Application of Machine Learning to Multi Antenna Transmission and Machine Type Resource Allocation

Don-Roberts U. Emenonye

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Electrical Engineering

R. Michael Buehrer, Chair
Carl B. Dietrich, Co-chair
Harpreet S. Dhillon

July 31st, 2020
Blacksburg, Virginia

Keywords: Machine Type Communication, Space Time Block Coding, Deep Learning, Reinforcement Learning

Application of Machine Learning to Multi Antenna Transmission and Machine Type Resource Allocation

Don-Roberts U. Emenonye

(ABSTRACT)

Wireless communication systems is a well-researched area in electrical engineering that has continually evolved over the past decades. This constant evolution and development have led to well-formulated theoretical baselines in terms of reliability and efficiency. However, most communication baselines are derived by splitting the baseband communications into a series of modular blocks like modulation, coding, channel estimation, and orthogonal frequency modulation. Subsequently, these blocks are independently optimized. Although this has led to a very efficient and reliable process, a theoretical verification of the optimality of this design process is not feasible due to the complexities of each individual block. In this work, we propose two modifications to these conventional wireless systems.

First, with the goal of designing better space-time block codes for improved reliability, we propose to redesign the transmit and receive blocks of the physical layer. We replace a portion of the transmit chain - from modulation to antenna mapping with a neural network. Similarly, the receiver/decoder is also replaced with a neural network. In other words, the first part of this work focuses on jointly optimizing the transmit and receive blocks to produce a set of space-time codes that are resilient to Rayleigh fading channels. We compare our results to the conventional orthogonal space-time block codes for multiple antenna configurations.

The second part of this work investigates the possibility of designing a distributed multi-agent reinforcement learning-based multi-access algorithm for machine type communication. This work recognizes that cellular networks are being proposed as a solution for the connectivity of machine type devices (MTDs) and one of the most crucial aspects of scheduling in

cellular connectivity is the random access procedure. The random access process is used by conventional cellular users to receive an allocation for the uplink transmissions. This process usually requires six resource blocks. It is efficient for cellular users to perform this process because transmission of cellular data usually requires more than six resource blocks. Hence, it is relatively efficient to perform the random access process in order to establish a connection. Moreover, as long as cellular users maintain synchronization, they do not have to undertake the random access process every time they have data to transmit. They can maintain a connection with the base station through discontinuous reception. On the other hand, the random access process is unsuitable for MTDs because MTDs usually have small-sized packets. Hence, performing the random access process to transmit such small-sized packets is highly inefficient. Also, most MTDs are power constrained, thus they turn off when they have no data to transmit. This means that they lose their connection and can't maintain any form of discontinuous reception. Hence, they perform the random process each time they have data to transmit. Due to these observations, explicit scheduling is undesirable for MTC.

To overcome these challenges, we propose bypassing the entire scheduling process by using a grant free resource allocation scheme. In this scheme, MTDs pseudo randomly transmit their data in random access slots. Note that this results in the possibility of a large number of collisions during the random access slots. To alleviate the resulting congestion, we exploit a heterogeneous network and investigate the optimal MTD-BS association which minimizes the long term congestion experienced in the overall cellular network. Our results show that we can derive the optimal MTD-BS association when the number of MTDs is less than the total number of random access slots.

Application of Machine Learning to Multi Antenna Transmission and Machine Type Resource Allocation

Don-Roberts U. Emenonye

(GENERAL AUDIENCE ABSTRACT)

Wireless communication systems is a well researched area of engineering that has continually evolved over the past decades. This constant evolution and development has led to well formulated theoretical baselines in terms of reliability and efficiency. This two part thesis investigates the possibility of improving these wireless systems with machine learning.

First, with the goal of designing more resilient codes for transmission, we propose to redesign the transmit and receive blocks of the physical layer. We focus on jointly optimizing the transmit and receive blocks to produce a set of transmit codes that are resilient to channel impairments. We compare our results to the current conventional codes for various transmit and receive antenna configuration.

The second part of this work investigates the possibility of designing a distributed multi-access scheme for machine type devices. In this scheme, MTDs pseudo-randomly transmit their data by randomly selecting time slots. This results in the possibility of a large number of collisions occurring in the duration of this slots. To alleviate the resulting congestion, we employ a heterogeneous network and investigate the optimal MTD-BS association which minimizes the long term congestion experienced in the overall network. Our results show that we can derive the optimal MTD-BS algorithm when the number of MTDs is less than the total number of slots.

Dedication

To all the teachers I have ever had

Acknowledgments

The list of individuals that have contributed to my education would be endless. To name and appreciate a few, I would like to start by thanking my supervisors Dr. R. Michael Buehrer and Dr. Carl B. Dietrich for providing the resources and inspiration to investigate interesting ideas. I am sincerely grateful for the opportunity to have worked with such seasoned and experienced individuals, their valuable feedback and expert guidance has enabled me to find solutions to difficult problems in an efficient manner.

In addition, I would like to thank Dr. Harpreet S. Dhillon for his patience and guidance during my graduate study, I would especially like to express gratitude to him for providing encouragements at the beginning of my graduate research.

I would also like to express additional thanks to Dr. R. Michael Buehrer and Dr. Harpreet S. Dhillon for providing two of the best classes I have ever taken in the form of Multi-Channel Communications and Stochastic Signals and Processes.

To Christopher, Rubayet, and Raghu, I am grateful for the wage free encouragement and technical guidance.

To Mr. Thornton, thank you for being a great friend and for the unscheduled car rides. It has been great sharing both a classroom and a work area with you.

To Frank, Aidin, Avik, Dmitri, Jet, Daniel, Will, Xavier, and Christina, getting to know you guys has been a terrific experience. Thank you for the great memories.

To Hilda, thank you for being a genuinely terrific and kind person and for your tremendous patience.

To my beloved parents, I am forever indebted to you because of the incredible sacrifices you have made.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Case for Learning-Based Systems

Although wireless networks have been extensively researched during the past decades, the promise of improved guaranteed quality of service, - reliability, and latency and new applications in future generation networks necessitates further investigation [1]. One area of possible research involves the modularized physical layer. Examples of the modular blocks include forward error correction, modulation, channel equalization, and orthogonal frequency division multiplexing. Over time, the optimization of these blocks has spawned into distinct and well-defined areas of research, and expert domain knowledge of these research areas has been established. Hence, recent research has yielded only marginal improvements in the performance gains, and most times these gains require complex analytical models that are intractable in real-world systems. Also, the modular structure of the physical layer may not be optimal, as is the case with the separation of modulation and coding [2].

However, in the machine learning world, model-free algorithms are being used to solve complex problems. Authors in [3], proposes a novel paradigm shift from the conventional design of the physical layer to an end-to-end machine-based design. This work led to a surge in the research of end-to-end machine learning-based systems. Leveraging this existing body of research, we aim to design space-time block codes that are more resilient to channel

impairments. Following previous works, we construct the transmit and receive aspects of the physical layer as a neural network. The symbols produced from the encoders are multiplexed in both space and time such that the final output is an integer multiple of the number of symbols in one time step. These symbol mappings are optimized by minimizing the mostly convex loss functions deployed in machine learning systems [4, 5]. We present our case by comparing our results with conventional space-time block codes which are submodular in design.

The second aspect of this work pertains to the design of a grant-free scheduling algorithm for deivces that operate without human intervention. This devices are usually referred to as machine type devices (MTDs). The expected prevalence of MTDs has led to a significant body of work directed at providing connectivity for such devices. Through this body of work, cellular networks have been largely proposed and discussed as a solution. However, MTDs operate in a different way than conventional cellular uses, hence, explicitly scheduling such device through conventional cellular networks might prove challenging. The first challenge relates to the suitability of the random access process. The first issue pertaining to the random access process is the sporadic, event-based nature of machine-type communications. The event-based nature of machine type devices can lead to similar packets arrival times - for instance, if there is a sudden change in the temperature, sensors reporting weather conditions will simultaneously have a burst of packets. This will lead to an increase in congestion and subsequently a high rate of collisions in the random access process. Second, after the random access process fails, devices need to back off then retry the entire random access procedure, this is unsuitable for machine type devices with stringent delay requirements. Lastly, with the exception of autonomous vehicles, the packets generated by machine type devices are small relative to the number of resource blocks used in the RA process, each RA slot has a size of 1.04MHz in the uplink, this is equivalent to six resource blocks. Therefore,

it can be very inefficient to explicitly schedule machine type devices.

We propose a multi-agent reinforcement-based solution. This solution exploits a heterogeneous cellular network in order to alleviate the congestion caused by the simultaneous transmission of packets by a large number of machine the devices. Results from this scheme show that the MTDs are able to self-organize in a distributed manner across multiple cells in order to eliminate the long term congestion.

1.2 Current Trends

1.2.1 End-to-End Learning Based Diversity Systems

Multi-antenna systems have been in existence for more than half a century. Single Input Multiple Output systems based on receive combining has excited since the 1950s [6]. Multi transmit antenna systems (MISO) in [7], takes advantage of its time division duplex TDD implementation to support transmit based selection diversity. Although SIMO and MISO techniques had been popularized, the effect of simultaneously using multiple transmit and receive antennas was not known. Authors in [8], showed that with independent channel gains and antennas spaced sufficiently at both the transmitter and the receiver, the capacity increases linearly as the number of transmitters and receivers increased

The Alamouti codes introduced in [9], provides a diversity achieving scheme with a linearly scale-able diversity order. Subsequent works in [10], develops space-time block codes using the mathematical concept of orthogonal designs and Hadamard matrices. Due to the orthogonal nature of these block codes, the decoding/receive process has linear complexity. Authors in [11] present an example list of block codes and their encoding and decoding algorithms. Considering this plethora of works, multi-channel communication is a well-researched

area, and any machine-learning-based algorithm must have significant performance benefits, for it to be considered for implementation.

Noting this subtle, but important point, wireless researchers have tried to reinvent the communication wheel with machine learning-based techniques. Researchers in [12], designed a support vector machine (SVM) based link adaptation scheme. Using a frame by frame error computation, devices perform link adaptation to find the optimal rate/reliability trade-off. In [13], a deep neural network model is trained offline based on channel statistics and used online for symbol detection in orthogonal frequency division systems. A machine learning-based localization scheme is developed in [14]. Although promising, most of the previously stated research has focused on optimizing modular blocks of the physical layer. Recently, researchers postulated that representing the entire physical layer as a single block and performing a single optimization process could provide some performance benefits while simultaneously reducing the complexity of the communication systems.

To this end, pioneering work on completely learning-based systems was investigated by [3]. The physical transmit and receive blocks modeled as one deep neural network produced competitive symbol error performance when compared with a conventional single transmit single receive systems. Convexity of loss functions enabled the batch by batch optimization of the network weights and bias parameters. End to End learning is easily extended to multi-channel communications for both diversity and multiplexing operations [15]. These end-to-end systems all assume that a differentiable channel model is present - allowing for backpropagation. For real-world channel models, this assumption becomes unpractical. To solve this problem, authors in [16] presents a policy gradient-based technique for training the receiver and the transmitter in an alternating sequence. The loss is supplied from the receiver to the transmitter using a lossless and unquantized feedback link. The effect of quantization on the loss is analyzed in [17].

1.2.2 Grant Free Resource Allocation for Machine Type Communications

Cellular networks have been proposed as a source of connectivity for machine-type devices. However, exploiting a network that was designed for human type communications for machine type communications has provided some interesting research challenges. An increasingly researched challenge is the random access procedure. Two challenges related to the application of the random access process to machine type communications are congestion and inefficiency. I) Congestion: Sensors measuring the same physical process could have similar transmit times. This will lead to an increase in congestion and subsequently a high rate of collisions in random access. II) Efficiency: MTDs generate packets that are small relative to the number of resource blocks used in the RA process. Therefore, it is inefficient to use the random access process to transmit packets of such sizes. Previous resource allocation solutions for MTDs can be grouped into coordinated access and grant free schemes.

Coordinated access schemes

Several coordinated access schemes were inspired by the 3GPP [18]. Leveraging this, authors in [19] proposed a cooperative access class barring (ACB) scheme. In [20], authors develop a dynamic ACB scheme to reduce congestion and delay. In [21], a Bayesian algorithm is used to estimate the backlogged machine type devices and the optimal ACB factor is determined. Aside access class barring, researchers in [22] proposes introducing a backoff timer. In this work, devices that experience a collision during the random process are encouraged to apply a backoff timer. Also, a random access attempt limit is proposed. Such a method clearly impacts latency. In [23], the 3GPP proposes dynamic allocation of RACH resources. During high traffic load cases, this algorithm increases the number of PUSCH

resources allocated for the random access procedure. The authors in [24, 25], proposed a frame-slotted Aloha type solution. In this work, MTDs restrict their transmission to dedicated slots on a frame by frame basis, however, this work does not takes into account the sometimes mission-critical nature of machine type communication. In such critical systems, if a packet arrives after the dedicated slots then the latency requirements can be violated while waiting for the next dedicated slots. In [26], a group-based access control scheme is developed. In contrast to the conventional random access scheme, where each cellular user attempts to connect to the network, this work divides devices into groups and a device is allotted the responsibility of performing the random access procedure. Leveraging on code division multiple access (CDMA), authors in [27] proposed a method of increasing the number of contention resources in the random access process. In [28], based on the received signal strength at various UEs, cells are divided into different spatial-groups. Preambles are generated according to this spatial grouping. In [29], UEs located in spatial-groups which are far apart are assigned identical preambles. In [30], a non-orthogonal random access (NORA) scheme is proposed. This scheme employs power control, time-of-arrival, and successive interference cancellation to decode identical preambles that are transmitted by different UEs in the same time slot. Authors in [31], propose combining a spatial-grouping scheme and NORA in order to increase the number of preambles available in the RACH process. The authors in [32], use the analog fountain codes to allow MTDs opportunistically access random access slots in order to satisfy their delay requirements. In [33], authors employ a bloom filtering based signature assignment scheme in order to reduce collision during the random access procedure. In [34], a collision avoidance scheme is developed based on the transmitting patterns of machine type devices. In [35], the probability of collision is reduced by attaching a unique MTD ID to each preamble transmission. All the works presented in this section still require significant uplink signalling which limits any potential deployments for MTCs.

Grant-free schemes

In this section, we present a discussion of the second type of solution for cellular-based machine the communications. These solutions are termed grant-free schemes. In these schemes, an MTD pseudo-randomly selects a resource block and transmits its data in that resource block. Clearly, a completely random grant-free approach will result in a significant amount of collisions, hence this is still an area of intensive research. Authors in [36], propose periodic pooling of resources in order to guarantee a certain quality of service (QoS) (reliability). Authors in [37], exploit behavioural game theory to model the distributed nature of a grant-free approach. In this work, the problem of satisfying the quality of service of distinct heterogeneous machine type device is phrased as a non-cooperative game. This non-cooperative solution satisfies the QoS requirements of the MTDs 96% of the time. The detection challenges in a fully grant-free aproach is solved by combine sparse signal processing and massive MIMO [38]. Authors in [39], formulates the problem of detecting the sporadic and massive number of concurrent request from machine type devices as a compressed sensing single measurement vector problem. This detection problem is also solved with machine learning [40]. In [41], sum rate bounds are derived in the uplink for massive MIMO aided machine type communications. More recently, a collaborative Q-learning based slotted ALOHA scheme was developed to ensure MTDs find unique time slots [42]. Although impressive, most of the discussed grant-free techniques still experiences non-negligible collisions. This is because to a large extent grant-free scheme involves randomly selecting a resource block for data transmission. To solve this problem, we present in chapter 4, an MTD-BS association algorithm which seeks to minimize or eliminate congestion.

1.3 Thesis Organisation and Contributions

This thesis consists of two parts. The first part involves modeling the physical layer as an autoencoder in order to generate new space-time block codes. The second part of this thesis involves applying reinforcement learning in order to reduce the congestion caused by a massive number of machine-type devices. More specifically:

- In the first part of this thesis, we expand on previous autoencoder models for the physical layer [3]. We show through simulations that we can eliminate the need for explicit pilots and channel estimation in space-time coding if we design the receiver as two distinct blocks. The first block called the pre-processing block extracts the complex values needed for combining from a batch of received symbols. This block also performs some sort of pseudo-combining at the receiver, after which the second block estimates the symbols sent at a specific time.

- We compare the symbol error rate performance obtained from the neural network based space-time coding scheme with the symbol error rate performance derived from classic orthogonal space-time block codes.

- In the second part of this thesis, we formulate the congestion problem caused by the concurrent transmissions from a massive number of machine type devices as a constrained combinatorial problem. We claim that the problem is intractable due to a lack of information at the base station.

- The optimization is reformulated as a finite MDP. We approximate the solution of the MDP using a distributed model-free algorithm.

- Lastly, we empirically validate the algorithm using the probability of collision, channel utilization, and access success probability. We show that the algorithm approximates

the optimal solution when the number of MTDs is less than the number of time slots.

1.3.1 Organization

The rest of this work is organized as follows: Chapter 2 provides the background needed to fully appreciate the work presented in the rest of this thesis. This chapter presents discussions about multi-antenna systems, machine-type devices, and machine learning techniques. Chapter 3 presents a discussion about the neural network based space-time block codes. The neural-network based system model and training schemes used is also presented in this chapter. Lastly, the simulation results are presented. Chapter 4 first provides the formulation of the congestion problem caused by concurrent transmissions from a massive number of machine type devices as a constrained combinatorial problem. In this chapter, the optimization problem is re-formulated as an MDP. A abse station association algorithm which approximates the solution to the MDP is also presented. Lastly, the simulation results are presented. Chapter 5 provides concluding remarks and a brief description of future research directions.

Chapter 2

Background

In order to to fully appreciate the work presented in this thesis, we provide background material in this chapter for several key concepts. First, we provide an introductory overview of multi-antenna systems and the muliplexing/diversity beneifts they provide. Second, we present a discussion about the type of communication devices that are regarded as "machine type" devices and we briefly present a discussion on the suitability of using cellular networks in order to provide connectivity for machine type communications. We highlight that the random access protocol is a bottleneck in the deployment of cellular networks for machine type connections. Third, because, most contributions made in this thesis are in the area of machine learning and/or reinforcement learning, we provide a definition and a simple introduction to the fundamental principles behind machine learning. More specifically, we define the various terms and discuss the different classifications of machine learning algorithms. A simple discussion is presented about the limitations and failings of machine learning algorithms. However, we also show that deep learning models are powerful function approximators used to learn relationships that are too complex for conventional machine learning methods. Along with an overview of how the neural models are stacked in order to generate a deep learning system, a discussion of the expressive elements needed for each neural element is presented. In addition, we present a brief reinforcement learning overview. A discussion is presented regarding model and model-free reinforcement learning approaches. This section concludes with a description of the popular Q-learning algorithm deployed in

chapter 4.

2.1 Multi-antenna Systems

Signal processing techniques that deploy multiple antennas at the transmitters and/or receivers are described as Multi Input Multi Output systems (MIMO). The additional degree of freedom present in communication systems due to multiple antennas is used to either improve system throughput or system reliability. MIMO techniques that exploit the additional degrees of freedom to increase throughput are referred to as spatial multiplexing. Spatial multiplexing takes advantage/exploits the rich multipath scattering in communication channels to increase the rate at which information is transmitted from one point to another. The other class of MIMO techniques combats the multipath effect of the communication channel in order to increase system reliability. These systems are termed diversity achieving systems. Although, MIMO is commonly used to refer to any system with multiple antennas at both the transmitter and the receive, some authors abuse this terminology and refer to systems with more than one antenna at only one side of the communication link as MIMO systems. In the strict sense, a communication model deployed with single transmit and multiple receive antennas is called SIMO - single input multi output. Similarly, a model with a single-receive multi-transmit configuration is called MISO - multiple input single output. Finally, in the literature a system model with N_t transmit antennas and N_r receive antennas is denoted as a $N_r \times N_t$ MIMO system.

The medium between the N_t transmit antennas and N_r receive antennas is referred to as the spatial wireless channel and arguably the most important phenomenon that affects this medium is fading. Fading causes the strength of the signal transmitted to vary with time and/or space. Most authors categorize fading into two groups: I) Small scale fading

characterizes the rapid fluctuation of signal strength over short distances and/or over a short period of time. Small scale fading occurs when the transmitted signal arrives along different paths, each path having its own inherent phase, Doppler shift, delay, and path attenuation. This category of fading can be further categorized dependent on the bandwidth of signal transmitted. A signal with bandwidth greater than or on the same order as the channel coherence bandwidth will experience frequency-selective fading, which leads to complications during demodulation. A simple solution is to use orthogonal frequency division multiplexing (OFDM) to divide the signal into many narrowband signals. A channel with a short coherence time (or a large Doppler) is termed a fast fading channel. Hence, fades will last for only a short period of time. II) Large scale fading occurs on a much longer time scale and over longer distances. Large scale fading is used to characterize the loss in signal amplitude due to the terrain of the environment.

As expected, fading can cause severe degradation in bit error rate performance. In order to combat this performance degradation, diversity achieving systems are usually deployed. Diversity involves transmitting or receiving different versions of the same signal such that all signal replicas experience independent fades. The intuition is that if every signal replica experiences independent fading, the probability that all replicas would be in a deep fade at the same time is greatly reduced. Diversity schemes include: I) Time diversity: The idea here is that the signals transmitted at different time slots can experience independent fades. II) Frequency diversity: This exploits the dependency of small scale fading on frequency. Here, signal replicas are spaced in frequency to encourage diversity. III) Spatial diversity: This involves exploiting the rich scattering of the channel and establishing different physical paths for transmitting the same information. The first type of spatial diversity involves sending a single stream and receiving it on multiple receive antennas. A crucial aspect of such diversity achieving systems is the receive combining. Examples of receive combining

techniques include: I) Selective combining: this technique involves selecting one of he received signals based on some metric. Most authors use the signal to noise ratio (SNR) as the metric of choice. II) Maximum ratio combining: this involves weighing the various received signals based on their SNR's and combining them. The weights are designed such that signals with better SNR are given a higher weight. III) Equal gain combining - This is a special version of maximum ratio combining with the caveats that all the weights are set to the same value. The discussed diversity techniques are referred to as receive diversity. It gets this terminology because processing and combining are done by the receiver. The performance of a receive diversity system is heavily dependent on how far apart the receive antennas are spaced. The farther apart the antennas are the more independent the fades. However, in cellular communications, engineers are limited by the available space. This is because mobile phones have a size beyond which they are impractical. This motivates the need for transmit diversity in the forward link or cellular downlink. However, achieving transmit diversity is a nontrivial task. Consider a multi transmit single antenna system, simply sending the same signal across all antennas doesn't provide diversity. To see this, consider the output of a matched filter with a single receive antenna:

$$r = \frac{s}{\sqrt{s}} \sum_{i=1}^{N_t} h_i + n \tag{2.1}$$

In this equation, r, represents the received signal sample at the output of our matched filter, h_i denotes the channel between the receiver and the $i-$th transmitter, s is the transmitted symbol, and n represents the noise impairments. Clearly, the received signal is just a scaled version of the transmitted signal. In fact, the SNR is no different than if only one antenna was used at the transmitter. This is because a fraction of the power is sent from each antenna and the "combining" happens in the air and is thus non-coherent. To correct this, some type of encoding needs to be performed at the transmitter. This encoding is known as

space-time coding. The most popular type of space-time coding for two transmit antennas is known as the Alamouti scheme. This scheme has attracted much popularity because it is the only rate one space-time block coding scheme for complex modulation. This means that on average one symbol is transmitted every time a step. At the receiver, maximum

Table 2.1: Alamouti Scheme

Time Index	Antenna 1	Antenna 2
t	s_1	s_2
$t+T_s$	$-s_2^*$	s_1^*

ratio combining is used to realize the diversity benefits. A common space time block code for higher number of transmit antennas is the orthogonal space time block coding (OSTBC) scheme [10] which is a generalization of the Alamouti scheme and is shown in Tables 2.2 and 2.3 for 3 and 4 antennas respectively. These codes gained popularity because the authors provided a decoding scheme which is linear in processing time.

Table 2.2: OSTBC for 3 transmit antennas

Time Index	Antenna 1	Antenna 2	Antenna 3
t	s_1	s_2	0
$t+T_s$	$-s_2^*$	s_1^*	0
$t+2T_s$	0	0	s_1
$t+3T_s$	0	0	$-s_2$

Table 2.3: OSTBC for 4 transmit antennas

Time Index	Antenna 1	Antenna 2	Antenna 3	Antenna 4
t	s_1	s_2	0	0
$t+T_s$	$-s_2^*$	s_1^*	0	0
$t+2T_s$	0	0	s_1	s_2
$t+3T_s$	0	0	$-s_2^*$	s_1^*

The second major MIMO technique is spatial multiplexing (SM). Spatial multiplexing involves sending distinct streams of data over a multipath channel. The number of streams

that can be effectively sent over such a channel is $min\{N_t, N_r\}$. Notice that in spatial multiplexing, multiple streams are sent over the same frequency resource at the same time, hence, this technique increases throughput without increasing the required bandwidth. In an SM-MIMO system, a precoder is used to map the stream of input data to the multiple transmit antennas. The precoder can be a simple serial to parallel converter. In eigenbeamforming, the precoder is a function of the communication channel and thus requires feedback from the receiver.

The second aspect of the SM-MIMO system is a postcoder or spatial decoder/detector. This block processes the signal received at the receive antennas. In spatial multiplexing, each received symbol can be viewed as a complex weighted sum of all the signals transmitted from the transmit antennas. The post-coder decouples each received data stream. Popular decoupling techniques include zero-forcing (ZF), zero-forcing with successive interference cancellation, minimum mean squared error (MMSE), and MMSE-SIC. Zero forcing can be intuitively described as inverting the channel and applying that channel inversion to the received signal vector. This method amplifies the environmental impairments (noise) which are usually modeled as a Gaussian random variable. The MMSE scheme is similar to the zero forcing scheme. The fundamental difference is a non zero value is added to the diagonals of the channel before inversion, this prevents the environmental impairments from being amplified. In the successive interference cancellation schemes, the MMSE or ZF postcoder is sequentially applied to the received signal. First, the receive antenna with the highest SNR is estimated and decoded, then the decoded symbols are processed and subtracted from the symbols of the remaining receive antennas. This process is iterated until all the receive streams have been processed. Arguably, the most important part of any SM-MIMO is the channel. Unlike in conventional communication systems, we want the channel to have a large scattering. The number of possible data streams is limited by the rank of the channel. If we

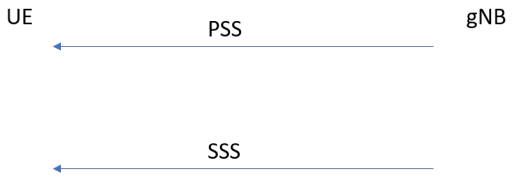
have a channel with a rank of 2, regardless of the number of transmit and receive antennas, we can only send two effective data streams.

2.2 Machine Type Devices

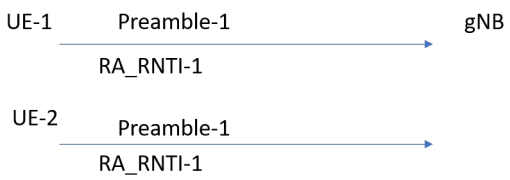
Internet of Things devices are expected to become increasingly prevalent [43] and the type of wireless communication that enables such IoT devices is referred to as "machine type" communications. As the name implies, machine type devices (MTD) are uniquely different from human type devices as the end users are machines, not humans. MTDs will find application in heavily automated tasks like smart metering [44], and in monitoring pressure and other environmental variables. Recently, cellular networks have been proposed as a means to provide inter-connectivity among machine type devices. However, cellular communication is a technology that has been optimized for the mainly downlink-centric cellular transmission. Therefore, the uplink-centric and sporadic nature of machine type transmissions is going to cause a few challenges. One issue lies in the use of the random access process used to assign resources. More specifically, the challenge pertaining to the random access process is the sporadic, event-based nature of machine type communications. The event-based nature of machine type devices can lead to correlated packets arrival times. For instance, if there is a sudden change in the temperature, sensors reporting weather conditions will simultaneously generate a burst of packets. This will lead to an increase in congestion and subsequently a high rate of collisions in the random access process. Therefore, the random access process in cellular networks needs to be improved before it can be used for MTDs.

2.2.1 Random Access

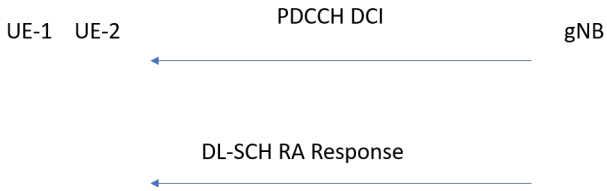
When a UE migrates into a region with cellular coverage, its first job is to synchronize with the base station (gNodeB). In order to achieve this task, it needs to decode the primary synchronization signals (PSS) and secondary synchronization signals (SSS). The former is responsible for slot synchronization while the later is responsible for frame synchronization. The PSS and SSS are both present in the 1st and 11th slots in every frame. The PSS occupies the last OFDM symbol in both slots, while the SSS is present in the symbol before the PSS.



The PSS provides the Physical Layer Cell Identity, $N_{ID}^{(1)} \in \{0, 1, 2\}$, while the SSS provides the Physical Layer Cell Identity Group $N_{ID}^{(2)} \in \{0, \cdots 167\}$. This two values are combined to form the cell ID: $N_{ID}^{cell} = 3 \times N_{ID}^{(1)} + N_{ID}^{(2)}$. The UE proceeds to download the Master Information Block (MIB) from the broadcast channel (BCH) .The MIB contains the information about the cell bandwidth. To complete the downlink synchronization process, the UE tunes to the (downlink shared channel) DL-SCH and downloads the system information block (SIB). The SIB contains parameters needed for initial access - the root sequence index for Zadoff Chu codes, zero correlation zone configurations, physical random access channel (PRACH) frequency, and the PRACH frequency offset. To illustrate the random access procedure, we focus on contention-based random access with two UEs. To start the random access process, the UE randomly transmits one of a number of orthogonal preambles. For illustration purposes, we assume that UE-1 and UE-2 select the same preamble and both transmit at the same time.

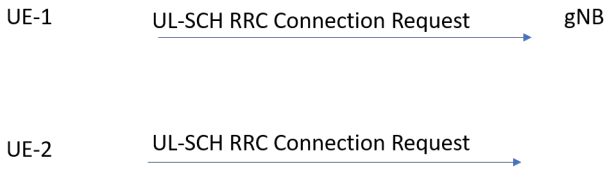


Due to the two UEs selecting the same preamble and transmitting at the same time, they have the same random access-radio network temporary identifier-1 (RA-RNTI-1). For simplicity, we assume that UE-2 loses out on this contention, i.e, the gNB correctly detects the preamble for UE-1, and fails to detect the preamble of UE-2. The gNB estimates the uplink transmission timing of UE-1 and derives RA-RNTI-1. Subsequently, an identifier (cell-RNTI) is assigned to UE-1, and this identifier serves as a placeholder used in addressing UE-1. The two UEs both tune to the physical downlink control channel (PDCCH) and receive the downlink control information (DCI). This information is addressed to RA-RNTI-1. The gNB provides the RA response on the DL-SCH. This RA-response carries the following: temporary C-RNTI-1, Timing Advance, Uplink Resource Grant. It should be emphasized that both UEs receive this message because it is addressed to RA-RNTI-1. Both UEs save the temporary C-RNTI-1 and they both apply the timing advance.

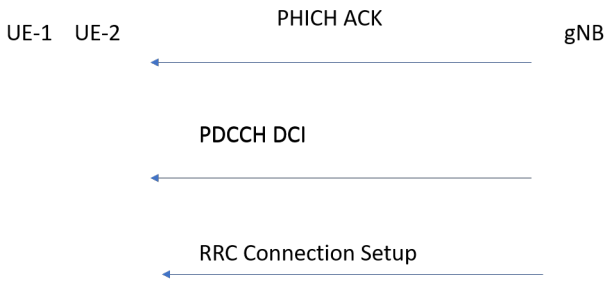


UE-1 and UE-2 don't have a permanent identity, hence, they independently generate two separate random numbers to represent their identities. They include this random identity in their respective radio resource control (RRC) connection request.

Once, this request is made, the two UEs independently start their "T300 timer". UE-2 has transmitted with the uplink resources and timing advance meant for UE-1, hence its transmission will most likely be lost in the resulting collision. The gNB accepts the

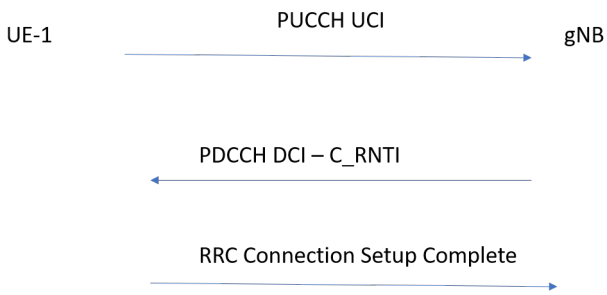


transmission from UE-1 and sends an ACK in (Acknowledgment). A resource assignment is made using the C-RNTI-1. An RRC Connection Setup message addressed with C-RNTI-1 is also sent by the gNB. Both UEs receives these messages.



The RRC Connection Setup addressed by C-RNTI-1 also contains the random number generated by UE-1. UE-1 notices the equivalence between the random number it generated and the random number it received and it sends a Hybrid ARQ. The RA process has succeeded for UE-1 and it stops its T300 timer. Uplink resource requests are made by UE-

1 on the PUCCH, it receives this resource on the PDCCH and finally sends a final RRC connection setup message.



The random number generated by UE-2 and that in the connection setup received by UE-2 don't match. Since, UE-2 can't confirm its identity, UE-2 realizes it has lost the contention. UE-2's T300 timer ends and it retries the random access process. It should be noted that each random access slot has a size of 1.08 MHz (six resource blocks).

2.3 Introduction to Machine Learning

Any algorithm that accepts and learns from data, such that it gets better at performing some task, T can be classified as a machine learning algorithm. The performance is measured and evaluated by a quantitatively defined-metric. Tasks that are difficult to model and solve but for which there is ample data are perfect candidates for a machine learning (ML) algorithm. A common misuse of terminology among researchers is to assume "learning", "task" and "training examples" all mean the same thing. To buttress the differences among these terms, consider an autonomous vehicle that has been trained to perform maneuvers on the highway. Performing maneuvers is the task, the data captured by the vehicular sensors which are usually in the form of pixels are the examples. More specifically, the distinct examples are defined by a feature vector, such that $x \in \mathbb{R}^p$, wherein this case, p represents the number of RGB values characterizing each image. A collection of examples is easily specified by a matrix $X \in \mathbb{R}^{n \times p}$. After specifying the task and collecting examples, we can write a learning algorithm or we can write a deterministic program to provide the vehicular nodes with maneuvering capabilities.

A common trend among machine learning researchers is to classify machine learning problems according to its intended task. A common classification of ML problems is: I) Classification problems: Classification problems involves learning a function mapping $f : \mathbb{R}^p \longrightarrow \{1, ..., k\}$. This category takes a training sample defined by its feature vector and

places the sample in one of k distinct groups. A popular example of such a problem is the identification of animals in pictures. In this problem, a sample example has a feature vector of RGB values and the output is a numerical value defining various classes. II) Regression problems: In this problem, given a training sample, an algorithm is tasked with predicting a continuous numerical value. Mathematically, the algorithm learns the function mapping $f : \mathbb{R}^p \longrightarrow \mathbb{R}$. A common and non-trivial example is predicting the price of future stock options, given a set of past examples $\{X,\mathbf{y}\}$, such that X represents examples and their feature vectors and y represents the corresponding past stock prices.

Another common discriminator of ML problems pertains to the mode of learning. Broadly speaking, the two types of learning algorithms are unsupervised learning and supervised learning algorithms. I) Supervised algorithm: Consider a function $y = f(\mathbf{x})$, a supervised learning algorithm finds an association from the training inputs denoted by $\mathbf{x}$ to corresponding labels y. This association is used to predict label values for previously unseen test data. Examples of supervised learning algorithms include linear regression, logistic regression, and support vector machines. II) Unsupervised learning: The fundamental difference between unsupervised and supervised learning is the absence of a target/label in the former. Unsupervised algorithms attempt to extract information about the underlying data distribution of the training samples. The algorithms learn to cluster training samples with related features into distinct groups. When a new test sample is provided, the algorithm must determine which group placement is most suitable for this previously unseen data. Examples of unsupervised learning include principal component analysis and $k-$means clustering. The authors in [45] present an extensive discussion on machine learning concepts.

Although ML has shown promise in various applications such as object/image classification, house pricing prediction, email classification/spam filtering, and online fraud detection; machine learning becomes less effective as the data dimensions increase i.e., what is known

as the curse of dimensionality. The increase in dimension size (length of feature vector) leads to an exponential increase in the number of possible different feature configurations and the probability that some configurations will not have an associated training example increases. Hence, the output of such examples needs to be approximated. This fundamental problem motivates the need for deep learning.

2.3.1 Deep Learning

Deep learning models are universal function approximators. Consider a function, f^* which denotes the relationship between a dataset X, and its target $\mathbf{y}$. A deep learning model attempts to use the dataset $\{X, \mathbf{y}\}$ to learn a set of parameters, θ which approximates the function f^*. A deep learning model produces the mappings,$\approx \mathbf{y} = f^*(\mathbf{x})$. Although there are other types of deep learning models, in its most basic form, the output of the deep learning model has no feedback loop and it is called a feedforward network. In practice, feedforward models have many layers and can be viewed as functions of functions.

$$f(x) = f^{(l)}(\cdots f^{(2)}(f^{(1)}(\mathbf{x}))) \tag{2.2}$$

The functions f^1 , f^2 , and f^l denotes the first, second, and output layers respectively. A single layer of the feedforward models also known as a perceptron and can be described with the following function:

$$\mathbf{y}^l = \Phi^l(\mathbf{W}^l\mathbf{x}^{l-1} + \mathbf{b}^l) \tag{2.3}$$

The weights, $\mathbf{W}^l \in \mathbb{R}^{N_l \times N_{l-1}}$, and the biases, $\mathbf{b}^l \in \mathbb{R}^{N_l}$, terms are the trainable parameters of the $l-$th layer - $\theta^l = \{\mathbf{w}^l, \mathbf{b}^l\}$. The input into the $l-$th layer, $\mathbf{x}^l \in \mathbb{R}^{N_{l-1}}$, is the output of the $(l-1)$th layer. The entire operation can be viewed as a linear transformation

followed by a nonlinear activation function. The non-linearity of the activation functions is the expressive power of the neural networks.

Table 2.4: Activation functions

Name	Function $\Phi(\mathbf{q})$	Range
Linear	q_i	$(-\infty, \infty)$
TanH	$\tanh(q_i)$	$(-1, 1)$
RELU [46]	$\max(0, q_i)$	$[0, \infty]$
Leaky RELU [47]	$\begin{cases} \alpha\mathbf{q}, & \text{for } \mathbf{q} < 0 \\ \mathbf{q}, & \text{for } \mathbf{q} \geq 0 \end{cases}$	$[-\infty, \infty)$
Sigmoid	$\frac{1}{1+e^{-q}}$	$(0, 1)$
ArcTan	$tanh^{-1}(q_i)$	$(-\infty, \infty)$
ELU [48]	$\begin{cases} \mathbf{q}, & \text{for } \mathbf{q} > 0 \\ \alpha e^{\mathbf{q}} - \alpha & \text{for } \mathbf{q} \leq 0 \end{cases}$	$(0, \infty)$
SoftPlus [49]	$In(1 + exp(\mathbf{q}))$	$(0, \infty)$
SoftMax [50]	$\frac{e^{\mathbf{q}_i}}{\sum_{j=0}^{j=N} e^{\mathbf{q}_j}}$	$(0, 1)$
Swish [51]	$\frac{\mathbf{q}}{1+e^{-\mathbf{q}}}$	$(0, \infty)$

In the absence of this expressive power, there would be no real gain in using a multiple layered neural network. The activation function operates on its input vector, by operating independently on each of its elements. Hence, $[\Phi(\mathbf{q})]_i = \Phi(\mathbf{q}_i)$. Table 2.4 presents a comprehensive list of common activation functions. Figures 2.1, 2.2, 2.3, 2.4, and 2.5 present graphical illustrations of the activation functions used in this work. In its most simplistic form, the operation of a deep learning model can be divided into two pieces- the forward pass and the backward pass. The forward pass produces an approximate output, $\hat{\mathbf{y}} = f(\mathbf{x}; \theta)$. This approximation along with a convex loss functions are used in the backward pass of the

neural network operation.

Table 2.5: Loss functions

Name	$l(\hat{\mathbf{y}}, \mathbf{y})$
Binary Entropy Loss	$-\mathbf{y}\log(\hat{\mathbf{y}}) - (1-\mathbf{y})\log(1-\hat{\mathbf{y}})$
Minimum Squared Error	$\|(\hat{\mathbf{y}} - \mathbf{y})\|^2$
Minimum Absolute Error	$\mid (\hat{\mathbf{y}} - \mathbf{y}) \mid$
Cross Entropy Loss	$-\sum_j^M \mathbf{y}_j \log(\hat{\mathbf{y}})$

The loss function produces a comparison between the neural network approximations, $\hat{\mathbf{y}}$, and the target variables, $\mathbf{y}$. Table 2.5 provides examples of common loss functions. It should be noted that the loss functions are application/problem specific. In Regression-based deep learning applications, both the target variable, $\mathbf{y}$, and the estimates produced by the neural network $\hat{\mathbf{y}}$ are continuous and real-valued, hence the minimum squared and minimum absolute error are suitable loss functions. For classification problems, the neural network produces a continuous real number between the range (0,1). This restriction in the range is usually enforced by applying the sigmoid (binary classification) or softmax (multinomial classifications) activation functions. The target variables in classification problems are either 1,0, i.e $\mathbf{y}_i \in \{0, 1\}$. In practice, this value is one-hot encoded such that the target vector for a single data sample consists of all zeros except at the index of the correct class label. The index representing the correct label contains a "1" as its value.

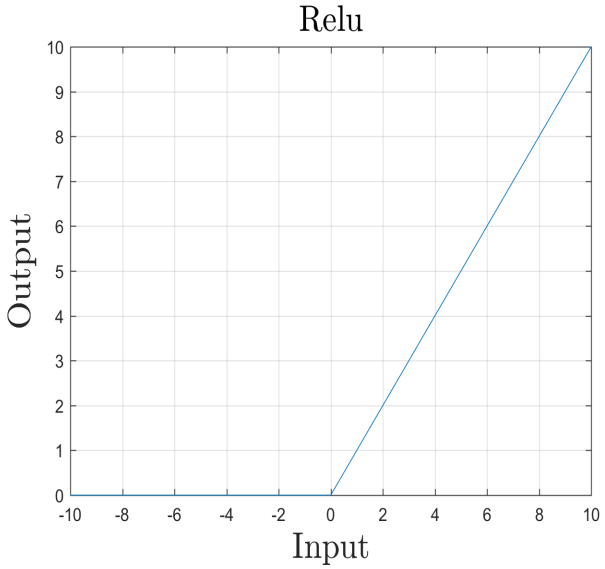


Figure 2.1: A RELU Activation Function.

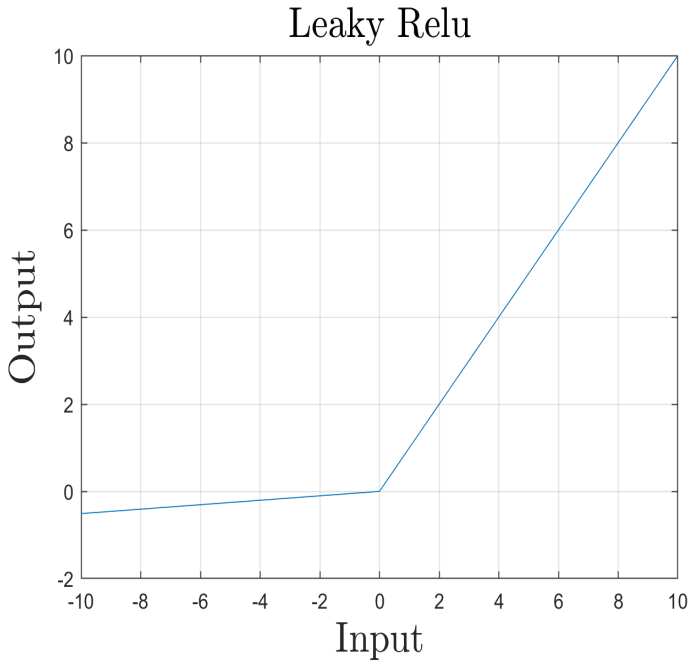


Figure 2.2: The Leaky RELU Activation Function.

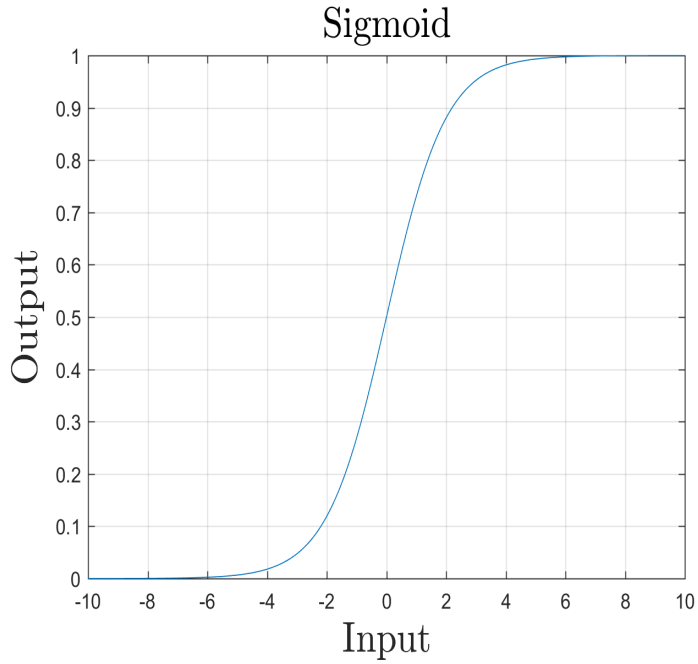


Figure 2.3: The Sigmoid Activation Function.

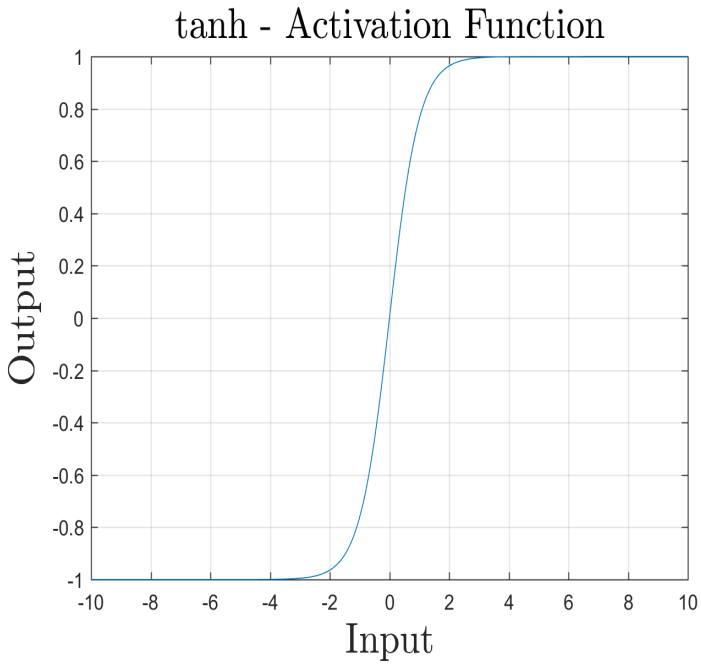


Figure 2.4: The Tanh Activation Function.

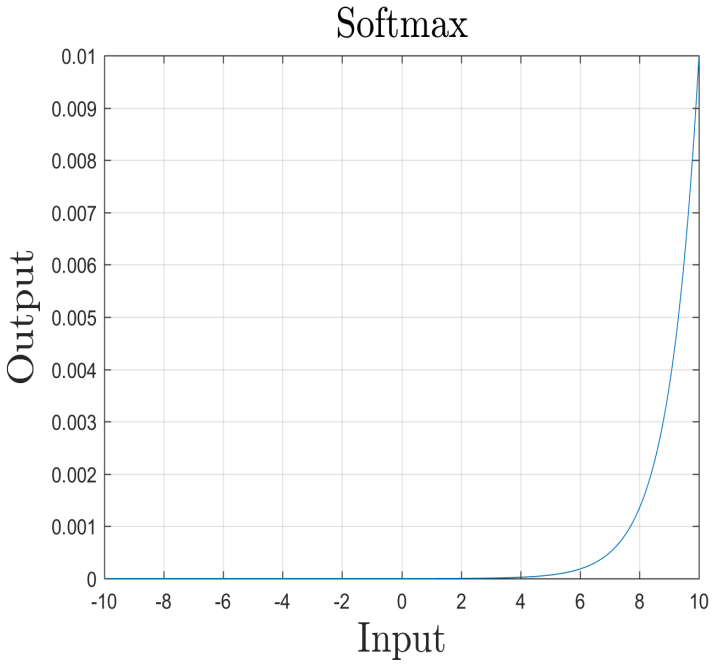


Figure 2.5: The Softmax Activation Function.

The backward pass entails propagating the loss from the outermost layer throughout the entire network in order to update the parameters of the neural network. The popular back-propagation algorithm provides an intuitive way of achieving this [52]. At a time $t+1$, with a learning rate, γ, the back-propagation algorithm can be described with the following equation:

$$\theta_{t+1} = \theta_t - \gamma \frac{\partial l(\hat{\mathbf{y}}, \mathbf{y})}{\partial \theta_t} \tag{2.4}$$

Although gradient descent has shown good performance [53, 54], researchers have recently proposed other propagation algorithms. Authors in [55], improve upon the gradient descent algorithm by dynamically updating the learning rate. This method is referred to as momentum, and provides gradient stability and accelerates descent across regions of low steepness. A simple parameter update with gradient descent and momentum is shown in Equation 2.5:

$$\begin{aligned} Q_{t+1} &= \beta Q_t + (1-\beta)\frac{\partial l(\hat{\mathbf{y}}, \mathbf{y})}{\partial \theta_t} \\ \theta_{t+1} &= \theta_t - \gamma Q_{t+1} \end{aligned} \tag{2.5}$$

$\beta \in [0, 1]$ is a hyperparameter that can be used to tune the performance of the learning system. Recently, the Adam algorithm [56] has seen its popularity increase among researchers over the past couple of years. This popularity is due to its ability to deal with stochastic objective functions while maintaining a simplistic implementation. The algorithm can be written using the following weight update:

while θ_t not converged **do**

$t \leftarrow t + 1$

$m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1)\frac{\partial l(\hat{\mathbf{y}},\mathbf{y})}{\partial \theta_{t-1}}$

$q_t \leftarrow \beta_2 q_{t-1} + (1 - \beta_2)(\frac{\partial l(\hat{\mathbf{y}},\mathbf{y})}{\partial \theta_{t-1}})^2$

$\tilde{m}_t \leftarrow m_t/(1 - \beta_1^t)$

$\tilde{q}_t \leftarrow q_t/(1 - \beta_2^t)$

$\theta_t \leftarrow \theta_{t-1} - \gamma \tilde{m}_t/(\sqrt{\tilde{q}_t} + \epsilon)$

end while

The $(\frac{\partial l(\hat{\mathbf{y}},\mathbf{y})}{\partial \theta_{t-1}})^2$ term represents elementwise squaring of the gradient vector. In addition, to the learning rate, γ; $\beta_1, \beta_2 \in [0, 1]$ and ϵ are also hyperparameters. For much of chapter 3, a deep neural model is employed and optimized through the Adam algorithm.

2.3.2 Reinforcement Learning

A novice would be forgiven for expecting previously discussed learning techniques to be all-encompassing of all the available learning techniques, but this is not the case. A special class of learning involves mapping circumstances to actions, in order to ensure that a numerical reward is being maximized. This category of learning is referred to as reinforcement learning. This model depicts a learner that can start off as myopic and sequentially becomes intelligent by leveraging its past experiences. A reinforcement learning system has no instruction manual, the learner is never told what to do, however, the system is designed as a closed-loop system such that the learner can receive numerical feedback on the consequences of its actions.

A distinction between reinforcement learning and supervised learning is the absence of labeled data. In supervised learning, the learning parameters are trained using labeled examples, subsequently, the learning algorithm with these learning parameters attempts to

predict the label of previously unseen data. Clearly, these parameters are trained to generalize, however, in interactive systems where the actions predicted by the algorithm generates new training samples and a new action sample space, it becomes important that learning systems have the ability to sequentially improve their parameters through experiences. To solve this problem, reinforcement learning models are used to learn from their past experiences. Obviously, learning from experience implies that the learning system must have a target goal and behavior. The target goal of any reinforcement learning problem is completely described by a reward signal. Reinforcement learning is supposed to provide an intuitive solution to the conundrum of goal-based learning from interaction.

In practice, reinforcement learning solutions are usually framed as Markov decision processes (MDPs). The formulation of a Markov decision process can be divided into an agent and its environment. The agent makes decisions based on a full or partial observation of the environment. The environment responds to the decisions by providing both a reward signal and new environmental situations to the agent. More concisely, at a discrete-time step, t, an agent observes either a full or partial representation of the environment, $S_t \in S$. Through a stochastic or deterministic policy conditioned on this observation $\pi(a_t|s_t)$, an action is selected, $A_t \in A(S_t)$. Note that, S, is the set of all possible states and $A(S_t)$ is the set of all possible actions provided that we are in state, S_t. Subsequently at the next time step, the agent receives a numerical reward, $R_{t+1} \in \mathbb{R}$ and transitions into a new state S_{t+1}.

This continuous interaction between the agent and environment creates a sequence of state, action, and reward. Notice that for a finite MDP, the cardinality of the set of all possible states, actions, and rewards must be finite. For this special finite MDP case, the rewards and states are random variables with properly defined discrete probability distribu-

tions. These random variables exhibit the Markov property:

$$p(s^{'}, r|s, a) = Pr\{S_t = s^{'}, R_t = r|S_{t-1} = s_{t-1}, A_{t-1} = a_{t-1}\} \tag{2.6}$$

where, $s^{'}$ denotes the future state. The Markovian property implies that current states include all the past information which would influence future states.

$$\sum_{s^{'} \in S} \sum_{r \in R} p(s^{'}, r|s, a) = 1, \forall s \in S, a \in A(s) \tag{2.7}$$

In Equation 2.6, p is a probability variable denoting the dynamics of the Markov decision process. With the Markovian property and the dynamics variable, we have all the information pertaining to the environment, and any relevant information can be generated. For example, the state transition probabilities can be computed with

$$p(s^{'}|s, a) = \sum_{r \in R} p(s^{'}, r|s, a) \tag{2.8}$$

while, the reward for a specific state action pair can be computed as

$$r(s, a) = \sum_{r \in R} r \sum_{s^{'} \in R} p(s^{'}, r|s, a) \tag{2.9}$$

In order, to formally and mathematically define the reward structure, consider the reward received from the environment after time step t: $R_{t+1}, R_{t+2}, \cdots, R_T$, where T indicates the final time step. The question arises: how do we want to maximize this sequence? An intuitive answer would be, to maximize the sum of the total sequence. This sum of sequential rewards is a simple form of the return.

$$G_t = R_{t+1} + R_{t+2} + \cdots + R_T \tag{2.10}$$

This naive approach could work well for a sequence with a finite length, $T < \infty$. For such sequences, the task ends when the agent moves into a terminal state, S^*. Reinforcement learning problems that cause agents to interacts with environments such that the generated sequence of states ends in a terminal state are termed episodic tasks and the resulting sequences are referred to as episodes. In contrast, tasks that are designed in such a way that there are no terminal states are referred to as a continuing task. For continuing tasks, the sum of the total reward would be infinite, hence researchers introduced a discounting factor, γ. In a continuing task, the agents select actions that maximize the cumulative sum of its immediate reward and a discounted sum of all future rewards.

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{2.11}$$

Equation 2.12 presents a generalised definition of the return. The discount factor can only take values between zero and 1, $0 \leq \gamma \leq 1$. With a discount factor of zero, the agent only considers immediate reward, and as this factor is increased from zero to one, the agent becomes less myopic. However, a necessary condition for boundedness is $\gamma < 1$. The return can be recursively written as:

$$G_t = R_{t+1} + \gamma G_{t+1} \tag{2.12}$$

A term prevalent to most reinforcement learning algorithms is the value function. The value function denotes the quality of a certain state in terms of future expected return. However, the expected reward is determined by the behavior of the agent - its policy. Hence, the value function is derived by evaluating its expected return as a function of its policies. Strictly speaking, the value function of a specific state, s, is the expected amount of reward

an agent can accrue if it starts out at that state and follows a certain policy, π.

$$v_\pi(s) = \mathbb{E}_\pi \left[G_t | S_t = s\right] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s\right], \forall s \in S \tag{2.13}$$

Likewise, an action value function can defined as the expected future return if an agent starts at a state, $S_t \in S$ takes action, $A_t \in A$ and subsequently follows a policy, π.

$$q_\pi(s, a) = \mathbb{E}_\pi \left[G_t | S_t = s, A_t = a\right] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a\right] \tag{2.14}$$

From equations 2.13 and 2.14, the state value function is derived by averaging all its action values.

$$v_\pi(s) = \sum_{a^{'} \in A(s)} \pi(a^{'}|s) q_\pi(s, a^{'}) \tag{2.15}$$

Similar to the return function 2.12, the value and action functions of a specific state has a recursive relationship with its successive states.

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi \left[G_t | S_t = s\right] \\ v_\pi(s) &= \mathbb{E}_\pi \left[R_{t+1} + \gamma R_{t+2} + \cdots | S_t = s\right] \\ v_\pi(s) &= \mathbb{E}_\pi \left[R_{t+1} + \gamma G_{t+1} | S_t = s\right] \\ v_\pi(s) &= \mathbb{E}_\pi \left[R_{t+1} | S_t = s\right] + \mathbb{E}_\pi \left[\gamma G_{t+1} | S_t = s\right] \end{aligned} \tag{2.16}$$

Using the law of total expectation;

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi\left[R_{t+1}|S_t = s\right] + \gamma\mathbb{E}_\pi\left[\mathbb{E}_\pi\left[G_{t+1}|S_{t+1}\right]|S_t = s\right] \\
v_\pi(s) &= \mathbb{E}_\pi\left[R_{t+1}|S_t = s\right] + \gamma\mathbb{E}_\pi\left[v_\pi(s_{t+1})|S_t = s\right] \\
v_\pi(s) &= \mathbb{E}_\pi\left[R_{t+1} + \gamma v_\pi(s_{t+1})|S_t = s\right]
\end{aligned} \tag{2.17}
$$

Similarly, the action value function can be expressed as

$$
\begin{aligned}
q_\pi(s,a) &= \mathbb{E}_\pi\left[G_t|S_t = s, A_t = a\right] \\
q_\pi(s,a) &= \mathbb{E}_\pi\left[R_{t+1} + \gamma R_{t+2} + \cdots|S_t = s, A_t = a\right] \\
q_\pi(s,a) &= \mathbb{E}_\pi\left[R_{t+1} + \gamma G_{t+1}|S_t = s, A_t = a\right] \\
q_\pi(s,a) &= \mathbb{E}_\pi\left[R_{t+1}|S_t = s\right] + \mathbb{E}_\pi\left[\gamma G_{t+1}|S_t = s, A_t = a\right]
\end{aligned} \tag{2.18}
$$

Again with the law of total expectation

$$
\begin{aligned}
q_\pi(s,a) &= \mathbb{E}_\pi\left[R_{t+1}|S_t = s\right] + \gamma\mathbb{E}_\pi\left[\mathbb{E}_\pi\left[G_{t+1}|S_{t+1}\right]|S_t = s, A_t = a\right] \\
q_\pi(s,a) &= \mathbb{E}_\pi\left[R_{t+1}|S_t = s\right] + \gamma\mathbb{E}_\pi\left[v_\pi(s_{t+1})|S_t = s, A_t = a\right] \\
q_\pi(s,a) &= \mathbb{E}_\pi\left[R_{t+1} + \gamma v_\pi(s_{t+1})|S_t = s, A_t = a\right]
\end{aligned} \tag{2.19}
$$

The last equation in 2.17 and 2.19 denotes the Bellman equation for v_π and q_π respectively. Notice that the bellman equations are dependent on the policy, π. Hence, for a finite MDP we can take advantage of the bellman equations in order to define an optimal policy, π^*. More specifically, a policy π^* is better than every other policy, π, if the value function $v_{\pi^*}(s)\ \forall s$ derived by the following policy, π^*, is equal or greater than the value functions derived from following any other policy, $v_\pi(s)\ \forall s$. Although every finite MDP has an optimal policy, it doesn't have to be unique. All equivalent polices share the same state-value

functions termed the optimal state-value function, v_*.

$$v_*(s) = \max_{\pi} v_\pi(s) \tag{2.20}$$

Obviously, the optimal state value function is equivalent to the expected return derived by selecting the action with highest action value.

$$\begin{aligned} v_*(s) &= \max_{a \in A(s)} q_{\pi_*}(s,a) \\ v_*(s) &= \max_{a} \mathbb{E}_{\pi_*}\left[R_{t+1} + \gamma G_{t+1} | S_t = s, A_t = a\right] \\ v_*(s) &= \max_{a} \mathbb{E}_{\pi_*}\left[R_{t+1} + \gamma v_*(S_{t+1}) | S_t = s, A_t = a\right] \\ v_*(s) &= \max_{a} \sum_{s^{'},r} p(s^{'}, r|s,a)\left[r + \gamma v_*(S_{t+1})\right] \end{aligned} \tag{2.21}$$

The Bellman optimality equation for $v_*(s)$ is represented by the last two equations . Similarly, the the optimality eqaution for $q_*(s)$ is

$$q_*(s,a) = \mathbb{E}\left[R_{t+1} + \gamma \max_{a^{'}} q_*(S_{t+1}, a^{'}) | S_t = s, A_t = a\right] \tag{2.22}$$

Given the optimal values for state value function $v_*(s)$, the best policy is one that is greedy with respect to this value function. According to Equation 2.21, acting greedily entails performing a one-step look ahead and evaluating all the value functions of all possible successor states and selecting the action that leads to the state with the highest expected return. It is easy to see that in a system with a large number of successor states, this one-step look ahead will become expensive. To minimize this computational cost, the optimal action-value function can be used. With, q_* the agent simply selects the action that maximizes the expected return, $q_*(s,a)$. In essence, the action value provides a locally available cache of the expected return. Hence, the one-step look ahead is eliminated at the cost of storing the ex-

pected return of each state-action value pair. Considering this, we still have the problem of obtaining the optimal action/state value functions. Since the Bellman equations represent a nonlinear system of equations, this problem can be solved through various readily available techniques. However, explicitly solving the Bellman optimality equations is analogous to performing an exhaustive search and this search depends on the following:

- We have full knowledge of the system dynamics, p.
- We have enough computation resources to evaluate all states.

Clearly, for most real-world problems with a high number of possible states, these assumptions are not applicable. Consequently, researchers usually settle for approximate solutions. Two examples of popular reinforcement learning approximations will be discussed in the following sections.

Dynamic Programming

Dynamic programming (DP) techniques provide the backbone for most of the algorithms used in reinforcement learning. Algorithms that fall under the dynamic programming framework employs the complete system dynamics of the environment as a Markov decision process in order to iteratively find the optimal policy. A pertinent idea to dynamic programming is the transformation of the Bellman equations into update rules, subsequently, these update rules are used to iteratively improve the approximations of the value functions. In order to evaluate a certain policy, first, the state-value function derived from that policy is transformed into an iterative update scheme. This transformation is used to sequentially compute the state-value function, provided that a given policy, π is followed. This is sequence is termed iterative policy evaluation. Mathematically, this sequential update is derived from the last expectation in 2.17:

$$
\begin{aligned}
v_{k+1}(s) &= \mathbb{E}_{\pi}\left[R_{t+1} + \gamma v_{k+1}(s_{t+1}) | S_t = s\right] \\
v_{k+1}(s) &= \sum_{a} \pi(a|s) \sum_{s^{'},r} p(s^{'}, r|s, a)\left[r + \gamma v_{k+1}(s_{t+1})\right]
\end{aligned}
\tag{2.23}
$$

It is important to state that regardless of the initial state-value, v_0, the sequence, $v_0, v_1, \cdots$ converges to v_{π} in the limit. Equation 2.23 operates by successively approximating the new state-value function from both the immediate reward and all possible previous successor states value function. The iterative policy evaluation algorithm is discussed below. Notice that, since this method only converges in the limit, a stopping criterion is usually imposed in practice.

Algorithm 1: Iterative Policy Evaluation

Result: The approximated state-value function $Va(s)$

Initialize the policy, π, the stopping criterion, δ and the state-value function, $Va(s)\ \forall s$;

while $\delta > \Omega$ **do**

  $\Omega \leftarrow 0$;

  foreach $s \in S$ **do**

    $v \leftarrow Va(s)$;

    $Va(s) \leftarrow \sum_a \pi(a|s) \sum_{s^{'},r} p(s^{'}, r|s, a)\left[r + \gamma Va(s^{'})\right]$;

    $\Omega \leftarrow max(\Omega, |v - Va(s)|)$;

  end

end

After approximating the state-value functions, we have an estimate of how good it is to start from the state, s, and follow a policy, $\pi(s)$. One question lingers - is this the best policy to follow? One way to answer this question is to select an action from a different

policy, $a = \pi^{'}(s) \neq \pi(s)$, and subsequently follow the existing policy, π. This is defined mathematically as

$$\begin{aligned} q_\pi(s, \pi^{'}(s)) &= \mathbb{E}_\pi \left[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s, A_t = a \right] \\ q_\pi(s, \pi^{'}(s)) &= \sum_{s^{'}, r} p(s^{'}, r | s, a) \left[r + \gamma v_\pi(S^{'}) \right] \end{aligned} \tag{2.24}$$

If the action value function calculated in this way is greater than the current state value function derived form following policy, π, when in-state s, then it would be better to always select $a = \pi^{'}(s)$ whenever this state is encountered. This idea is a simplification of the policy improvement theorem. The policy improvement theorem states, for all states, $s \in S$, if there are two nonstochastic polices, π and $\pi^{'}$, which returns:

$$q_\pi(s, \pi^{'}(s)) \geq v_\pi(s) \tag{2.25}$$

then

$$v_{\pi^{'}}(s) \geq v_\pi(s) \tag{2.26}$$

This theorem paves the way for iterative policy improvement. If we consider a new policy described as:

$$\pi^{'}(s) = \arg\max_a q_\pi(s, a) \tag{2.27}$$

By construction this new policy satisfies the policy improvement theorem, hence its state value function, $v_{\pi^{'}}(s)$ is equal or greater than the state value function derived from the old policy. Therefore, a new policy that performs better or equal to old policy can be iteratively generated by acting greedily on the state value functions of the old policy. This process is called policy improvement. If the new policy generates value functions equal to the value

function of the old policy, $v_{\pi'}(s) = v_{\pi}(s)$ then from Equation 2.27, we have:

$$\begin{aligned} v_{\pi'}(s) &= \max_a \mathbb{E}_{\pi}\left[R_{t+1} + \gamma v_{\pi}(S_{t+1}) | S_t = s, A_t = a\right] \\ v_{\pi'}(s) &= \max_a \sum_{s',r} p(s', r|s, a)\left[r + \gamma v_{\pi}(S')\right] \end{aligned} \tag{2.28}$$

Clearly, this equation is equal to the Bellman optimality equation 2.21. Therefore, both the new and old policies are optimal, $v_{\pi'}(s) = v_*(s)$. We can conclude that an iteration of policy evaluation will generate a strictly better policy except when the old policy is already optimal. This fact provides us with the policy iteration algorithm.

The policy iteration algorithm can be described as:

- Upon evaluating a policy, π, through its state value functions, $v_{\pi}(s)$.
- We improve this policy by acting greedily with respect to the new value functions - we get a new policy π'.
- We evaluate this policy to get its state value functions, $v_{\pi'}(s)$.
- This policy is improved again by taking greedy actions, this results in a new policy, π''
- Once again this policy is evaluated to get, $v_{\pi''}(s)$.
- This process is continued until policy evaluation generates a policy, such that the new state value functions and the previous state value functions are equal and therefore optimal.

The pseudo-code for this algorithm can be found in [57]. It is clear that every sequence of policy iteration involves evaluating the current policy, v_{π}. However, since v_{π} is derived only in the limit, this becomes a very expensive procedure. Hence, this problem led to the

formulation of another class of solutions referred to as value iteration. Value iteration is a special kind of policy iteration where the policy evaluation is truncated after a single sweep of all the states. Surprisingly, the value iteration equation is equivalent to transforming the Bellman optimality equation into an update rule:

$$\begin{aligned} v_{k+1}(s) &= \max_a \mathbb{E}_\pi \left[R_{t+1} + \gamma v_k(S_{t+1}) | S_t = s, A_t = a \right] \\ v_{k+1}(s) &= \max_a \sum_{s',r} p(s', r|s, a) \left[r + \gamma v_k(s') \right] \end{aligned} \tag{2.29}$$

The value iteration algorithm aggregates a single policy evaluation iteration with a single pass of policy improvements. The complete pseudo-code for value iteration is found in [57]. Although policy and value iteration provides a method of both policy evaluation and policy improvement, they are both constrained due to their dependence on the availability of the system dynamics. In other words, dynamic programming methods are dependent on the complete knowledge of the system model. Since a derivation of the complete system dynamics are usually intractable for most interesting problems. Model-free solutions are used to circumvent this challenge by approximating the system dynamics from experienced data.

Model Free Algorithms

Model-free algorithms are usually classified as either: I) Monte Carlo based (MC) algorithms or II) Temporal Difference (TD) algorithms. TD and Monte Carlo algorithms update an estimate, V of the actual value functions, v_π, based on experiences gained by following a policy, π. The difference in the two algorithms lies in the manner of update. Monte Carlo algorithms wait until the episodes terminate and return, G_t is available. MC algorithms

approximate the actual state value function given by Equation 2.17 as

$$V(S_t) = V(S_t) + \alpha \left[G_t - V(S_t) \right] \tag{2.30}$$

MC methods can be slow since the learning episodes could take a long time to terminate. Temporal difference algorithms provide a simple alternative. TD algorithms provide an update by bootstrapping the current return, R_t, with an estimate of the next state action value, $V(S_t)$. In other words, temporal difference learning approximate Equation 2.17, as

$$V(S_t) = V(S_t) + \alpha \left[R_{t+1} + \gamma V(S_{t+1}) - V(S_t) \right] \tag{2.31}$$

As with dynamic programming, after evaluating the state value function derived from a policy, π, the next step is to improve on that policy. However, unlike dynamic programming the state value functions are not sufficient for policy improvement. With a model (dynamic programming), during policy improvement, a simple look-ahead provides the best action to take in order to achieve the maximum reward. Without a model, this simple look ahead is not possible without computing the action-value functions. Hence, model-free methods are largely dependent on estimating the action-value functions, $Q(S_t, A_t)$. Estimating the state-value functions through TD learning can be described as

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t) \right] \tag{2.32}$$

Notice that if the policy which generates the next action, A_{t+1} is deterministic, learning from experience can be greatly hindered. This is because some actions will never be selected and the corresponding state-action values will never improve with experience and

this undermines the entire goal of learning. To solve this problem and ensure that all state-action pairs are visited, a probabilistic constraint is usually imposed on the learning process. This constraint demands that at every time step, the agent has a non-zero chance of picking every possible action. The manner in which this constraint is imposed defines two classes of learning: I) on-policy: here, the agent attempts to improve the policy used to generate the S_t, A_t, R_t, S_{t+1} sequence, and II) off-policy: the agent improves a policy that is different from the policy producing the sequence transitions. In its most simplified form, the on policy update scheme for TD learning is given by Equation 2.32, with the slight modification, that the next action, A_{t+1}, need for bootstrapping is derived according to, $A_{t+1} \leftarrow \begin{cases} \arg\max_a Q(S_{t+1}, a), & \text{With probability } 1-\epsilon \\ \text{any action a}, & \text{With probability } \epsilon \end{cases}$ This is referred to as ϵ-greedy selection, with optimization hyperparameter, ϵ. The resulting algorithm is referred to as Sarsa [57]. The off-policy TD algorithm [58] has a slight modification.

$$Q(S_t, A_t) = Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right] \tag{2.33}$$

Here, the present state is picked according to the $\epsilon-$greedy strategy while the q-value used for bootstrapping is selected according to a greedy policy. This algorithm is clearly off-policy, with the behavioral policy specified by the $\epsilon-$greedy strategy, and with a completely greedy target policy. The resulting algorithm is referred to as Q-learning. Chapter 4 employs a variation of this Q-learning scheme.

Chapter 3

Learning Based Multi-Antenna Transmission Systems

The modularized nature of communication systems has proved an efficient and effective model. However, researchers have recently investigated the possibility of exploiting potential gains by combining the modularized functions into a single function [3]. A feedforward autoencoder system presents us with a method of achieving this goal. To this end, this chapter investigates the possibility of an end to end diversity achieving system for different transmit and receive configurations.

To present a learned diversity system for combating the effects of a previously unknown channel, we train the learned system offline by providing the encoder with a batch of symbol indices, the encoder produces symbol representations for each index. The number of representations generated depends on both the designated number of transmit antennas and the code rate. The symbols are passed through the multiplicative effects of a Rayleigh channel and additive effects of Gaussian noise. At the decoder, the original symbols are extracted. Subsequently, the loss is computed through a suitable convex loss function, and backpropagation is used to update the weights of both the encoder and decoder. For training purposes, we use offline training which makes this backpropagation possible. During testing, random and previously unseen instantiations of the channel are used to evaluate the symbol error rate at the varying signal to noise ratios. This allows us to determine if the system has

generalized the response to the wireless channel. The noise power is used to vary the signal to noise ratio.

The transmitter design mandates that a matrix of complex representations is generated for each symbol index. This unique structure employs us to investigate the possibility of combating the impact of channel correlation at both the transmit and receive antennas. To investigate the impact of correlation, we train the encoder and decoder with independent instantiations from a Rayleigh distribution. After training, the learned system is tested in the presence of correlation at either the transmit or receive antennas. Our results indicate that while the performance of the learned system is not hampered by correlation among the transmit antennas, it degrades substantially with correlated receive antennas.

Lastly, we recognize that in the real-world the wireless propagation environment is constantly changing. Hence, a learned system might operate in a different wireless propagation environment than the propagation environment used during training. Considering this, any learned system should be resilient to changes in the channel. To investigate the robustness of our system, we present the symbol error rate achieved at varying signal to noise ratios when there is a mismatch in the distribution used during training and testing. The encoder and decoder are trained with independent instantiations from a Rayleigh distribution. After, training, the learned system is tested by sampling from a Nakagami distribution. Results show that the learned system is robust to variations in the wireless propagation environment.

3.1 System Model

A learned system - completely described by a neural network - tries to communicate one out of a possible of $T = 2^k$ symbols. The learned system can be visualized as a three-part system consisting of the encoder/transmitter, decoder/receiver, and the channel [see Figure 3.1].

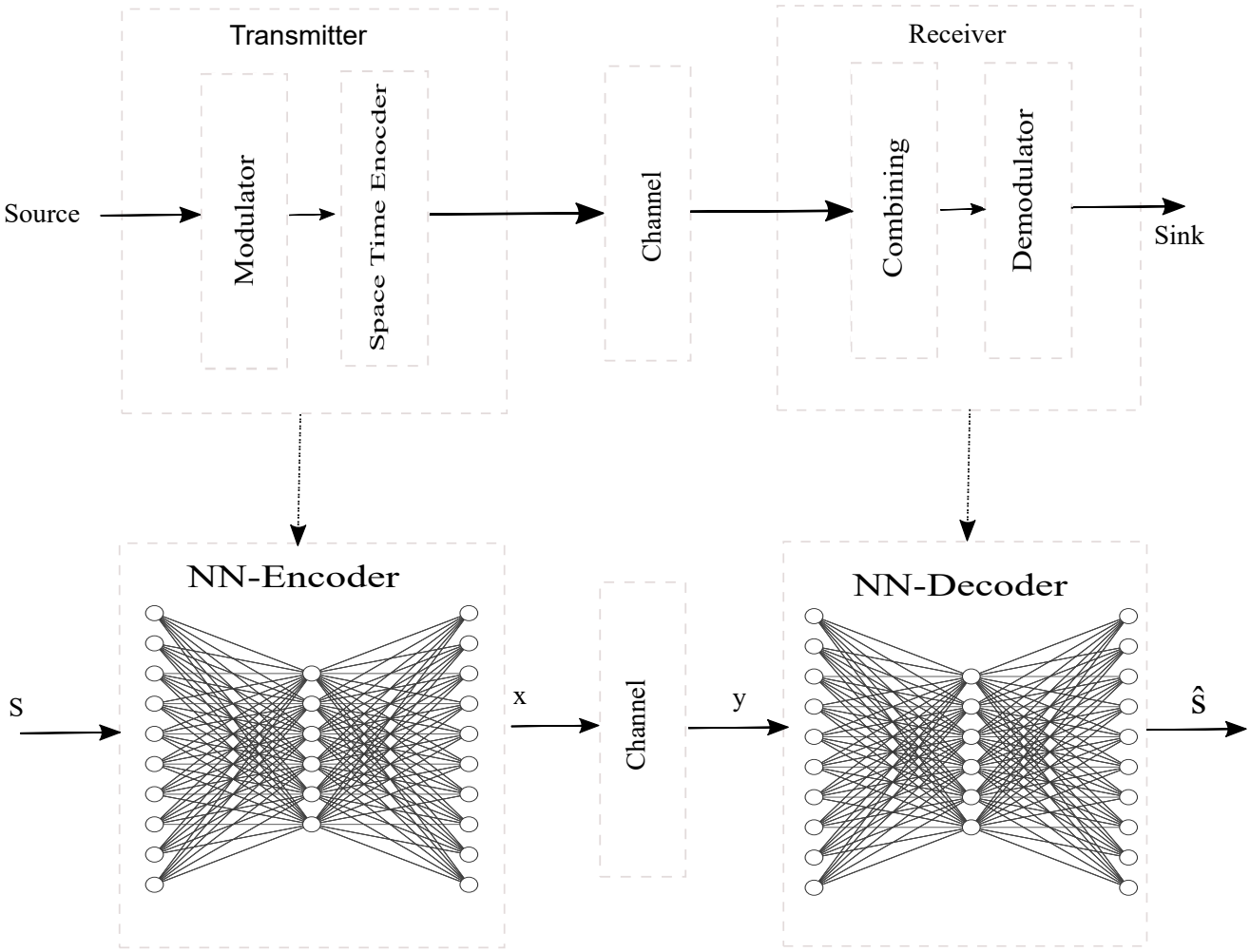


Figure 3.1: Autoencoder-Based System Layout.

Unlike previous works, a differentiable channel model is not necessary. The need for a differentiable channel model arises due to gradient evaluation during backpropagation, we

circumvent this by by-passing the channel during backpropagation. The gradient from the input neural layer of the decoder is passed to the output layer of the encoder. This operation is plausible because backpropagation is carried out by the product rule. During learning, a symbol, s is fed into the encoder and reconstructed at the decoder, $\hat{s}$. The autoencoder is modeled with a deep feedforward neural network, the weights and bias are modeled with a parameter θ. The encoder operation is described by the following function.

$$\mathbf{S} = f_T(s; \theta_T) \tag{3.1}$$

Each symbol, s, is encoded by the transmitter parameters, θ_T, across space and time to form a matrix of symbols

$$\mathbf{S}_i = \begin{pmatrix} \tilde{s}_{11} & \tilde{s}_{12} & \dots & \tilde{s}_{1K} \\ \tilde{s}_{21} & \tilde{s}_{22} & \dots & \tilde{s}_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{s}_{N_t1} & \tilde{s}_{N_t2} & \dots & \tilde{s}_{N_tK} \end{pmatrix} \tag{3.2}$$

where the matrix $\mathbf{S} \in \mathbb{C}^{Nt \times K}$, and its individual entries $\tilde{s}_{jl}$ represent the learned representation on the $j-$th antenna at the $l-$th time instants. Clearly, every input symbol to the transmitter occupies K different channel uses. Hence, the symbol code rate is easily specified by $\frac{1}{K}$. At a single time instant, a single column of the matrix in Equation 3.2 is transmitted. We define such a column as $\mathbf{x}$. The received signal at each time instant, modeled as $\mathbf{y}_i = \mathbf{H}_i\mathbf{x}+\mathbf{n}_i$, where $\mathbf{x} \in \mathbb{C}^{Nt}$, $\mathbf{H}_i \in \mathbb{C}^{Nr \times Nt}$ represents the channel impairments modeled by Rayleigh fading, and $\mathbf{n}_i \in \mathbf{C}^{Nr}$ represents Gaussian noise. The decoder has two blocks: the preprocessing block and the signal recovery block. The decoder also takes as input, K sequence of received signals, $\mathbf{Y} = \{\mathbf{y}_k\}_{k=1}^{k=K}$. The decoder produces a probability

vector, where the components of $\mathbf{v}$ represent the estimated posterior probabilities for each possible message. The decoder operation can be represented using the following function:

$$\mathbf{v} = f_R(\mathbf{Y}; \theta) \tag{3.3}$$

Finally, the symbol is estimated as $\hat{s} = \arg\max_i [\mathbf{v}]_i$. $[\mathbf{v}]_i$, denotes the $i-$th element of a vector $\mathbf{v}$.

3.1.1 Deep Learning Based Transmitter Design

The transmitter is designed using a feedforward neural network. A vector of input symbols of batch size M is passed through an embedding layer followed by a single hidden dense layer and an output layer. The embedding layer is essentially a lookup table. It converts positive integers into a row vector of fixed size, T. It is essentially a more efficient implementation of one-hot encoding. The elements of the lookup table are initialized randomly using Xavier's method [59]. Xavier's algorithm initializes a vector by selecting random variables from a uniform distribution. The endpoints of the uniform distribution is specified by [-d, d], where $d = \sqrt{\frac{6}{2T}}$. For a specific symbol in a batch of size M, the vector output, $\mathbf{s}_i$, of the last dense layer has a dimension of $2 \times N_t \times K$. This vector is reshaped to generate complex numbers, and it is multiplexed across all the antennas and along K time instants to produce a matrix, $\mathbf{S}_i \in \mathbb{C}^{Nt \times K}$. Elements in $\mathbf{S}_i$ are analogous to modulated symbols.

Table 3.1: Learned Transmitter - Neural Network Model

Layer Index	Shape of Neurons	Number of Neurons
Input Layer	$T \times T$	T^2
Middle Layer	$[2N_tT] \times T$	$[2N_tT^2]$
Output Layer	$[2N_tK] \times [2N_tT]$	$4N_t^2KT]$

At each time instant, the transmit power is normalised across each antenna, such that the transmit power along a single antenna is ≈ 1. Figure 3.2 provides an illustration of the transmitter.

To provide intuition about the symbols produced by the encoder, we provide constellation plots for a 2×2 system with a code rate, $1/K = 1$.

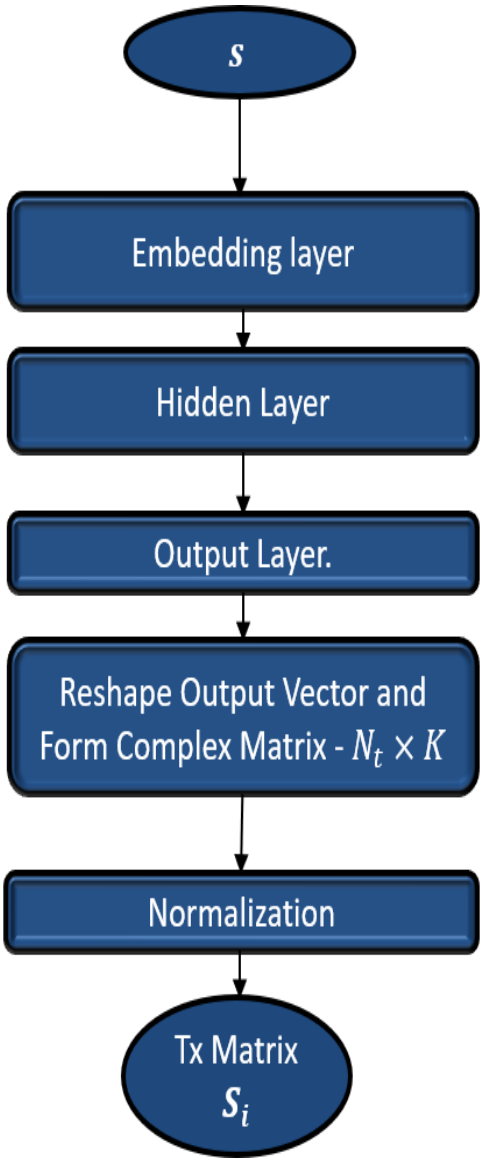
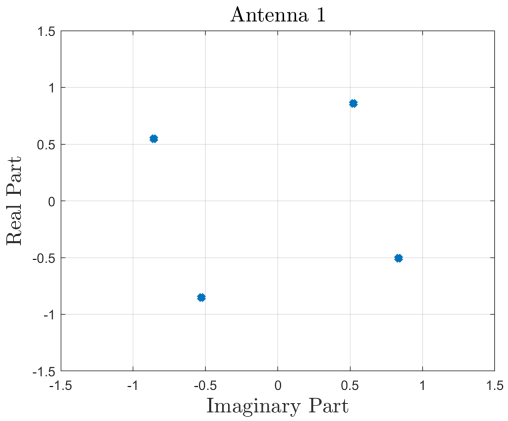
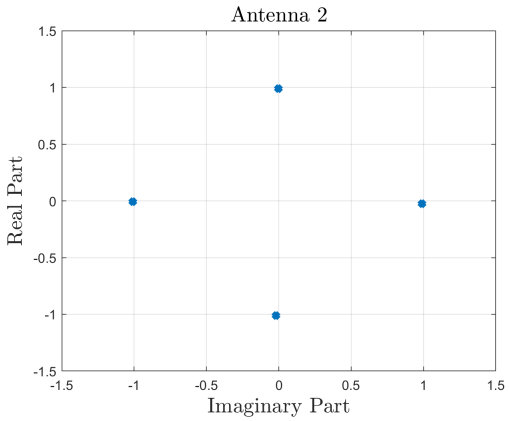


Figure 3.2: Learned Transmit system.

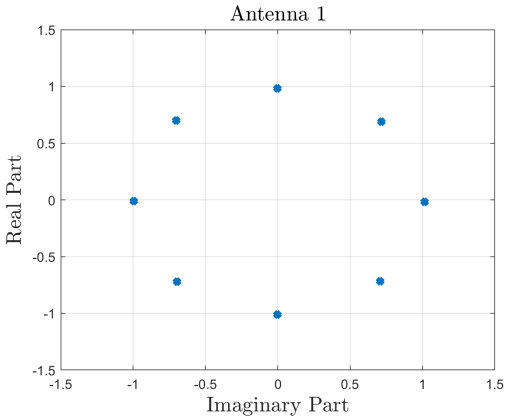


(a) 4-Ary Constellation for Antenna 1

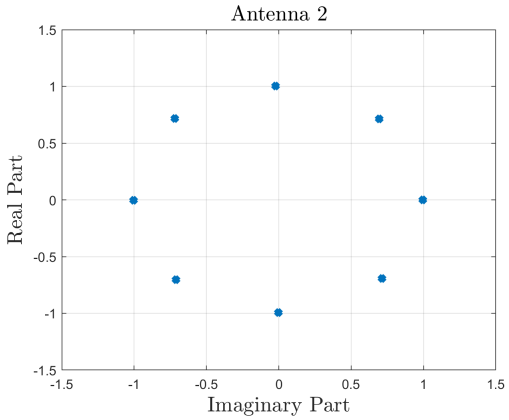


(b) 4-Ary Constellation for Antenna 2

Figure 3.3: 4-Ary Transmit Constellation

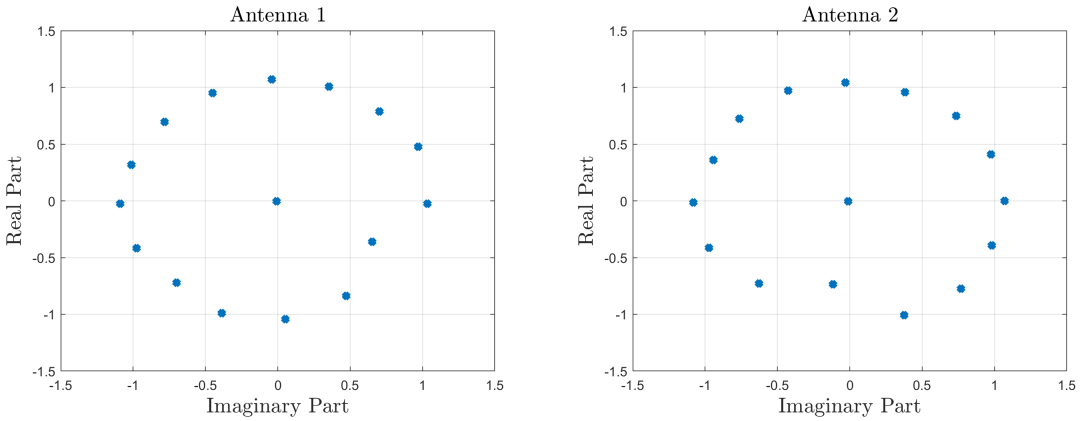


(a) 8-Ary Constellation for Antenna 1



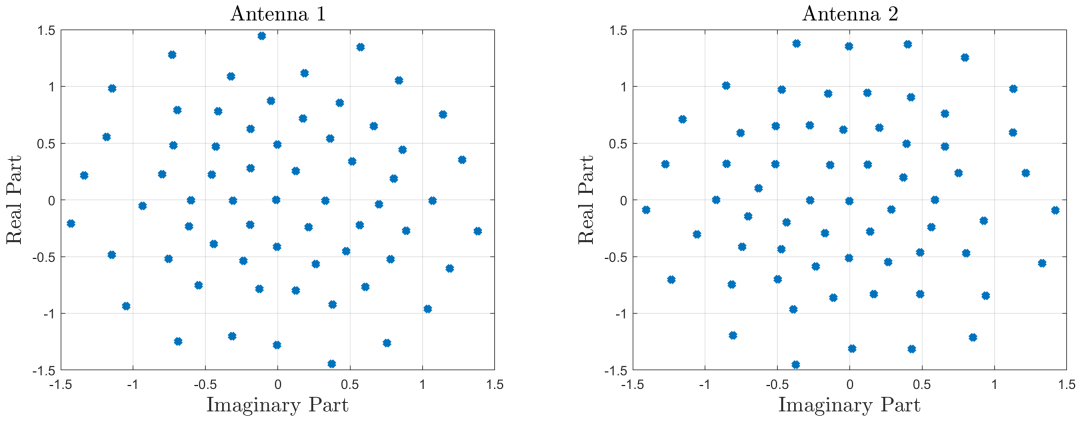
(b) 8-Ary Constellation for Antenna 2

Figure 3.4: 8-Ary Transmit Constellation



(a) 16-Ary Constellation for Antenna 1 (b) 16-Ary Constellation for Antenna 2

Figure 3.5: 16-Ary Transmit Constellation



(a) 64-Ary Constellation for Antenna 1 (b) 64-Ary Constellation for Antenna 2

Figure 3.6: 64-Ary Transmit Constellation

3.1.2 Deep Learning Based Receiver Design

The receiver is also designed using a feed-forward neural network, the signals across the receive antennas at each time step is represented by $\mathbf{y}_i$, where $i = 1, \cdots, K$. $\mathbf{y}_i$ is corrupted by noise and the multiplicative effects of Rayleigh fading channel.

$$\mathbf{y}_i = \mathbf{H}_i\mathbf{x} + \mathbf{n}_i \tag{3.4}$$

For a specific transmitted symbol s_i, the received signal across space and time can be modeled by a matrix, $\mathbf{Y}_i$. The matrix $\mathbf{Y}_i$ represents the received symbols across all antennas during a sequence of K time instants.

$$\mathbf{Y}_i = \begin{bmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \dots & \mathbf{y}_K \end{bmatrix}^T \tag{3.5}$$

The decoder (receiver) has two branches - a preprocessing block and a signal recover block. Table 3.2 presents a description of the depth and design of the pre-processor .

Table 3.2: Pre-processing - Neural Network Model

Layer Index	Shape of Neurons	Number of Neurons
Input Layer	$30\times$ $[2N_r]$	$[40N_r]$
Middle Layer	22×30	[660]
Output Layer	$[2N_r^2]$ $\times 22$	$[44N_r^2]$

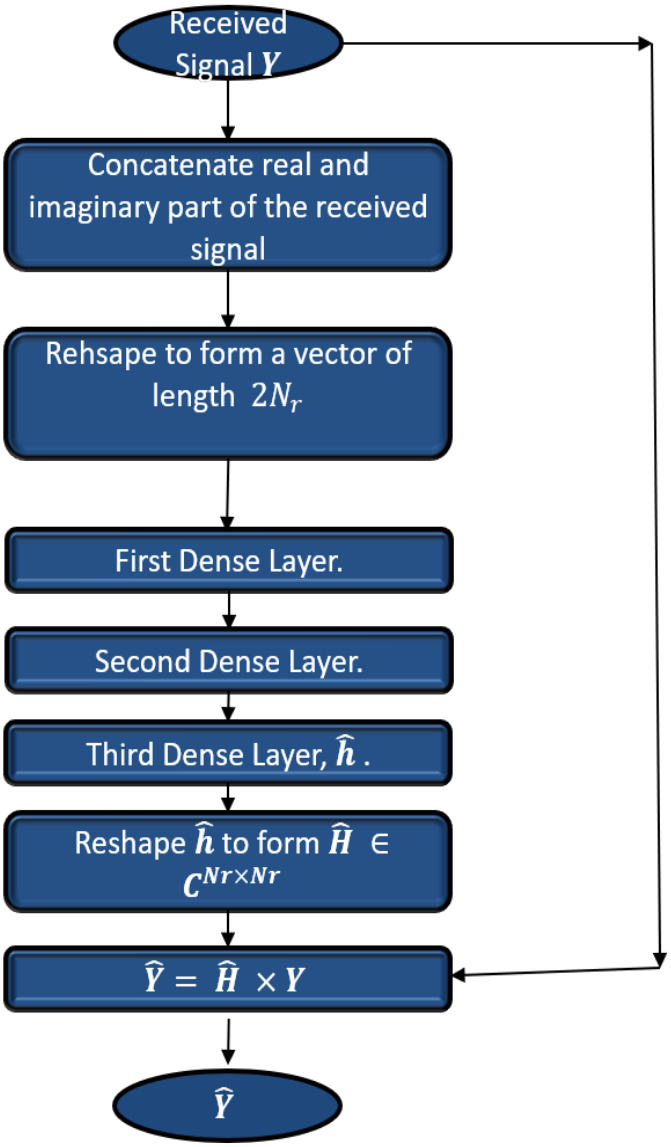


Figure 3.7: Learned Pre-processing system.

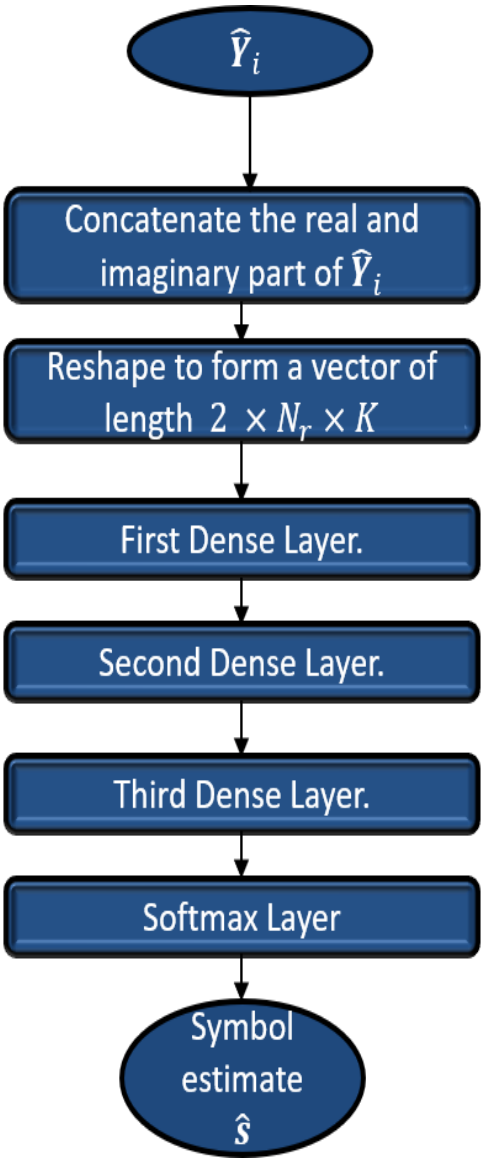


Figure 3.8: Learned Decoder system.

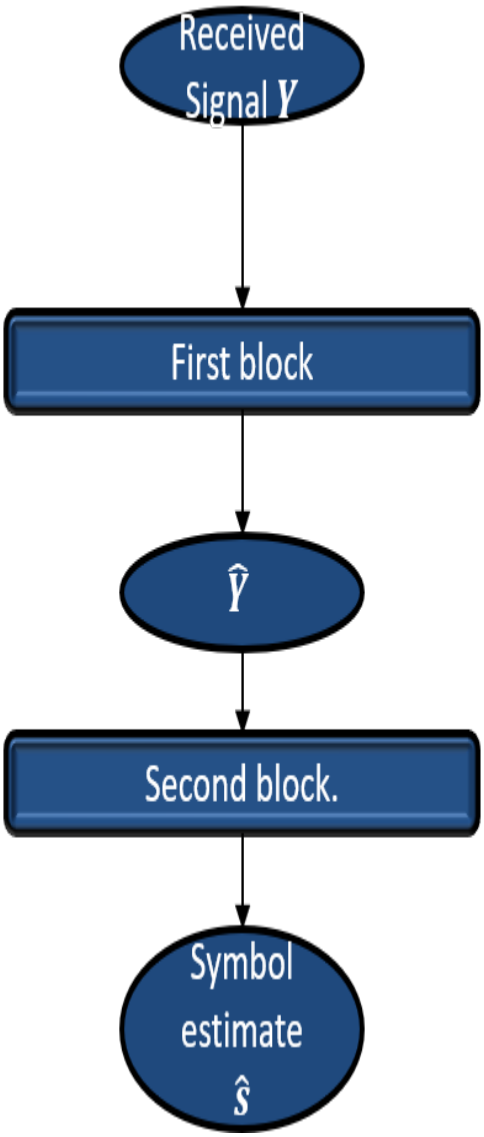


Figure 3.9: Overall Decoder system.

The pre-processing part of the decoder is depicted in 3.7, and takes as its input $\mathbf{Y}$. The real and imaginary part of the matrix are concatenated, and the matrix is reshaped to form a vector of length $2N_rK$. The vector is passed through three feed-forward layers. After each of the three layers, the expressive element of the neural network is provided by a relu activation function. Although this block is synonymous with the channel estimation and/or channel equalization blocks in conventional systems, we do not need pilots for channel estimation. The expressive nature of the neural networks allows for the channel to be directly inferred from a batch of received symbols. Notice, this incurs a delay as the entire batch must be received before any of its symbols are decoded. The output vector of the last neural network layer has a size of $2N_t^2$. The real vector is transformed into a complex vector with a size of $N_t \times N_r$. This vector is reshaped into matrix form, $\hat{\mathbf{H}} \in \mathbb{C}^{N_r \times N_t}$ which is used for channel equalization.

$$\widehat{\mathbf{Y}} = \mathbf{Y} \times \widehat{\mathbf{H}} \tag{3.6}$$

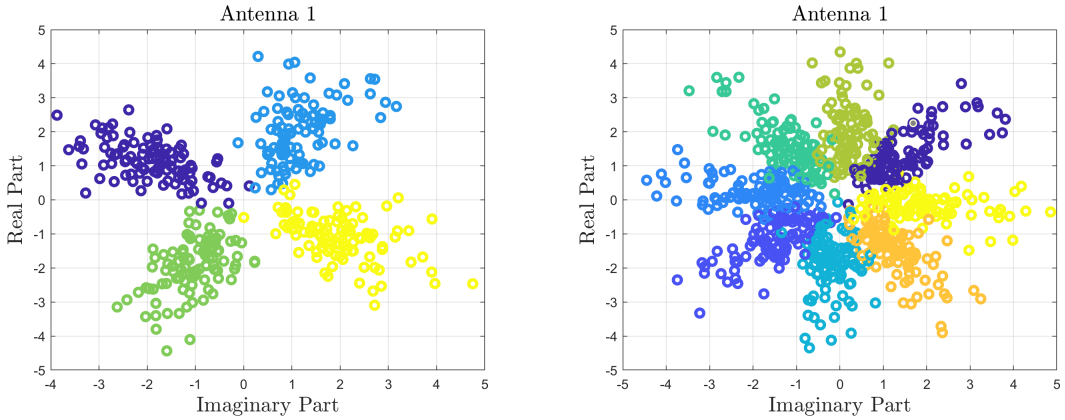
To illustrate the output of the pre-processing block, we plot the complex constellation produced from Equation 3.6, for varying values of signal to noise ratio [3.13, 3.17, and 3.21]. For these plots, the following configurations is used: $N_t = 2 = N_r$ and $K = 1$

The real and imaginary part of this matrix is concatenated to form a vector of size $2 \times N_t \times K$. This serves as input for the second branch of the decoder.

Table 3.3: Second Decoder Branch - Neural Network Model

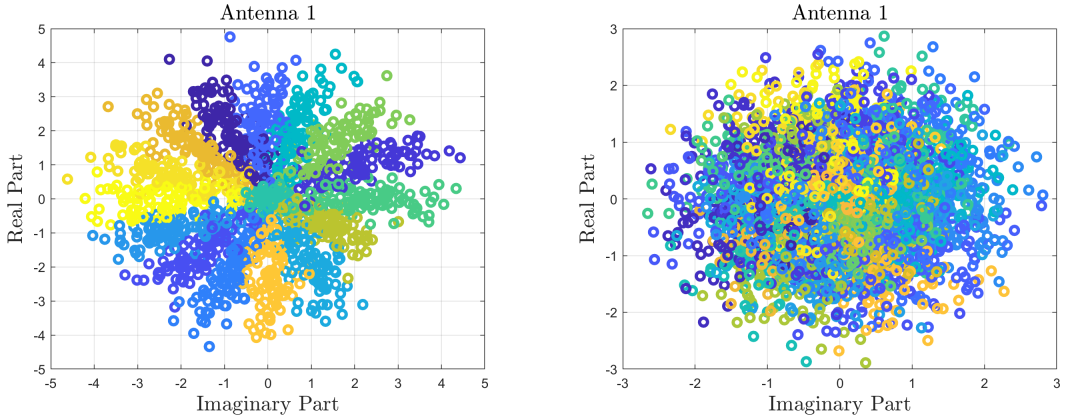
Layer Index	Shape of Neurons	Number of Neurons
Input Layer	$[N_tT] \times [2N_tK]$	$[2N_t^2KT]$
Middle Layer	$[N_tT] \times [N_tT]$	$[N_t^2T^2]$
Output Layer	$[T] \times [N_tT]$	$[N_tT^2]$

The following figures show the output of the pre-processing block.



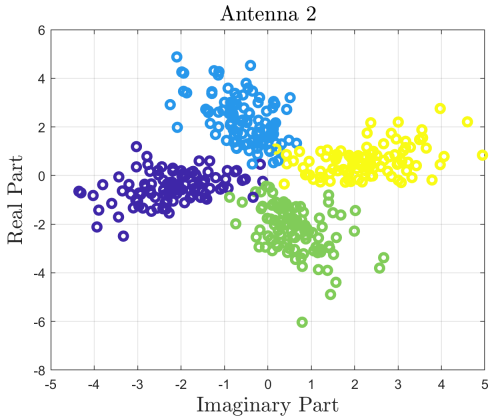
(a) 4-Ary Constellation for Rx Antenna 1 (b) 8-Ary Constellation for Rx Antenna 1

Figure 3.10: Constellation after Equalisation

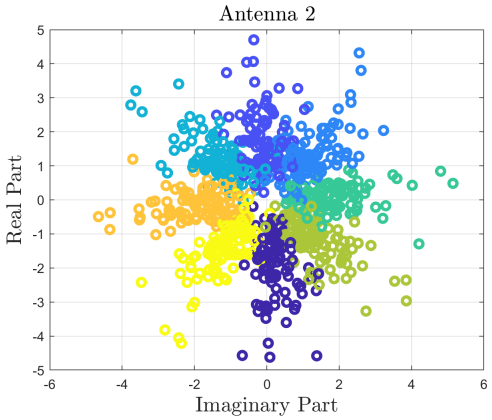


(a) 16-Ary Constellation for Rx Antenna 1 (b) 64-Ary Constellation for Rx Antenna 1

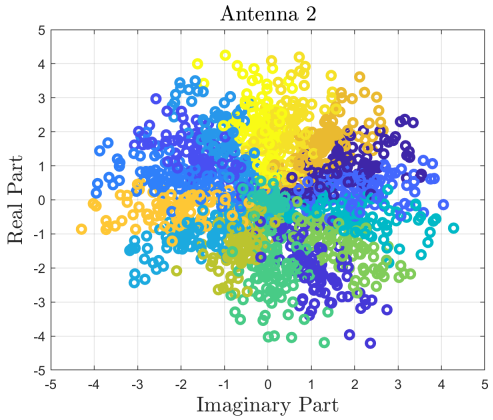
Figure 3.11: Constellation after Equalisation



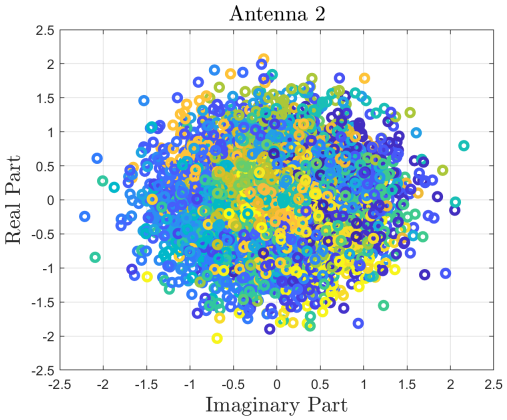
(a) 4-Ary Constellation for Rx Antenna 2



(b) 8-Ary Constellation for Rx Antenna 2



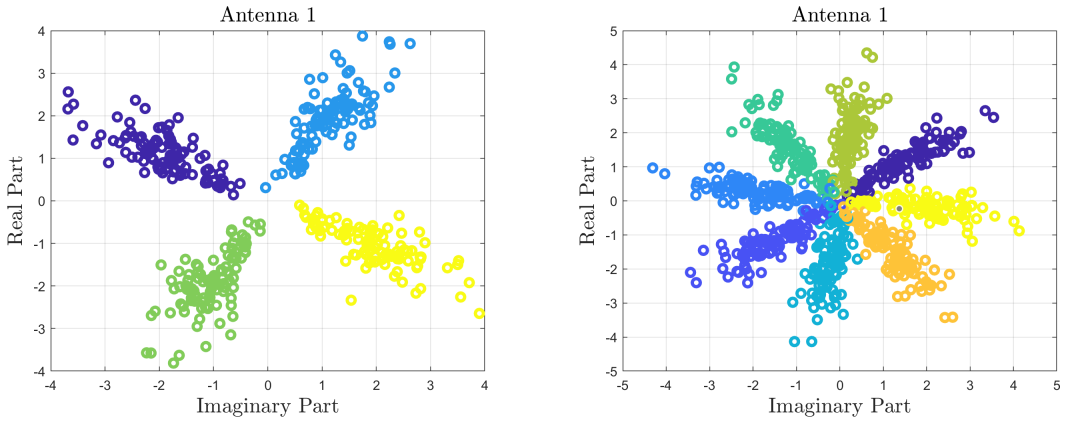
(a) 16-Ary Constellation for Rx Antenna 2



(b) 64-Ary Constellation for Rx Antenna 2

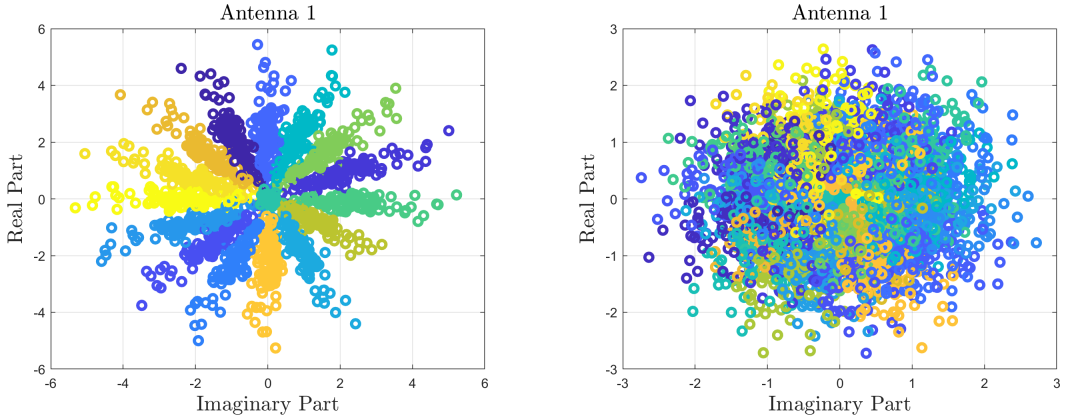
Figure 3.13: Constellation after Equalisation

At an SNR of 7dB, the following constellation plots depicts the complex encodings of the neural networks.



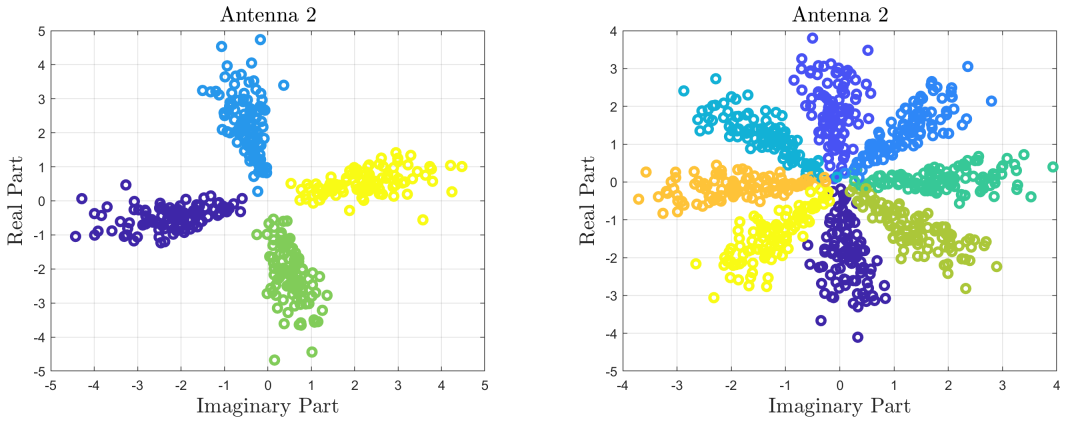
(a) 4-Ary Constellation for Rx Antenna -1 (b) 8-Ary Constellation for Rx Antenna 1

Figure 3.14: Constellation after Equalisation



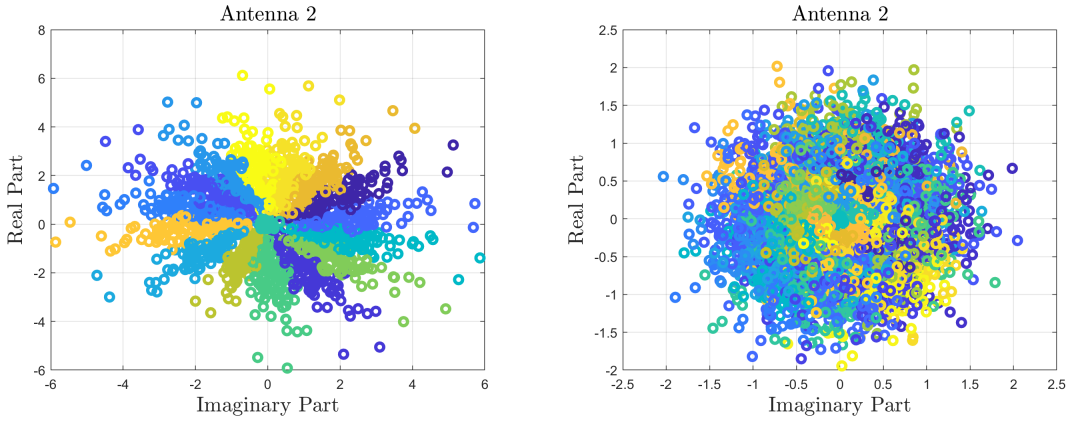
(a) 16-Ary Constellation for Rx Antenna 1 (b) 64-Ary Constellation for Rx Antenna 1

Figure 3.15: Constellation after Equalisation



(a) 4-Ary Constellation for Rx Antenna 2 (b) 8-Ary Constellation for Rx Antenna 2

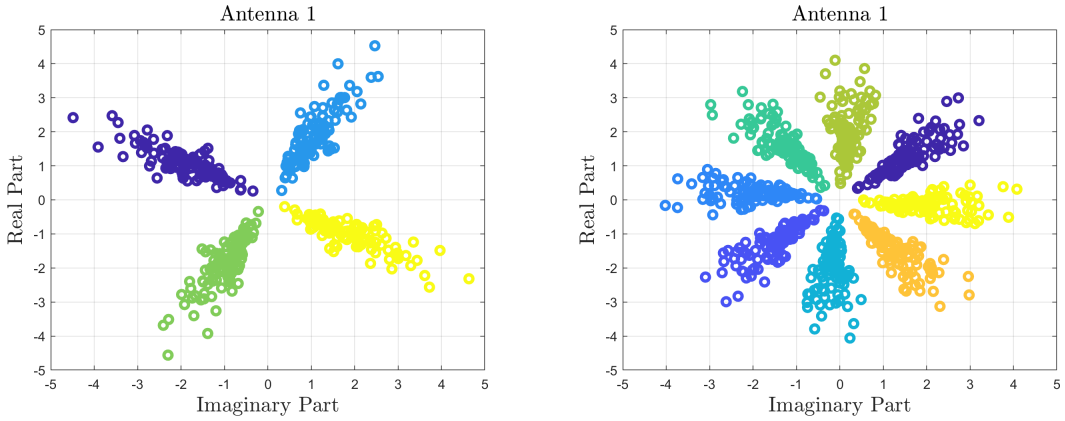
Figure 3.16: Constellation after Equalisation



(a) 16-Ary Constellation for Rx Antenna 2 (b) 64-Ary Constellation for Rx Antenna 2

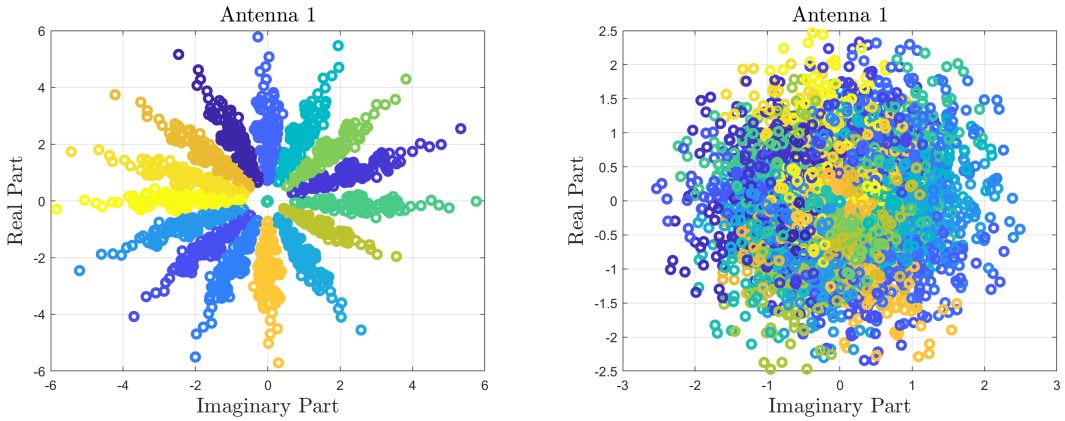
Figure 3.17: Constellation after Equalisation

At an SNR of 18dB, the constellation after pre-processing is given by the following fiugres:



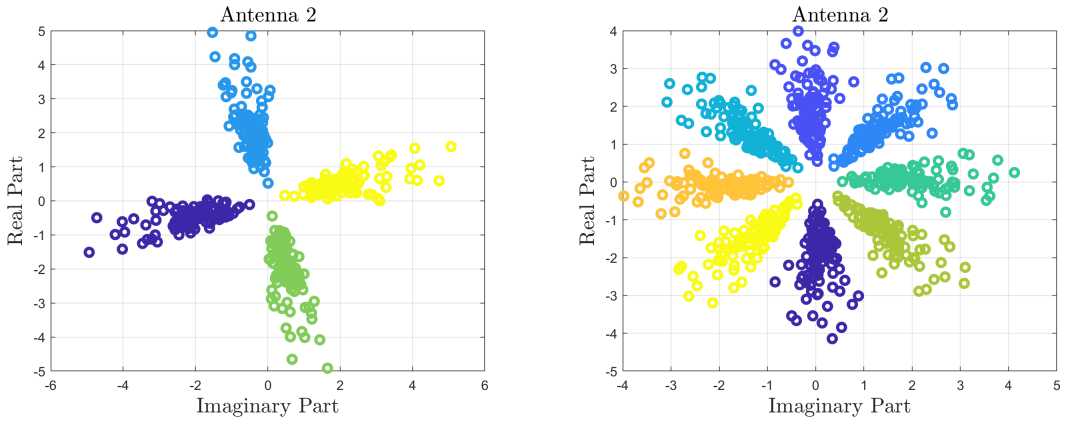
(a) 4-Ary Constellation for Rx Antenna 1 (b) 8-Ary Constellation for Rx Antenna 1

Figure 3.18: Constellation after Equalisation



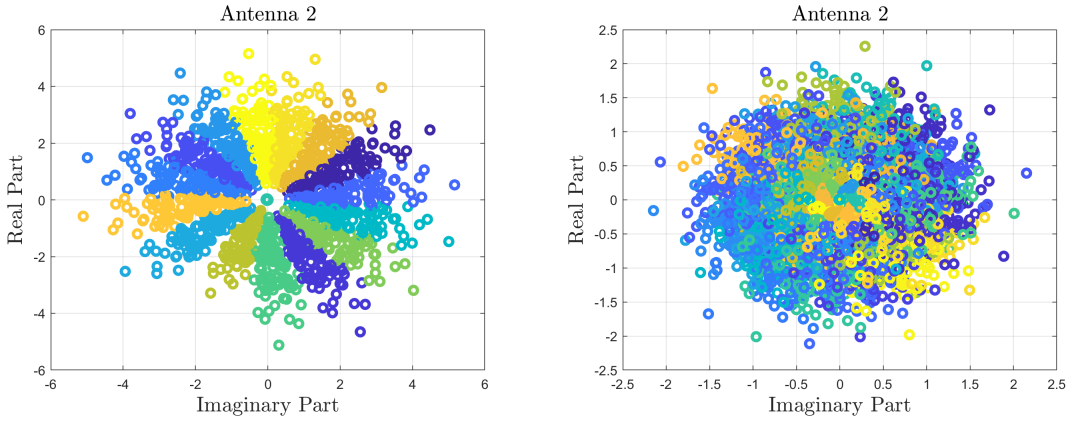
(a) 16-Ary Constellation for Rx Antenna 1 (b) 64-Ary Constellation for Rx Antenna 1

Figure 3.19: Constellation after Equalisation



(a) 4-Ary Constellation for Rx Antenna 2 (b) 8-Ary Constellation for Rx Antenna 2

Figure 3.20: Constellation after Equalisation



(a) 16-Ary Constellation for Rx Antenna 2 (b) 64-Ary Constellation for Rx Antenna 2

Figure 3.21: Constellation after Equalisation

The first two layers of the second decoder branch use a swish activation function. The last layer is activated by a softmax layer. The softmax layer provides T pseudo-probabilities, each denoting the likelihood that the decoded symbol is one out of a possible of T symbols that could have been transmitted. The transmitted symbol, s is approximated as the index with the highest pseudo probability. This is essentially a classifier.

3.2 Training Scheme

The training scheme for this learning system occurs offline and the training specifications are listed in Table 3.4. Tensorflow and Python are used to implement the Neural network architecture. Tensorflow is selected over the more simplified Keras, because of the need to design a scheme for handling complex numbers. For schemes, where the total number of possible symbols is less than 32, a personal desktop with a CPU can be used, however for schemes of higher modulation order, a workstation with a GPU is suggested. Our device is equipped with 24GB NVIDIA Quadro M6000. The cross entropy loss is chosen as the loss function. A training batch size of 20000 symbols is used in a flat fading Rayleigh environment. The training scheme is iterated 100000 times and the neural parameters are optimized using the Adam algorithm [56]. A new complex Rayleigh channel is generated on each iteration. The generated Rayleigh channel is a matrix with independent entries, i.e, during training we assume uncorrelated transmit and receive antennas.

Table 3.4: List Of Hyperparameters

Hyper-parameters	Value
Learning Rate	0.001
Iterations (epochs)	100000
Batch size	20000

The neural network is first trained at an SNR of 15dB, this allows it to properly detect and classify the coded symbols. After 100000 iterations, the neural network is optimized at 10dB, this allows it to detect symbols at low SNR.

3.3 Simulation

During testing, various scenarios are evaluated. First, we test on the same channel distribution that was used during training. In this scenario, the channel is randomly generated by sampling from a Rayleigh distribution. The channel matrix has independent matrix elements, this implies that there is no correlation at the transmit or receive antennas. At the transmitter, a batch of random symbol indices is generated on each iteration. The symbol indices are encoded by the transmitter and the resulting symbols are passed through the multiplicative effects of the randomly generated channel. The resulting signal is distorted by adding Gaussian noise with a specific noise power. The decoder operates in batches. The decoder delays pre-processing until a batch of symbols with a predefined length arrives at the receiver. The decoder receives these symbols and tries to reconstruct the original symbol index. Subsequently, the symbol error rate is computed for different noise powers.

Second, we evaluate the impact of correlated transmit and receive antennas. In this scenario, correlation is induced at either the transmitter or receiver using the Kronecker correlation model.

$$\mathbf{H}_c = \mathbf{R}_{rx}^{1/2}\, \mathbf{H}\, \mathbf{R}_{tx}^{1/2} \tag{3.7}$$

where $\mathbf{H}$ is a channel matrix with independent entries, $\mathbf{R}_{rx}$ is a matrix representing the correlation among the receive antennas, $\mathbf{R}_{tx}$ is a matrix representing the correlation among the transmit antennas, and $\mathbf{H}_c$ is the resulting correlated channel used for testing.

Lastly, we investigate the robustness of our system by testing on a distribution that is different from the distribution used during training. Testing is carried out on channels sampled from a Nakagami-m distribution with varying m values. Note that an independent Rayleigh channel is equivalent to a Nakagami-m channel with an m factor of 1 and the Nakagami channels can be used to approximate a Ricean channel. We present symbol error rate performance curves for varying values of m.

3.3.1 Simulation With Independent Rayleigh Fading Channels Encountered During Validation

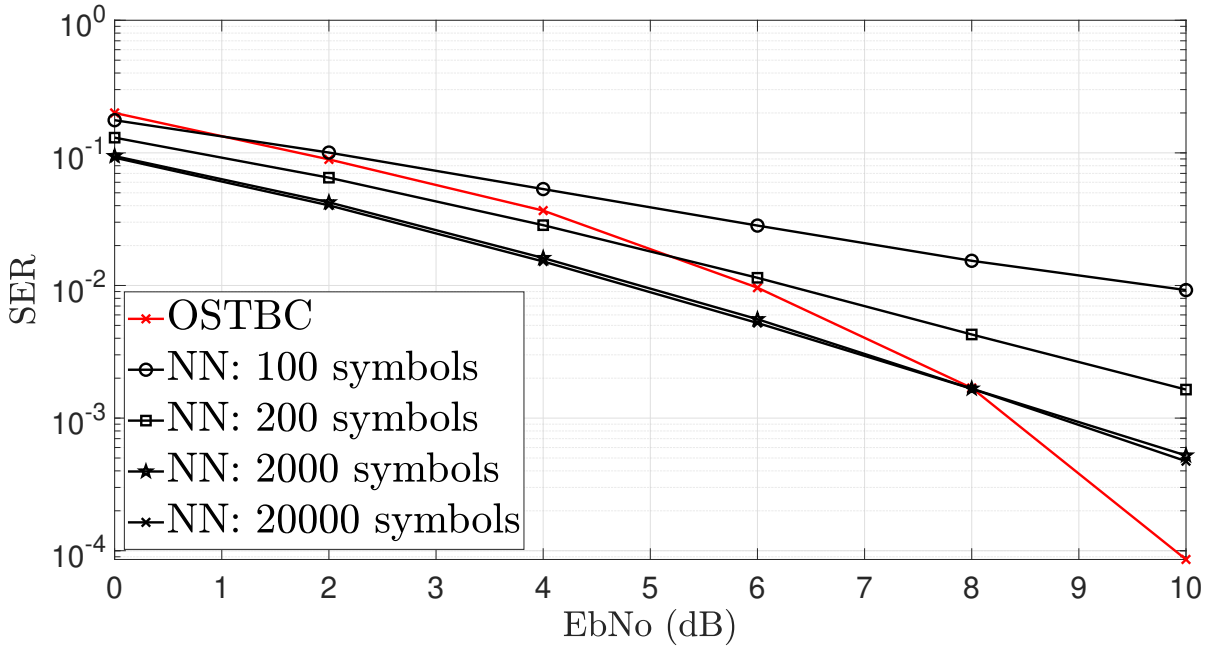


Figure 3.22: SER of 3×3 MIMO Scenario with information rate of 2 bits/s trained and tested on independent Rayleigh channels. The OSTBC serves as a baseline and employs perfect channel estimation at the receiver.

Figure 3.22 compares the performance of the learned system with symbol error rate performance resulting from a system using orthogonal space-time block codes [10]. Clearly,

for larger batch sizes, $M = 2000, 20000$, and at low signal-to-noise ratios, the neural network-based system provides a gain of approximately 2dB. However, for energy per bit-to-noise ratio $\frac{E_b}{N_o}$ values greater than 8dB, the OSTBC provides better error rates. For a batch size of $M = 200$, a similar trend is observed, except that the $\frac{E_b}{N_o}$ threshold point at which the neural network stops being beneficial is much lower. For a batch size of $M = 100$, there are no practical benefits of employing a neural based solution. The OSTBC code in this figure uses 16-QAM as its modulation scheme of choice, while the NN based systems can transmit one of 16 possible symbols. The two curves are both rate one-half codes, hence 2 bits are transmitted every time instant. In Figures 3.23 and 3.24, the neural network also provides substantial performance gains at low SNRs, although the threshold points at which the neural network stops being beneficial is much higher. In Figure 3.23, for a batch size of $M = 2000, 20000$, this threshold is at an $\frac{E_b}{N_o}$ value of 14dB. For a batch size of $M = 100$ and $M = 200$, the $\frac{E_b}{N_o}$ threshold is at 10dB and 12dB respectively.

In Figure 3.24, for batch sizes of $M = 2000, 20000$, and at low signal-to-noise ratios, the neural network provides about 5dB of improvement over conventional OSTBCs. For these larger batches, the neural network stops being beneficial at an $\frac{E_b}{N_o}$ value of 18dB. For smaller batches, the neural network also provides substantial benefits at low signal-to-noise ratios, however, the neural network stops being beneficial at a much lower $\frac{E_b}{N_o}$. The performance curves in Figure 3.23 employ modulation schemes that produce 64 distinct symbols while the performance curves in Figure 3.24 employ modulation schemes that produce 256 distinct symbols. Note that the OSTBC in both curves achieves this by using 64-QAM and 256-QAM in Figure 3.23 and Figure 3.24 respectively. The codes in both curves are rate one-half codes, hence, the systems evaluated in Figure 3.23 transmits 3 bits every time instant, while the systems in Figure 3.24 transmits 4 bits every time instant.

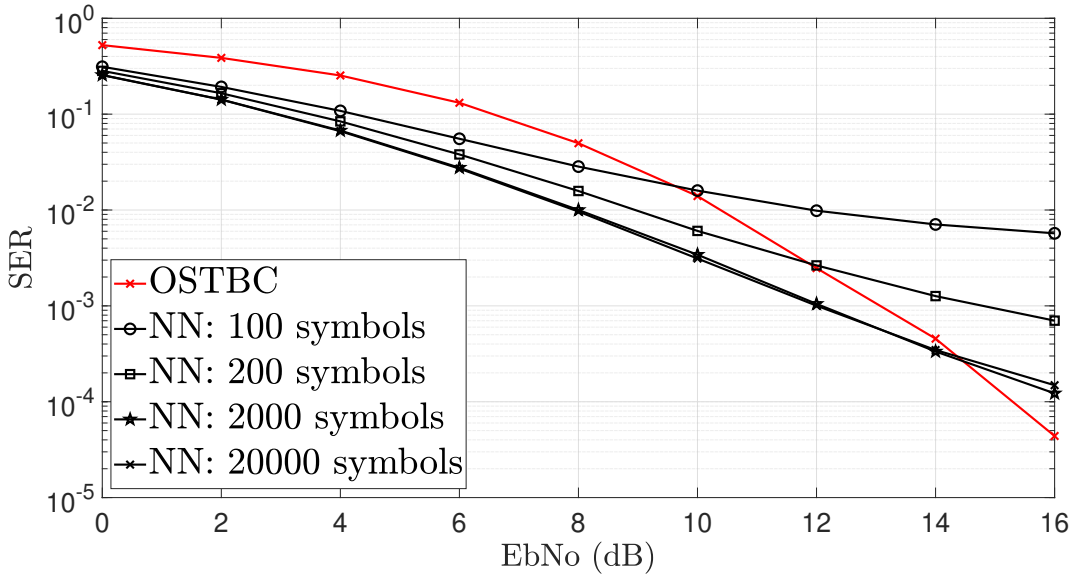


Figure 3.23: SER of 3×3 MIMO Scenario with information rate of 3 bits/s trained and tested on independent Rayleigh channels. The OSTBC serves as a baseline and employs perfect channel estimation at the receiver.

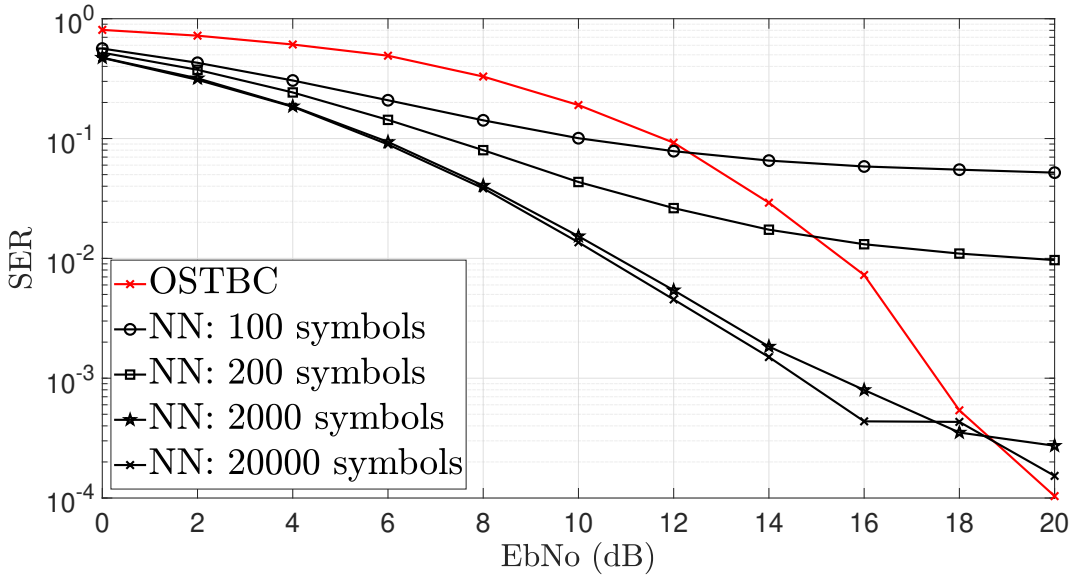


Figure 3.24: SER of 3×3 MIMO Scenario with information rate of 4 bits/s trained and tested on independent Rayleigh channels. The OSTBC serves as a baseline and employs perfect channel estimation at the receiver.

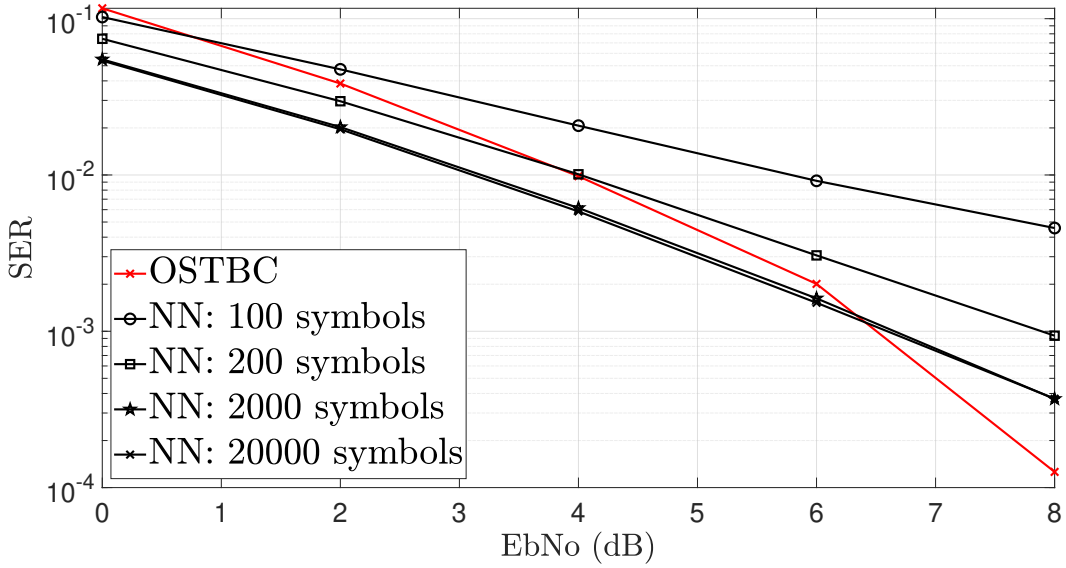


Figure 3.25: SER of 4×4 MIMO Scenario with information rate of 2 bits/s trained and tested on independent Rayleigh channels. The OSTBC serves as a baseline and employs perfect channel estimation at the receiver.

Figures [3.25, 3.26, and 3.27] provide performance evaluations for a different antenna configuration. Each of the systems which performance depicted in Figure 3.25 have the same rate and number of symbols as the corresponding systems depicted in Figure 3.22. For, larger batch sizes, $M = 2000, 20000$, the neural system provides an improvement at low $\frac{E_b}{N_o}$, but after a threshold of 6.5dB the conventional system clearly outperforms the neural based systems. For a batch sizes of $M = 100$, and $M = 200$, the neural network provides no substantial benefit. Irrespective of batch size, the neural systems depicted in Figure 3.26, provides a 2dB performance gain at low $\frac{E_b}{N_o}$. The OSTBC system used in this plot employs 64-QAM, while the neural system can transmit one of 64 different symbols. Note that the code rate is one-half, hence the 3 bits is transmitted every time instant.

The systems in Figure 3.27, show a similar trend in the performance comparison of both the conventional system and the neural-based system.

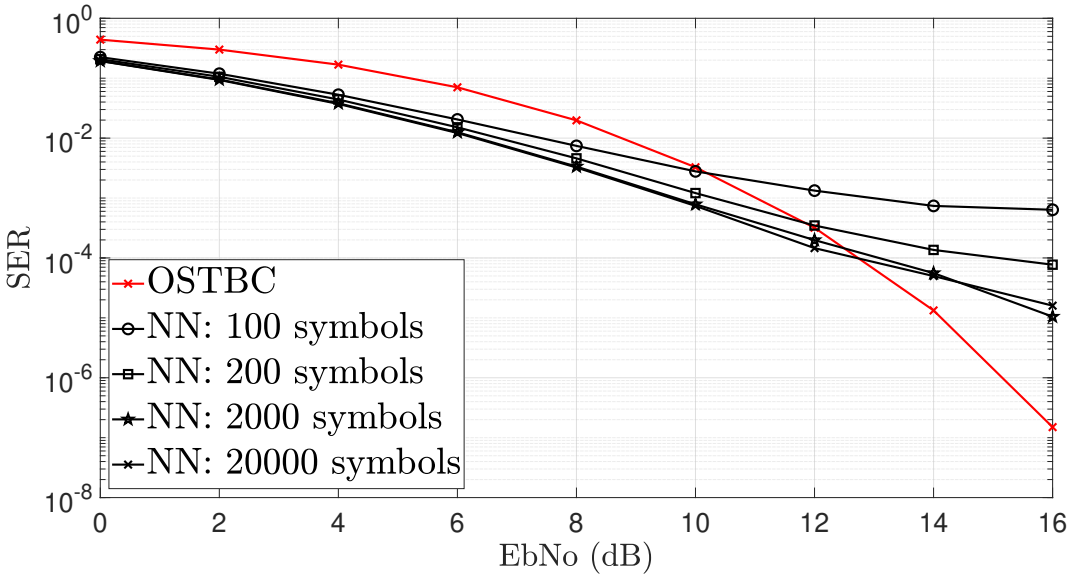


Figure 3.26: SER of 4×4 MIMO Scenario with information rate of 3 bits/s trained and tested on independent Rayleigh channels. The OSTBC serves as a baseline and employs perfect channel estimation at the receiver.

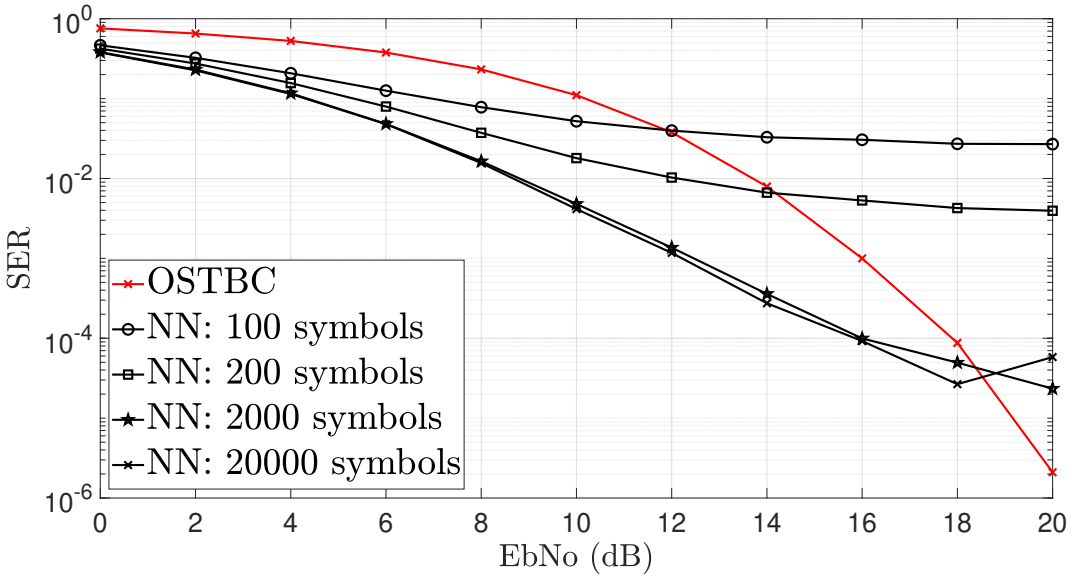


Figure 3.27: SER of 4×4 MIMO Scenario with information rate of 4 bits/s trained and tested on independent Rayleigh channels. The OSTBC serves as a baseline and employs perfect channel estimation at the receiver.

3.3.2 Simulation With Correlated Rayleigh Fading Channels Encountered During Validation

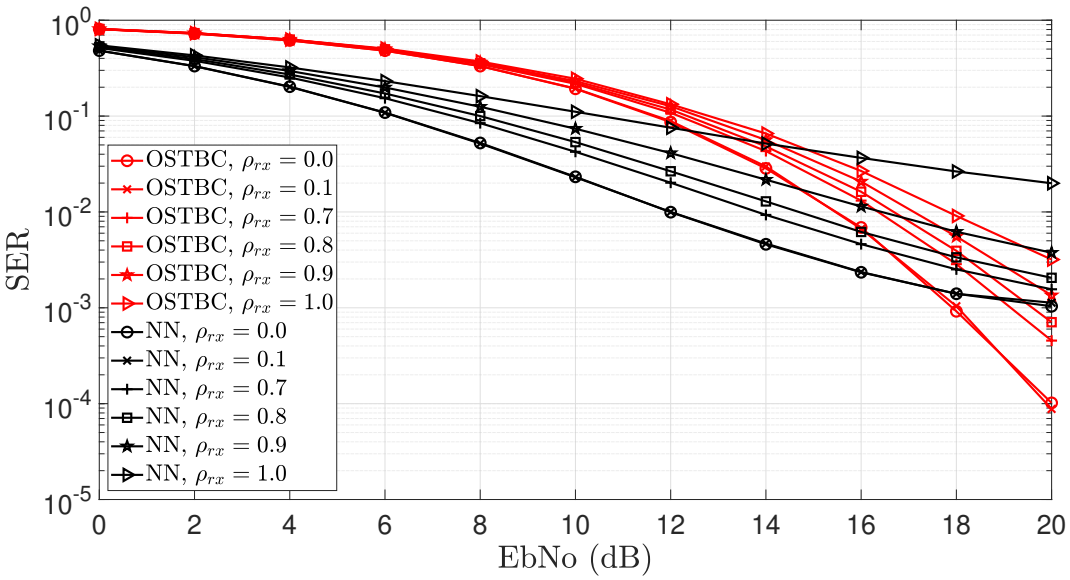


Figure 3.28: SER of 3×3 Scenario with Information rate of 4 bits/s with varying degrees of correlation induced among the receive antennas. The OSTBC serves as a baseline and employs perfect channel estimation at the receiver.

Highly correlated receive antennas means that the probability of obtaining independent symbol replicas is reduced. Hence, the symbol error rate performance should deteriorate with an increase in the correlation among the receive antennas. This is depicted in Figure 3.28. In this figure, with low correlation among the receive antennas (correlation values of 0.1), the symbol error rate performance is identical to the symbol error rate performance with no correlation. As the correlation values increases, the performance of the NN based system degrades rapidly. The worst performance is observed for perfectly correlated receive antennas. Although OSTBC curves show a similar performance trend, the NN based system always produces better symbol error rate performance at low signal to noise ratio. However, the threshold at which the NN based system stops providing benefit is hampered with receive correlation. This threshold decreases from an $\frac{Eb}{No}$ value of 17.5dB at low correlation to an

$\frac{Eb}{No}$ value of 12.5dB at perfect correlation.

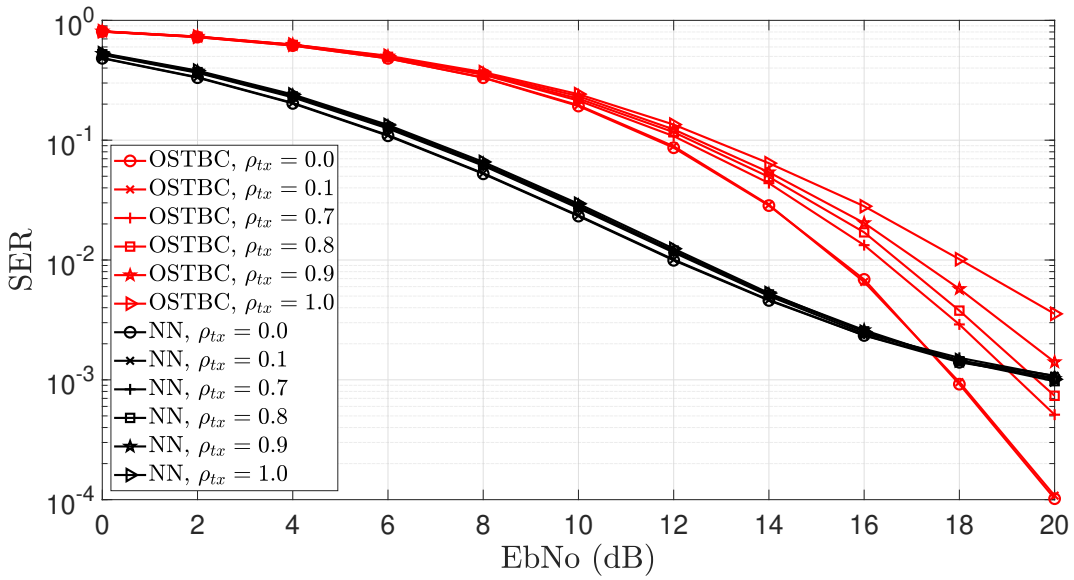


Figure 3.29: SER of 3×3 Scenario with Information rate of 4 bits/s with varying degrees of correlation induced among the transmit antennas. The OSTBC serves as a baseline and employs perfect channel estimation at the receiver.

In Figure 3.29, as expected transmit correlation hampers the symbol error rate performance of the conventional OSTBCs. The performance is substantially worse when the transmitters are perfectly correlated. However unlike the conventional OSTBCs, the NN based system doesn't degrade with an increase in transmit correlation. At first glance, this might not seem intuitive, but it is important to note that the neural network produces a matrix of complex values for each symbol index. Hence the complex representations transmitted across each antenna at each time instant all correspond to the same symbol index and by definition are correlated. Although the OSTBCs are affected by transmit correlation, beyond a certain signal to noise ratio they still perform better than the learned system.

3.3.3 Simulation With Heterogeneous Distributions Encountered During Validation

Figure 3.30 presents the symbol error rate performance for a learned system trained on independent Rayleigh channels and validated across different distributions. The conventional OSTBCs employs perfect channel estimation across all validation channels. Regardless of the distribution encountered during validation, at low SNR the learned system outperforms the conventional OSTBCs. The learned and conventional system validated in Nakagami fading channels with an $m-$factor of 0.5 provide the best performance, this is because the fading encountered in this channels is most severe and the learned and conventional system exploit the increased multipath in order to provide increased diversity. With an $m-$factor of 1, the channels sampled from a Nakagami distribution are equivalent to channels sampled from a Rayleigh distribution. The learned and conventional system validated in Ricean fading

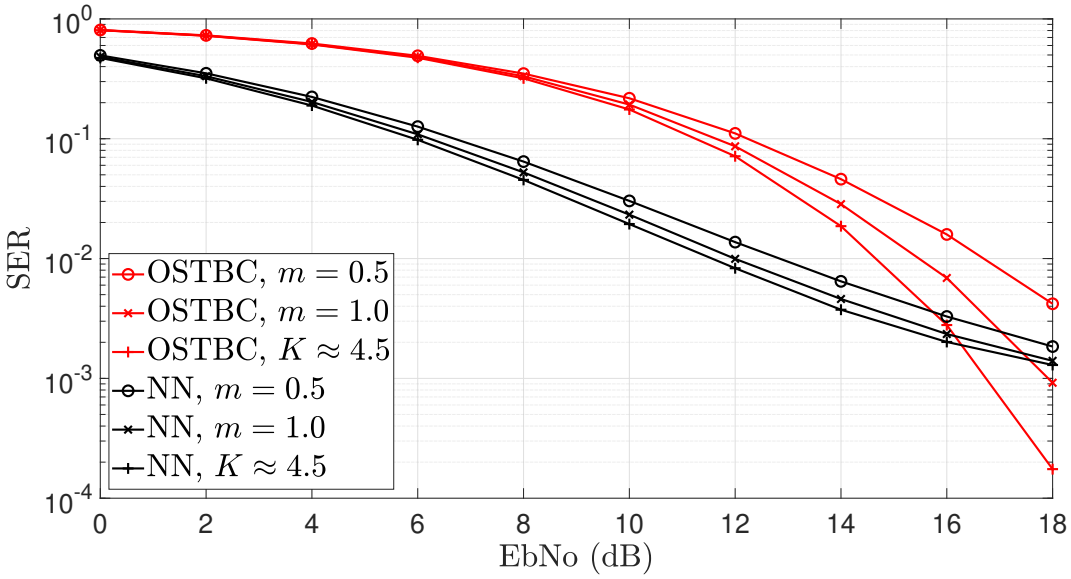


Figure 3.30: SER of 3×3 MIMO Scenario with information rate of 4 bits/s trained on independent Rayleigh channels and tested across three different channels - Nakagami channel with an m factor of 0.5, Nakagami channel with an m factor of 1, and Ricean channel with a Ricean factor of 4.5

channels provide the least performance in terms of symbol error rate. This is due to the presence of a line of sight component. The line of sight component decreases the diversity that can be achieved due to multiple transmit and/or multiple receive antennas.

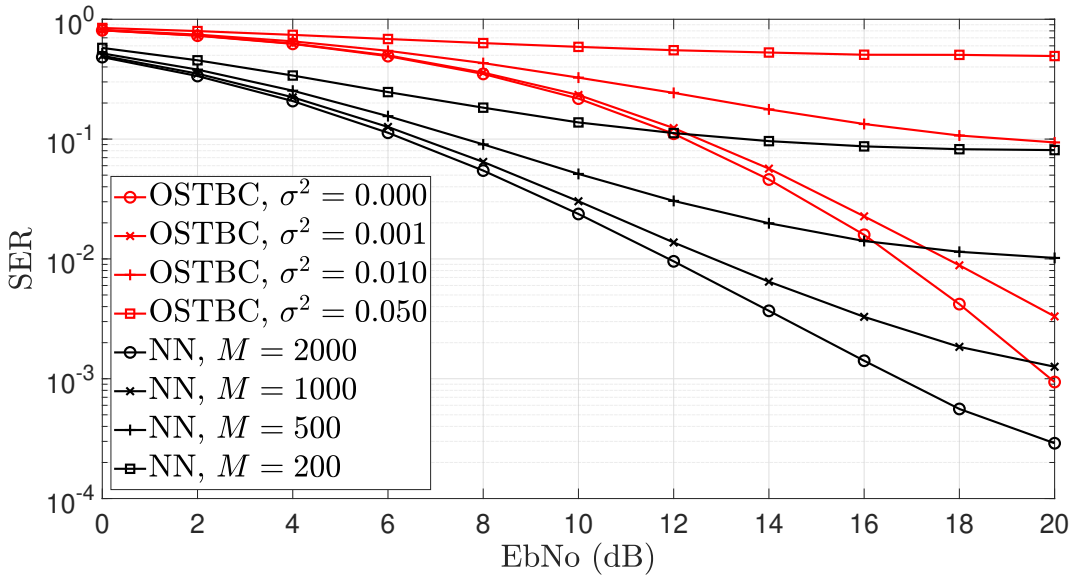


Figure 3.31: SER of 3×3 MIMO Scenario with information rate of 4 bits/s trained on independent Rayleigh channels and tested on a Nakagami channel with an m factor of 0.5. The impact of varying the decoding bach sizes of learned system is compared with the impact of channel estimation errors in the conventional OSTBCs

Figures [3.31, 3.32, 3.33] compares the variation in performance of the learned system due to varying batch sizes with the variation in performance of the conventional OSTBCs due to errors in channel estimation at the receiver. For the conventional OSTBCs, the errors in channel estimation are sampled from a Gaussian distribution.

$$\hat{\mathbf{H}} = \mathbf{H} + \mathbf{E} \tag{3.8}$$

where $\hat{\mathbf{H}}$ is the estimated channel used for maximum ratio combining, $\mathbf{H}$ is the actual channel, and $\mathbf{E}$ is a matrix of errors sampled from a Gaussian distribution with zero mean

and σ standard deviation.

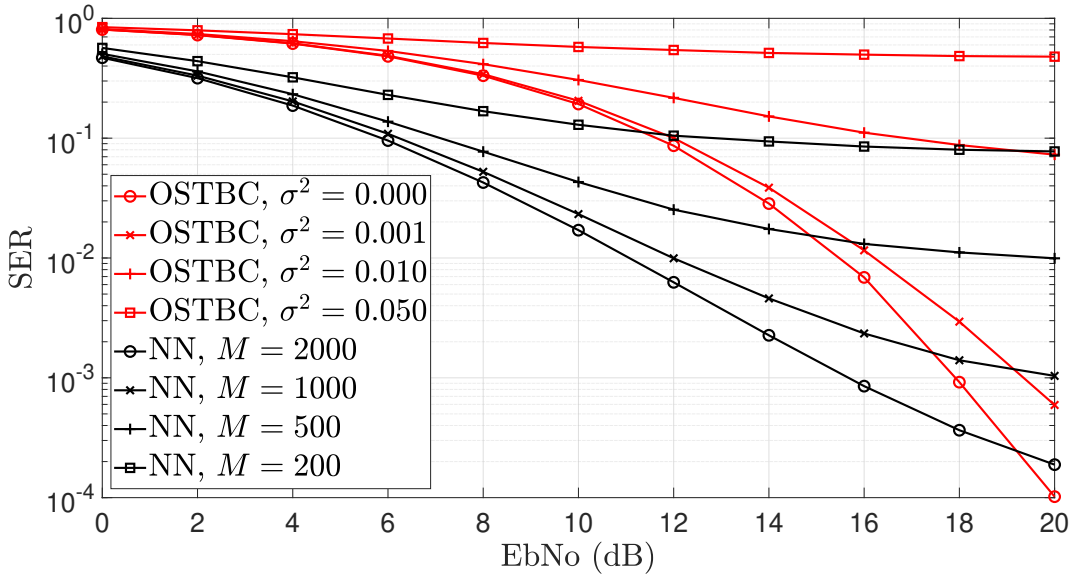


Figure 3.32: SER of 3×3 MIMO Scenario with information rate of 4 bits/s trained on independent Rayleigh channels and tested on a Nakagami channel with an m factor of 1. The impact of varying the decoding bach sizes of learned system is compared with the impact of channel estimation errors in the conventional OSTBCs

Evidently from these figures, the symbol error rate performance of the conventional block codes deteriorates rapidly as the variance of the channel estimation errors increases. Similarly for the learned system, as the size of the decoding batch decreases, the symbol error rate performance deteriorates rapidly. It is important to note that for a small batch size of $M = 200$ symbols and at low SNRs, the learned system outperforms the conventional OSTBCs irrespective of the variation in channel estimation errors. More significantly, irrespective of validation distribution, at low SNRs, the learned system with a small batch size outperforms the conventional OSTBCs with perfect channel estimation.

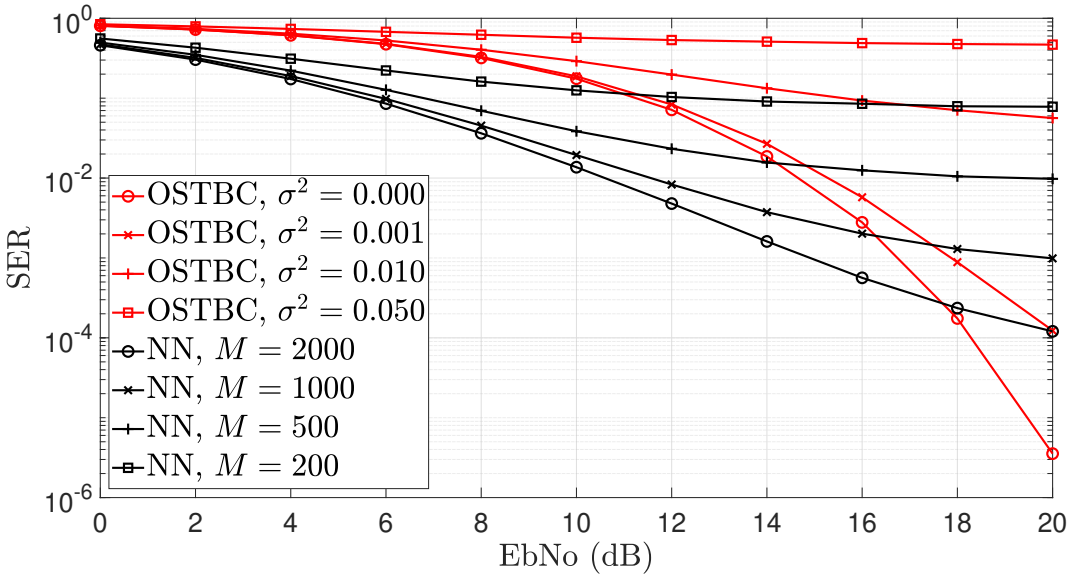


Figure 3.33: SER of 3×3 MIMO Scenario with information rate of 4 bits/s trained on independent Rayleigh channels and tested on a Ricean channel with a Ricean factor of 4.5. The impact of varying the decoding bach sizes of learned system is compared with the impact of channel estimation errors in the conventional OSTBCs

3.4 Conclusion

This chapter has evaluated the possibility of designing machine learning-based space-time block codes. In this chapter, we first present the neural network model used to generate the STBC codes, and, we also present a visual depiction of the symbols produced at the output of the transmitter. Second, we presented a thorough description of the training scheme used. Third, we present simulation results for various transmit and receive configurations across different bit rate configurations. Through simulation, we provide insight into the effect of correlation at either the transmit or receive antennas. We show the robustness of our model by validating the symbol error rate performance on a distribution different from the distribution used during training. Lastly, we compare the performance drop due to variation in batch sizes at the learned decoder with the performance drop due to channel estimation errors at conventional wireless receivers.

Chapter 4

Reinforcement Learning for Massive Machine Type Communications

Proponents of next generation wireless systems envision the use of the random access slots to support massive machine type communications. Thus, the deployment of a massive number of these machine type devices will yield to a non-zero probability of a large number of simultaneous transmission during a single random access slot. Hence, it is imperative we find a way to alleviate the resulting congestion. This chapter investigates the possibility of exploiting dense heterogeneous deployments of base stations in order to reduce the long term congestion caused by concurrent requests from massive machine type communications. The heterogeneous system model includes the deployment of large cells (i.e., those with large coverage areas) as well as small cells that are used to cover smaller areas with large numbers of devices, so called "hotspots". The former type of cell is typically termed a "macro cell" while the latter are termed micro cells or even pico cells, depending on the size. An important aspect of heterogeneous cellular networks is that the macro cells and pico (or micro) cells typically have coverage areas that overlap to some degree. This results in the possibility of multiple MTD-base station association. We investigate the optimal MTD-gNB association which minimizes the long term congestion experienced in the overall cellular network.

To address this problem, we formulate the optimization problem as a finite Markov Decision Process (MDP). Hence, the cardinality of the set of all state, action, and reward pairs

is finite. This places a restriction on the resulting state space. More specifically, the present state space must include all the information needed to make a decision at the current time (Markovian Property). The characterization of the problem as a finite MDP mandates that there exists a policy capable of producing the optimal state-action value function. However, as indicated in previous chapters, deriving this optimal solution is dependent on the availability of the system dynamics which are unknown apriori and are nontrivial to determine. To circumvent the need for the complete system dynamics, we use a model-free algorithm to learn the optimal state-action value functions. Without the system dynamics, the model-free algorithm needs to optimize the current MTD-gNB association while simultaneously exploring other possibly sub-optimal MTD-gNB associations. To this end, we employ an off-policy algorithm that optimizes a target policy for producing the best MTD-gNB association while using a behavioral policy to explore other associations and their consequent rewards. We validate this work by computing the probability of collision, channel utilization, and access success probability of the learned system. We show empirically that the optimal performance is obtained by the resulting approach when the total number of slots is greater than or equal to the total number of MTDs. When the total number of MTDs is greater than the number of slots, the performance of the computed metrics degrades rapidly. To provide a rough intuition about the performance of the off-policy algorithm, we show graphically the learned MTD-BS association.

4.1 System Model

We consider a heterogeneous cellular network with M base stations having some overlap in their regions of overage as depicted in figure 4.1. We assume the set of all base stations $\mathcal{M} = \{1, \cdots, M\}$ are equipped with single antennas, and the set of all MTDs,

$\mathcal{N} = \{1, \cdots, N\}$, are simultaneously trying to access the cellular network through the random access slots. The set of base stations that can provide cellular access to a specific MTD is referred to as $\delta-$suitable base stations. We postulate that an MTD can willfully attach to any of the $\delta-$suitable base stations. In other words, the MTD can migrate from one cell to another if the base station of the new cell is within a certain distance. Exploiting this assumption, MTDs try to find the optimal MTD-BS association that minimizes the long term congestion in the cellular network. In other words, instead of minimizing the congestion in specific cells at a specific time instant, we design a self-optimizing network that gradually learns the optimal set of MTD-BS association vectors, $\mathbf{x} = \{\mathbf{x}_m\}_{m=1}^{m=M}$ which minimizes the overall cellular long-term congestion across all cells.

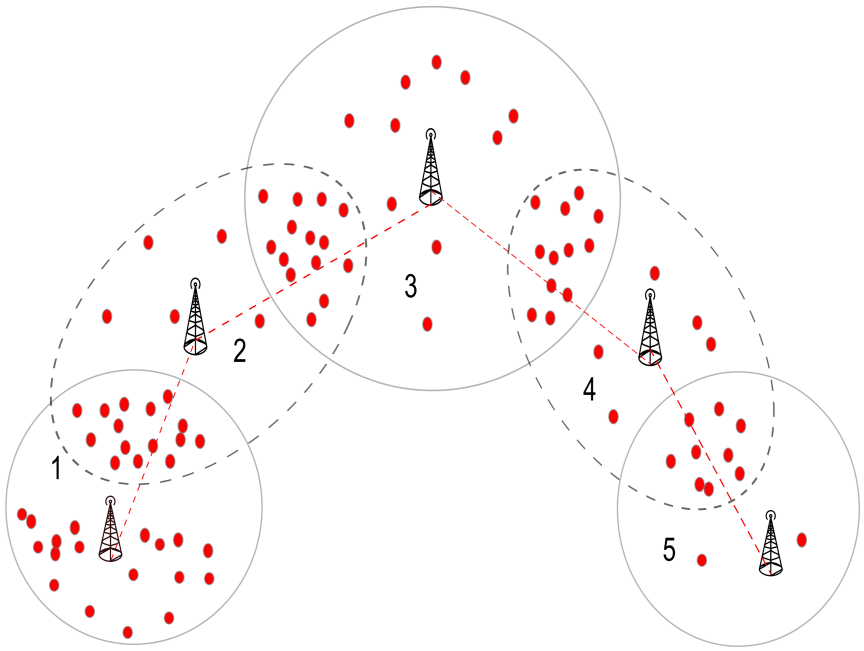


Figure 4.1: Cellular deployments with overlapping regions of coverage. The $5-$th cell is uncongested, but the first cell has a lot of MTDs attached to it. The MTD-BS algorithm learns to distribute the MTDs in a manner that minimzes the overall network congestion. The dotted line depicts micro and pico cells.

The elements in vector, $\mathbf{x}_m$ denote the machine type devices that associated with the $m-$th base station. If we denote, the number of random access opportunities in the m-th cell as T_m, the problem of minimizing congestion can be formulated as:

$$\begin{aligned} \min_{\mathbf{x}_m} \quad & \lim_{T\to\infty} \sum_{t=1}^{T} g(\sum_{m=1}^{M}(\|\mathbf{x}_m(t)\| > T_m(t))) \\ \text{s.t.} \quad & D_{m,n}(g([\mathbf{x}_m]_n)) \leq \delta_m \quad \forall m, n \\ & \sum_{m=1}^{M}[\mathbf{x}_m]_n \leq 1, \quad n = 1, \ldots, N \\ & [\mathbf{x}_m]_n \leq 1, \quad \forall m, n \end{aligned} \tag{4.1}$$

Note that $g(.)$ denotes the indicator function, $\text{g(y)} = \begin{cases} 1, & \text{if y is True} \\ 0, & \text{Otherwise} \end{cases}$

$[\mathbf{x}]_n$ represents the $n-$th element of the vector $\mathbf{x}$, δ_m indicates the coverage radius of the $m-$th base station. If the indicator function, $g([\mathbf{x}_m]_n)$, returns 1; the, D_{mn}, function calculates the distance between $n-$th MTD and the $m-$th base station otherwise it returns "∞".

The set $\mathbf{x} = \{\mathbf{x}_m\}_{m=1}^{m=M}$ is not known at any base station, because no base station has knowledge of the set of MTDs that attempt a transmission at a particular time slot. Hence this problem is intractable. To circumvent this challenge, at each MTD we formulate the optimization problem as an MDP, with a reward system designed to mimic the objective function in Equation 4.1. Concretely, we define a time instant relative to the frame number. Note that the intrinsic finite nature of a frame length [60], guarantees that the number of random access opportunities will be finite, this ensures that the cardinality of action space is alwats finite. This is a necessary condition for the existence of a solution to the algorithm

proposed in the next sections. We consider a scenario where each base station has K random access slots in a single 5GNR-based frame. Hence, in the entire cellular network, there are MK random access opportunities at a single time instant. Clearly, MTDs can only select random access slots from base stations which are $\delta-$suitable. A specific random access slot from the $m-$th base station is denoted by K^m and it is δ-suitable for the $n-$th MTD, if it doesn't violate the constraint $D_{m,n}(g([\mathbf{x}_m]_n)) \leq \delta_m$. This constraint is viable because the signal strength from a specific base station attenuates according to the distance. Hence, each MTD can estimate δ from the received signal strength from each base station. The state space of $n-$th MTD is described by its index, i.e., $S_n(t) = (n)$, based on the current state space, an action, $A_n(t) \in \mathcal{A}_n(S_n(t)) = \{K_0, K_1, K_2, \cdots, K_{MK}\}$ is taken. The first action indicates that the MTD chooses not to transmit, and the other MK actions indicate the slot selected for transmission. Notice, that some MK slots are ineligible for selection, and selection of ineligible action attracts a negative reward and no transmission occurs. The reward signal is termed as follows:

- If the $n-$th MTD selects an illegal action it applies a negative reward, R_n^{ill}.
- Otherwise if the $n-$th MTD receives an ack from the base station following its transmission, it applies a positive reward, R_n^{succ}.
- Otherwise, the $n-$ thMTD applies a negative reward, R_n^{fail}.

This reward structure produces three discrete rewards, hence the cardinality of the set of all possible (state, action, reward) pairs is finite. The solution to the optimization problem is the optimal policy, π^* which produces a sequence of actions that maximizes the cumulative reward averaged across all the MTDs. Specifically, a policy refers to a stochastic description of the likelihood of taking action $\mathbf{A}(t) \in \mathcal{A}(\mathcal{S}(t))$, when in state, $\mathcal{S}(t)$. $\mathcal{A}(\mathcal{S}(t))$ is set of actions available when in state $\mathcal{S}(t)$. Clearly, $\mathcal{A}$ indicates the set of the action space of all

MTDs, $\mathcal{A} = \{\mathcal{A}_n\}_{n=1}^{n=N}$, and $\mathcal{S}$ indicates the set of the state space of all MTDs, $\mathcal{S} = \{S_n\}_{n=1}^{n=N}$. Note that this stochastic mapping is a probability distribution, $\sum_{\mathbf{A}(t)} \pi(\mathbf{A}(t)|\mathcal{S}(t)) = 1$. Starting at state, $S_n(0)$ and following a policy, π_n, the long term average reward derived at the $n-$th MTD can be expressed as:

$$\overline{R_n^{\pi_n}} \quad = \lim_{T\to\infty} \sup \frac{1}{T+1} \sum_{t=0}^{T} \mathbb{E}[R_n(t)|S_n(0)] \tag{4.2}$$

The expectation is taken with respect to the environmental conditions. Environmental conditions like the number of available slots are greatly influenced by the actions of the other MTDs. The final goal is to maximize the weighted sum rewards:

$$\pi^* = \arg\max_{\pi} \sum_{n=1}^{n=N} \theta_n \overline{R_n^{\pi_n}} \tag{4.3}$$

where $\theta_n \geq 1$ and $\sum_{n=1}^{n=N} \theta_n = 1$. θ_n can be set equal to $1/N$ in order to ensure all the MTDs are of equal importance. Equation 4.3 will be approximated by maximizing the individual rewards, $\overline{R_n^{\pi_n}} \quad \forall n$, in an online and distributed manner. In the subsequent sections, we will empirically show that when the number of devices across all cells, N, is less than the total number of random access slots, MK, the solution found by solving this MDP in an online and distributed manner closely approximates the solution derived for the optimization problem in 4.1, if the required association vector sets, $\mathbf{x} = \{\mathbf{x}_m\}_{m=1}^{m=M}$, was provided by some oracle.

4.2 Proposed Solution

The distributed algorithm is largely based on the Q-learning algorithm [58] with randomly initialized state-action value functions. At each time step, each MTD evaluates a particular MTD-BS association through its state-action value functions. The particular action to be evaluated is derived from the behavior policy, π^b. The selected behavior policy is the decaying $\epsilon-$greedy and it provides a means of complementing the exploitation of the optimal MTD-BS association with an infrequent exploration of the cellular environment to determine if there is a better MTD-BS association that provides a better reward. The policy can be described as

$$\pi^b(A_n|S_n(t)) = \begin{cases} A_n = \arg\max_a Q(S_n(t), a), & \text{with probability } 1 - \epsilon(t) + \frac{\epsilon(t)}{|\mathcal{A}_n(S_n(t))|} \\ A_n(t) \neq \arg\max_a Q(S_n(t), a)), & \text{with probability } \epsilon(t)/|\mathcal{A}_n(S_n(t))| \end{cases}$$

The decay factor, $\epsilon(t)$ is time dependent and it is computed through $\epsilon(t) = \frac{1}{\log(t+1)}$.

$$Q(S_n(t), A_n(t)) \leftarrow Q(S_n(t), A_n(t)) + \alpha[R_n(t) + \gamma \max_a Q(S_n(t+1), A_n(t+1)) - Q(S_n(t), A_n(t))] \tag{4.4}$$

The max operator ensures that the action selected from the next state follows the target policy, this algorithm is clearly an off-policy algorithm. For this algorithm to converge and produce the optimal state-action value function, the following necessary conditions must be met:

- The MDP must be finite, this restriction is satisfied from the problem formulation described in the previous sections.
- All the state-actions are continuously visited and their value functions are continuously updated. This is easily satisfied by selecting the right, ϵ value for the behavioural

policy.

- Since the states have no terminal value, the discount factor must be less than one for the algorithm to converge, $\gamma < 1$.

Although the problem formulation satisfies the above listed necessary conditions needed for convergence, the exact theoretical optimal value is not attained. This is due to the constant-step size, α used in Equation 4.4. The constant step size violates the following stochastic approximation constraints [57]:

$$\begin{aligned} \sum_{t=1}^{\infty} \alpha_t &= \infty \\ \sum_{t=1}^{\infty} \alpha_t^2 &< \infty \end{aligned} \tag{4.5}$$

Notice that the first constraint means that the step size is large enough to overcome the initial conditions. This constraint is clearly satisfied with a constant step size parameter. However, the second constraint mandates that eventually, the step size on each iteration of the algorithm will become small enough to guarantee convergence. A constant step size would violate this constraint. The lack of guaranteed theoretical convergence might seem unsatisfactory, but it is important to note that the MTD-BS association problem at each machine type device is non-stationary due to the actions of the other MTDs and attaining a final q^* through a decreasing step size is not desirable as the system must be able to continually adapt to changes in its environment. Hence, this violation is beneficial. Regardless of the absence of guaranteed theoretical based convergence, we empirically will show in the next sections that within a tolerable margin of error, the Q-values do converge.

4.3 Simulation Results

This section validates the performance of the proposed multi cell MTD-BS association algorithm. We base our simulation on parameters specified by the 3GPP for LTE based machine type communication [60]. This document describes in detail the number of RACH opportunities present for different frame structures, the number of preambles present, it also provides a description of metrics such asthe probability of collision. These metrics are needed in order to evaluate any algorithm designed for MTC. As depicted in 4.2, this simulation deploys seven Macrocells and three picocells. To model the practical use of picocells which is to extend cell coverage or to alleviate congestion [61], the picocells are deployed at the cell edge of the Macrocell. Each of the three picocells can associate with 20 percent of the total number of MTDs in the cellular network.

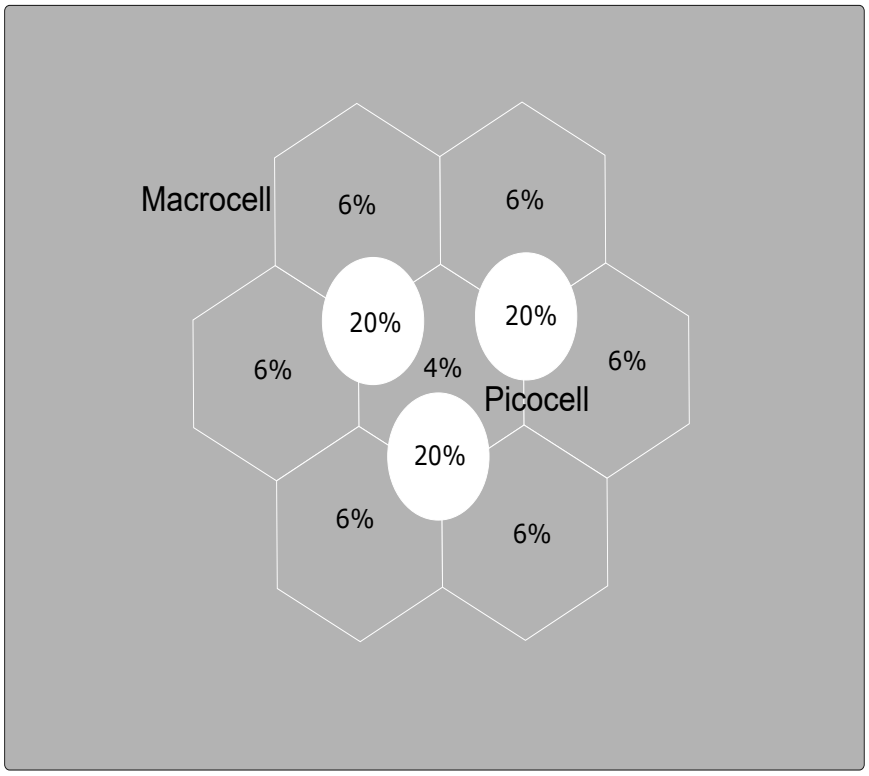


Figure 4.2: Heterogeneous cellular network with seven Macrocells complemented with 3 picocells placed at the congestion hot spots.

All N MTDs are deployed with processors capable of performing simple addition and multiplication operations, the locations of the base stations are also assumed to be hardcoded into the MTD during deployment. The LTE simulation parameters are as described in the Table 4.1.

Table 4.1: Simulation Parameters

Parameters	Value
Total Number of Preambles	1
Total Number of MTDs, N	100,200,300,400,500,600,700
Total Number of Macrocells	7
Total Number of picocells	3
RA Slots/frame	50
Distribution of MTDs	Except the center cell which is $\delta-$suitable for 4% of the total MTDs, every other Macrocell is assumed to be $\delta-$suitable for 6% of the total MTDs. Each of the pico cells are $\delta-$suitable for 20% of all the MTDs in the network.

We evaluate the proposed MTD-BS association scheme in terms of the following metrics:

- System level probability of collision: This metric evaluates the system wide congestion as the ratio between the number of random access slots that are selected more than once across all cells and the total number of random access slots across all cells.
- System level channel utilization: This metric evaluates the efficiency in system wide utilization of the random access slots. It is evaluated as the ratio between the number of random access slots used for a successful transmission and the total number of

available random access. Both the former and the latter parameters are measured across all cells

- System level access success probability: This is simply the ratio between number of bits/s/Hz that is successfully transmitted across all cells and the number of bits/s/Hz that could be transmitted if we had an infinite number of random access slots.

In addition, we also compute cell-specific versions of some of the above-listed metrics. We present the equivalent cell-specific versions for the tightly congested cells in this chapter and present the metrics for uncongested cells in Appendix A. We compare the performance of our scheme to the following baselines: I) cell-specific random allocation scheme , II) the collaborative distributed Q-learning scheme presented in [42] - this scheme uses Q-learning to find unique time slots, III) oracle - This is the optimal association that can be achieved if we had a perfect knowledge of the set $\mathbf{x}$. It is derived using an exhaustive search, IV) Lastly, we provide a cell specific oracle - this scheme finds the optimal MTD-slot assignment for each cell. We remark that the last two baselines can't be implemented in practice. Lastly, we provide a chart showing the redistribution of MTDs from congested cells to other less congested $\delta-$suitable cells.

Figure 4.3 depicts the overall channel utilization ratio of the proposed multi-cell association algorithm in comparison with the random selection and collaborative RL schemes [42]. Also, an overall oracle based scheme and cell-specific oracle based schemes are provided as a baseline. The multi-cell association scheme is able to approximate the best solution in terms of overall channel utilization when the number of MTDs is less than the number of time slots. Beyond this threshold, the multi-cell algorithm can't find unique slots for MTD transmissions, and it assigns more than one user to a slot. The collaborative RL scheme performs substantially worse than the proposed solution when the number of MTDs in the

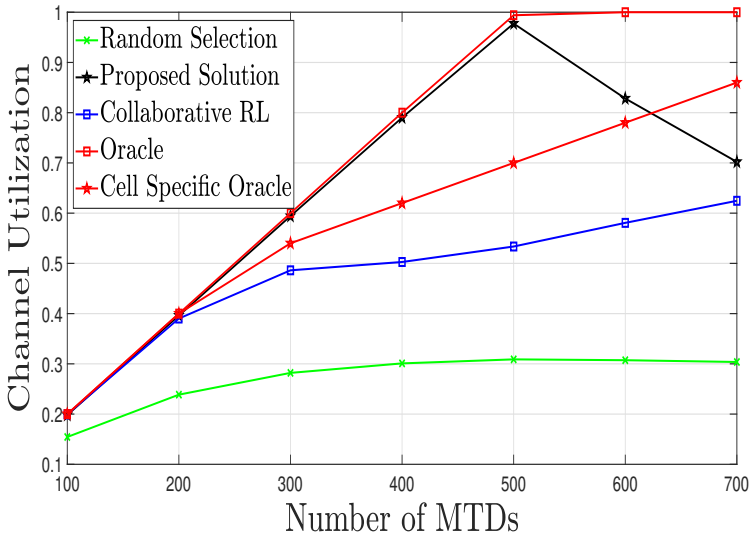


Figure 4.3: Channel Utilization Ratio.

most congested cells is greater than the number of random access opportunities in those cells. In Figure 4.3, this happens at $N = 300$. At this point, the number of MTDs in cells eight, nine, and ten, i.e. $N_8 = 60, N_9 = 60, N_{10} = 60$, is greater than the number of available time slots, 50. Notice that the distributed nature of the collaborative RL scheme ensures that it performs worse than a cell-specific oracle based scheme. This happens because the oracle is centralized and has information that the RL-based algorithms do not have. The oracle knows the total number of MTDs trying to access a specific base station and hence it assigns slots such that the number of served users is less than or equal to the number of available time slots. The oracle does not attempt to serve MTDs when there are no free slots. We depict the channel utilization for the most congested cells in Figures 4.4, 4.5, and 4.6. We employ the same baselines used in Figure 4.3. Notice that when the number of users is significantly less than the total number of time slots, K. The per-cell utilization of the proposed algorithm is lower than that of the oracle based solution as depicted in Figures 4.4, 4.5, and 4.6. This is explained by the fact that our algorithm splits the MTDs across multiple cells while both of the oracle based solutions maximize cell capacity. The oracle

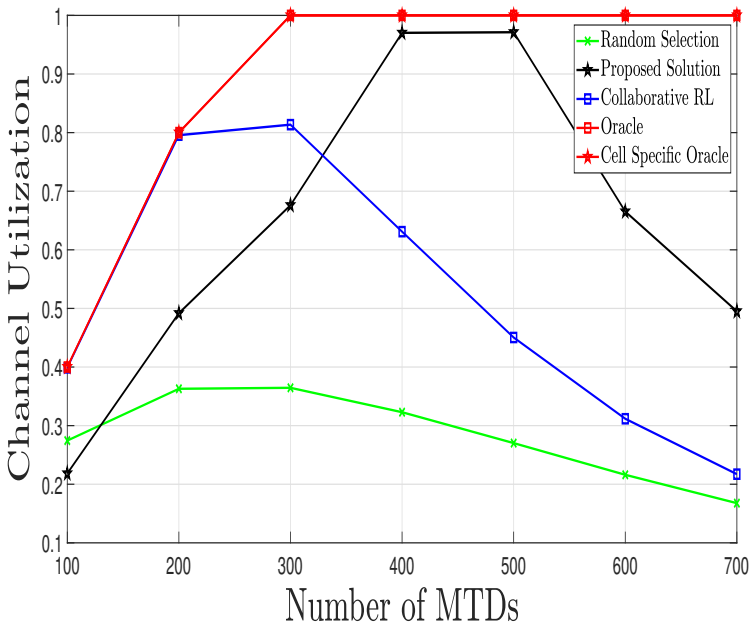


Figure 4.4: Channel Utilization Ratio for the 8−th Cell.

scheme only redistributes MTDs across eligible cells when the current cell capacity has been maximized. This difference in operation is also evident by the channel utilization in the less congested cells [see Figures 4.7, 4.8]. Channel utilization plots for other cells can be found be in Appendix A.1

When the network is highly congested across all cells, the per cell utilization of the proposed solution rapidly drops off because MTDs can no longer find a suitable time slot. Hence, collisions occur and the time slots are underutilized. Notice, that the collaborative RL scheme saturates earlier than the multi-cell association scheme. This is because the proposed scheme can redistribute MTDs across multiple cells. The redistribution of MTDs across cells can be depicted using Figure 4.9 and Figure 4.10.

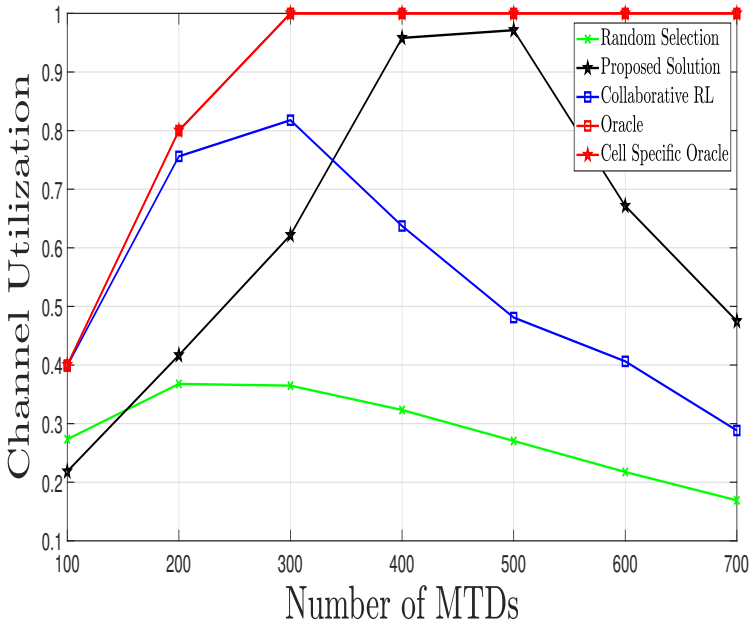


Figure 4.5: Channel Utilization Ratio for the 9−th Cell.

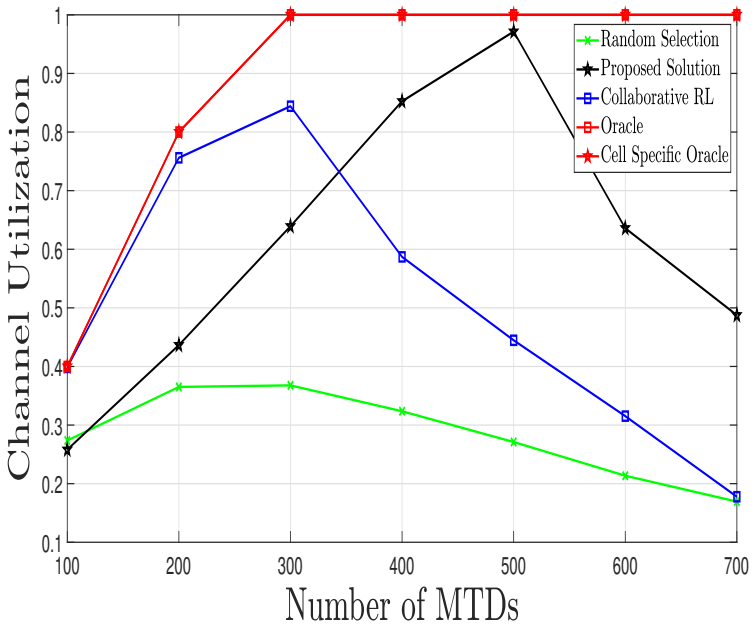


Figure 4.6: Channel Utilization Ratio for the 10−th Cell.

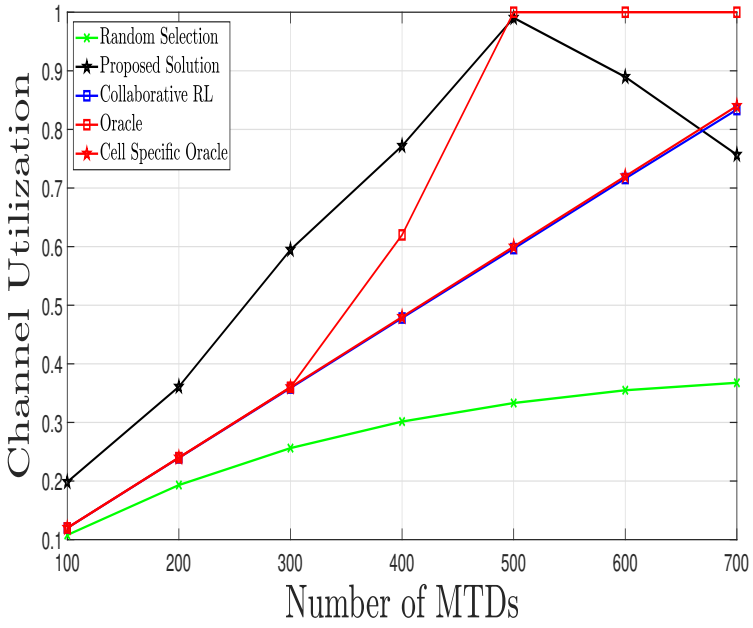


Figure 4.7: Channel Utilization Ratio for the 3−th Cell.

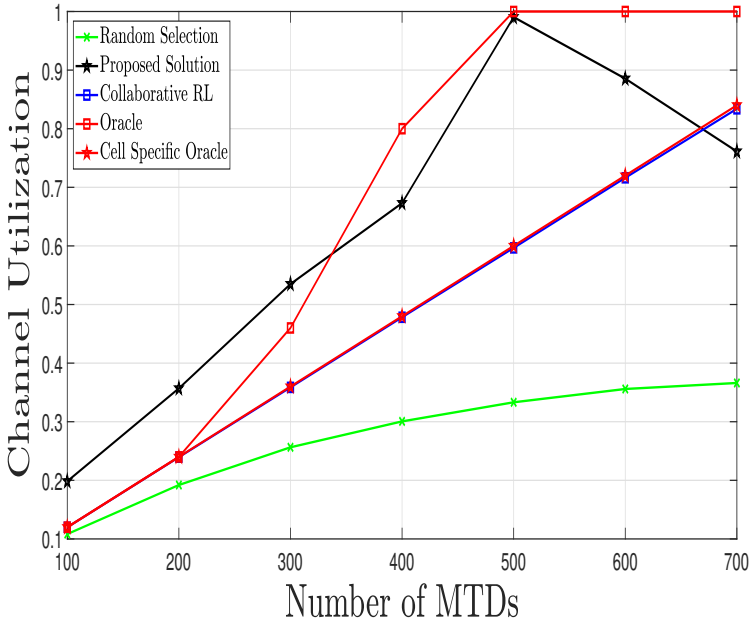
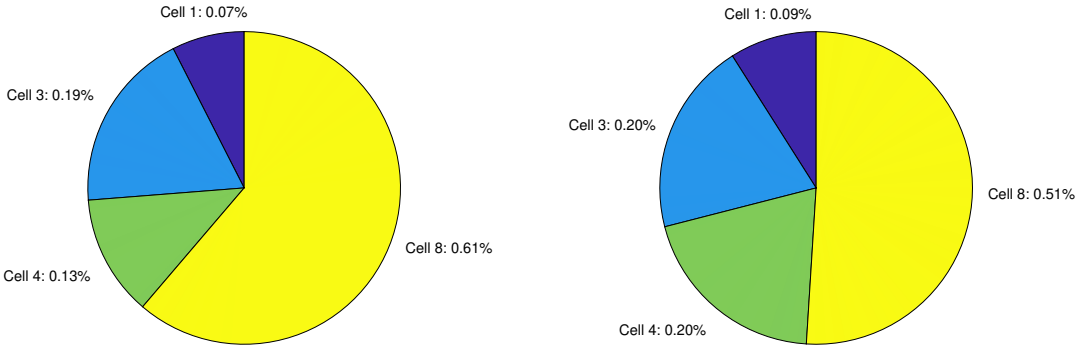
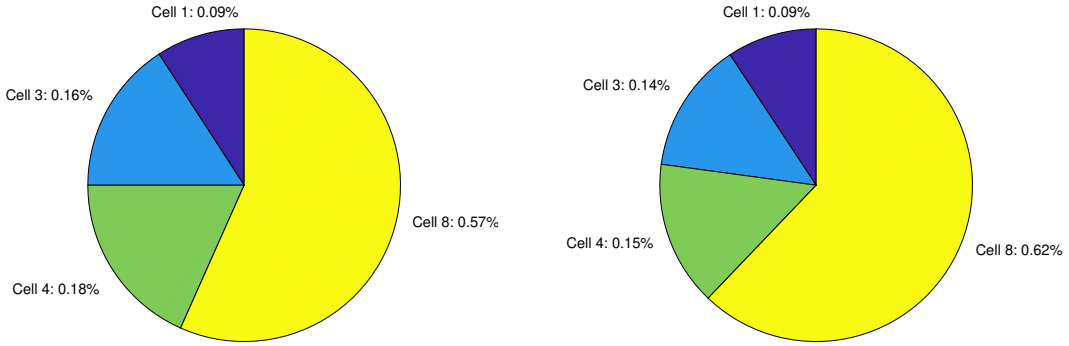


Figure 4.8: Channel Utilization Ratio for the 4−th Cell.



(a) $N = 400$, $K = 500$ (b) $N = 500$, $K = 500$

Figure 4.9: Learned redistribution of MTDs attached to Cell 8 to other $\delta-$suitable cells



(a) $N = 600$, $K = 500$ (b) $N = 700$, $K = 500$

Figure 4.10: Learned redistribution of MTDs attached to Cell 8 to other $\delta-$suitable cells

The probability of the selection of a single slot by multiple MTDs is depicted in Figure 4.11. We provide a comparison between the proposed MTD-BS association algorithm and the collaborative RL scheme.

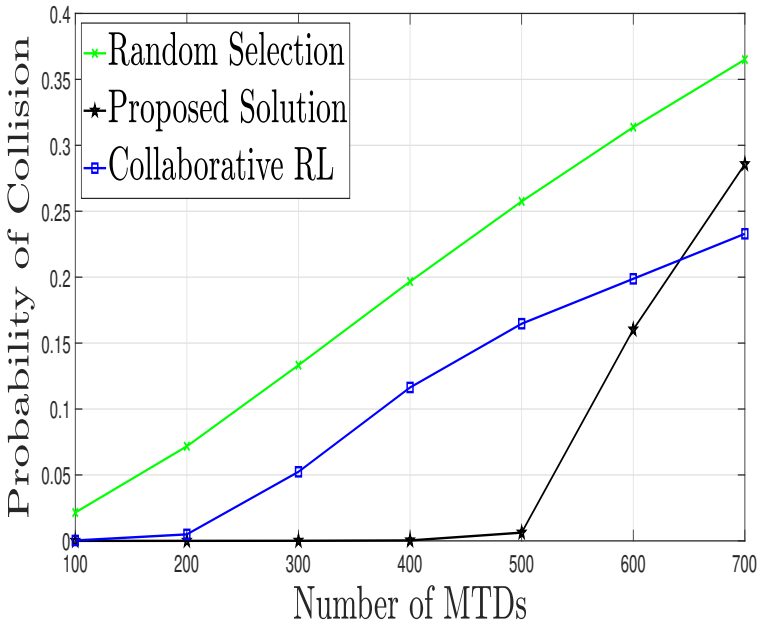


Figure 4.11: Probability of Collisions across all Cells.

The random selection algorithm is provided as a baseline. When the total MTD count, N is less than or equal to the total number of slots, K, the proposed scheme has a near-zero probability of collision. When the number of MTDs is greater than this threshold, MTDs are unable to find uncongested cells and the probability of collision increases rapidly. The collaborative algorithm experience significant collisions when N, is greater than 200. Again, this is due to the fact that above this threshold, the cells, 8, 9, and 10 are extremely congested and the load can't be redistributed because there are no free slots in any cells. A cell-specific depiction of the collision probability for the three most congested cells is presented in Figure 4.12, 4.13 and 4.14. As expected, these plots show a gradual cell wise deterioration in performance of the proposed algorithm when the total number of users is greater than the

total number of slots. The collision probability for uncongested cells is presented in appendix A.2. One interesting observation is that the collaborative RL algorithm never experiences collisions in uncongested cells, while the proposed algorithm experiences a significant number of collisions in the uncongested cells when $N > K$. This is due to the redistribution of MTDs from the congested cells to the uncongested cells. We can view this more concretely using the following scenario: when the total number of users, $N = 600$, and there are $K = 500$ random access opportunities uniformly distributed across 10 possible cells. The 1st, 3rd, 4th and 8−th cells initially have $N_1 = 24, N_3 = 36, N_4 = 36$ and $N_8 = 120$ MTDs respectively, but each cell has only $K_8 = 50$ random access slots. The 8−th cell is obviously congested, but some of its MTDs can attach to the 1st, 3rd and 4th cells, [see Figure 4.2]. If the 8th cell redistributes its MTDs across these cells, according to the following percentages $9\%, 16\%$ and 18% , [see Figure 4.10a]. The 3rd cell now has $N_3 = 60$ MTDs, and it is congested. Therefore, the probability of a collision could be substantial. This is depicted in Figure A.8.

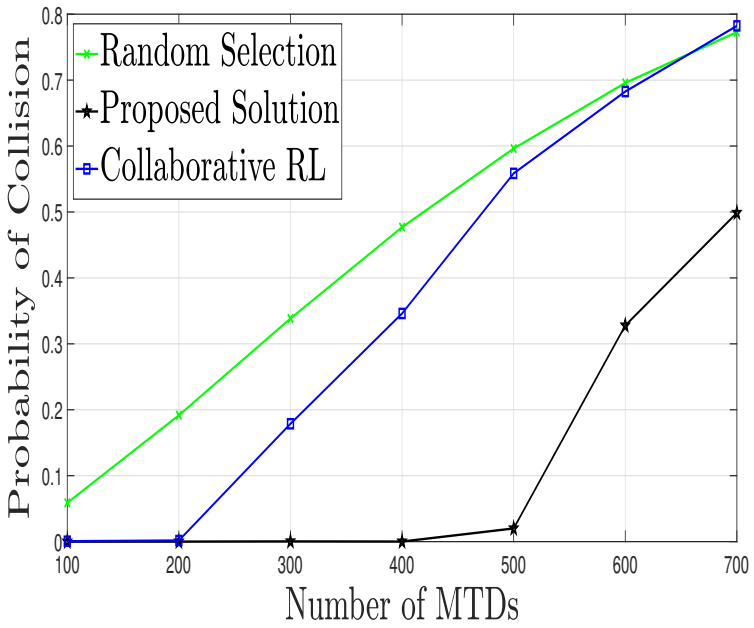


Figure 4.12: Probability of Collisions for the 8−th Cell.

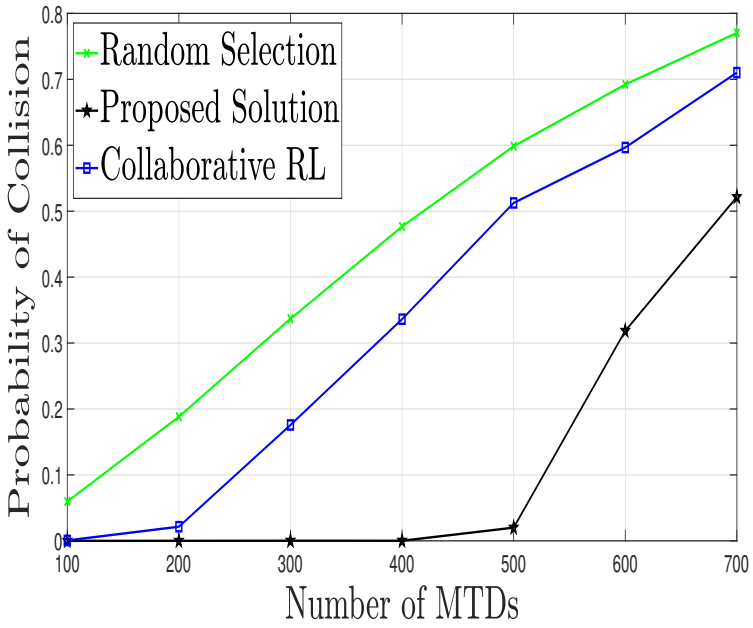


Figure 4.13: Probability of Collisions for the 9−th Cell.

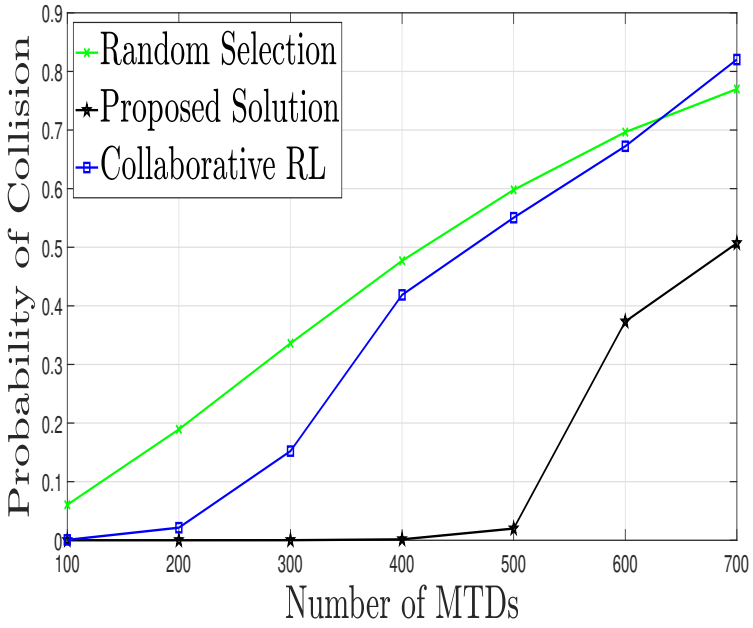


Figure 4.14: Probability of Collisions for the 10−th Cell.

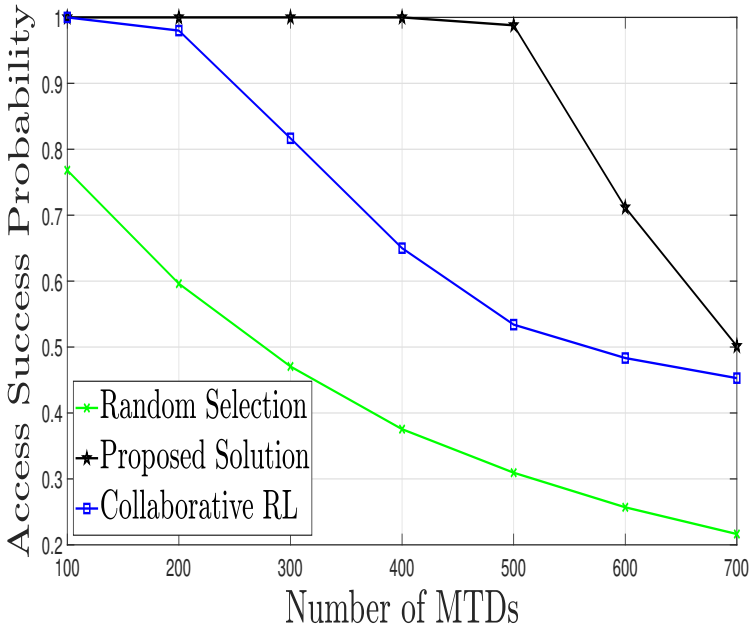


Figure 4.15: Access Success Probability.

One of the constraints in random access for machine type communication is the limited number of slots. To provide an intuition on the efficiency of the proposed algorithm, we employ the performance metric shown in Figure 4.15) to describe the ratio between the number of bits transmitted and the number of bits that could be transmitted if we had an infinite number of slots. Notice, that when $N \leq K$, we have a near optimal performance.

As depicted in Figure 4.16, the variation among the selected state-action value is minimal after 70 seconds or 70000 frames. This indicates a pseudo convergence of actions after a certain time interval.

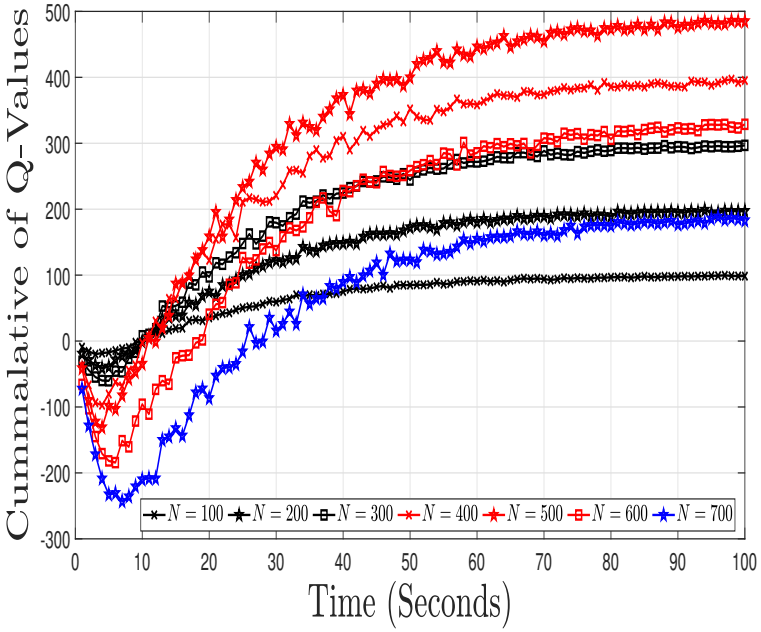


Figure 4.16: Cumulative of the State Action Value that is Selected every Second by All MTDs while Following the MTD-BS off-policy Association Algorithm.

4.4 Conclusion

This chapter formulates the task of deploying a heterogeneous network to eliminate the congestion caused massive number of transmissions from machine type devices as a constrained combinatorial optimization problem. The constrained optimization problem is re-formulated as an MDP. The solution to the MDP is approximated using a distributed model-free algorithm. The work done is empirically validated through the probability of collision, channel utilization and success access probability metrics.

Chapter 5

Conclusions and Future Work

In this thesis, we have investigated two applications of machine learning to wireless systems. First, we investigated the possibility of designing a neural network-based space-time coding scheme. We describe in detail the neural network based system model and the training scheme used to generate the neural network-based based STBCs. We provide a comparison of the symbol error rate performance obtained from the neural network with the symbol error rate performance obtained from conventional OSTBCs. We note that during decoding, the conventional OSTBCs employs perfect channel estimation and maximum ratio combining at the receiver. On the other hand, through the pre-processing block, the neural network infers a complex matrix from a batch of received symbols. After this, it uses this complex matrix to combine the set received symbols. More specifically, at the neural decoder, we eliminate pilot placement but incur additional delay at the pre-processing stage. Second, we investigate the task of eliminating the long term congestion caused by concurrent data transmissions from a massive number of machine type devices in a heterogeneous cellular network. An important aspect of heterogeneous cellular networks is that the macro cells and pico (or micro) cells typically have coverage areas that overlap to some degree. This results in the possibility of multiple MTD-base station association. We investigate the optimal MTD-gNB association which minimizes the long term congestion experienced in the overall cellular network. To address this problem, we formulate the optimization problem as a combinatorial problem and state that the problem is intractable due to the unavailability of the set of all

association vectors at any base station. We reformulate the optimization problem as a finite MDP and develop a distributed MTD-BS association scheme to approximate the solution of the resulting MDP. We validate this work empirically through the probability of collision, channel utilization, and access success probability of the resulting system. Empirically results indicate that the optimal solution is achieved when the number of time MTDs is less than or equal to the number of time slots.

A line of possible research is to extend the learned system to the large antenna regime. An interesting application of this thesis would be investigating if the batch pre-processing block extends to massive MIMO. The potential to use the pre-processing block developed for massive MIMO will reduce the required pilot overhead. Another potential area is to extend the state space of the reinforcement learning solutions in order to include the latency requirements of MTDs. Lastly, an investigation can be carried out on the possibility of a non-orthogonal resource sharing reinforcement learning-based framework.

Bibliography

[1] 3GPP, "Technical Specification Group Services and System Aspects," *Study on enhancement of 3GPP Support for 5G V2X Services*, vol. 3GPP TR 22.886, 2018.

[2] E. Zehavi, "8-PSK trellis codes for a Rayleigh channel," *IEEE Transactions on Communications*, vol. 40, no. 5, pp. 873–884, 1992.

[3] T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[4] J. O. Berger, *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media, 2013.

[5] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Advances in neural information processing systems*, 2018, pp. 8778–8788.

[6] D. G. Brennan, "Linear Diversity Combining Techniques," *Proceedings of the IRE*, vol. 47, no. 6, pp. 1075–1102, 1959.

[7] A. Wittneben, "Analysis and comparison of optimal predictive transmitter selection and combining diversity for DECT," in *Proceedings of GLOBECOM '95*, vol. 2, 1995, pp. 1527–1531 vol.2.

[8] N. Chiurtu, B. Rimoldi, and E. Telatar, "On the capacity of multi-antenna Gaussian channels," in *Proceedings. 2001 IEEE International Symposium on Information Theory (IEEE Cat. No.01CH37252)*, 2001, pp. 53–.

[9] S. M. Alamouti, "A simple transmit diversity technique for wireless communications," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 8, pp. 1451–1458, 1998.

[10] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1456–1467, 1999.

[11] ——, "Space-time block coding for wireless communications: performance results," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 3, pp. 451–460, 1999.

[12] R. Daniels and R. W. Heath, "Online adaptive modulation and coding with support vector machines," in *2010 European Wireless Conference (EW)*, 2010, pp. 718–724.

[13] H. Ye, G. Y. Li, and B. Juang, "Power of Deep Learning for Channel Estimation and Signal Detection in OFDM Systems," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, 2018.

[14] X. Wang, L. Gao, S. Mao, and S. Pandey, "CSI-Based Fingerprinting for Indoor Localization: A Deep Learning Approach," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 1, pp. 763–776, 2017.

[15] T. J. O'Shea, T. Erpek, and T. C. Clancy, "Physical layer deep learning of encodings for the MIMO fading channel," in *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2017, pp. 76–80.

[16] F. A. Aoudia and J. Hoydis, "Model-Free Training of End-to-End Communication Systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 11, pp. 2503–2516, 2019.

[17] J. Song, B. Peng, C. Häger, H. Wymeersch, and A. Sahai, "Learning Physical-Layer

Communication With Quantized Feedback," *IEEE Transactions on Communications*, vol. 68, no. 1, pp. 645–653, 2020.

[18] 3GPP, "Technical Specification Group Services and System Aspects," *TECHNICAL SPECIFICATION*, vol. 3GPP TS 22.101, 2017.

[19] S. Lien, T. Liau, C. Kao, and K. Chen, "Cooperative Access Class Barring for Machine-to-Machine Communications," *IEEE Transactions on Wireless Communications*, vol. 11, no. 1, pp. 27–32, 2012.

[20] S. Duan, V. Shah-Mansouri, and V. W. S. Wong, "Dynamic access class barring for M2M communications in LTE networks," in *2013 IEEE Global Communications Conference (GLOBECOM)*, 2013, pp. 4747–4752.

[21] H. S. Jang, H. Jin, B. C. Jung, and T. Q. S. Quek, "Recursive Access Class Barring for Machine Type Communications with PUSCH Resource Constraints," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–6.

[22] X. Yang, A. Fapojuwo, and E. Egbogah, "Performance Analysis and Parameter Optimization of Random Access Backoff Algorithm in LTE," in *2012 IEEE Vehicular Technology Conference (VTC Fall)*, 2012, pp. 1–5.

[23] G. T. R. WG2, "MTC simulation results with specific solutions ," *3rd Generation Partnership Project (3GPP)*, vol. 3GPP TS 22.101, 2017.

[24] C. Stefanovic and P. Popovski, "ALOHA Random Access that Operates as a Rateless Code," *IEEE Transactions on Communications*, vol. 61, no. 11, pp. 4653–4662, 2013.

[25] F. Vazquez-Gallego, J. Alonso-Zarate, A. M. Mandalari, O. Briante, A. Molinaro, and G. Ruggeri, "Performance evaluation of reservation frame slotted-aloha for data collec-

tion m2m networks," in *European Wireless 2014; 20th European Wireless Conference*, 2014, pp. 1–6.

[26] G. Farhadi and A. Ito, "Group-Based Signaling and Access Control for Cellular Machine-to-Machine Communication," in *2013 IEEE 78th Vehicular Technology Conference (VTC Fall)*, 2013, pp. 1–6.

[27] N. K. Pratas, H. Thomsen, . Stefanović, and P. Popovski, "Code-expanded random access for machine-type communications," in *2012 IEEE Globecom Workshops*, 2012, pp. 1681–1686.

[28] H. S. Jang, S. M. Kim, K. S. Ko, J. Cha, and D. K. Sung, "Spatial Group Based Random Access for M2M Communications," *IEEE Communications Letters*, vol. 18, no. 6, pp. 961–964, 2014.

[29] T. Kim, H. S. Jang, and D. K. Sung, "An Enhanced Random Access Scheme With Spatial Group Based Reusable Preamble Allocation in Cellular M2M Networks," *IEEE Communications Letters*, vol. 19, no. 10, pp. 1714–1717, 2015.

[30] Y. Liang, X. Li, J. Zhang, and Z. Ding, "Non-Orthogonal Random Access for 5G Networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 7, pp. 4817–4831, 2017.

[31] H. S. Jang, H. Park, and D. K. Sung, "A Non-Orthogonal Resource Allocation Scheme in Spatial Group Based Random Access for Cellular M2M Communications," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 5, pp. 4496–4500, 2017.

[32] M. Shirvanimoghaddam, Y. Li, M. Dohler, B. Vucetic, and S. Feng, "Probabilistic Rateless Multiple Access for Machine-to-Machine Communication," *IEEE Transactions on Wireless Communications*, vol. 14, no. 12, pp. 6815–6826, 2015.

[33] N. K. Pratas, C. Stefanovic, G. C. Madueno, and P. Popovski, "Random Access for Machine-Type Communication Based on Bloom Filtering," in *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1–7.

[34] A. E. Kalor, O. A. Hanna, and P. Popovski, "Random Access Schemes in Wireless Systems with Correlated User Activity," in *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2018, pp. 1–5.

[35] N. Zhang, G. Kang, J. Wang, Y. Guo, and F. Labeau, "Resource Allocation in a New Random Access for M2M Communications," *IEEE Communications Letters*, vol. 19, no. 5, pp. 843–846, 2015.

[36] G. Corrales Madueño, . Stefanović, and P. Popovski, "Reliable Reporting for Massive M2M Communications With Periodic Resource Pooling," *IEEE Wireless Communications Letters*, vol. 3, no. 4, pp. 429–432, 2014.

[37] N. Abuzainab, W. Saad, C. S. Hong, and H. V. Poor, "Cognitive Hierarchy Theory for Distributed Resource Allocation in the Internet of Things," *IEEE Transactions on Wireless Communications*, vol. 16, no. 12, pp. 7687–7702, 2017.

[38] Z. Chen, F. Sohrabi, and W. Yu, "Sparse activity detection for massive connectivity," *IEEE Transactions on Signal Processing*, vol. 66, no. 7, pp. 1890–1904, 2018.

[39] L. Liu, E. G. Larsson, W. Yu, P. Popovski, C. Stefanovic, and E. de Carvalho, "Sparse Signal Processing for Grant-Free Massive Connectivity: A Future Paradigm for Random Access Protocols in the Internet of Things," *IEEE Signal Processing Magazine*, vol. 35, no. 5, pp. 88–99, 2018.

[40] K. Senel and E. G. Larsson, "Joint User Activity and Non-Coherent Data Detection in

mMTC-Enabled Massive MIMO Using Machine Learning Algorithms," in *WSA 2018; 22nd International ITG Workshop on Smart Antennas*, 2018, pp. 1–6.

[41] E. de Carvalho, E. Björnson, J. H. Sørensen, E. G. Larsson, and P. Popovski, "Random Pilot and Data Access in Massive MIMO for Machine-Type Communications," *IEEE Transactions on Wireless Communications*, vol. 16, no. 12, pp. 7703–7717, 2017.

[42] S. K. Sharma and X. Wang, "Collaborative Distributed Q-Learning for RACH Congestion Minimization in Cellular IoT Networks," *IEEE Communications Letters*, vol. 23, no. 4, pp. 600–603, 2019.

[43] R. Taylor, D. Baron, and D. Schmidt, "The world in 2025 - predictions for the next ten years," in *2015 10th International Microsystems, Packaging, Assembly and Circuits Technology Conference (IMPACT)*, 2015, pp. 192–195.

[44] S. K. Samanta and C. K. Chanda, "Wireless power network design through smart grid transmission system model," in *2015 International Conference on Energy, Power and Environment: Towards Sustainable Growth (ICEPE)*, 2015, pp. 1–5.

[45] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[46] A. F. Agarap, "Deep learning using rectified linear units (relu)," *arXiv preprint arXiv:1803.08375*, 2018.

[47] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1, 2013, p. 3.

[48] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[49] C. Dugas, Y. Bengio, F. Bélisle, C. Nadeau, and R. Garcia, "Incorporating second-order functional knowledge for better option pricing," in *Advances in neural information processing systems*, 2001, pp. 472–478.

[50] J. S. Bridle, "Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters," in *Advances in neural information processing systems*, 1990, pp. 211–217.

[51] P. Ramachandran, B. Zoph, and Q. V. Le, "Swish: a self-gated activation function," *arXiv preprint arXiv:1710.05941*, vol. 7, 2017.

[52] P. J. Werbos, "Applications of advances in nonlinear sensitivity analysis," in *System modeling and optimization*. Springer, 1982, pp. 762–770.

[53] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 2, pp. 1–25, 2017.

[54] N. Jia and E. Y. Lam, "Machine learning for inverse lithography: using stochastic gradient descent for robust photomask synthesis," *Journal of Optics*, vol. 12, no. 4, p. 045601, 2010.

[55] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural networks*, vol. 12, no. 1, pp. 145–151, 1999.

[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[57] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.

[58] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[59] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[60] 3GPP, "Study on RAN improvements for machine-type communications," *TECHNICAL SPECIFICATION*, vol. 3GPP TS 37.868, 2017.

[61] S.-Y. Lien, T.-H. Liau, C.-Y. Kao, and K.-C. Chen, "Cooperative access class barring for machine-to-machine communications," *IEEE Transactions on Wireless Communications*, vol. 11, no. 1, pp. 27–32, 2011.

Appendices

Appendix A

Reinforcement Learning for Massive Machine Type Communication

A.1 Channel Utilization Ratio for uncongested Cells

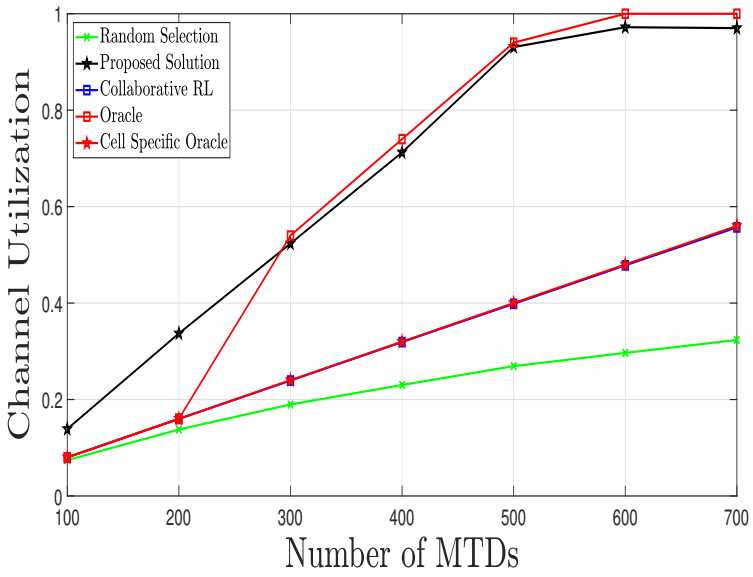


Figure A.1: Channel Utilization Ratio for the 1−th Cell.

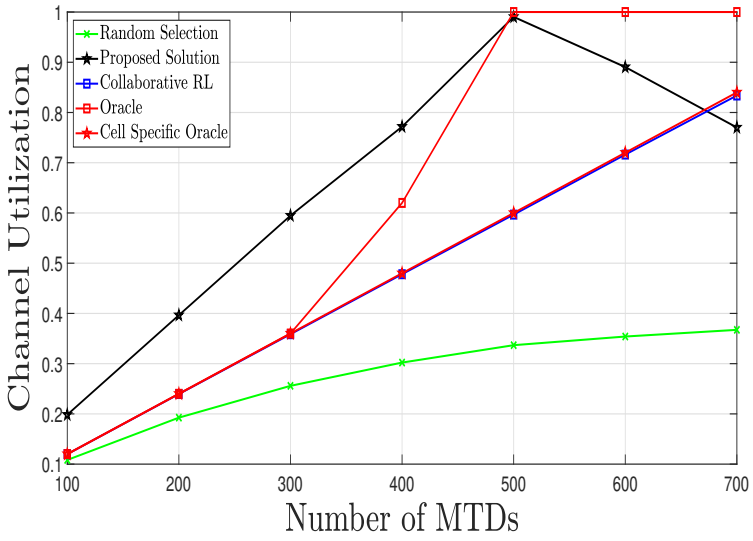


Figure A.2: Channel Utilization Ratio for the 2−th Cell.

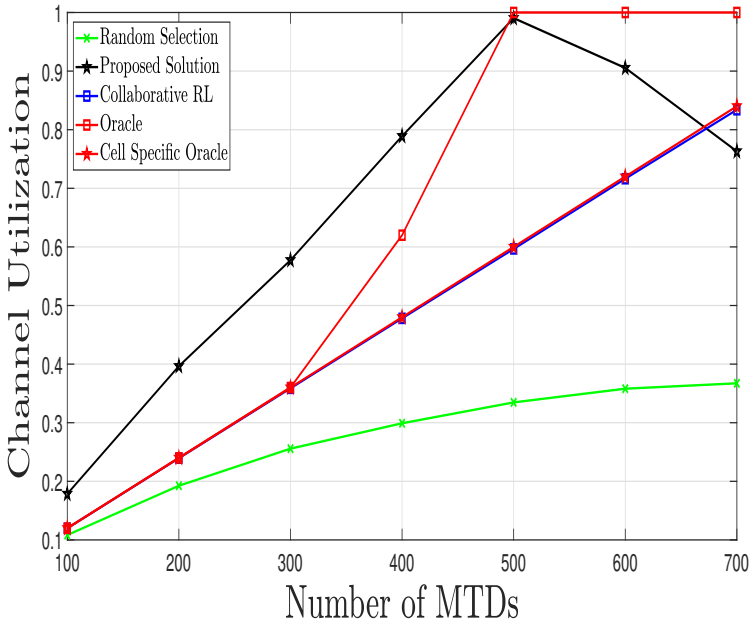


Figure A.3: Channel Utilization Ratio for the 5−th Cell.

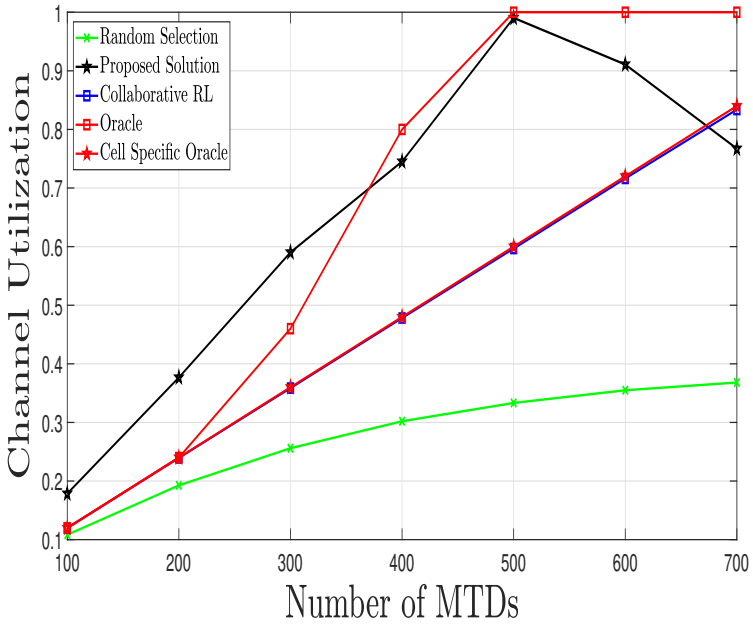


Figure A.4: Channel Utilization Ratio for the 6−th Cell.

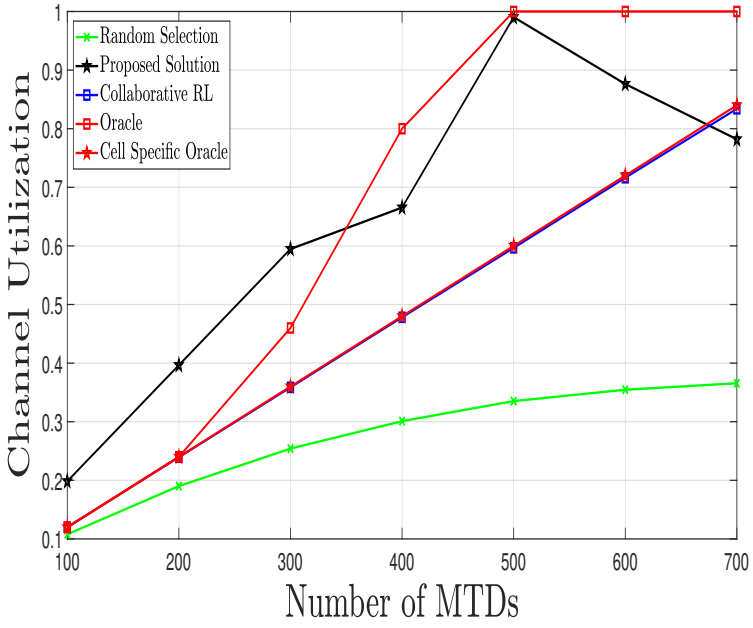


Figure A.5: Channel Utilization Ratio for the 7−th Cell.

A.2 Probability of Collision for uncongested Cells

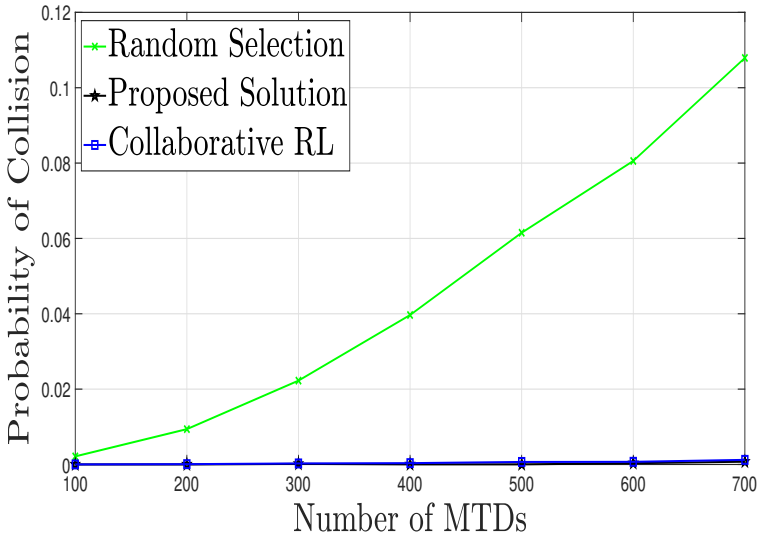


Figure A.6: Probability of Collisions for the 1−th Cell.

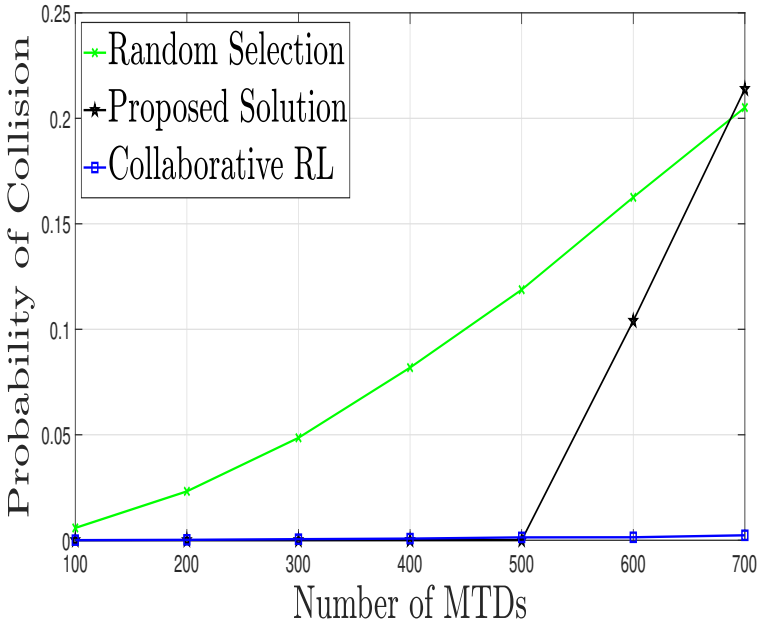


Figure A.7: Probability of Collisions for the 2−th Cell.

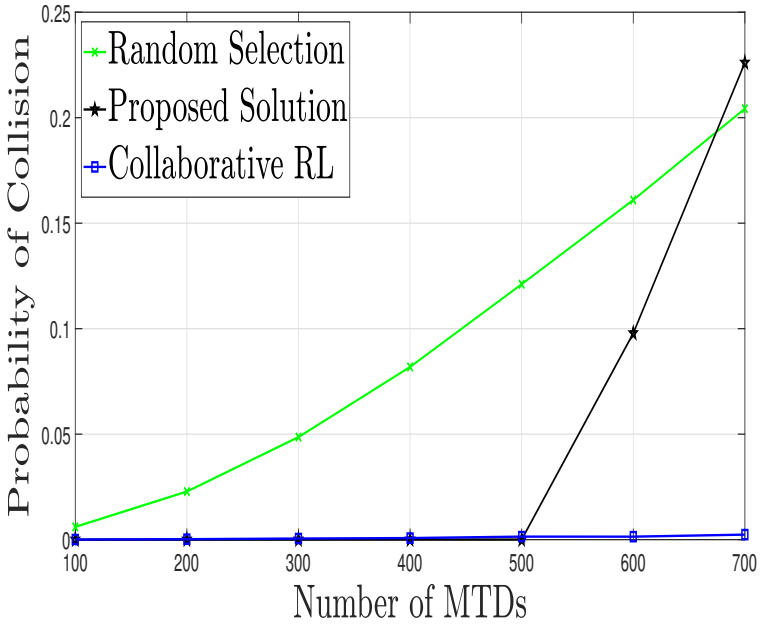


Figure A.8: Probability of Collisions for the 3−th Cell.

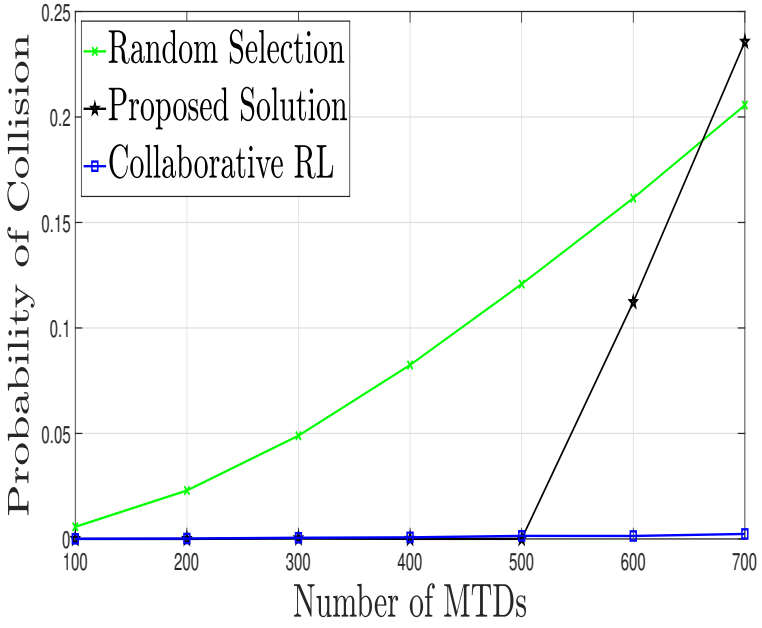


Figure A.9: Probability of Collisions for the 4−th Cell.

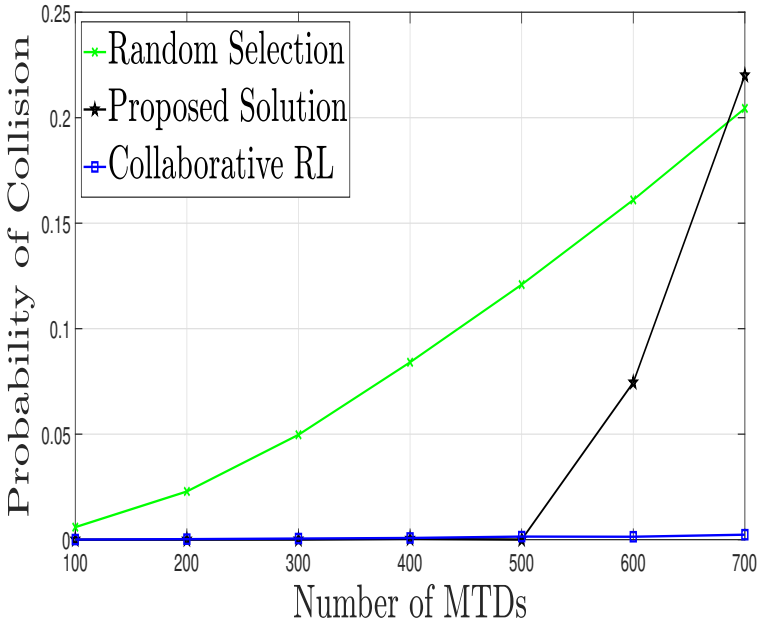


Figure A.10: Probability of Collisions for the 5−th Cell.

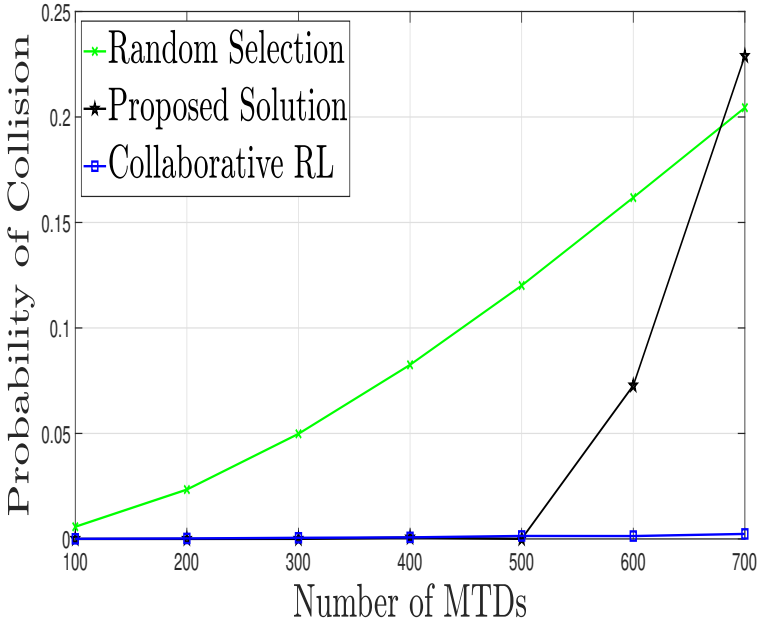


Figure A.11: Probability of Collisions for the 6−th Cell.

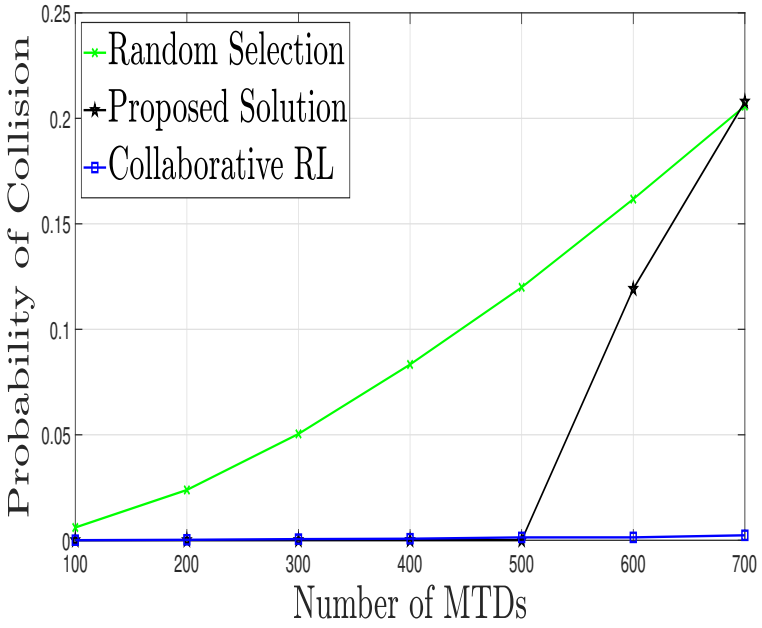


Figure A.12: Probability of Collisions for the 7−th Cell.