receiving messages over a channel by using a transmitter and a receiver. The traditional wireless communication system needs many signal processing processes such as channel coding, modulation, channel estimation, demodulation, etc., between transmitter and receiver to achieve reliable communication [8]. In contrast, the deep learning-based SIC approaches successively extract users' signals without explicitly doing channel estimation, demodulation, and removal of decoded signals. The deep learning-based SIC (DL-SIC) employs a deep neural network (DNN) used at every SIC stage of decoding users. In particular, one DNN with a fully connected layer is used to decode one user.

Considering the above motivation, we implement the DL-SIC for the uplink MIMO-NOMA system in this thesis. This thesis aims to mitigate the error propagation and receiver complexity of traditional SIC operations and, at the same time, reduce the processing complexity and latency of block structure type wireless communication systems by using deep neural networks (DNNs). Inspired by the exciting research of DL-SIC for quadrature phase shift keying (QPSK) users [9], we upgrade the architecture of DNNs for higher-order modulation. After that, we propose a new layout for the DL-SIC design. We use bit error rate (BER) and total mean square error (MSE) as the analysis metrics.

The main objectives of the thesis can be listed as follows.

• Perform a thorough literature review on deep learning and study the deep learning approaches for physical layer communications, especially for multiuser decoding with SIC.

• Study the deep learning-based SIC system proposed by [9] with QPSK modulated users and compare it with the traditional equivalent system.

• Develop the DL-SIC receiver inspired by [9] for users with higher-order quadrature amplitude modulation (QAM). We implemented 16QAM in this thesis.

• Develop a DL-SIC model for higher-order modulation with a new proposed layout to decode the users' signals with comparable performance to the existing conventional systems.

1.2 Thesis Outline

The thesis includes six chapters. In this chapter, we have described the requirements and challenges of conventional successive interference cancellation receivers and how deep learning can support the development of the successive interference cancellation of the MIMO-NOMA system. Chapter 2 presents the background literature about deep learning and an overview of

deep learning-based models in wireless communication. In Chapter 3, we implement the deep learning-based SIC for the uplink of the MIMO-NOMA system of [9] with slight modification and compare and analyze it with the conventional minimum mean square error (MMSE) based SIC. In Chapter 4, firstly, we upgrade the DNN architecture inspired by [9] to decode higher-order modulation users. After that, we implement the proposed DL-SIC considering users transmitting with hierarchical modulation. We also implement zero-forcing (ZF) based SIC for the traditional MIMO-NOMA system to compare and analyze with the proposed system. Future work and conclusion are discussed in Chapters 5 and 6, respectively.

2. BACKGROUND AND LITERATURE REVIEW

The various approaches to the NOMA system have been proposed in recent years as potential multiple access techniques for 5G and beyond. Unlike traditional orthogonal multiple access (OMA) systems, NOMA can serve many users while maintaining the same degree of freedom (DOF) [10] and achieving spectral efficiency. In particular, it has been investigated that NOMA can achieve a greater ergodic sum rate in a single-input single-output scenario than the OMA scheme by using a fixed resource allocation strategy [11]. The idea of NOMA has been applied to multiple-input multiple-output systems, known as MIMO-NOMA, to achieve higher spectral efficiency. The author of [10] investigates the performance gain of NOMA over OMA in uplink single-cell systems in both single-antenna and multi-antenna scenarios. The asymptotic ergodic sum rate is analyzed for a sufficiently large number of users for both schemes.

In recent years, applying deep learning to physical layer communication has been proven to achieve high performance in many ways. There are several attempts where DL is used to operate specific parts of the physical layer communication and as well as to operate in the end-to-end communication form. Some works related to deep learning-based NOMA systems will be discussed in the literature review in this chapter.

In contrast to conventional ML algorithms, a deep learning algorithm can predict, classify, or make decisions based on data without being explicitly programmed. DL optimizes a given training target by using multi-layer structures and nonlinear processing units from raw input data [12]. DL can help to achieve better performance over physical layer communication than conventional ML algorithms. As with ML algorithms, deep learning has essential characteristics, such as deep modularization, that significantly enhance the ability to extract features and adapt the structure [13]. Some well-known DL models are the deep neural network (DNN), convolutional neural network (CNN), and generative adversarial network (GAN). Deep learning employs a deep neural network (DNN) to investigate data representation in each layer.

Firstly, we discuss how deep learning can be a possible approach to the challenges of physical layer communication systems, deep neural network basics, some essential deep learning libraries, activation functions, and loss functions. We then present deep learning approaches in physical layer communication. Finally, we discuss the overviews of background literature that have proposed a deep learning approach to successive interference cancellation in the MIMO-NOMA system.

2.1 Deep Learning as a Potential Approach to the Challenges of Physical Layer Communication

- Most signal processing algorithms in traditional communication systems are based on statistics and information theory and are frequently provably optimal for tractable mathematical models. These are typically linear, stationary, and have Gaussian statistics [14]. On the other hand, a practical system may have imperfections and non-linearities that are difficult to represent analytically [6]. As a result, systems or algorithms that can accomplish communication tasks in the absence of defined channel models are critical. Despite complex channel conditions, deep neural networks have been approved to be a universal function approximator [15] with excellent algorithmic learning ability [13].

They can optimize end-to-end performance using simple training approaches rather than having well-defined models. Therefore, a DL approach communication system that does not require a mathematically tractable model could be the solution to the challenges of the current communication system, which heavily depends on mathematical models.

- Since traditional MIMO data detection algorithms are iterative reconstruction procedures [16] and could result in a computational bottleneck in real-time, upgraded systems such as massive MIMO and mmWave require real-time huge data processing capabilities. As a result, these systems need parallel signal processing architecture to achieve efficiency and accuracy [11].

  On the other hand, NNs can be substantially parallelized on concurrent architectures and easily implemented with low-precision data types [17], "learned" algorithms could be executed quicker with less energy cost than their manual equivalents [14]. Parallel processing architectures with distributed memory such as graphics processing units (GPUs) and specialized chips for NN inferences have proven to be very energy efficient and capable of providing significant computational throughput when fully applied in concurrent algorithms [14]. GPUs typically have thousands of cores and excel at quick matrix multiplications, which are critical for training neural networks.

- Mobile networks generate large amounts of various sorts of data at a rapid rate [12]. Handling massive amounts of data is a key feature of deep learning because the inherent nature of distributed and parallel computing architectures provides computation speed and processing capacity [13].

- Traditional communication systems have a connecting architecture of multiple processing blocks (modulation, demodulation, channel estimation, etc.). Figure 2.1 depicts the block diagram of a conventional communications system, which includes source coding and decoding, channel coding and decoding, modulation, demodulation, channel estimation, and signal detection. These are designed and optimized locally within each block. Furthermore, the structure of the communication system could vary depending on the environment to achieve optimality. Global optimality cannot be ensured [6]. The DL approach communication systems do not rely on the artificial block structure to achieve global optimality since they can optimize end-to-end performance without demanding individual processing blocks for specific tasks of the communication system.

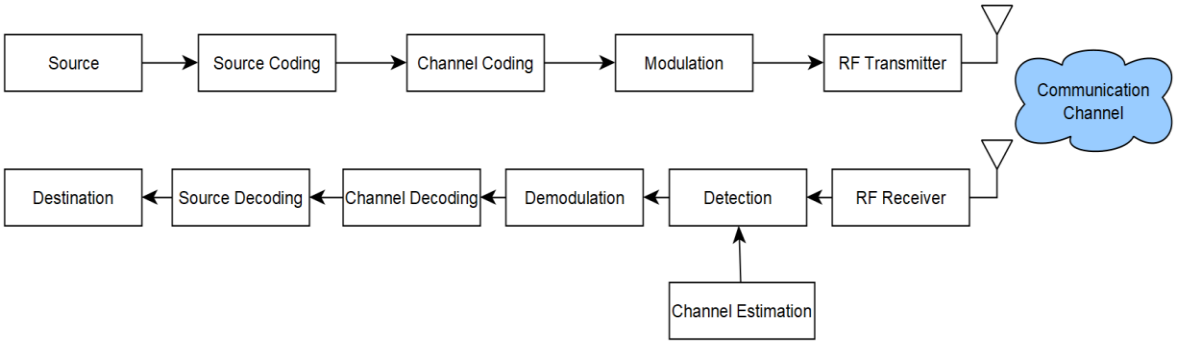
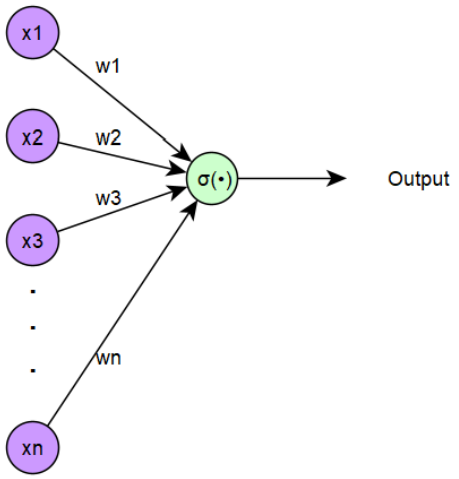


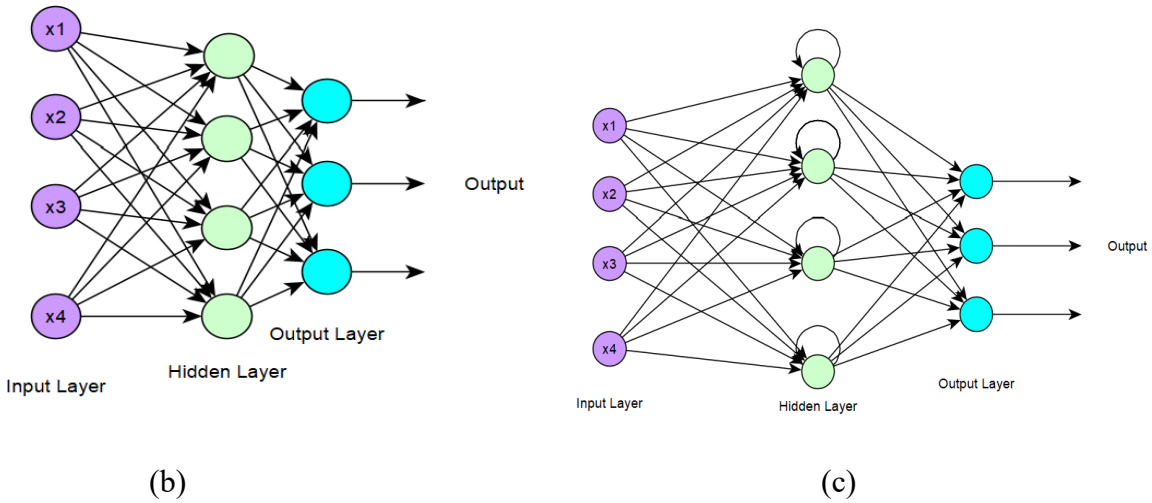
Figure 2.1. Block diagram of a typical communication system.

2.2 Deep Neural Network Basics

The concept of using neural networks (NNs) to control machines intelligently dates back to 1942, when an early neural network model was developed to simulate the status of a single neuron. Figure 2.2 shows the representation of a single neuron, a feedforward neural network (FNN), and a recurrent neural network (RNN).



(a)



(b) (c)

Figure 2.2. (a) A single neuron, (b) Feedforward neural network, and (c) Recurrent neural network.

The NN is generated when the inputs are connected to multiple neurons producing multiple outputs and forming a layered network. A perceptron with one input layer and one output layer can be considered a simple NN. In order for the perceptron to produce a value close to the target value, a loss function must be established, such as square error or cross-entropy. Despite only being used for linearly separable problems, a single perceptron is capable of introducing a nonlinear function as a universal approximator by adding hidden layers and neurons between the input and output layers. This multi-layer perceptron (MLP) can be referred to as NN with multiple hidden layers [13]. Therefore, the basic type of DL models is feedforward neural networks (FNNs) or multi-layer perceptrons (MLPs).

Deep neural networks' main goal is to approximate complex functions using a combination of predetermined unit operations (neurons). For example, if $f(.)$ is considered to be the function of a neural network, the operation of $f(x) \rightarrow y$ is the mapping of an input x to a category y. Depending on the model's structure, the operations are commonly described by a weighted combination of a certain group of hidden units with a nonlinear activation function [12]. These procedures, along with the output units, are referred to as layers. The feedforward network's first layer is known as the input layer, and the last layer is known as the output layer. The layers that exist between the input and output layers are known as hidden layers because their output behavior is unknown from outside of the neural network. An objective function can be nearly any form, such as mapping images to their labels (classification), computing future stock prices based on previous values (regression), or even determining the next ideal chess move according to the current state of the board (control) [12].

We can use supervised, unsupervised, reinforcement learning, and other machine learning approaches to build a deep neural network. Deep learning architectures have performed admirably in all these areas. The difference between supervised learning and unsupervised learning is that supervised learning has trained the model with prior knowledge of what the output for the fed input should be. In other words, supervised learning uses the labeled data sets,

the pair of the input and the expected output, while unsupervised learning does not use the labeled data sets. Supervised learning is typically used for classification tasks when the input is mapped to the output labels. On the other hand, in reinforcement learning (RL), the learner has to discover which action yields the most reward with trial and error. The two significant features of RL are the trial and error search and delayed rewards [18].

The two types of DNN are feedforward neural network (FNN) and recurrent neural network (RNN). Each neuron in FNN is connected to the adjacent layers but not to the neurons within each layer. On the other hand, both the current inputs and hidden states from the previous time step are used to determine the outputs of RNN layers. For the applications such as natural language processing (NLP), RNN tries to provide NNs with memory because the outputs depend on the current inputs and previously accessible information [13].

Because RNN is time-dependent, non-stationary errors may appear throughout the training process. Short-term memory is an issue for RNN. A specific type of RNN known as long short-term memory (LSTM) has also been developed to reduce certain unwanted information in the network. The bidirectional RNN and LSTM are some examples of regularly used RNNs [6].

Another DL-based model is a convolutional neural network (CNN) derived from a fully connected feedforward network. CNN introduces the concept of designing DNN architectures based on the needs of specific scenarios. The idea of CNN is adding convolutional and pooling layers before feeding into a fully connected network, as shown in Figure 2.3 [13]. It uses a number of locally connected kernels, also called filters, to capture correlations between data regions. It allows for preventing parameter growth mainly when applied to image applications. For example, the input may comprise thousands or millions of pixels when processing an image. However, small important characteristics such as edges with kernels that occupy just tens or hundreds of pixels may need to detect. Through this, the model manages to keep fewer parameters, decreasing the model's memory requirements and improving its statistical efficiency [19].

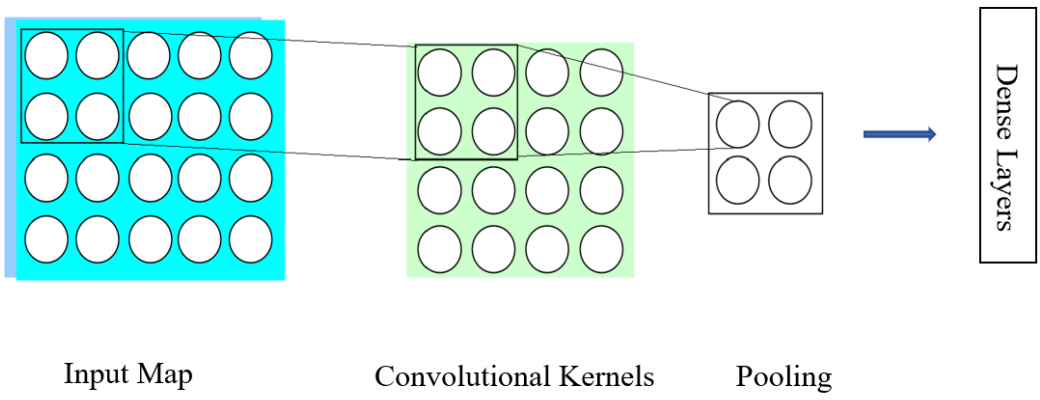


Figure 2.3. Convolutional neural network.

2.2.1 Training a Neural Network

Consider a feedforward neural network (NN) with n inputs, one hidden layer with m neurons $\{e_1, e_2, \dots, e_m\}$ and an output layer with q neurons $\{o_1, o_2, \dots, o_q\}$. In the hidden layer, there are n weights in each neuron, one for each input. The bias value of the m neurons can be described as $B = [b_1, b_2, \dots, b_m]^T \in \mathbb{R}^{m\times 1}$. The inputs of the NN can be denoted as $A = [a_1, a_2, \dots, a_n]^T \in R^{n\times 1}$ and the weight matrix between A and m numbers of neurons can be expressed as

$$W = \begin{bmatrix} \omega_1 \\ \vdots \\ \omega_m \end{bmatrix} \in \mathbb{R}^{m\times n}, \tag{2.1}$$

where ω_m is the weight vector of the m^{th} neuron for n inputs and can be expressed as $\omega_m \in \mathbb{R}^{1\times n}$. Each input is multiplied by the corresponding weight of each neuron, and these input and weight multiplication pairs are summed up. The neurons can be expressed as nonlinear activation functions, $s(.)$. The output of the m^{th} neuron can be calculated as,

$$\hat{y}_m = s(\omega_m A + b_m). \tag{2.2}$$

The output vector of the hidden layer can be described as $\hat{Y} = [\hat{y}_1, \hat{y}_2, \dots, \hat{y}_m]^T \in \mathbb{R}^{m\times 1}$ and they are also the inputs for each neuron of the output layer. The weight matrix between the $\hat{Y}$ and q neurons can be expressed as,

$$\widehat{W} = \begin{bmatrix} \widehat{\omega}_1 \\ \vdots \\ \widehat{\omega}_q \end{bmatrix} \in \mathbb{R}^{q\times m}, \tag{2.3}$$

where $\widehat{\omega}_q$ is the weight vector of the q^{th} neuron for m inputs and can be expressed as $\widehat{\omega}_q \in \mathbb{R}^{1\times m}$. The bias value of the q neurons can be described as $\hat{B} = [\hat{b}_1, \hat{b}_2, \dots, \hat{b}_q]^T \in \mathbb{R}^{q\times 1}$. Finally, the output layer can be represented as $\hat{O} = \left[\hat{o}_1, \hat{o}_2, \dots, \hat{o}_q\right]^T \in \mathbb{R}^{q\times 1}$ and the output from the q^{th} neuron can be calculated as

$$\hat{o}_q = s(\widehat{\omega}_q \hat{Y} + \hat{b}_q). \tag{2.4}$$

Figure 2.4 shows a simple neural network with three inputs, $n = 3$, two neurons in the hidden layer, $m = 2$, and a single neuron in the output layer, $q = 1$.

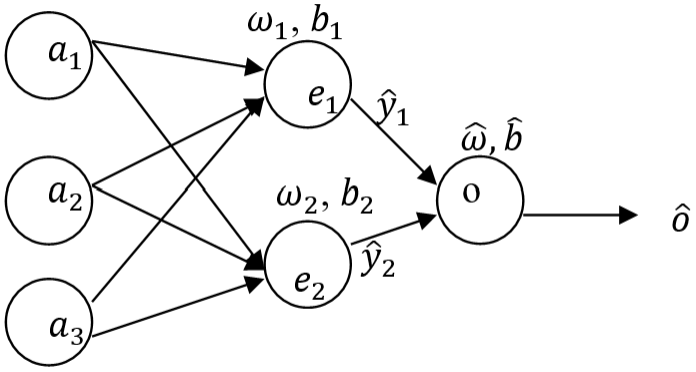


Figure 2.4. A simple FNN.

The depth of the neural network corresponds to the number of iterative operations done on input data via the transfer functions of sequential layers. The width of the neural network is directly related to the memory required by each layer and also refers to the number of output activations per layer or the average across all layers [14]. The labeled data is used during training to update the weight sets properly. A loss function is defined in the training process to measure the gap between the predicted output and the target output. The goal of training the neural network is to achieve the weight sets by which the neural networks can give as much accurate output prediction as possible. The most commonly used loss functions are mean-squared error (MSE) and categorical cross-entropy. The optimization algorithm minimizes the loss function and optimizes the desired weight sets. Some popular optimization algorithms are stochastic gradient descent (SGD), SGD with momentum, and ADAM, derived from adaptive moment estimation.

ADAM is the optimization algorithm that we mainly use in this thesis. The method is developed for efficient stochastic optimization by combining two prior optimation algorithms, AdaGrad and RMSProp. It just requires first-order gradients and consumes little memory. With ADAM, the individual adaptive learning rates for different parameters are computed using estimations of the gradients' first and second moments [20].

2.3 Deep Learning Libraries

Developing a DL model from scratch is not an easy task. It necessitates forwarding behaviors, gradient propagation procedures at each layer, and CUDA coding for GPU parallelization. In recent years, deep learning (DL) has grown in popularity, with applications ranging from image, video, and speech recognition to natural language processing (NLP). This increasing popularity and use case of DL has led to the development of several tools, algorithms, and dedicated libraries that make it simple to design and train large NNs. TensorFlow [21], Pytorch [22], and Caffee [23] are among such tools, which enable high-level algorithm definition in a variety of programming languages or configuration files, automatic differentiation of training loss functions across arbitrarily large networks, and compilation of the network's

forward and backward passes into hardware optimized concurrent dense matrix algebra kernels. [12]. Some of the popular DL libraries are summarized below.

- Tensorflow
  The google brain team research organization initially introduced Tensorflow. This open-source library supports numerical computation and large-scale machine learning that operates in heterogeneous environments. It can be used to interpret many algorithms, including training and inference algorithms for deep neural networks. It has been applied to various applications across computer science and other areas, such as speech recognition, computer vision, robotics, information retrieval, natural language processing, geographic information extraction, and computational drug discovery [21]. TensorFlow supports several languages to construct and execute a TensorFlow graph. While Python's application programming interface (API) is currently the most complete and easiest to use, there are some languages that Tensorflow API supports, such as C++, Java, and Javascript.
- Keras
  Keras [24] is built upon the TensorFlow platform, which provides higher-level programming interfaces. It is a simple, user-friendly, and highly productive interface for solving machine learning problems, especially for modern deep learning environments. It can efficiently execute low-level tensor operations on CPU, GPU, or TPU [24].
- Pytorch
  PyTorch is an open-source library built on the platform of Torch. Developers originally developed the program in Lua but later improved it in Python [22]. PyTorch is known for achieving high execution speed even when working with large and complex graphs. Aside from CPUs and GPUs, it is also highly flexible, allowing it to run on simplified processors. PyTorch is becoming more popular thanks to its straightforward way of building neural networks.
- Caffe
  Caffe [23] is a dedicated open-source deep-learning framework developed by Berkeley artificial intelligent research (BAIR). It supports deep learning implementations on mobile operating systems, including iOS and Android, and the training of neural networks on multiple GPUs within distributed systems [12].

2.4 Activation Functions

An activation function is applied to specify the transformation of the weighted sum of the input into an output from a node or neuron in a network layer. There are the nonlinear and linear activation functions. Nonlinear activation functions are the most commonly used activation functions. The choice of activation function significantly impacts the neural network's capability and performance, and different stages of deep learning models may require different activation functions. The followings some popular nonlinear activation functions are described.

The sigmoid function is one of the commonly used activation functions. The output curve of the sigmoid seems to look like an S-shape and exists between (0 and 1). It can be expressed mathematically as

$$Sigmoid(x) = \frac{1}{(1+e^{-x})}. \tag{2.5}$$

Tanh is similar to the sigmoid activation function. It also produces an S-shaped output curve, but the Tanh function ranges from -1 to 1. It can be written as

$$tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}. \tag{2.6}$$

The Rectified Linear Unit (ReLU) is often used in deep learning algorithms as an activation function. When a negative input is provided, the function returns 0. If a positive input is provided, the function returns that value. It can be expressed as

$$\begin{array}{l} 0 \;\; if \; x \leq 0, \text{ and} \\ x \;\; if \; x > 0. \end{array} \tag{2.7}$$

The exponential linear unit (ELU) is a variant of ReLU that yields a better output for $x < 0$. It can be expressed as

$$\begin{array}{ll} \beta(e^{x}-1) & if \; x \leq 0, \text{ and} \\ x & if \; x > 0, \end{array} \tag{2.8}$$

where β is a positive value and controls the value at which an ELU saturates when the inputs are negative.

SELUs stand for the Scaled Exponential Linear Units [25] and can be expressed mathematically

$$\begin{array}{ll} s\beta(e^{x}-1) & if \; x \leq 0, \text{ and} \\ sx & if \; x > 0, \end{array} \tag{2.9}$$

where the parameters $s = 1.0507$ and $\beta = 1.6733$ are frequently used [12].

The softmax activation is often used as the output layer in classification networks since the result can be viewed as a probability distribution. The softmax function for the input vector with n elements (n category) can be expressed as

$$Softmax(x)_i = \frac{e^{x_i}}{\sum_{i=1}^{n} e^{x_i}}. \tag{2.10}$$

2.5 Loss Functions

The objective function has to be minimized or maximized according to the model's target. When we minimize it, we may refer to it as the cost function, loss function, or error function [26]. The loss function will output a large number when the model approximates improperly. If the model predicts a proper approximation, it will result in a lower number. Since we investigate our model with various approaches or algorithms, the output of the loss function can tell us if we are getting closer to our desired outcome or not. The loss function learns to minimize the prediction error during training using some optimization function.

Depending on the type of learning, loss functions can be divided into two major categories: regression losses and classification losses. Regression is the process of forecasting a specific value that is continuous, for example, estimating housing prices and stock prices. On the other hand, we attempt to predict output from a set of finite categorical values in classification. The classification includes dividing the dataset into distinct classes based on the different parameters in such a way that previously unknown data can be assigned to one of the classes. The following are some commonly used loss functions.

- Mean Square Error (MSE)
  MSE is one of the most commonly used loss functions because it is simple and easy to implement. To get the mean squared error (MSE), first, take the difference between the model's approximation '$\hat{s}$' and the target value 's' and then calculate the mean of the square of that difference. For the *n* training examples, MSE can be expressed as

$$MSE = \frac{\sum_{i=1}^{n}(s_i - \hat{s}_i)^2}{n}. \tag{2.11}$$

- Binary Cross-entropy Loss or Log Loss
  The commonly used loss function for binary classification problems is binary cross-entropy loss, also known as log loss. Binary classification is a problem in which we must categorize observations into one of two labels based on their features. For example, assume we have some images of clothes and need to organize them into two classes, one for pants and one for tops. It can be interpreted as

$$Log\ Loss = \frac{-1}{n}\sum_{i=1}^{n} s_i\, log(\hat{s}_i) + (1 - s_i)log(1 - \hat{s}_i). \tag{2.12}$$

- Categorical Cross-entropy Loss
  Categorical cross-entropy is a loss function used in multi-class classification tasks. It can be written as

$$Loss = -\sum_i s_i\, log(\hat{s}_i). \tag{2.13}$$

2.6 Literature Reviews

As previously mentioned, the essential requirement of a communications system is the reliable delivery of a message from source to destination through a communications channel. In practice, a traditional communications system is built as a block structure. This section discusses some literature on DL application to physical layer communication.

2.6.1 Deep learning Based Joint Channel Estimation and Signal Detection

In recent years, deep learning (DL) has emerged as an efficient channel estimation approach in wireless communication systems. Conventionally, channel estimation and signal detection are independent activities at the receiver. Before transmitted symbols are detected, the channel state

information (CSI)) is estimated via pilot transmission. The transmitted symbols can then be recovered at the receiver using the estimated CSI. The paper [26] presents a DL-based strategy for joint channel estimation and signal identification in orthogonal frequency-division multiplexing (OFDM) systems, which use DL to handle wireless OFDM channels end-to-end. A deep learning model is trained offline using the received signals corresponding to the transmitted data and pilots. The trained model is then utilized to decode the online transmitted data directly without explicitly doing channel estimation. The proposed method is proven to be more robust than the traditional methods when fewer pilots are utilized, the cyclic prefix (CP) is omitted, and nonlinear clipping noise is present.

The authors in [27] investigate and compare the performance of DL-based channel estimation to that of conventional methodologies such as least-squares (LS) and linear minimum mean-squared error (LMMSE) estimators. This paper offers a theoretical investigation of DL-based channel estimation for single-input multiple-output (SIMO) systems. Since DNN with ReLU activation function is mathematically similar to a piecewise linear function, the DL estimator can efficiently achieve universal approximation to a large family of functions by utilizing piecewise linearity. The other DL approach channel estimation method can be found in [28]. The authors of [28] developed the MMSE channel estimator for conditionally normal channel models. After that, the CNN estimators are defined using the framework of this MMSE estimator.

2.6.2 Deep Learning Approach MIMO-NOMA System

Current NOMA systems suffer from limitations: high computing complexity and significant changes in wireless channel conditions make studying the channel characteristics and defining the optimal allocation strategy extremely challenging. To overcome this fundamental limitation, novel deep learning (DL)-aided NOMA system is proposed in [29], in which a single base station serves several NOMA users with random deployment. The DL approach method is used to learn the NOMA systems in completely unknown channels in an end-to-end manner. A long short-term memory (LSTM) network based on DL is included in a standard NOMA system, allowing the proposed technique to detect channel properties automatically. In particular, the DL-aided NOMA system trains and tests the proposed NOMA system framework for automatic encoding, decoding, and channel detection in an additive white gaussian noise (AWGN) channel. Simulation results with the proposed scheme are more robust and efficient than those with the conventional approach.

Faster than Nyquist (FTN) and non-orthogonal multiple access (NOMA) have been identified as potential methods for achieving better spectral efficiency and massive connectivity. In [30], the author proposed a DL-aided novel sliding-window detection method for FTN NOMA. The proposed detector achieves higher accuracy than the minimum mean squared error-frequency domain equalization (MMSE-FDE).

The paper [3] presents DL-based SIC decoding for downlink MIMO-NOMA. The MIMO-NOMA system's precoding and SIC decoding are jointly optimized (or trained) in this approach to minimize the total mean square error of the users' signals. Deep neural networks are applied to build the precoder and SIC decoders so that transmitted signals for multiple downlink users can be properly precoded at the transmitter using superposition coding, and each user's signal is extracted using SIC decoding. Except for the first step, two DNNs are employed at each SIC step, one for reconstructing the previously decoded signal and the other for decoding the current

user's signal. One DNN is required to decode the users' signal since there is no previously decoded user.

The authors of [31] developed a DL technique for detecting downlink signals in MIMO-NOMA. A deep neural network (DNN) is employed to work as the entire SIC receiver and decode the transmitted signals for all downlink users in a single slot. The DNN's input is the received signal of each received antenna. The typical DNN output layer is fully connected and uses the softmax function as an output layer with one-hot encoding. However, signals from multiple transmitting antennas should be decoded in a single slot to detect MIMO-NOMA signals. As a result, the proposed output layer was created to construct groups. The number of groups corresponds to the number of transmitting antennas, and the number of neurons of each group corresponds to the number of one-hot encodings. The effect of power allocation and modulation type on the performance of the proposed strategy was investigated.

In [4], convolutional neural networks (CNNs) are used to reconstruct the desired signal of uplink users in the MIMO channel. The proposed scheme can instantaneously decode multiple users' information in a cluster without any conventional communication signal processing steps. In [9], a DNN with fully connected layers is employed at each SIC step to decode a single user's data. Aside from the input and output layers, DNN has two hidden layers. DNN's output layer generates the decoded bits of the corresponding user. Except for the initial SIC step, the input layer is fed by the received combined signal and the previously decoded signals of all users. DNNs are trained to map the received signal to the bit sequences emitted. Since we mainly refer to the proposed scheme [9] to extend the DL-based SIC systems for higher order modulation, the detailed implementation and discussions can be seen in Chapter 3.

3. DEEP LEARNING-BASED SIC WITH QPSK MODULATION

As discussed in Chapter 2, the deep learning approach to MIMO-NOMA systems has had significant interest in recent years due to its promising performance. When investigating the deep learning-based SIC, the deep neural network is applied to operate such communication processes as estimation of the channel, detection, decoding of the signal, and removal of the decoded signal. We extended the existing deep learning-based SIC for higher-order modulation and compared it with the conventional MIMO NOMA system in the Rayleigh fading channel.

In this chapter, we have implemented deep learning-based SIC for the uplink MIMO-NOMA system with QPSK modulation in the Rayleigh fading channel proposed by [9] with a slight modification. Firstly, we introduce the concept of deep learning-based SIC by formulating the system model in Section 3.1. The implementation of DNN for the SIC receiver has been carried out in Section 3.2. The discussion of the test and results is in the following Section 3.3. We compare the results with conventional NOMA SIC with QPSK modulation. For result analysis, we use the BER metric.

3.1 System Model

We consider the single-cell uplink MIMO-NOMA system, which includes M receiving antenna and K users, as shown in Figure 3.1. The figure shows that the users transmit to the base station using the same frequency resources with different transmission powers.

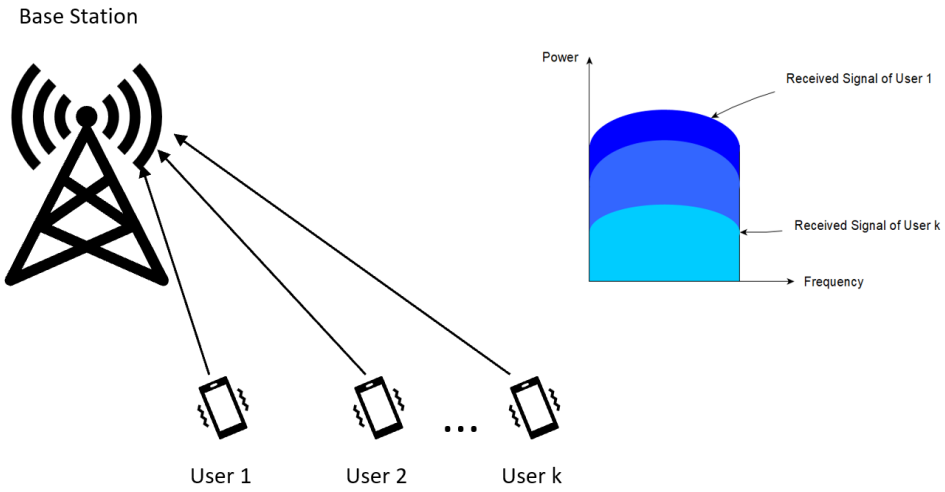


Figure 3.1. Single-cell NOMA uplink system.

As inspired by the analysis in [32], [9], the users transmit the multiple frames, which combine the pilot and data symbols, at a consecutive time. Transmission of each frame is during the coherent time interval, so the channel impulse response over a single frame is considered unchanged. In particular, each frame contains J pilot symbols and N data symbols. We label

symbols with subscripts L as pilot symbols and D as data symbols. At BS, the receiving data signals of all users at any frame can be represented by

$$\boldsymbol{Y^D} = \boldsymbol{HVX^D} + \boldsymbol{G^D}, \tag{3.1}$$

where $\boldsymbol{X^D}$ denotes the data matrix and can be expressed as $\boldsymbol{X^D} = [\, X_1^D, X_2^D, \dots, X_K^D]^T \in \mathbb{C}^{K\times N}$. X_k^D denotes the data vector of user k's modulated symbol and can be represented as $X_k^D = \left[\, X_{k,1}^D, X_{k,2}^D, \dots, X_{k,N}^D\right] \in \mathbb{C}^{1\times N}$ and $E\left(\left|X_{k,n}^D\right|^2\right) = 1$. $\boldsymbol{H}$ is the channel matrix between all the K users, and M receive antennas of BS and is denoted as $\boldsymbol{H} = \left[h_1, h_2, \dots, h_K\right] \in \mathbb{C}^{M\times K}$ where $h_k = \left[h_{k,1}, h_{k,2}, \dots, h_{k,M}\right] \in \mathbb{C}^{M\times 1}$ is the channel vector between BS and the k^{th} user. We consider a Rayleigh fading scenario. $\boldsymbol{V}$ is the diagonal matrix and can be expressed as $\boldsymbol{V} = diag(\,\lambda_1, \lambda_2, \dots, \lambda_k)$. Let us assume P as the peak total transmission power of all users, such as $\sum_{k=1}^{K} \lambda_k \le P$. $\boldsymbol{G^D} \sim \mathcal{CN}(0, N_0 I) \in \mathbb{C}^{M\times N}$is additive white gaussian noise at BS for data transmission.

Similarly, the receiving pilot signals of all users at any frame can be represented by

$$\boldsymbol{Y^L} = \boldsymbol{HVX^L} + \boldsymbol{G^L}, \tag{3.2}$$

where $\boldsymbol{X^L}$ denotes the pilot matrix and can be expressed as $\boldsymbol{X^L} = [\, X_1^L, X_2^L, \dots, X_K^L]^T \in \mathbb{C}^{K\times J}$. X_k^L denotes the pilot vector of user k's modulated symbol and can be represented as $X_k^L = \left[\, X_{k,1}^L, X_{k,2}^L, \dots, X_{k,J}^L\right] \in \mathbb{C}^{1\times J}$ and $E\left(\left|X_{k,n}^L\right|^2\right) = 1$. $\boldsymbol{G^L} \in \mathbb{C}^{M\times J}$ is additive white gaussian noise at BS for pilot transmission.

For the power allocation strategy, we consider equal power allocation as in [9]. An equal power allocation strategy is also typically selected for massive machine-type communication (mMTC) [10]. In addition, it does not necessarily require allocating more power to users with the weaker channel. Power allocation strategy depends on the targeted capacity region for users. Therefore, the user with a weaker channel can be allocated power higher than or less than or equal to the users with the more robust channel [33]. Hence, the equal power allocation factor for each user can be represented as $\lambda_k = \frac{P}{K}\ \forall k \in \{1,2,\dots,K\}$. The channel coefficients between the users and the BS antenna are considered to be independent and identically distributed. Let us assume the channel gain is in descending order in contrast to the users' index, i.e., $\|h_1\| \ge \|h_2\| \ge \cdots \|h_K\|$.

The signal detection at the receiver can be carried out by recovering the received signal. In a traditional MIMO-NOMA receiver, extracting the desired signal from the received signal follows the steps of channel estimation, detection, and demodulation. Figure 3.2 presents the procedure of how a traditional SIC receiver works. The signals of users will be decoded according to their signal strength. The highest signal strength user will be decoded first, and then the decoded signal is removed from the combined signal. After that, the second-highest signal strength user will be decoded. The operation continues till the user with the lowest signal strength is decoded. The number of SIC operations is based on the number of users.

Channel estimation can be done during pilot transmission by using X_k^L pilot symbols. Successive interference cancellation for the MIMO-NOMA system can be carried out by using a general ZF-SIC and MMSE SIC scheme. The estimated data vector for the k^{th} user decoded by ZF-SIC can be expressed as

$$\hat{X}_k^D = Z_k \boldsymbol{Y^D}, \tag{3.3}$$

where $Z_k = \hat{\mathrm{h}}_k{}^H (\hat{\mathrm{h}}_k \hat{\mathrm{h}}_k{}^H + I)^{-1}$ and $\hat{\mathrm{h}}_k$ is the channel estimation vector of the k^{th} user such that $\hat{\mathrm{h}}_k = \left[\hat{\mathrm{h}}_1, \hat{\mathrm{h}}_2, \dots, \hat{\mathrm{h}}_M\right]^T \in \mathbb{C}^{M\times 1}$. After that, the decoded signal is subtracted from the received signal, and then the next $(k+1)^{th}$ user is decoded. In the same way, the detection for the k^{th} user using MMSE-SIC can be expressed in general form as

$$\hat{X}_k^D = W_k \boldsymbol{Y^D}, \tag{3.4}$$

where $W_k = \hat{\mathrm{h}}_k{}^H (\hat{\mathrm{h}}_k \hat{\mathrm{h}}_k{}^H + \rho^{-1} I)^{-1}$and ρ is the received SNR at the receiver. ρ can be calculated as

$$\rho = \frac{P_r}{N_0}. \tag{3.5}$$

Furthermore, the signal-to-interference-plus-noise ratio (SINR) analysis [34] for the k^{th} user $(k \neq 1)$ can be interpreted as

$$SINR_k = \frac{\lambda_k \rho |h_k|^2}{\sum_{j=1}^{k-1} \lambda_j \rho |h_j|^2 + 1}. \tag{3.6}$$

Meanwhile, the SINR calculation for the first user can be expressed as

$$SINR_1 = \lambda_1 \rho |h_1|^2. \tag{3.7}$$

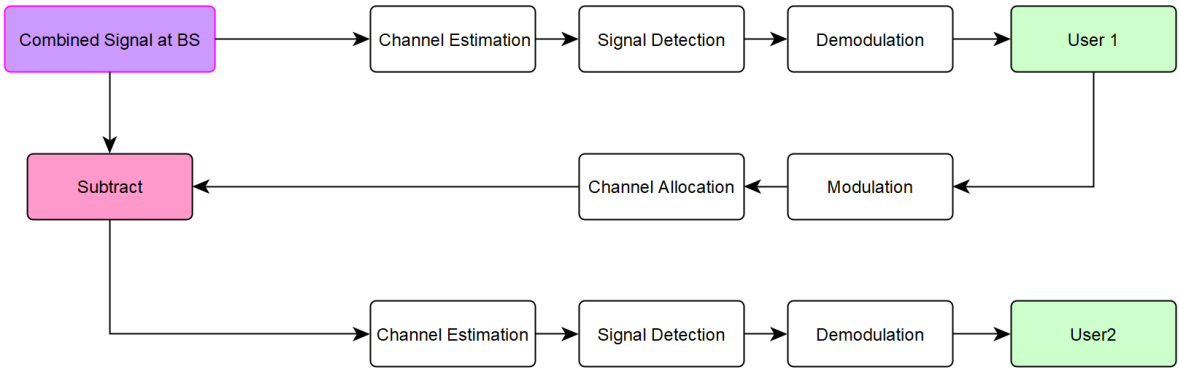


Figure 3.2. Traditional SIC receiver architecture.

3.2 Implementation of DNN for SIC Receiver

As discussed in the previous section, the traditional SIC receiver estimates CSI first according to the corresponding transmitted pilot symbols. Then, the received signal can be

reconstructed by using the estimated CSI. In the DL-based SIC approach, the DNN is trained during pilot transmission and used to recover the transmitted bits directly without explicitly estimating the channel condition or subtracting the decoded signal. In this section, we extend the existing research and implement the DNN design for the MIMO-NOMA SIC proposed by [9].

A DNN with fully connected layers is used at each SIC step to decode a single user's data. DNN is implemented with two hidden layers besides the input and output layers. The output layer of DNN produces the decoded bits of the corresponding user while the input layer is fed by the received combined signal and the previously decoded signal of all users, except for the first SIC step. DNNs are trained to map the received signal to the transmitted bit sequences. Like in the traditional SIC scheme, the user with the highest channel condition will be decoded first. Then, the second strongest user will be decoded at the next SIC step. This way, the operation is done successively until the lowest user is decoded. The output nodes of each DNN are based on the modulation order *C*. For example, the binary phase shift keying (BPSK) modulation needs two nodes, and QPSK modulation requires four.

Figure 3.3 shows the DL-based SIC architecture for two users. The following modulation block after each SIC of DNN is applied to modulate the decoded bits of the corresponding user. The modulated symbols of that user are fed into the input of DNN at the next SIC operation.

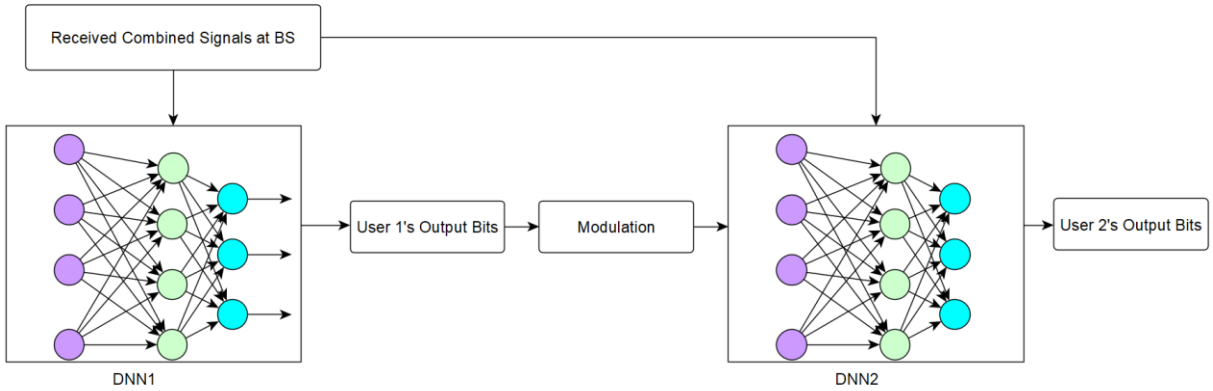


Figure 3.3. Deep learning-based SIC architecture.

DNN has Q fully connected layers, and the output layer of each DNN use the softmax function. If we consider an input vector $x = \{x_1, x_2, \dots, x_c\} \in \mathbb{R}^{C\times 1}$ and then, the output vector $t = \{t_1, t_2, \dots, t_C\}$ can be represented with real values between 0 and 1 that sum to 1 [35]. The Adam optimizer minimizes the categorical cross-entropy loss function between the output value and the training target. The loss function [9], [35] can be expressed as

$$loss = -\sum_{j=1}^{J}\sum_{c=1}^{C} B_{jc} log\ (t_{jc}), \tag{3.8}$$

where B_{jc} is a binary indicator ground truth such that $B_{jc} = 1$ if and only if the j^{th}sample is a member of the c^{th} class. Symbol t_{jc} denotes the softmax's output probability that the j^{th} input belongs to the c^{th} class.

Let us consider $f_k(.)$ as the processing function of the k^{th} DNN. Since all the complex signals of users can be separated into real and imaginary parts, the function of the k^{th} DNN, $f_k(.)$ can be defined as $f_k(.) : \mathbb{R}^{2(\mathrm{M+k-1})} \rightarrow \{0,1\}^{\log_2(\mathrm{C}\times 1)}$ that maps the received combined signal and the previously decoded signal of the $k-1$ users to the transmitted bit of the k^{th}user [9]. The weight and bias matrix of the k^{th} DNN is updated after training pilot symbols and utilized to approximate the function of the k^{th} DNN. Exponential linear units (ELU) and rectified linear units (ReLU) are used as the activation functions of the first and second hidden layers, respectively.

3.3. Test and Results

In this section, experimental simulations are conducted to analyze the implemented DL-SIC performance. Consider an uplink MIMO-NOMA system with M = 2 antennas, $J = 960$ pilot symbols, N = 3840 data symbols, K = 2 users, and P = 1 Watt. All users use QPSK modulation. Model training and testing have been done by using TensorFlow in Python. We have trained the models over 50 epochs using randomly generated 960 pilot symbols. Then, the trained models were tested with 3840 data symbols in the 0 dB to 20 dB SNR range. Each DNN is trained using the SGD method with the Adam optimizer with a learning rate of 0.001. Detailed parameters and settings for numerical analysis are described in Table 1.

Table 1. Detailed parameters and setting for DL-SIC with QPSK modulation.

Parameter	**Value/Type**
Number of users	2
transmit antennas	1
receive antennas	2
Optimizer	ADAM
Learning rate	0.001
Number of symbols	3840
Number of symbols	960
Number of hidden layers	2
Activation function (hidden layer 1)	ELU
Activation function (hidden layer 2)	ReLU
Activation function (Output layer)	Softmax
Number of training epochs	50
Power allocation factor	0.5
Modulation	QPSK
Programming	Python, Matlab
Framework	Tensorflow
Communication channel	Rayleigh fading

For the comparison, we use Matlab to implement the MMSE-SIC for traditional MIMO-NOMA SIC. We assume perfect channel estimation for the MMSE-SIC. All users use QPSK

modulation. Figure 3.4 shows the BER versus SNR comparison of conventional SIC and DL approach SIC for user 1 and user 2. It indicates that the DL-based SIC performs better than the traditional MMSE-SIC across almost all SNR ranges. The BER performances coincide between 0 and 2 dB in both systems. At 10^{-3} BER level user 1 and user 2 with DL-SIC reach approximately 11 dB and 12 dB, respectively. Meanwhile, those with MMSE-SIC get that BER level about at 15 dB and 16 dB. DL-SIC achieved around 4 dB gains compared to MMSE-SIC in both users' cases.

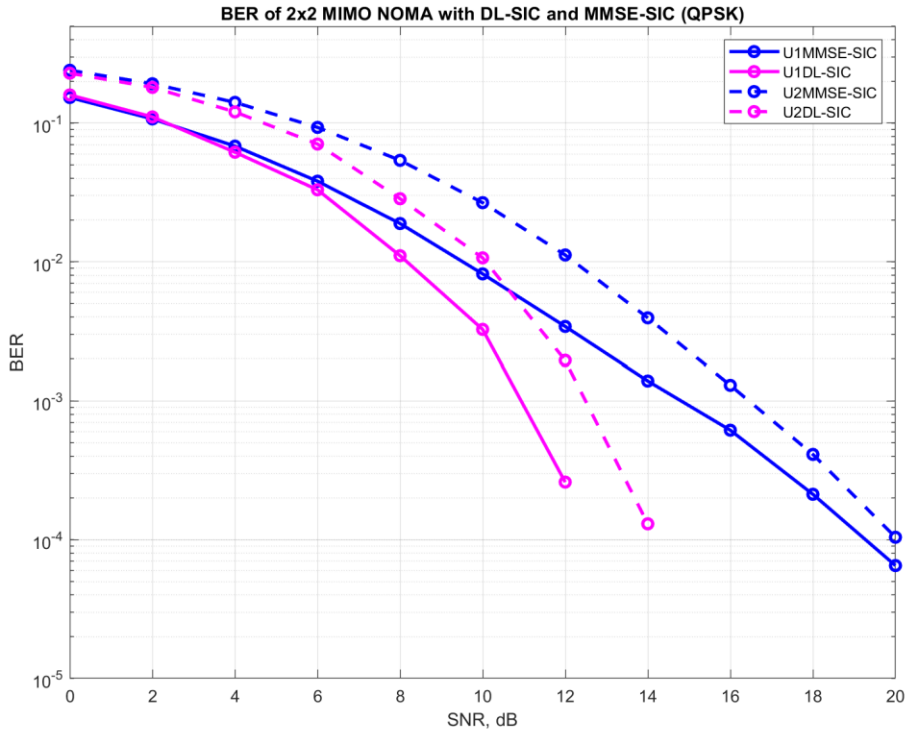


Figure 3.4. BER versus SNR for user 1 and user 2 with DL-SIC and MMSE-SIC (QPSK).

4. DEEP LEARNING-BASED SIC FOR HIGHER-ORDER MODULATION

In this chapter, we investigate the deep learning approach for the MIMO-NOMA system for higher-order modulation. We propose a new approach to decode all the users' signals without depending on the modulation order. As we have implemented in Chapter 3 for exiting DL-based SIC [9], the output of each DNN at every SIC step depends on the modulation order of the transmitting user. Therefore, in this thesis, we implement the new DNN architecture, which can adapt to decode the users with different modulations.

Firstly, we upgrade the DNN architecture inspired by [9] to decode higher-order modulation users. After that, we implement our newly proposed DNN design. In this thesis, we apply 16QAM for higher-order modulation. We also implement ZF-SIC for traditional SIC using the same number of pilot symbols (training symbols) and data symbols (testing symbols) over the Rayleigh fading channel. The BER and total MSE metrics are used to analyze and compare the results of different methods.

This chapter consists of four sections. Following a short introduction in Section 4.1, Section 4.2 implements the DL model for higher-order modulation by upgrading the model with QPSK modulation. In Section 4.3, we propose a different layout for DL-SIC. The proposed system's results, which are trained and tested in the different scenarios, are analyzed and compared with the based-line system ZF-SIC in the following subsections. In Section 4.4, we test our proposed system without feeding the input vectors of the previously decoded symbols to the next SIC steps. The comparisons and discussions are also presented.

4.1 Introduction

As we investigated in Chapter 3, deep learning can help decode NOMA users' signals without necessarily doing channel estimation, demodulation, and subtracting off the previously decoded symbols. In Chapter 3, we implemented the system by assuming both users transmit using QPSK modulation. In this chapter, we propose two DL-SIC schemes for higher-order modulation.

The system model is generally the same as in the previous chapter, where a single-cell MIMO-NOMA with M receiving antenna system serves K uplink users. The users are now capable of transmitting with 16QAM modulation. Training of DNN occurs during pilot transmission, and testing occurs during data transmission. Without loss of generality, we consider the channel gain in descending order in contrast to the users' index, i.e., $\|h_1\| \geq \|h_2\| \geq \cdots \|h_K\|$. The decoding order follows the order of channel gain. The user with the highest channel gain is decoded first.

4.2 DL-SIC for Higher-Order Modulation Using One DNN at Each SIC Step

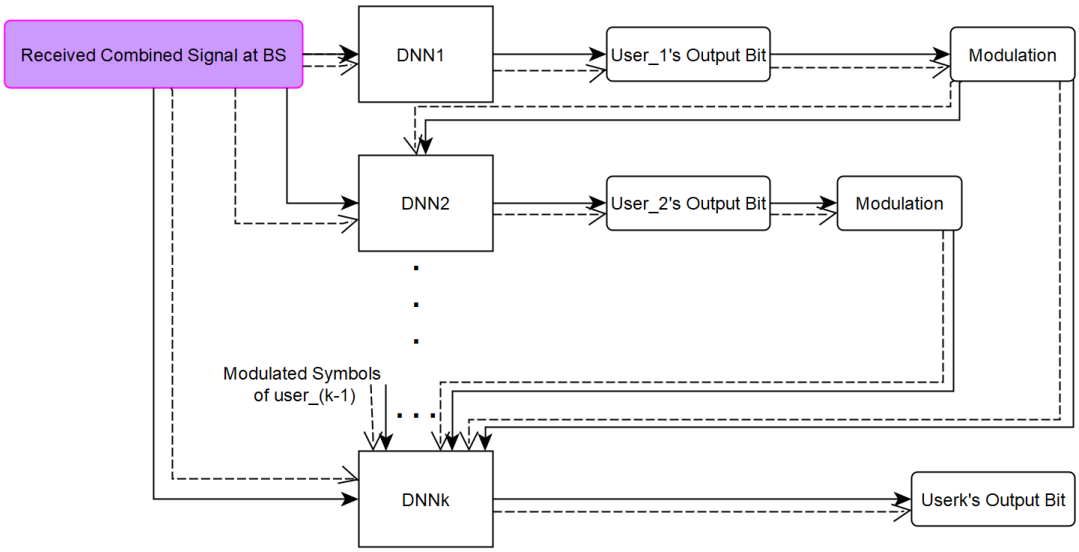


Figure 4.1. DL approach SIC operation for K users during the training and testing phase.

For this proposed method, we upgrade each DNN by adding more hidden layers and more nodes at each hidden layer. Figure 4.1 shows the SIC operation of the K users in both the training and testing phases. The dashed line represents the training phase, and the solid line represents the testing phase. The modulation block at each SIC step is to modulate the decoded bits and feed those modulated symbols as input in the next SIC operation, as we have done in the previous chapter. As shown in the figure, we still use one DNN at every SIC step as in the chapter 3 model.

4.2.1 Implementation

The DNN design we implemented for this proposed scheme includes six layers. These six layers are one input layer, four hidden layers, and one output layer. All the layers are fully connected. At the first SIC step, the input layer receives the combined signals of K users. At the k^{th} SIC step, the input layer receives the combined signals of K users and the previously decoded symbols of all $(k-1)$ users. Since the complex received symbol can be separated into real and imaginary parts, the number of input layer cells of the first DNN is $2M$, and that of the k^{th} DNN is $2(M)+2(k-1)$. Each hidden layer includes 256 nodes. While the first hidden layer uses ELU as an activation function, all other hidden layers apply ReLU as an activation function.

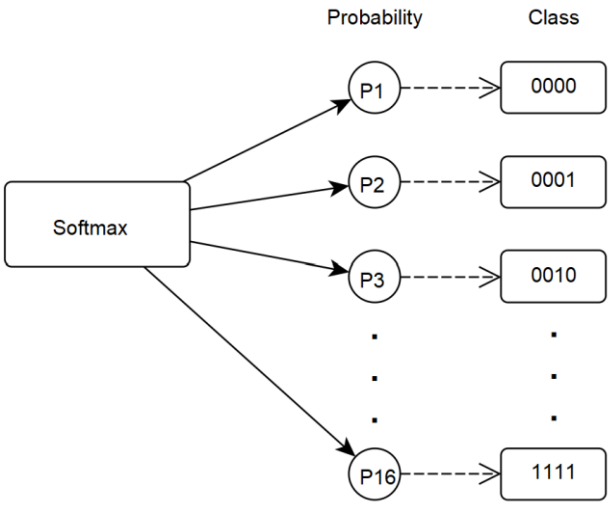


Figure 4.2 The output layer with softmax for 16QAM.

The softmax function as the output layer produces the bits of the decoded user. The number of nodes in the output layer depends on the modulation order, *C*. In our case, sixteen categorical classification needs at the output layer, such as 0000, 0001, 0010 to 1111. All these classes are needed to be one-hot encoded. Figure 4.2 depicts the output layer with the softmax function for sixteen categorical classes, where P_1 to P_{16} represent the output probability of each class. Since the output layer still depends on the modulation order, we have to consider that the users in this system use the same modulation scheme.

4.2.2 Test and Results

In this section, experimental simulations are conducted to analyze the performance of the proposed DL-SIC for higher-order modulation. Consider the number of receive antennas $M = 2$, $J = 1000$ pilot symbols, $N = 100000$ data symbols, $K = 2$ users, and $P = 1$ Watt. We apply equal power allocation for both users. All users use 16QAM modulation. Model training and testing are implemented in Python using TensorFlow. We have trained the models over 100 epochs using pilot symbols. Then, the trained models are tested with data symbols in the 0 dB to 20 dB SNR range. Both pilot and data symbols are randomly generated symbols. Each DNN is trained using the SGD method with the Adam optimizer with a learning rate of 0.001.

We implement the ZF-SIC for Rayleigh's fading channel for the baseline comparison in Matlab. In this case, we use a power allocation factor is 0.8. We use the same number of pilot and data symbols. The detailed parameters and setting for DL-SIC is described in Table 2. Figure 4.3 shows the BER versus SNR comparison of conventional ZF-SIC and DL approach SIC for user 1 and user 2. It can be clearly seen that the proposed DL- SIC for 16QAM has better performance across all SNR ranges. At 10^{-1} BER level user 1 and user 2 with DL-SIC reach approximately 7 dB and 8 dB, respectively. Meanwhile, those with ZF-SIC can get that BER level at about 20 dB and larger. We also produce the total MSE curve over different SNR ranges. Total MSE can be represented as $\sum_{k=1}^{K} E\left\{\left\|X_k^D - \hat{X}_k^D\right\|^2\right\}$ [9].

Table 2. Detailed parameters and setting for DL-SIC with 16QAM.

Parameter	**Value/Type**
Number of users	2
transmit antennas	1
receive antennas	2
Optimizer	ADAM
Learning rate	0.001
Number of data symbols	100000
Number of pilot symbols	1000
Number of hidden layers	4
Activation function (hidden layer 1)	ELU
Activation function (hidden layer 2,3,4)	ReLU
Activation function (output layer)	Softmax
Number of training epochs	100
Power allocation factor	0.5
Modulation	16QAM
Programming	Python, Matlab
Framework	Tensorflow
Communication channel	Rayleigh fading

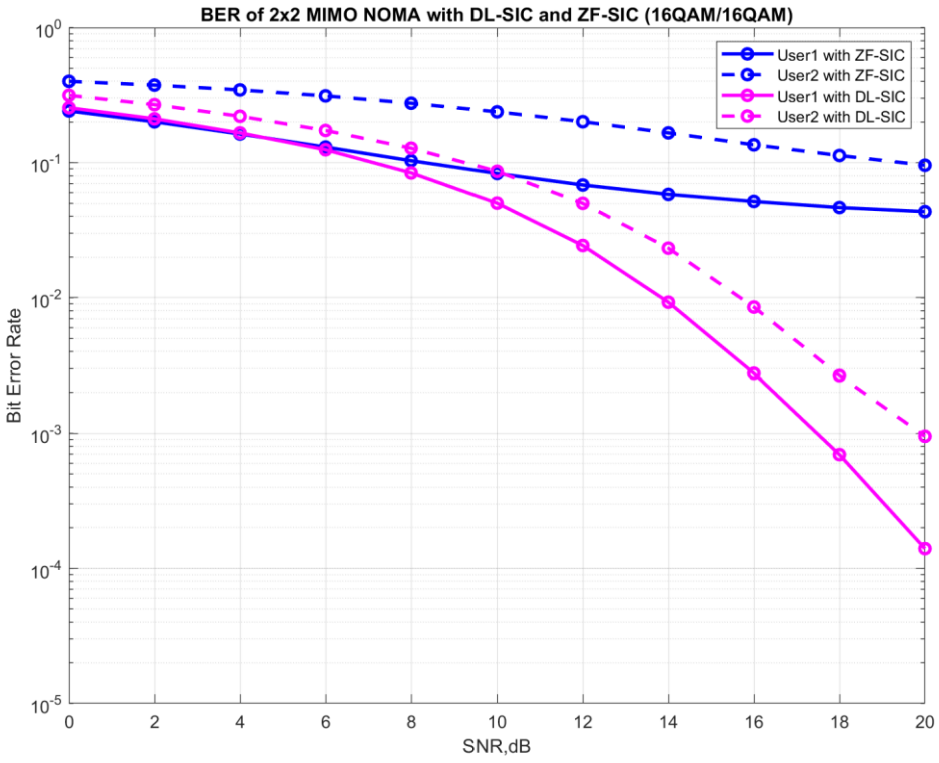


Figure 4.3. BER versus SNR for user 1 and user 2 with DL-SIC and ZF-SIC (16QAM).

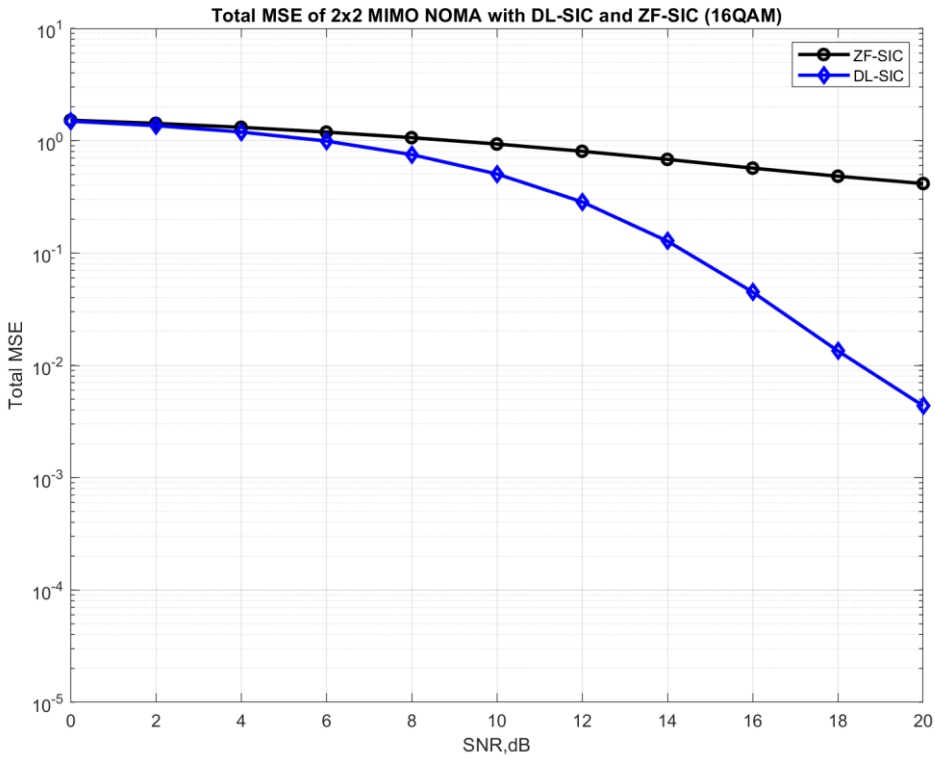


Figure 4.4. Total MSE versus SNR with DL-SIC and ZF-SIC (16QAM).

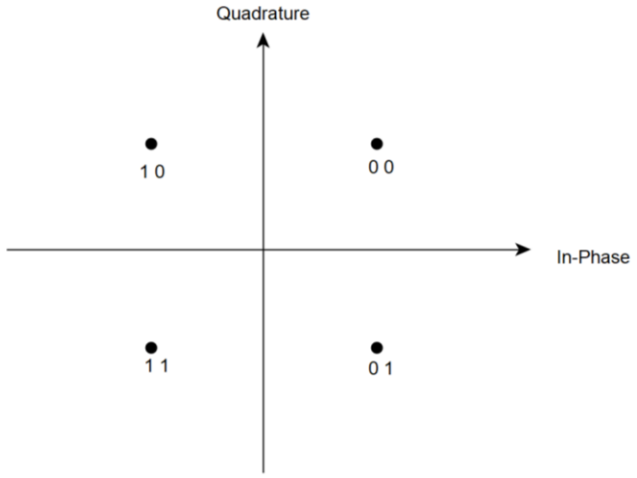
Figure 4.4 represents the total MSE versus SNR with DL-SIC and ZF-SIC methods. The figure shows that the proposed DL scheme and the traditional ZF-SIC have similar performances, around 0 to 2 dB. Overall, the proposed DL-SIC gets a much lower MSE curve almost all over the SNR range.

4.3 DL-SIC for Higher-Order Modulation Using Two DNNs at Each SIC Step

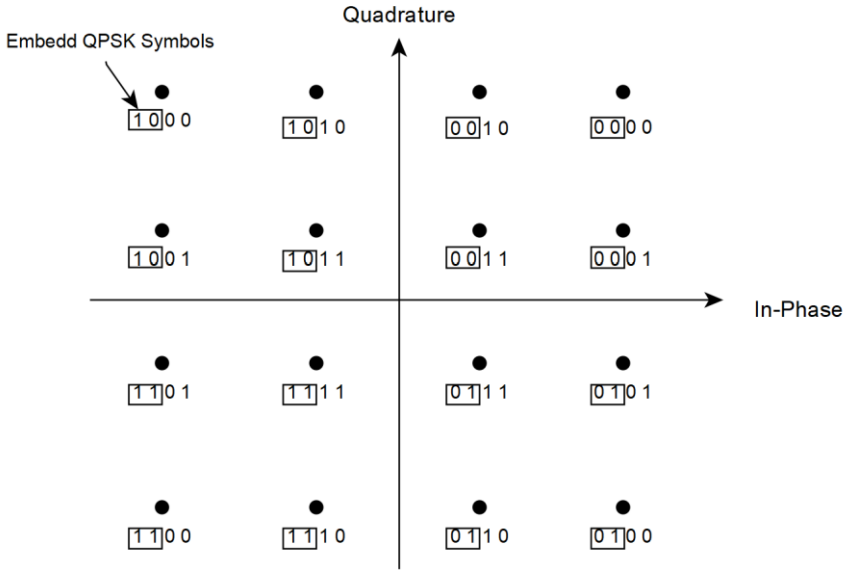
In this section, we propose different DL-based SIC for higher-order modulation. In our proposed DL receiver, we use two DNNs for each SIC step. Generally, we consider the first DNN to recover the first two bits and the second DNN to recover the last two bits of 16QAM. In order to make our proposed design more efficient, we adopt a hierarchical 16QAM modulation scheme. The output of the first DNN could be two bits of the QPSK symbol or the first two bits of 16QAM symbols according to the user's modulation order. If the user uses QPSK modulation, only one DNN is needed to decode the user's information and does not need to use the second DNN. If the user transmits with 16QAM modulation, both DNNs need to be used to recover all the user's information bits.

The hierarchical modulation was initially proposed in [36] to apply in digital broadcasting for HDTV, in which the users receive the information according to their channel capacity. For this thesis, we adopted the hierarchical modulation scheme using QPSK symbols to form 16QAM symbols inspired by [7]. In that research, hierarchical modulation is used in upgrading the digital broadcast receiving system, where the upgraded receiver is applied to receive the secondary

constellation, and the exiting receiver is still able to decode the basic constellation. The authors refer to the QPSK symbols as the basic constellation and the last two bits of hierarchical 16QAM as the secondary constellation. Figure 4.5 depicts the basic QPSK modulation constellation and the combined constellation of 16QAM. As shown in the figure, the embedded QPSK symbols can be found in each symbol of 16QAM within the same quadrant.



(a)



(b)

Figure 4.5. (a) QPSK constellation, (b) Hierarchical 16QAM constellation with embedded QPSK symbols.

4.3.1 Implementation

In this proposed model, two DNNs are implemented for SIC operation. According to the hierarchical nature of modulation, we can see the four bits of 16QAM symbols as two QPSK symbols. Therefore, the DNN model, which can learn the QPSK symbol, can be used to learn the first and second bits of 16QAM symbols. We adopt the same 16QAM constellation for hierarchical modulation as in Figure 4.5 (b). The figure shows that the first two bits of every 16QAM symbol are the QPSK symbol.

Each DNN design we implemented for this proposed scheme includes four layers. These four layers are one input layer, two hidden layers, and one output layer. All the layers are fully connected. At the first SIC step, the input layer receives the combined signals of K users. At k^{th} SIC step, the input layer receives the combined signals of K users and the previously decoded symbols of all $(k-1)$ users. Since the complex received symbol can be separated into real and imaginary parts, the number of input layer cells of DNNs of the first SIC step is $2M$, and that of the k^{th} SIC step is $2(M)+2(k-1)$. Each hidden layer consists of 100 nodes. The first hidden layer uses ELU, and the second one uses ReLU as the activation function.

Figure 4.6 shows the proposed DL-SIC receiver for two users. The dashed line represents the training phase (pilot transmission), and the solid line represents the testing phase (data transmission). The softmax function as the output layer produces the bits of the decoded user. The number of nodes in the output layer is four. This model has four categorical classifications at the output layer, such as 00, 01, 10, and 11. All these classes are needed to be one-hot encoded. According to the user modulation order, the receiver decides whether to use both DNNs or not. If the user transmits using QPSK modulation, only one DNN needs to decode the user's signal such that the output bits of the first DNN is the decoded bits for that user. Otherwise, both DNNs must be used to decode all transmitted bits of the user. In that case, the output bits of the second DNN have to attach to the output bit of the first DNN in order to form the complete 16QAM symbol. Then, the decoded bits are modulated again in order to feed the input layer of the next SIC step together with the received combined signals. In the next section, we will test our proposed system without feeding the input vectors of the previously decoded symbols.

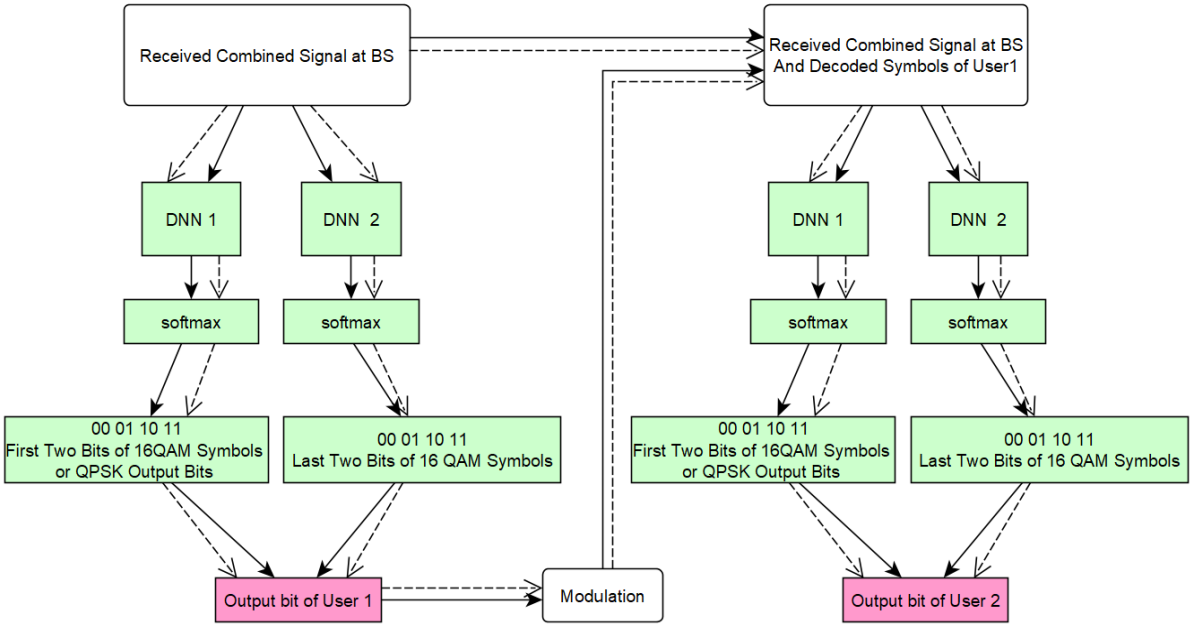


Figure 4.6. Proposed DL-SIC receiver for two users during the training and testing phase.

4.3.2 Test and Results

Experimental simulations are conducted to analyze the proposed DL-SIC model's performance. Consider the number of receive antennas $M = 2, J = 1000$ pilot symbols, $N = 100000$ data symbols, $K = 2$ users, and $P = 1$ Watt. Model training and testing have occurred in Python by using TensorFlow. We have trained the models for 100 epochs using 1000 pilot symbols. Then, the trained models are tested with 100000 data symbols over the 0 dB to 20 dB SNR range. Note that, at every SIC Step, our training target for the first DNN is the labeled encoded first two bits of 16QAM symbols (QPSK symbols), and that for the second DNN is the labeled encoded last two bits of 16QAM symbols. We use the Adam optimizer to minimize the categorical cross-entropy loss between the output and the training target. The learning rate is 0.001.

We have trained and tested the proposed DNN model on several occasions. These are shown as follows.

- Both users use QPSK modulation.
- Both users use 16QAM.
- User 1 uses 16QAM, and user 2 uses QPSK modulation.

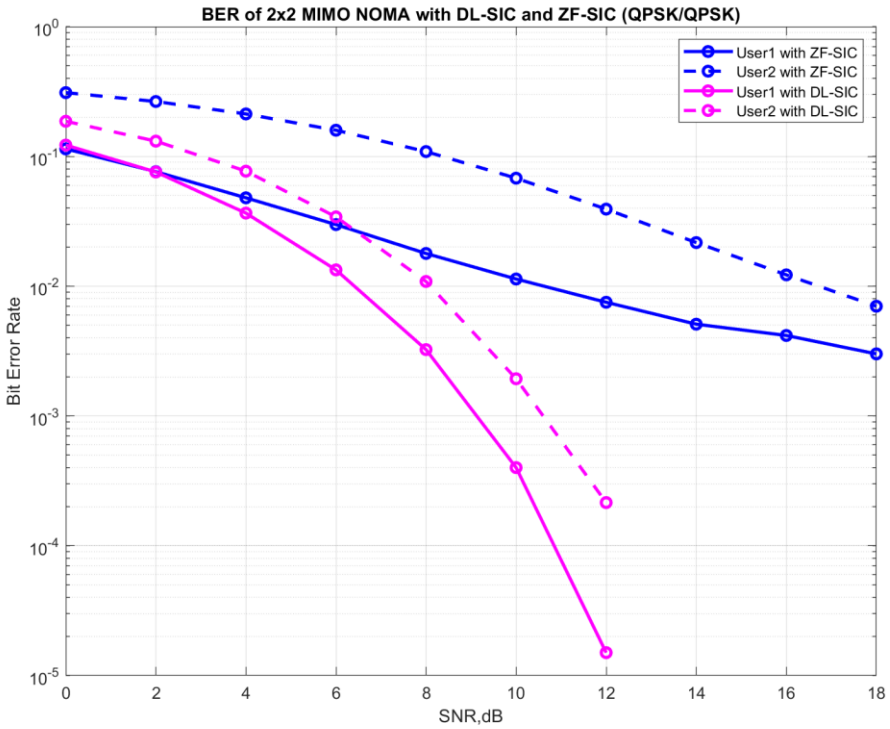


Figure 4.7. BER versus SNR for user 1 and user 2 with DL-SIC and ZF-SIC (QPSK/QPSK).

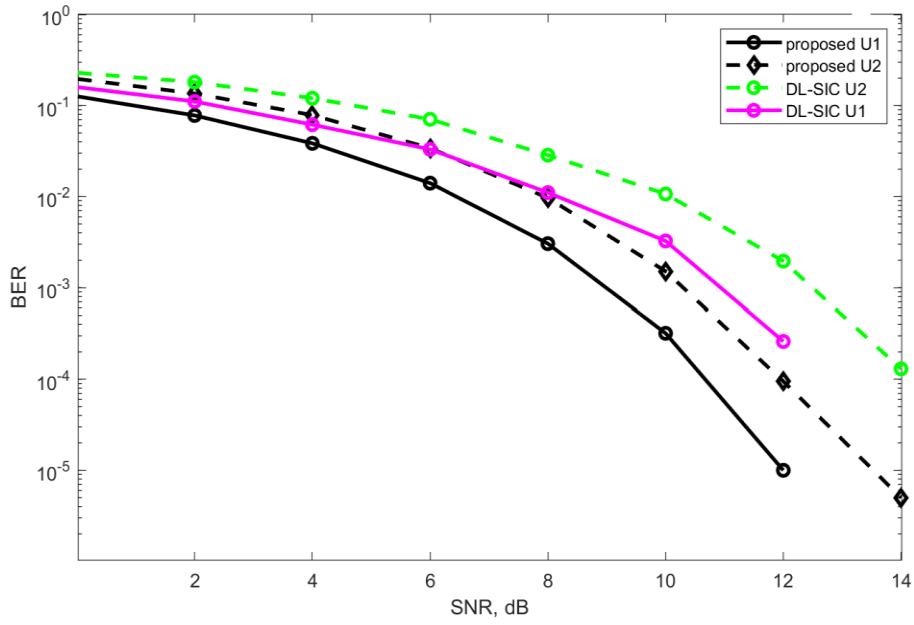


Figure 4.8. Comparison of BER versus SNR for user 1 and user 2 with DL-SIC and proposed method (QPSK/QPSK).

In order to validate the proposed model performance with the model we have implemented in Chapter 3 for QPSK modulation, firstly, we trained the model where both users use QPSK modulation. Figure 4.7 depicts the BER versus SNR comparison of conventional ZF-SIC and DL approach SIC for user 1 and user 2, where both users use QPSK modulation. The resultant BER curve of our proposed scheme has a similar performance to the Chapter 3 model. It indicates that our proposed system can decode the user's signal well by using the first DNN at each SIC step. Figure 4.8 compares the proposed model with the Chapter 3 model. The figure shows that the proposed model has achieved better performance than the model in Chapter 3. It is approved that the proposed DL-SIC can decode the user signal by using one DNN or two according to the user modulation order.

The total MSE performances where both users use QPSK and 16QAM modulation are depicted in Figure 4.9 and Figure 4.11. The proposed DL-SIC achieves the total MSE 10^{-4} at approximately 12 dB when both users transmit QPSK symbols. It results in the total MSE 10^{-4} at approximately 20 dB when both users transmit 16QAM symbols.

Figure 4.10 shows the BER performance over different SNR ranges with both DL-SIC and ZF-SIC for two users. Both users transmit using 16QAM. In this case, as we have designed, two DNNs are used to decode the user's 16QAM symbols at every SIC step. It can be seen that the proposed DL- SIC for 16QAM has better performance across all SNR ranges. At 10^{-1} BER level user 1 and user 2 with DL-SIC reach approximately 8 dB and 9 dB, respectively. Meanwhile, those with ZF-SIC can get that BER level at about 20 dB and larger. The proposed model achieves the BER 10^{-4} for user 1 and user 2 at approximately 18 dB and 20 dB, respectively. It proves that the proposed DNN model can learn the users' signals well all over the SNR range.

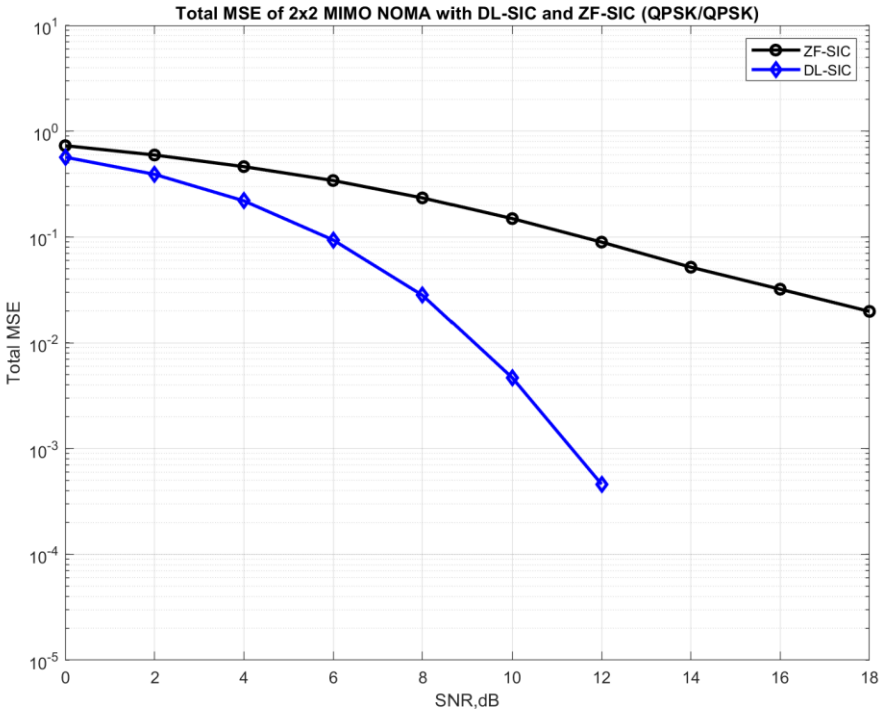


Figure 4.9. Total MSE versus SNR with DL-SIC and ZF-SIC (QPSK/QPSK).

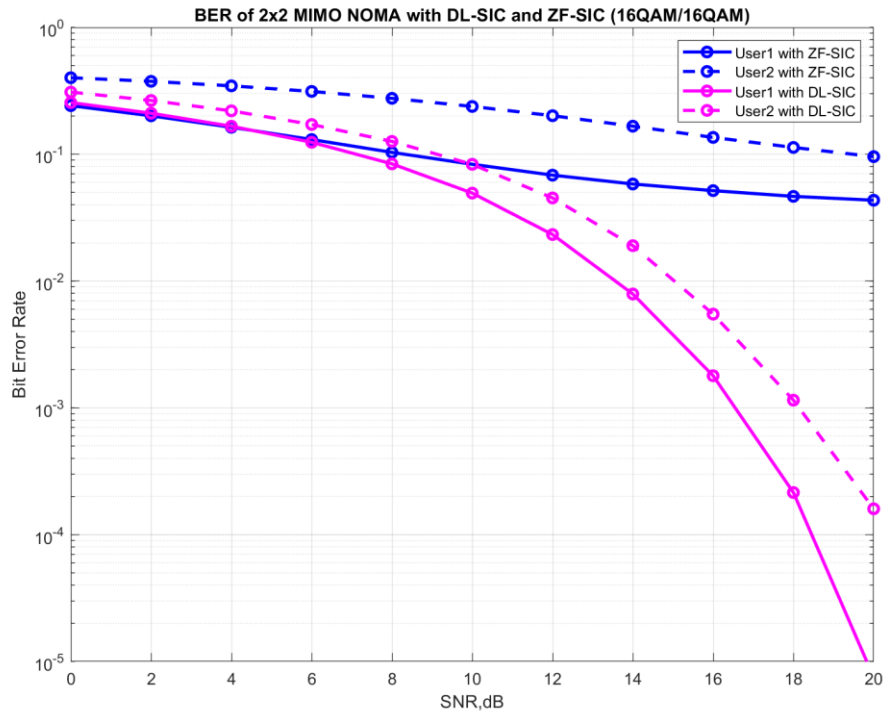


Figure 4.10. BER versus SNR for user 1 and user 2 with DL-SIC and ZF-SIC (16QAM/16QAM).

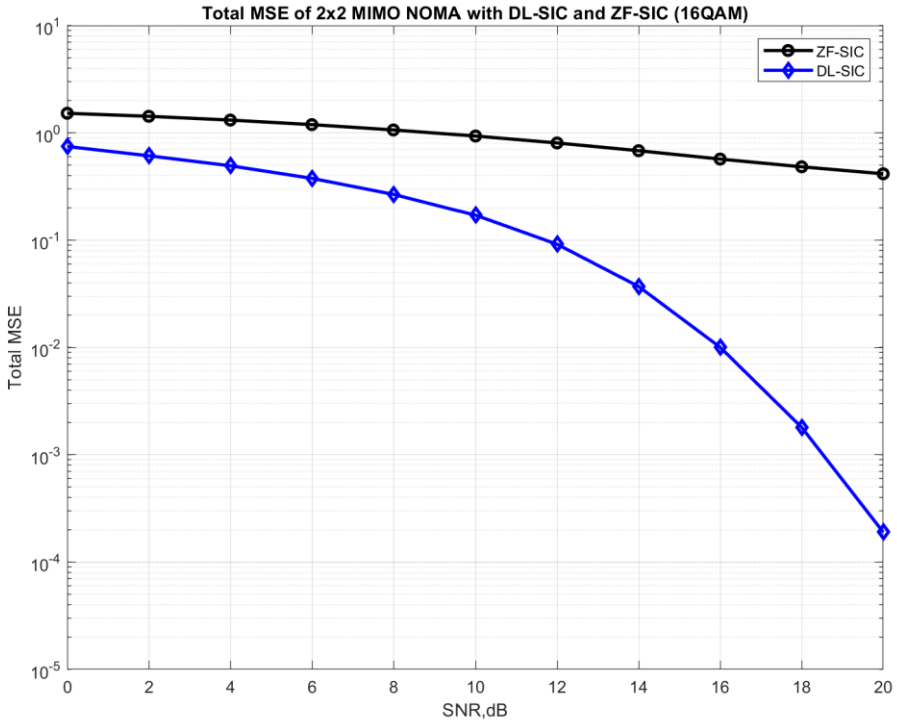


Figure 4.11. Total MSE versus SNR with DL-SIC and ZF-SIC (16QAM/16QAM).

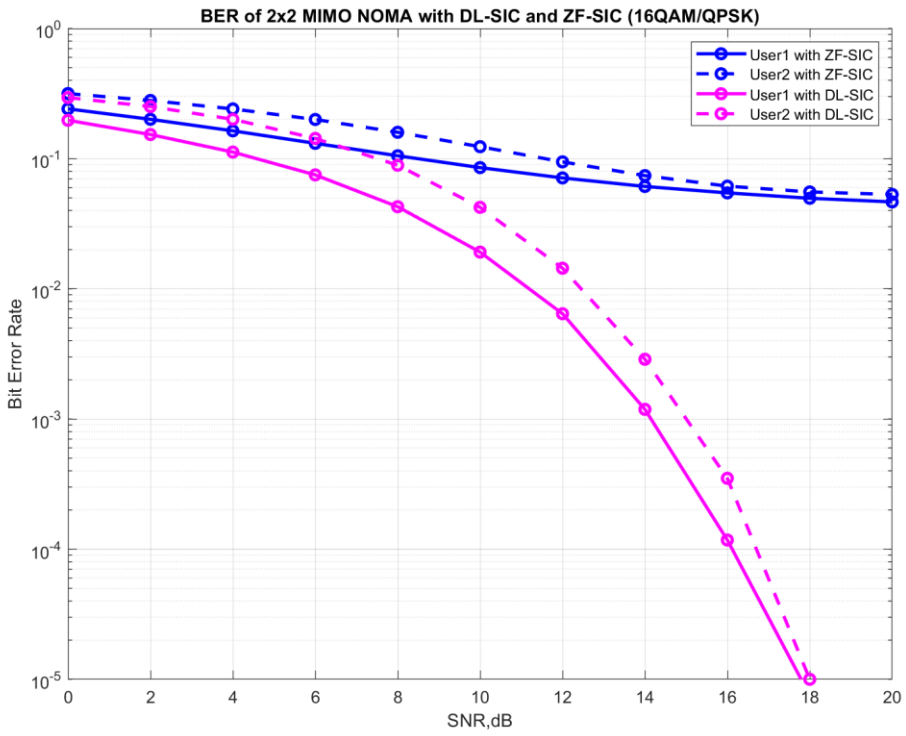


Figure 4.12. BER versus SNR for user 1 and user 2 with DL-SIC and ZF-SIC (16QAM/QPSK).

The BER performance for user 1 with 16QAM and user 2 with QPSK modulation is depicted in Figure 4.12. To balance NOMA transmission, we allocate 80% of power to user 1 and 20% of power to user 2. Both users achieve 10^{-5} error rate at 18 dB with the proposed DL method. Figure 4.13 shows the total MSE performance all over the SNR range. We have experimented with our proposed DL-SIC on different occasions where users use two modulations. The performance curves indicate that the proposed scheme can decode the users' signals of QPSK symbols or 16QAM symbols at the same SIC step by deciding whether to use two DNNs or not. The receiving system becomes more robust than the proposed model in Section 4.2. We will compare those two models in Section 4.4.

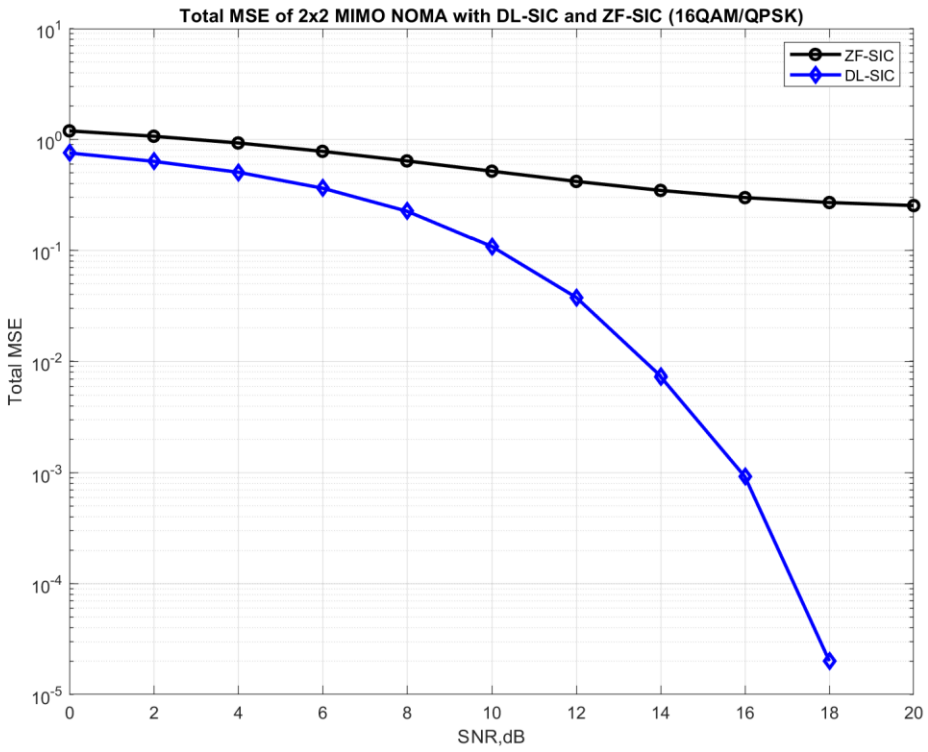


Figure 4.13. Total MSE versus SNR with DL-SIC and ZF-SIC (16QAM/QPSK).

4.4 Comparisons and Discussion

This section compares the DL-SIC model in Sections 4.2 and the proposed model. In addition, we also investigate the proposed DL-SIC model without adding the previously decoded symbols to the next SIC step, which means feeding the previously decoded symbols to the DNN for the next SIC operation. Therefore, modulation tasks after decoding each user's signal do not require.

The comparison of the DL-SIC scheme implemented in Sections 4.2 (first) and the proposed model is illustrated in Figure 4.14. The figure shows that the proposed model of Section 4.3 performs better than Section 4.2. At the 10^{-3} BER, user 2 achieves 18 dB with the first model while user 2 reaches 20 dB. The performance difference is approximate 2 dB for both user 1 and user 2. Even though the first model uses one DNN to decode user signal at each SIC step, the width of the neural network is twice the width of the second model. Each DNN of the first model has four hidden layers, while the DNNs we have implemented in the second model use two hidden layers. Thus, the height of the neural network used in the first model is also two times bigger except for input and output layers. This show that the proposed model can achieve better performance with two times smaller shapes DNNs, which means that the computational complexity of the neural network is two times reduced.

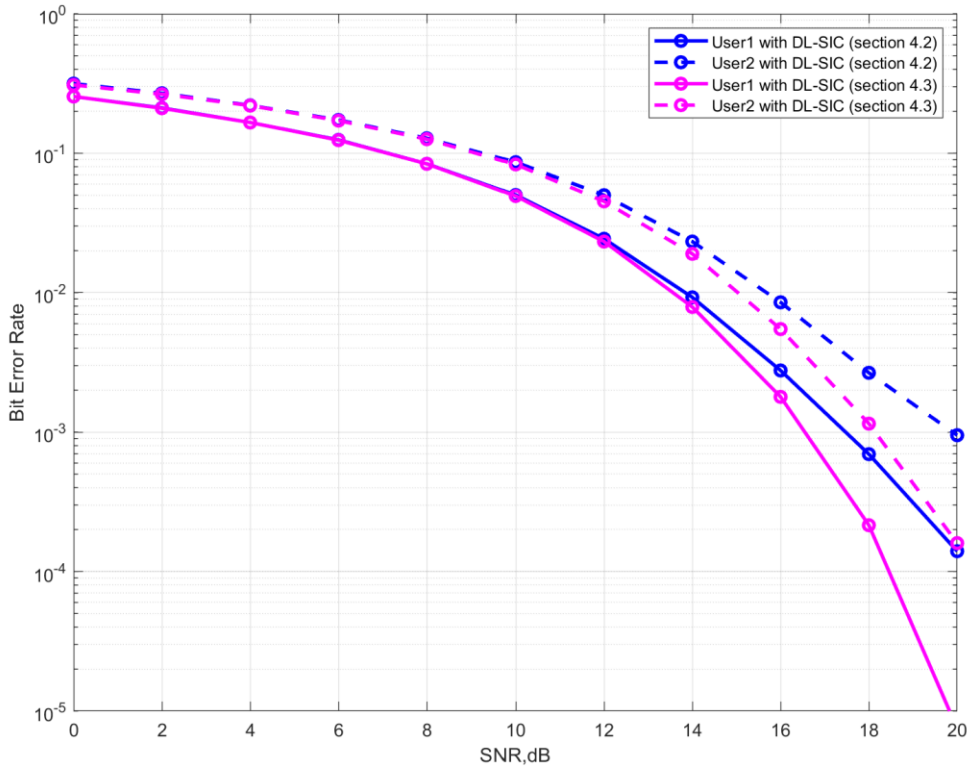


Figure 4.14. Comparison of BER versus SNR of two implemented DL-SIC models (16QAM/16QAM).

In Figure 4.15, the BER performance of a DL-SIC receiver without attaching the decoded signal of user 1 (QPSK) to user 2 (QPSK) SIC operation is compared to the proposed model. User 1 achieves the same BER performance because it is the first SIC operation, and there is no previously decoded user. However, the performance curve for user 2 behaves differently without the decoded symbols of user 1. At 10^{-4} BER, user 2 achieves 12 dB by attaching decoded symbols and 13 dB without the previously decoded symbols. The performance difference is about 1 dB.

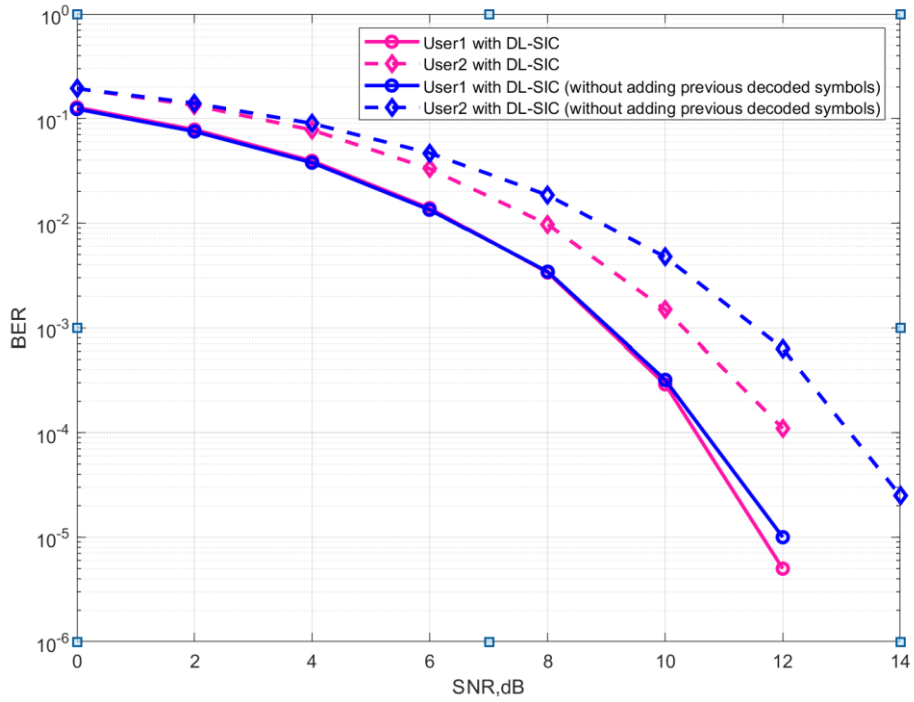


Figure 4.15. Comparison of BER versus SNR with and without adding previously decoded symbols (QPSK/QPSK).

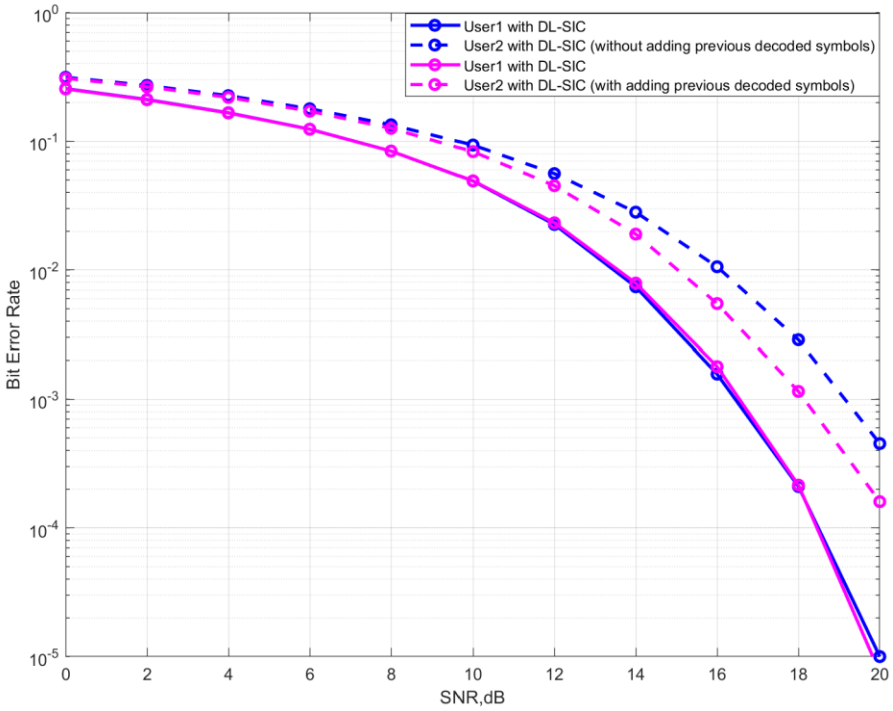


Figure 4.16. Comparison of BER versus SNR with and without adding previously decoded symbols (16QAM/16QAM).

Figure 4.16 compares the BER performance of a DL-SIC receiver without adding the decoded signal of user 1 (16QAM) at user 2 (16QAM) SIC operation to the proposed model. User 1 achieves the same BER performance. However, the performance curve for user 2 is altered without introducing user 1's decoded symbols. User 2 achieves 18 dB at 10^{-3} BER by attaching decoded symbols to the received combined signals, while it achieves 19 dB without attaching the decoded symbol.

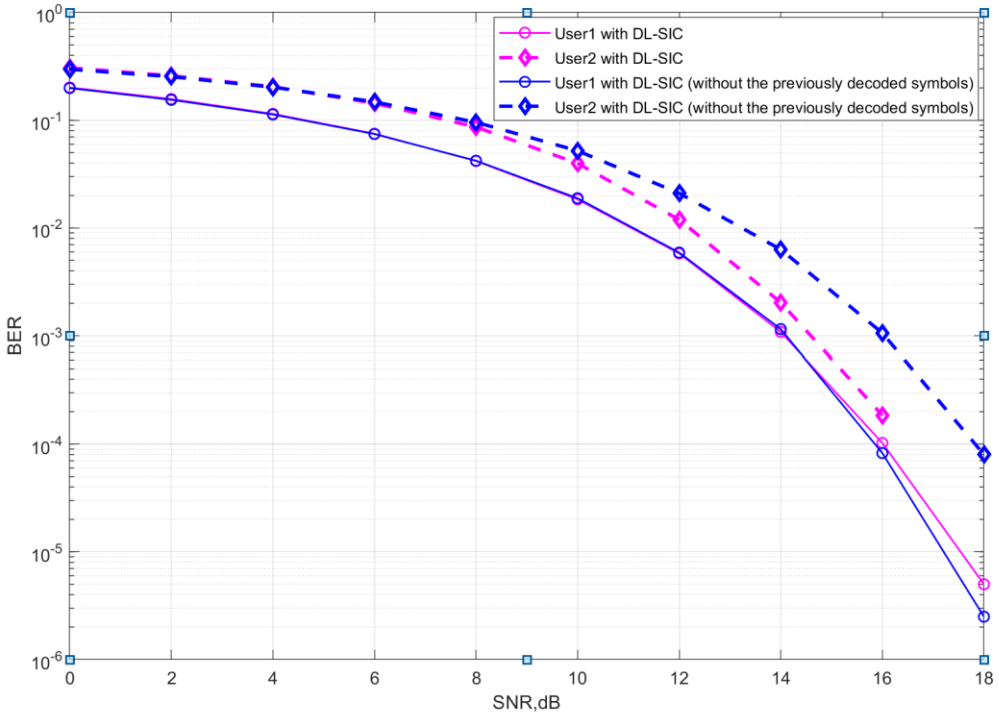


Figure 4.17. Comparison of BER versus SNR with and without adding previously decoded symbols (16QAM/QPSK).

Figure 4.17 compares the proposed model to the BER performance of a DL-SIC receiver without the decoded signal of user 1 (16QAM) at user 2 (QPSK). As in the previous scenario, the performance curve for user 2 is degraded without introducing decoded symbols from user 1. By attaching decoded symbols to the received combined signals, user 2 achieves 15 dB at 10^{-3} BER, whereas user 2 achieves 16 dB without attaching the decoded symbol. The performance difference is about 1 dB. The performance difference is approximately 1 dB. It demonstrates how adding previously decoded symbols assist DNN in learning properly for the next SIC operation. When the number of users grows, it may have a significant impact. In such a case, we should consider the complexity and performance trade-offs.

5. FUTURE WORK

We implemented deep learning-aided SIC operation for uplink MIMO-NOMA systems in Rayleigh fading channel. We investigated our proposed model with the approach of training during pilot transmission and testing during data transmission. Generally, the users transmit multiple frames containing the pilot and data symbols sequentially. Because each frame is transmitted throughout the coherent time interval, the channel impulse response across a single frame is assumed to be unchanged. All the models we have presented in this thesis have been trained offline using supervised learning. Training the model with an online learning approach for specific channel environments and conditions could be a future research area to explore. Implementing the DL-SIC with other fading channels to investigate their performance in different fading scenarios should also be considered as future work.

Another consideration is improving our proposed model so it can decode considerably higher modulation schemes. Our current DL-SIC design should apply to scenarios where the users transmit 64 QAM symbols. Because the current model can decode the users' symbols, either QPSK or 16QAM, by utilizing one or two trained DNNs, attempting to decode the 64 QAM symbols can be accomplished by treating the 64 QAM symbols as a combination of QPSK symbols and 16QAM symbols. Another possible extension is upgrading the DL-SIC model for millimeter wave MIMO-NOMA systems. In this thesis, we consider an equal power allocation strategy. However, the optimal power allocation strategy is important when the number of users grows. The DL-SIC system should be extended by applying a DL-based dynamic power allocation strategy in the future. Another thing we should explore is the learning strategy; deep reinforcement learning (DRL) should be considered rather than supervised learning. DRL is also thought to be a promising alternative for dealing with resource allocation challenges.

6. CONCLUSION

Wireless networks and related services are critical components of today's computerized environment and tremendously impact our daily lives. Data generated by significantly increasing devices have a variety of forms and have a nature of complex correlations [5]. As the 5G era begins, consumers and businesses define processes and channels to improve efficiency and livelihood. As a result, 5G-enabled applications and use cases such as autonomous vehicle control, intelligent transportation systems, smart agriculture, and manufacturing cell automation are starting to emerge. Because of these various bodies of wireless communications, the existing communication mechanism requires improvements in low latency, dependability, and availability, among other things [1]. On the other hand, deep learning can handle a massive amount of data and improve performance by using hierarchical feature extraction. It can further provide fast and highly accurate network analysis and management with the help of GPU-based parallel computing, overcoming the run-time limits of traditional mathematical approaches [12].

In recent years, many approaches to the NOMA system have been presented as potential multiple access solutions for 5G and beyond. In contrast to traditional orthogonal multiple access (OMA) systems, NOMA can service several users while keeping the same degree of freedom (DOF) [10] and achieving spectral efficiency. To achieve higher spectral efficiency, the NOMA concept has been applied to multiple-input multiple-output systems known as MIMO-NOMA. Current NOMA systems have some limitations, such as high computing complexity and a difficult optimal allocation strategy according to the significant changes in the wireless channel.

This thesis proposed a deep learning-aided uplink MIMO-NOMA system specifically for SIC operation. The main purpose of this thesis was to investigate the deep learning-based SIC for MIMO NOMA systems with higher order modulation. In Chapter 3, we implemented, with minor modifications, a deep learning-based SIC for the uplink MIMO-NOMA system using QPSK modulation in the Rayleigh fading channel proposed by [9]. The DL-based SIC was shown to outperform the equivalent MMSE-SIC at nearly all SNR levels. As we have implemented in Chapter 3, the output layer of the previous DL-SIC model [9] depends on the modulation order of the transmitting user.

In Chapter 4, two DL-SIC methods for higher order modulation schemes were proposed. To begin, we upgraded the Chapter 3 model's DNN architecture with QPSK modulation to decode higher-order modulation users. After that, we proposed a new SIC receiver layout that uses two DNNs in each SIC step. On three separate occasions, we trained and tested the proposed DNN model: both users use QPSK modulation; both use 16QAM; user 1 uses 16QAM, and user 2 uses QPSK modulation. After experimenting with our proposed DL-SIC on different occasions, the performance curves indicate that the proposed scheme can decode the users' signals of QPSK symbols or 16QAM symbols at the same SIC step by deciding whether to use two DNNs or not. The receiving system becomes more robust than the first proposed model.

We also simulated the model when the previously decoded symbol of user 1 does not apply to DNN when decoding the symbols of user 2. The BER curves showed that user 2 experiences more error propagation than user 2 with previously decoded symbols. The performance difference is about 1 dB. It shows how adding previously decoded symbols aids DNN in properly learning for the next SIC operation. When the number of users increases, the impact may be significant. In this case, we must weigh the complexity and performance trade-offs.

The DL-SIC can also help limit the SIC error propagation problem to some extent. In addition, the advantages of DL-SIC over conventional SIC are low latency processing due to the parallelized computing architecture and lower complexity as a result of decoding the user's signal without the need to estimate channel coefficients or explicitly subtract the decoded signal. To summarize, the numerical results showed that the BER and total MSE performance of the MIMO NOMA system with DL-approach SIC outperformed that of the conventional SIC.

7. References

[1] Ericsson (2017) 5G systems - Enabling the transformation of industry and society. White Paper UEN 284 23-3251 rev B, Ericsson.

[2] Islam S. M. R., Avazov N., Dobre O. A., and Kwak K. (2016) Power-Domain Non-Orthogonal Multiple Access (NOMA) in 5G Systems: Potentials and Challenges. IEEE Communications Surveys & Tutorials, vol. 19, no. 2.

[3] Kang J., Kim I., and Chun C. (2019) Deep Learning-Based MIMO-NOMA with Imperfect SIC Decoding. IEEE Systems Journal.

[4] Chuan L., Qing C., and Xianxu L. (2020) Uplink NOMA signal transmission with convolutional neural networks approach. Journal of Systems Engineering and Electronics, vol. 31.

[5] Alsheikh M. A., Niyato D., Lin S., Tan H. P., and Han Z. (2016) Mobile big data analytics using deep learning and Apache Spark. IEEE network.

[6] Qin Z., Ye H., Li G. Y., and Juang B. F. (2019) Deep Learning in Physical Layer Communications. IEEE Wireless Communications, vol. 26, no. 2.

[7] Tse D., and Viswanath P. (2005). Fundamentals of Wireless Communication. Cambridge. Cambridge University Press.

[8] Proakis J., and Salehi M. (2007) Digital Communications. McGraw-Hill Education; 5th edition.

[9] Aref M. A., and Jayaweera S. K. (2019) Deep Learning-aided Successive Interference Cancellation for MIMO-NOMA. GLOBECOM 2020 - 2020 IEEE Global Communications Conference.

[10] Wei Z., Yang L., Ng D. W. K., and Yuan J. (2018) On the Performance Gain of NOMA over OMA in Uplink Single-cell Systems. IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, UAE.

[11] Ding Z., Yang Z., Fan P., and Poor H. V. (2014). On the performance of non-orthogonal multiple access in 5G systems with randomly deployed users. IEEE Signal Processing Letters.

[12] Zhang C., Patras P., and Haddadi H. (2019) Deep learning in mobile and wireless networking: A survey. IEEE Communications Surveys Tutorials.

[13] Wang T., Wen C., Wang H., Gao F., Jiang T., and Jin S. (2017) Deep learning for wireless physical layer: Opportunities and challenges. China Communications 14.

[14] O'Shea T., and Hoydis J. (2017) An introduction to deep learning for the physical layer. IEEE Transactions on Cognitive Communications and Networking 3.

[15] Hornik K., Stinchcombe M., and White H. (1989) Multi-layer feedforward networks are universal approximators. Neural Networks 2.

[16] Samuel N., Diskin T., and Wiesel A. (2017) Deep MIMO detection.

[17] Vanhoucke V., Senior A., and Mao M. Z. (2011) Improving the speed of neural networks on CPUs. In Proc. Deep Learning and Unsupervised Feature Learning NIPS Workshop.

[18] Sutton R. S., and Barto A. G. (2018). Reinforcement learning: An introduction. MIT Press.

[19] Goodfellow I., Bengio Y., and Courville A. (2016) Deep learning. MIT Press.

[20] Kingma D. P., and Jimmy B. (2015) Adam: A Method for Stochastic Optimization.

[21] Abadi M., Agarwal A., Barham P., Brevdo E., Chen Z., Citro C., Corrado G. S., Davis A., Dean J., Devin M., Ghemawat S., Goodfellow I. J., Harp A., Irving G., Isard M., Jia Y., Józefowicz R., Kaiser L., Kudlur M., Levenberg J., Mané D., Monga R., Moore S., Murray D. G., Olah C., Schuster M., Shlens J., Steiner B., Sutskever I., Talwar K., Tucker P. A., Vanhoucke V., Vasudevan V., Viégas, F. B., Vinyals O., Warden P., Wattenberg M., Wicke M., Yu Y., and Zheng X. (2016). TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems.

[22] Paszke A., Gross S., Chintala S., Chanan G., Yang E., DeVito Z., Lin Z., Desmaison A., Antiga L., and Lerer A. (2017) Automatic differentiation in PyTorch. In: NIPS Autodiff Workshop.

[23] Jia Y., Shelhamer E., Donahue J., Karayev S., Long J., Girshick R.B., Guadarram S., and Darrell T. (2014) Caffe: Convolutional architecture for fast feature embedding. In: ACM Multimedia.

[24] Chollet F. et al. (2015), Keras. https://keras.io.

[25] Günter K., Thomas U., Andreas M., and Sepp H. (2017) Self-normalizing neural networks. In Advances in Neural Information Processing Systems.

[26] Ye H., Li G. Y., and Juang B. (2018) Power of deep learning for channel estimation and signal detection in OFDM systems. IEEE Wireless Communications Letters 7.

[27] Hu Q., Gao F., Zhang H., Jin S., and Li G. Y. (2021) Deep Learning for Channel Estimation: Interpretation, Performance, and Comparison. IEEE Transactions on Wireless Communications, vol. 20.

[28] David N., Wolfgang U., and Thomas W. (2017) Deep channel estimation. In Proc. 21st International ITG Workshop on Smart Antennas.

[29] Vaezi M., Schober R., Ding Z., and Poor H. V. (2019) Nonorthogonal Multiple Access: Common Myths and Critical Questions," IEEE Wireless Communications Magazine, vol. 26, no. 5.

[30] Gui G., Huang H., Song Y., and Sar H. (2018) Deep Learning for an Effective Non-orthogonal Multiple Access Scheme. IEEE Transactions on Vehicular Technology, vol. 67, no. 9.

[31] Pan J., Ye N., Wang A., and Li X. (2020) A Deep Learning-Aided Detection Method for FTN-Based NOMA. Hindawi Wireless Communications and Mobile Computing, vol. 2020.

[32] Lin C., Chang Q. and Li X. (2019) A Deep Learning Approach for MIMO-NOMA Downlink Signal Detection. Sensors (Basel), vol. 19, no. 11.

[33] Wei Z., Ng D. W., and Yuan J. (2018) Joint Pilot and Payload Power Control for Uplink MIMO-NOMA with MRC-SIC Receivers. IEEE Communications Letters, vol. 22, no. 4.

[34] Aldababsa M., Toka M., Gökceli S., Karabulut K. G., and Kucur O. (2018). A Tutorial on Non-orthogonal Multiple Access for 5G and Beyond. Wireless Communications and Mobile Computing.

[35] Bishop C. M. (2006) Pattern Recognition and Machine Learning. Springer, New York, NY.

[36] Ramchandran K., Ortega A., Uz K. M., and Vetterli M. (1993) Multiresolution broadcast for digital HDTV using joint source/channel coding. IEEE J. Select. Areas Commun.