Gated Transformer-Based Architecture for Automatic Modulation Classification

Antorip Sahu

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

Computer Engineering

December 14, 2023

Blacksburg, Virginia

Keywords: artificial intelligence, deep learning, neural networks, transformer, automatic modulation classification

Gated Transformer-Based Architecture for Automatic Modulation Classification

Antorip Sahu

(ABSTRACT)

This thesis delves into the advancement of 5G portable test-nodes in wireless communication systems with cognitive radio capabilities, specifically addressing the critical need for dynamic spectrum sensing and awareness at the radio receiver through AI-driven automatic modulation classification. Our methodology is centered around the transformer encoder architecture incorporating a multi-head self-attention mechanism. We train our architecture extensively across a diverse range of signal-to-noise ratios (SNRs) from the RadioML 2018.01A dataset. We introduce a novel transformer-based architecture with a gated mechanism, designed as a runtime re-configurable automatic modulation classification framework, which demonstrates enhanced performance with low SNR RF signals during evaluation, an area where conventional methods have shown limitations, as corroborated by existing research. Our innovative single-model framework employs distinct weight sets, activated by varying SNR levels, to enable a gating mechanism for more accurate modulation classification. This advancement in automatic modulation classification marks a crucial step toward the evolution of smarter communication systems.

Gated Transformer-Based Architecture for Automatic Modulation Classification

Antorip Sahu

(GENERAL AUDIENCE ABSTRACT)

This thesis delves into the advancement of wireless communication systems, particularly in developing portable devices capable of effectively detecting and analyzing radio signals with cognitive radio capabilities. Central to our research is leveraging artificial intelligence (AI) for automatic modulation classification, a method to identify signal modulation types. We utilize a transformer-based AI model trained on the RadioML 2018.01A dataset. Our training approach is particularly effective when evaluating low-quality signals using a gating mechanism based on signal-to-noise ratios, an area previously considered challenging in existing research. This work marks a significant advancement in creating more intelligent and responsive wireless communication systems.

Dedication

In loving memory of my dear mother, whose endless love and tireless dedication inspire me every day.

Acknowledgments

I wish to extend my heartfelt gratitude to my advisor, Professor Carl Dietrich. His mentorship, rooted in his vast expertise, greatly enriched my academic journey at Virginia Tech. His unwavering support and guidance, alongside our numerous discussions about projects like PORT5+ and CORNET upgradation, as well as the array of tasks under the Wireless@VT research group, were instrumental in my growth and learning. Further, I am highly appreciative of Professor Creed Jones. His mentorship allowed me to serve as a teaching assistant, and I am grateful for the myriad insights he bestowed upon my research. My academic journey was also enhanced by Professor Kendall Giles, whose course on Cybersecurity and the Internet-of-Things broadened my perspective in this domain. I owe a profound gratitude to John Ghra from the Electrical & Computer Engineering Department at Virginia Tech. He provided crucial computing resources that proved pivotal for my experimental work and subsequent analysis. Additionally, he imparted invaluable knowledge on system administration and computer programming. His teachings broadened my horizons, introducing me to a multitude of perspectives and nuances in the field of computer engineering. My time at the Wireless@VT lab was highly rewarding through my colleagues' consistent technical guidance and feedback. Shashank Jere frequently guided me in approaching the thesis topic with his valuable insights. Amit Dutta and Pratheek Upadhyaya deserve special mention for their directions and support. Furthermore, I wish to acknowledge my colleagues and friends at Virginia Tech, Ayush Chaturvedi and Shano Ezzell, who made my graduate school experience memorable. Their camaraderie played a significant role in my academic journey. Lastly, and most profoundly, I extend my gratitude to my father. His unwavering love, unyielding support, and continuous encouragement have been the bedrock of my endeavors.

Contents

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
AMC	Automatic Modulation Classification
AMR	Automatic Modulation Recognition
ANN	Artificial Neural Network
BRNN	Bidirectional Recurrent Neural Network
CCI	Commonwealth Cyber Initiative
CPU	Central Processing Unit
CR	Cognitive Radio
DL	Deep Learning
DNN	Deep Neural Network
FFT	Fast Fourier Transformation
GELU	Gaussian Error Linear Unit
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
IoT	Internet-of-Things

IQ	In-Phase and Quadrature
LSTM	Long Short-Term Memory
MIMO	Multiple-Input and Multiple-Output
ML	Machine Learning
NLP	Natural Language Processing
OFDM	Orthogonal Frequency-Division Multiplexing
ReLU	Rectified Linear Unit
RF	Radio Frequency
RNN	Recurrent Neural Network
SNR	Signal-to-Noise Ratio
UE	User Equipment
VGG	Visual Geometry Group
ViT	Vision Transformer

Chapter 1

Introduction

Wireless communication technologies, supported by sophisticated hardware innovations and advanced software capabilities, are central to the seamless connectivity and efficient exchange of information in our interconnected world. With the advent of Internet-of-Things (IoT), small-scale hardware devices have been optimized for multi-functionality, enabling improved machine intelligence applications. Concurrently, deep learning architectures have been studied to enhance automatic modulation classification (AMC) in wireless system receivers. Utilizing state-of-the-art neural networks, these architectures facilitate the identification and classification of diverse modulation schemes, improving system adaptability and reliability in variable wireless communication environments. This approach supports ongoing efforts to optimize and secure wireless communication technologies amid inherent stochastic challenges in a more holistic and adaptable learning approach, thereby reducing reliance on traditional professional solutions that depend on manual extraction of expert features, specialized to operate optimally only within specific constraints.

1.1 Overview

Wireless communication systems are complex systems marked by a mix of hardware, such as transmitters, receivers, and antennae, and software algorithms responsible for channel allocation, congestion control, and security protocols. The stochastic behavior of wireless networks emanating from signal fading, noise, path loss, interference, weather effects, and human-induced fluctuations demands that we develop theoretical and empirical frameworks that can encapsulate such unpredictability while optimizing security and reliability.

The advent of 5G technologies has heralded a new era of wireless communication, marked by unprecedented advancements in hardware infrastructure and software capabilities. These enhancements are aimed at fulfilling the diverse requirements of modern applications, ranging from enhanced mobile broadband (eMBB), massive machine-type-communication (mMTC) [4], and ultra-reliable and low-latency communications (URLLC) [5]. 5G has brought about around 10 to 100 times faster communication speeds when compared to 4G [6].

5G hardware infrastructure includes advanced antenna systems with massive MIMO, small cells, and mmWave bands. 5G software capabilities introduce network slicing, edge computing, software-defined networking (SDN), and Network Function Virtualization (NFV). Integrated advancements in 5G culminate in enhanced security, low latency for autonomous vehicles, and exponential growth in IoT devices.

The introduction of gNodeB (gNB) is instrumental in realizing the enhanced capabilities promised by 5G technologies, such as higher data rates, reduced latency, and increased connectivity. It is responsible for radio resource management, User Equipment (UE) connection (such as smartphones and IoT devices), signal processing (modulation, demodulation, encoding, decoding, transmission, and reception), beamforming, mobility management, and security. The gNBs are predominantly perceived as static infrastructural entities in cellular networks. However, leveraging advancements in low-power, high-performance computing, there's a flourishing research interest in realizing mobile and portable gNBs. This innovation aims to facilitate dynamic network deployments, catering to evolving connectivity demands.

In scenarios involving dynamic spectrum sensing, RF signal surveillance, and low SNR transmissions in highly noisy environments, the radio receivers in gNBs tend to be less effective. Solutions based on artificial intelligence (AI) hold tremendous potential to offer enhanced precision and adaptability, revolutionizing automatic modulation classification.

The convergence of sophisticated hardware technologies and innovative software approaches in 5G has facilitated a transformative communication ecosystem. Thus, 5G is a pivotal milestone in the evolving landscape of wireless communication technologies, propelling society closer to a seamlessly interconnected future.

1.2 Motivation

The advancement of Cognitive Radios (CRs) within the domain of Software Defined Radios (SDRs) has been notably reinforced by the accessibility of cost-effective computational resources. Pivotal progress in the Internet of Things (IoT), culminating seamlessly in the development of smart cities, mandates the evolution of nuanced and refined software proficiencies that reduce the excessive dependency on hardware. This pivotal shift fosters profound modularity, allowing software components to function with significant autonomy from hardware limitations, facilitating a streamlined pathway for upgrades and maintenance activities.

Radio infrastructures are transforming significantly, shifting from a predominantly hardware-centric focus towards a more flexible, software-oriented approach. This evolution is epitomized by the advent of cognitive radios, which embody a rich array of capabilities essential for spearheading advancements in wireless communication technologies. In this context, our research diligently explores the development and optimization of portable test nodes. These nodes are intricately equipped with gNodeB (gNB) functionalities, meticulously designed to enhance 5G communications utilizing energy-efficient hardware solutions. Such pioneering innovations herald a broad spectrum of impactful applications, extending their influence across diverse domains ranging from specialized military operations to various civilian utilities and services.

Our discussion navigates the fundamental development of 5G portable test-nodes, highlights their foundational architecture, and evaluates preliminary test results. An enriched software capability landscape, augmented by the strategic integration of artificial intelligence within cognitive radios, necessitates a concerted exploration of innovative and resilient models tailored for RF signal modulation classification, amongst other emergent use cases. Thus, our inquiry delves profoundly into this vibrant research frontier, scrutinizing the performance metrics and potentialities of transformer-based architectural paradigms in cognitive radio environments.

The transformer architecture, initiated by the Google Brain team in 2017 [1], heralded a paradigm shift in processing sequences by leveraging a ”self-attention” mechanism, enabling each element in a sequence to interconnect with every other element comprehensively in that sequence. This innovation imbued the model with a holistic dataset visibility during the training phase. The advent of transformers significantly revolutionized the field of Natural Language Processing (NLP), surpassing previous standards set by state-of-the-art neural network architectures such as RNNs, LSTMs, and GRUs, and exhibiting extraordinary performance in various NLP tasks. Further extrapolation of this architecture into the field of image processing, as manifested in Vision Transformers [7], has also yielded remarkable results.

Navigating the intricate landscape of artificial intelligence, a crucial challenge is achieving a refined generalization ability beyond simple perceptual limits, capturing the core essence of human intelligence. Achieving such a level of cognitive agility necessitates the incorporation of a substantial relational inductive bias [8] to approximate human-like intellectual faculties.

Transformers exemplify a compelling architectural foundation characterized by universality and adaptability to diverse data modalities, positioning themselves as an ideal candidate for undertaking multi-modal tasks [9].

This thesis is motivated by the potential of transformers in advancing neural network architectures, with a specific emphasis on examining their suitability and performance within the context of automatic modulation classification of RF signals, particularly under the challenging conditions of low signal-to-noise ratio (SNR) scenarios.

1.3 Objectives

The primary objectives of this research are outlined below:

- **Development of 5G Portable Test-Nodes (PORT5+):** Explore and design portable test-nodes, focusing on integrating 5G communication capabilities to adapt and function effectively in various use case scenarios, and discuss preliminary test results.

- **RF Signal Modulation Classification:** Design a deep learning model using transformer-based architecture for automatic modulation classification of RF signals.

- **Evaluation in Low SNR Scenarios:** Evaluate the performance of the transformer-based model for low SNR RF signal modulation classification.

1.4 Thesis Organization

This thesis is meticulously structured to facilitate a coherent and progressive unfolding of the research journey undertaken in developing 5G portable test-nodes, distinctly tailored to a specialized use case: automatic modulation classification.

Chapter 2 offers a concise yet informative exploration of the ongoing efforts and advancements in developing 5G portable test-nodes. This chapter, while encompassing essential details, briefly highlights the evolving developments and delineates the supportive role of these test-nodes in facilitating AI-related data collection and research.

In Chapter 3, a comprehensive literature review is conducted, exploring a multitude of deep learning frameworks and models historically employed in this domain. The analysis reveals a noticeable gap in the implementation of transformer encoder architectures for automatic modulation classification, indicating a nascent stage of exploration and application in this specific area.

In Chapter 4, the focus is placed on examining the theoretical background and mathematical foundations of the transformer architecture. We emphasize its proficiency in managing sequence-to-sequence data in machine translation contexts, particularly due to its success in Natural Language Processing (NLP). Thus, we establish a robust foundational understanding, facilitating the extraction and application of relevant insights to enhance the architecture designated for automatic modulation classification.

In Chapter 5, a methodical implementation focuses on two pivotal components crucial for constructing a transformer-based automatic modulation classification model. Initially, the chapter provides a clear analysis of key factors involved in dataset selection, laying down the essential groundwork required for the subsequent model development. The chapter then delves into the implementation strategies essential for actualizing the transformer-based architecture.

In Chapter 6, we examine the various experiments performed on the selected dataset. The chapter focuses on empirical analysis, providing crucial data-driven insights for model evaluation and selection.

In Chapter 7, the conclusions are drawn from the research findings, summarizing key insights and their implications. Furthermore, it outlines prospective directions for future research, identifying areas where further exploration and development could be pursued to enhance the field.

In Appendix A, we present a comparative visualization of a low SNR RF Signal (at -20 dB) and a high SNR RF Signal (at 30 dB) from the RADIOML 2018.01A Dataset [3]. In Appendix B, we present some of the additional mathematical formulae not discussed in Chapter 4, and are crucial for understanding the implementation of the transformer-based architecture. In Appendix C, we present the training curves for all the trained models we evaluate in this thesis. In Appendix D, we present the training curves for additional training for the model performance improvement. In Appendix E, we present the tables with the classification accuracy both with respect to the SNR and the RF Signal modulation types.

Chapter 2

PORT5+ Development

2.1 PORT5+: Introduction

The **5G Portable Test-Nodes (PORT5+)** development initiative is meticulously designed in an incremental yet exhaustive exploration and deployment of 5G communication systems in an edge-computing scenario. Commencing with an elemental configuration and progressively incorporating increased complexities, the project aims to meticulously scrutinize the intricacies of Standalone (SA) 5G New Radio (NR) paradigms within targeted spectral bands of 2.4 GHz and 3.5 GHz. This phased methodology affords a systematic evaluation mechanism, thereby ensuring a multivariate optimization of robustness, scalability, and overall system performance, which is imperative for realizing ubiquitous, high-speed communication networks.

This project is sponsored by the Commonwealth Cyber Initiative (CCI), a distinguished consortium dedicated to propelling advancements in cybersecurity and cyber-physical systems. This prestigious sponsorship signifies the project's alignment with CCI's mission to foster groundbreaking research, thus serving as a pivotal enabler in accelerating scientific discovery and innovation. The research and development activities are supervised by Dr. Nishith Tripathi, Dr. Carl Dietrich, and Dr. Jeff Reed, under the aegis of Wireless@VT.

The research framework of the PORT5+ initiative has a sequence of three configurations, allowing for the incremental introduction of system complexities. These configurations are not merely incremental but also holistic, designed to address various facets of 5G communications including near real-time Radio Access Network Intelligent Controllers (Near-RT RIC) and non-real-time Radio Access Network Intelligent Controllers (Non-RT RIC), and mobility management in a low power portable setup. Each stage of configuration presents a unique set of experimental setups and parameters, thereby facilitating a rigorous empirical evaluation aimed at optimizing both system functionality and operational performance. The PORT5+ Architecture Network Diagram for the final proposed configuration is showcased in Figure 2.1.

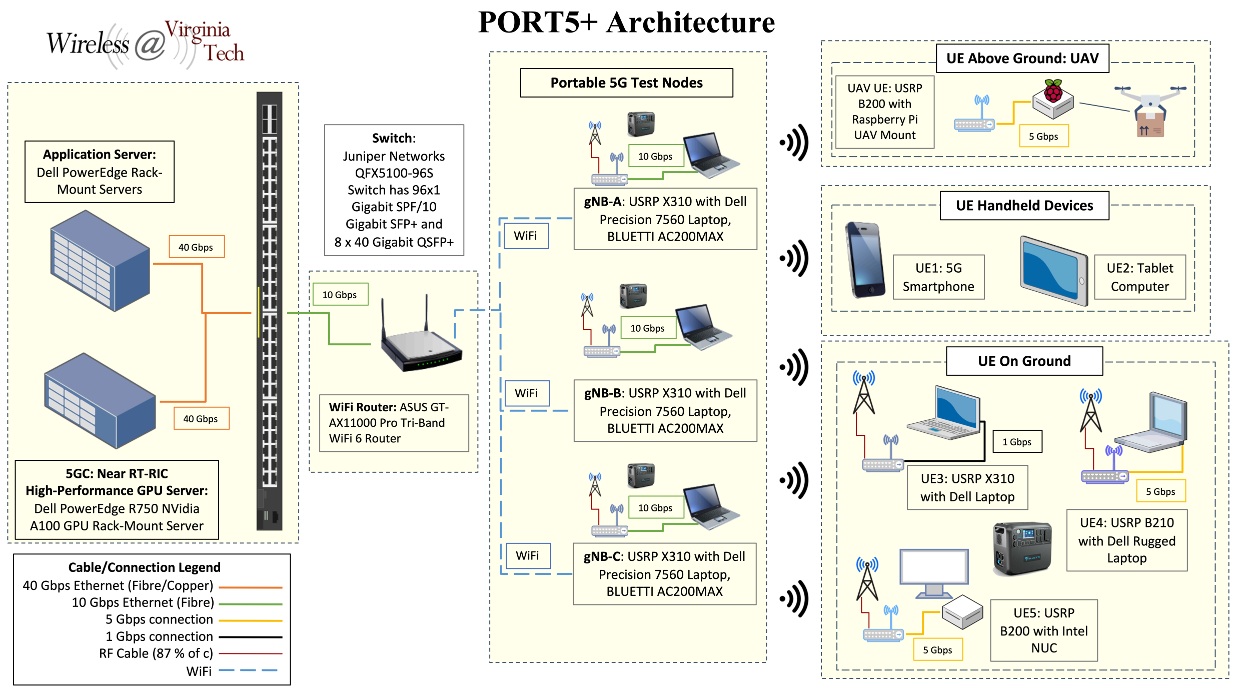

Wireless @ Virginia Tech
PORT5+ Architecture
Application Server: Dell PowerEdge Rack-Mount Servers
40 Gbps
40 Gbps
5GC: Near RT-RIC High-Performance GPU Server: Dell PowerEdge R750 NVidia A100 GPU Rack-Mount Server
Switch: Juniper Networks QFX5100-96S Switch has 96x1 Gigabit SPF/10 Gigabit SFP+ and 8 x 40 Gigabit QSFP+
10 Gbps
WiFi Router: ASUS GT-AX11000 Pro Tri-Band WiFi 6 Router
Portable 5G Test Nodes
10 Gbps
gNB-A: USRP X310 with Dell Precision 7560 Laptop, BLUETTI AC200MAX
WiFi
10 Gbps
gNB-B: USRP X310 with Dell Precision 7560 Laptop, BLUETTI AC200MAX
WiFi
10 Gbps
gNB-C: USRP X310 with Dell Precision 7560 Laptop, BLUETTI AC200MAX
WiFi
UE Above Ground: UAV
UAV UE: USRP B200 with Raspberry Pi UAV Mount
5 Gbps
UE Handheld Devices
UE1: 5G Smartphone
UE2: Tablet Computer
UE On Ground
1 Gbps
UE3: USRP X310 with Dell Laptop
5 Gbps
UE4: USRP B210 with Dell Rugged Laptop
5 Gbps
UE5: USRP B200 with Intel NUC
Cable/Connection Legend
40 Gbps Ethernet (Fibre/Copper)
10 Gbps Ethernet (Fibre)
5 Gbps connection
1 Gbps connection
RF Cable (87 % of c)
WiFi

Figure 2.1: The PORT5+ Architecture Network Diagram

2.2 PORT5+: Objective and Scope

2.2.1 Primary Objective

The project aims to deliver portable test-nodes capable of facilitating SA 5G NR communications with escalating complexities across three configurations.

2.2.2 Technical Scope

- Implementation of Near-RT RIC and Non-RT RIC.
- Utilization of OpenAirInterface (OAI) [10] and other open-source platforms.
- Operability in 2.4 GHz and 3.5 GHz frequency bands.

2.3 PORT5+: Development Methodology

The project employs a sprint-based methodology divided into three key configurations, each an extension of its predecessor. We discuss the common characteristics for all configurations below:

- Standalone 5G New Radio with 5G Core (5GC).
- Near-RT RIC and Non-RT RIC configuration on a high-performance server.
- Target Frequency Range: 2.4 GHz and 3.5 GHz.
- OAI open-source software for Radio Access Network.
- UE-gNB separation for radio far-field operation.

2.3.1 Configuration 1: Preliminary Stage

- Test Environment: Indoor (in the lab).
- High-performance laptop coupled with USRP X310 for gNB (gNB-A).
- A UE comprised of a laptop connected to USRP X310.
- An option to test multiple UEs and note any performance degradation.
- Implementation of Near-RT RIC on a dedicated GPU server.
- UE and gNB-A communicate using 5G NR-based OAI.
- Near-RT RIC and gNB-A to communicate through Wi-Fi.
- 5GC and gNB-A to communicate through WiFi.
- 5GC connects to the internet through VT Wi-Fi.

2.3.2 Configuration 2: Intermediate Stage

- Test Environment: Outdoor.
- All components in Configuration 1 with only one UE.
- Incorporation of two additional gNBs (gNB-B and gNB-C), each using high-performance laptops.
- Near-RT RIC interfacing gNB-A, gNB-B, and gNB-C through Wi-Fi.
- Mobility between gNBs Test Scenarios: handover, RRC reestablishment, cell reselection.

2.3.3 Configuration 3: Final Stage

- Test Environment: Outdoor (on the field).
- All components in Configuration 2.
- Inclusion of diverse UEs, ranging from 5G-enabled smartphones to UAVs outfitted with USRP B210 and Raspberry Pi.
- UEs and gNBs communicate using 5G NR-based OAI.
- Near-RT RIC and Non-RT RIC interface with gNBs through Wi-Fi.

2.4 PORT5+: Implementation & Test

Pratheek Upadhyaya, a graduate student researcher at Wireless@VT, implemented Configuration 1, and conducted preliminary tests. The implementation & results from these evaluations are discussed in the following subsections:

2.4.1 Configuration 1: Implementation

PORT5+: Hardware Implementation

USRPs sourced from Ettus Research (parent company: National Instruments) are used for gNBs and UEs (for all configurations). The USRPs tested are B210, X310, N310. It was observed that X310 was unstable for long periods of transmission.

PORT5+: Software Implementation

5G SA Network utilizes open source 5G software stack: RAN (v2023.w27) & CN (v1.5.1) from OpenAirInterface (OAI).

PORT5+: Implementation Parameters

The parameters listed below are consistently applied across all deployment scenarios.

- **Core Network Parameters** are presented in Table 2.1.

Core Network Parameter	**Value/Details**
MCC	001
MNC	01
DNN	oai

Table 2.1: Core Network Parameters

- **gNB Parameters** are presented in Table 2.2.

gNB Parameter	**Value/Details**
Mode of Operation	Time Division Duplexing (TDD).
Band	41
Frequency of Operation	2.574 GHz
SCS	30 KHz
Number of PRBs	106 (40 MHz channel bandwidth)
Number of data/transmission layers	1 (1x1 configuration)

Table 2.2: gNB Parameters

2.4.2 Configuration 1: Test Results

Both the gNB and UE are based on USRPs, and interface with the OAI gNB/UE stack, respectively. Initially, a single gNodeB and single UE setup were tested. The observations are detailed in Table 2.3.

Observation	**Value/Details**
gNodeB detection range	12 meters radius approx.
DL/UL Throughput (UDP Traffic)	Avg. 60/12 Mbps (DL/UL)
Peak DL Throughput (UDP Traffic)	77 Mbps
DL/UL Throughput (TCP Traffic)	Avg. 30/8 Mbps (DL/UL)
Latency (UDP Traffic)	12 ms
Latency (TCP Traffic)	30 ms

Table 2.3: Observations: Single gNB & Single UE

Additionally, a single gNB and multiple UEs setup was tested. It was noted that the UE throughput diminishes when the gNB processes requests from multiple UEs. This performance issue could potentially be addressed by increasing the number of CPU threads utilized by the gNB. The observations are presented in Table 2.4.

Observation	**Value/Details**
UE 1: DL Throughput (UDP Traffic)	30 Mbps
UE 1: UL Throughput (UDP Traffic)	6 Mbps
UE 2: DL Throughput (UDP Traffic)	18 Mbps
UE 2: UL Throughput (UDP Traffic)	4 Mbps

Table 2.4: Observations: Single gNB & Multiple UEs

2.5 PORT5+: Future Work

Currently, PORT5+ is undergoing development, and initial tests have shown promising results. While work continues, there remains a need to evaluate Configuration 2, and Configuration 3. Future experiments might include exploring MIMO configurations, testing handover processes for UE mobility, and fine-tuning hardware parameters & network parameters for increased range & enhanced throughput.

2.6 PORT5+: Applications in AI

PORT5+ embodies an inventive conceptualization, amalgamating Portable Test-Nodes with advanced 5G capabilities. It is currently evolving through the nascent stages of research, residing at the intersection of communication technologies and artificial intelligence. This novel convergence of ideas holds substantial promise in the expansive field of artificial intelligence. Portable Test-Nodes can function as powerful hardware-oriented agents, catering to two specialized use cases: (i) Online Learning, and (ii) Federated Learning.

(i) Online Learning: In fluctuating environmental conditions and variable geo-locations of portable Test-Nodes, a robust online learning paradigm surfaces. This adaptive architecture enables the core AI infrastructure to embody flexibility and responsiveness, adeptly navigating rapid shifts and changes. Integrated seamlessly with centralized servers, this system is instrumental in the enhancement and fine-tuning of AI applications. The portable test-nodes play a crucial role as information relays, facilitating essential training attributes effectively. This adaptable structure empowers a centralized AI system with the necessary agility and adaptability. As a result, it promotes the generation of inferences intricately tuned to the prevailing conditions, ensuring optimal accuracy and relevance in real-time scenarios.

(ii) Federated Learning: Portable Test-Nodes are equipped with compute and memory of small-scale servers, enabling them to proficiently process RF signals received at remote locations. This allows for meticulously examining the data accrued and the environmental-specific noise at the locality where the test-node is implemented, facilitating a decentralized approach. This methodology circumvents the need for data transfer to a centralized location, thereby preserving data privacy and integrity. They empower learning and research without ceding control of vital data to entities beyond jurisdictional boundaries.

Chapter 3

Review of Literature

Peng et al. [11] conducted an extensive review on utilizing Deep Learning (DL) to classify RF signal modulation types. The survey explores various methods implemented for signal representation and preprocessing, which are some of the essential steps for effective modulation classification. The authors delve into different techniques in which signals are represented and preprocessed before classification, providing an in-depth understanding of the application of DL in this domain.

In the domain of sequence representation, given that signal samples are received sequentially, Peng et al. [11] have categorized RF signal representation for Deep Learning (DL) into four broad groups: (i) *In-Phase and Quadrature Sequences*, (ii) *Amplitude and Phase Sequences*, (iii) *Fast Fourier Transformation Sequences*, and (iv) *Amplitude Histogram Sequences.*

Various research endeavors have advanced the field significantly using in-phase and quadrature (IQ) sequences. Initially proposed by O'Shea et al. [12], the use of IQ sequences has enabled the identification of signals across numerous digital and analog modulation schemes by segmenting them through a 128-point rectangular window (128-point in-phase and 128-point quadrature sequence) and some of the raw samples from this dataset are illustrated in [13]. Subsequent research, such as [2], has expanded and refined these datasets by contemplating more modulation schemes and conducting analysis with CNN, VGG [14] and ResNet [15], which has shown promising results and have facilitated effective modulation classification, particularly when ample training data is accessible. Huang et al. [16] have applied three different data-augmentation techniques on the dataset in [12] to study the performance of CNN, ResNet, DenseNet [17], and LSTM [18, 19]. Liu et al. [20] evaluates the performance of Convolutional Long Short-Term Deep Neural Network (CLDNN) [21] on the same dataset [12]. Zhang et al. [22] perform a comparison with pre-processed signal data on CNN and LSTM. Ma et al. [23] implements a hybrid model composed of CNN and LSTM to make the model lightweight. Ke et al. [24, 25] have implemented an LSTM-based denoising autoencoder to propose a framework that reconstructs the signal from noisy received signals for classification.

Hermawan et al. [26] propose a CNN-based architecture to comply with the latency requirements for beyond-5G communications. Yashashwi et al. [27] introduce a distortion correction module to improve the accuracy of CNN-based modulation recognition. DL methods are evaluated for signal modulation classification in OFDM systems by [28, 29, 30]. Shi et al. [31] propose a DL method to eliminate the problem of phase offset (PO) in automatic modulation recognition (AMR). In [32], Ali et al. have used an autoencoder-based DNN on MATLAB simulated signals. Meng et al. [33] have exemplified the benefit of parallel computation to speed up the inference process through DL methods for automatic modulation classification (AMC). Zheng et al. [34] have evaluated voting-based, confidence-based, and feature-based fusion techniques to solve the problem of varying signal length. A deep learning model generalized for channel identification is implemented in [35]. Under uncertain noise conditions, deep learning methods (DNN and RNN) are studied and evaluated in [36, 37]. Wang et al. [38] apply DL-based AMC techniques for MIMO systems.

Another method of characterizing received signals is by Amplitude and Phase Sequences. Rajendran et al. [39] explore the method of implementing an LSTM model on amplitude and phase sequences, which are generated from the dataset in [12] with modification by varying samples per symbol and sample length parameters. Rajendran et al. [40] apply the

Amplitude and Phase Sequences representation for processing modulation classification in their proposed spectrum monitoring initiative.

Fast Fourier Transformation (FFT) Sequences are created by mapping time-domain IQ signal to frequency-domain data, resulting in two real-valued sequences. Mossad et al. [41] compared the performance of IQ representation with the FFT representation of the dataset in [42].

Amplitude Histogram Sequences are constructed by counting signal data in intervals by dividing the entire range of data. Distinct signatures are obtained through this method and are explored in [43, 44]. Zhao et al. [45] implement a popular machine-learning technique of random forests on features from signal Amplitude Histograms and compare the computational complexities with DL-based methods.

Peng et al. [11] further discuss the combined representation of received signals in four categories: (a) *Features and Sequences*, (b) *Image and Sequences*, (c) *Multiple Images*, and (d) *Multiple Sequences*. Wang et al. [46] use CNN to classify IQ representation and constellation diagrams to identify modulation for low SNR signals. Wu et al. [47] apply CNN-based AMC on two image representations of signals: cyclic spectra and constellation diagrams, achieving identical or even better results with reduced learning parameters and training time compared to state-of-the-art DL-based methods.

The authors in [48] have proposed a model based on ResNeXt [49] to classify signals with low SNRs of -20 dB to 0 dB with higher accuracy on the dataset in [12, 42]. Zhange et al. [50] apply feature fusion techniques by transforming signals to two-dimensional images by Smoothed Pseudo Wigner-Ville Distribution (SPWVD) and Born-Jordan Distribution (BJD), then extract features using a CNN-based model, and then by fusing the image features and hand-crafted features and utilizing a multi-modality model, get high-performance accuracy at low SNR.

Li et al. [51] have applied transformers [1] to the dataset in [12] and present their experimental results to show the efficacy of this DL architecture. They use a popular variant of transformers, the Vision Transformer (ViT) proposed by Dosovitskiy et al. [7]. They observe an accuracy improvement from pure LSTM and CLDNN for high SNR signals.

Liang et al. [52] have applied the attention mechanism on ResNeXt [49] on a dataset in [12] and have the robustness of the model and higher modulation classification accuracy.

Sathyanarayanan et al. [53] have used SNR partitioning to classify signal modulation and have observed considerable improvement, although not at low SNR levels, with CNN architecture. Wu et al. [54] have applied attention to their deep learning method to solve intra-class confusion during modulation classification.

Zhang et al. [55] have implemented a lightweight DL model with considerably fewer parameters to achieve high recognition. Chen et al. [56] propose a multi-scale CNN to improve classification accuracy by extracting multi-scale feature maps for low computation complexity.

Transformer-based architectures with gating mechanisms have been studied in different fields with very promising results and can be referred to in [57, 58]. Transformer-based architecture has also been used by researchers at Wireless@VT lab for jamming detection and classification as presented by Jere et al. [59].

Chapter 4

Transformers: Theoretical Background

Transformers garnered particular attention after Vaswani et al. [1] published their influential paper ”Attention Is All You Need”, which focused on developing a sequence-to-sequence model for Natural Language Processing (NLP) for the purpose of machine translation. The dominant architecture before transformers were RNNs, LSTMs, and GRUs.

RNNs were an advancement toward the modeling and analysis of sequential or time series data over traditional feed-forward networks for the purpose of learning and prediction tasks. RNNs are a type of conventional feed-forward artificial neural networks (ANNs) capable of handling sequential data by storing historical information to understand the dependency between the data points through the concept of *memory* [60]. The various types of RNNs that are applied to machine learning problems are Bidirectional Recurrent Neural Networks (BRNN), Gated Recurrent Units (GRU), and Long Short-Term Memory (LSTM). BRNNs accept inputs from future time steps to enhance network accuracy. GRUs handle the vanishing gradient problem with reset and update gates to find relevant information to be stored for future predictions. LSTMs also addressed the vanishing gradient problem with input, output, and forget gates to determine relevant information to be stored.

Vaswani et al. [1] proposed a transformer architecture with the attention mechanism that was groundbreaking research in Natural Language Processing on text sequences for machine translation. The architecture is illustrated in Figure 4.1.

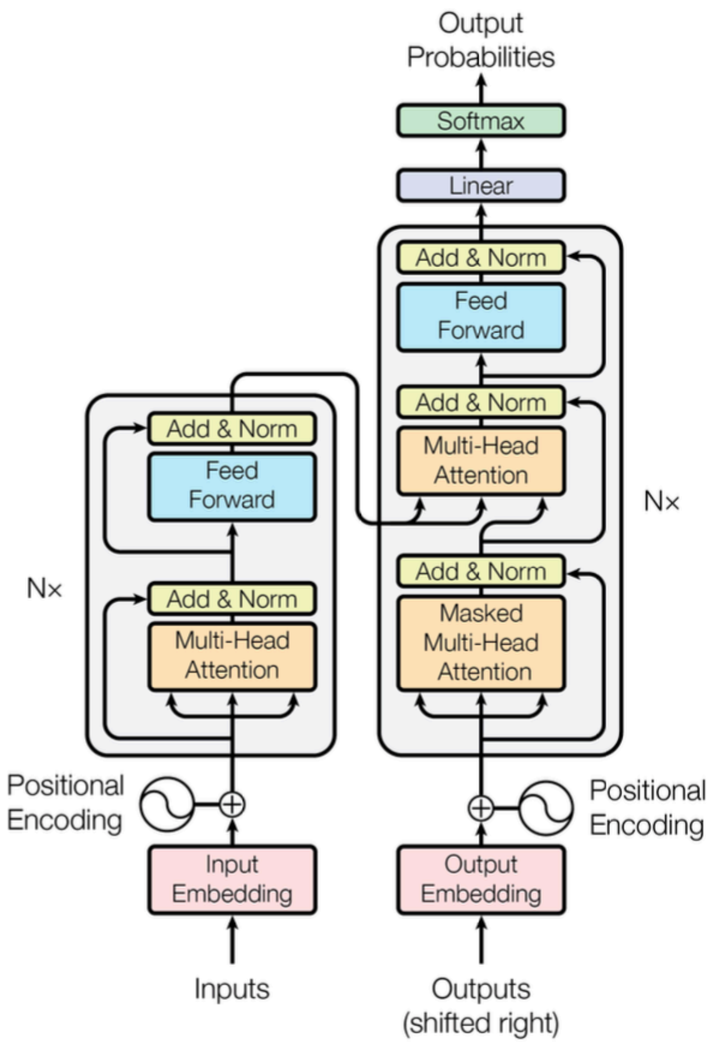


Figure 4.1: Transformer Architecture by Vaswani et al. [1]

Before we delve into the details of the transformer architecture proposed by Vaswani et al. [1], it is imperative to look back at the history of attention and its relevance to the field of machine learning and artificial intelligence, particularly in encoder-decoder architecture. The encoder-decoder architecture is often applied to sequence-to-sequence tasks, which is commonplace in language processing. An attention-based mechanism solves the problem of the decoder missing information from input sequences of varying lengths, resulting in a fixed-length vector.

The study of ”attention” has its roots in psychology. A recent study by Lindsay (2020) [61] aptly connects the concept of ”Attention” in psychology to machine learning. The author states that artificial attention is gaining popularity in artificial neural networks. Particularly, the connection between encoder and decoder models in artificial neural networks require a sort of attention mechanism and is often employed in state-of-the-art neural network paradigms for sequence modeling and prediction.

Bahdanau et al. (2014) [62] identified a problem that when a variable-length input sequence of words is input to generate a fixed-length vector, it affects the information from the source, which would cascade down to the reduced performance of the model with increasing input length. They suggested an additive attention mechanism to generate an attention score by searching for the positions in the source sentence for information in terms of relevance at the encoder. The generated context vector is fed to the decoder, and as a result, there was significant performance improvement in their model, irrespective of the sentence length.

Luong et al. (2015) [63] suggested improvements over the model proposed by Bahdanau et al. [62] with the introduction of global and local attention for a sequence of words in a sentence. The global approach attends to all the words in the source sentence, and the local approach attends to a subset of a few selected words to predict the outcome sentence. The ”global attention” used the idea of multiplicative attention instead of additive attention in the model proposed by Bahdanau et al. [62].

Vaswani et al. [1] define the attention function as a mapping of query and key-value pairs to an input, all of which are vectors. The output is a weighted sum of values, and the weight assigned is computed using the query and the corresponding key.

The attention mechanism in the transformer architecture proposed by Vaswani et al. [1] is

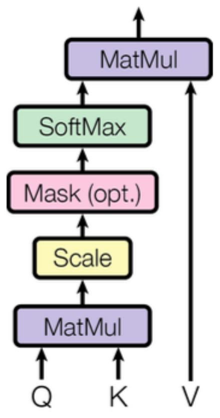


Figure 4.2: Scaled Dot-Product Attention by Vaswani et al. [1]

the **Scaled Dot-Product Attention**, as illustrated in Figure 4.2, and can be represented as follows for matrices $\mathbf{Q}$ for queries, $\mathbf{K}$ for keys, and $\mathbf{V}$ for values:

$$\text{attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right) \cdot \mathbf{V} \tag{4.1}$$

The following is the step-by-step computation of **Scaled Dot-Product Attention** [60]:

1. The alignment scores are computed by the multiplication of matrix $\mathbf{Q}$ (size of $m \times d_k$) and matrix $\mathbf{K}$ (size of $n \times d_k$), where d_k is the dimension of vectors $\mathbf{q}$ and $\mathbf{k}$ to generate a resultant matrix (size of $m \times n$):

$$\mathbf{Q}\mathbf{K}^T = \begin{bmatrix} e_{11} & e_{12} & \cdots & e_{1n} \\ e_{21} & e_{22} & \cdots & e_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ e_{m1} & e_{m2} & \cdots & e_{mn} \end{bmatrix} \tag{4.2}$$

2. The alignment scores are scaled by $\frac{1}{\sqrt{d_k}}$ to reduce the effect of the dot product magni-

tude growing considerably large:

$$\frac{\mathbf{QK}^T}{\sqrt{d_k}} = \begin{bmatrix} \frac{e_{11}}{\sqrt{d_k}} & \frac{e_{12}}{\sqrt{d_k}} & \cdots & \frac{e_{1n}}{\sqrt{d_k}} \\ \frac{e_{21}}{\sqrt{d_k}} & \frac{e_{22}}{\sqrt{d_k}} & \cdots & \frac{e_{2n}}{\sqrt{d_k}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{e_{m1}}{\sqrt{d_k}} & \frac{e_{m2}}{\sqrt{d_k}} & \cdots & \frac{e_{mn}}{\sqrt{d_k}} \end{bmatrix} \tag{4.3}$$

3. The softmax function is applied to compute the weights:

$$\text{softmax}(\frac{\mathbf{QK}^T}{\sqrt{d_k}}) = \begin{bmatrix} \text{softmax}(\frac{e_{11}}{\sqrt{d_k}} & \frac{e_{12}}{\sqrt{d_k}} & \cdots & \frac{e_{1n}}{\sqrt{d_k}}) \\ \text{softmax}(\frac{e_{21}}{\sqrt{d_k}} & \frac{e_{22}}{\sqrt{d_k}} & \cdots & \frac{e_{2n}}{\sqrt{d_k}}) \\ \vdots & \vdots & \ddots & \vdots \\ \text{softmax}(\frac{e_{m1}}{\sqrt{d_k}} & \frac{e_{m2}}{\sqrt{d_k}} & \cdots & \frac{e_{mn}}{\sqrt{d_k}}) \end{bmatrix} \tag{4.4}$$

4. In the last step, the computed weights are applied to the matrix $\mathbf{V}$ (size of $n \times d_v$)

$$\text{softmax}(\frac{\mathbf{QK}^T}{\sqrt{d_k}}) \cdot \mathbf{V} = \begin{bmatrix} \text{softmax}(\frac{e_{11}}{\sqrt{d_k}} & \frac{e_{12}}{\sqrt{d_k}} & \cdots & \frac{e_{1n}}{\sqrt{d_k}}) \\ \text{softmax}(\frac{e_{21}}{\sqrt{d_k}} & \frac{e_{22}}{\sqrt{d_k}} & \cdots & \frac{e_{2n}}{\sqrt{d_k}}) \\ \vdots & \vdots & \ddots & \vdots \\ \text{softmax}(\frac{e_{m1}}{\sqrt{d_k}} & \frac{e_{m2}}{\sqrt{d_k}} & \cdots & \frac{e_{mn}}{\sqrt{d_k}}) \end{bmatrix} \cdot \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1d} \\ v_{21} & v_{22} & \cdots & v_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ v_{m1} & v_{m2} & \cdots & v_{md} \end{bmatrix} \tag{4.5}$$

Vaswani et al. [1] propose a **Multi-Head Attention Mechanism** as illustrated in Figure 4.3, as an advancement to **Scaled Dot-Product Attention**. This mechanism projects $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ with different learned projection h times. The final result is generated by concatenating the h outputs. This helps to gather information from different subspaces of the data representation, which would otherwise not be possible with a single attention head.

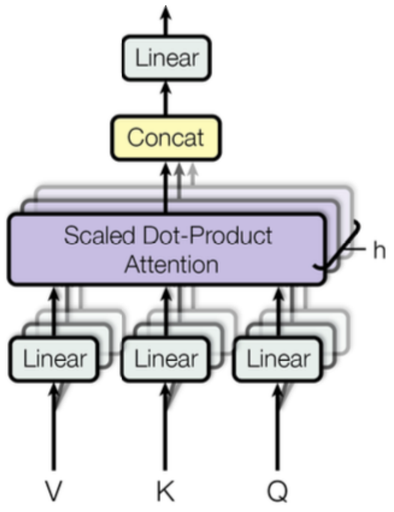


Figure 4.3: Multi-Head Attention Mechanism by Vaswani et al. [1]

The mathematical representation is as follows:

$$\text{multihead}(Q, K, V) = \text{concat}(\text{head}_1, \ldots, \text{head}_h)W^O \tag{4.6}$$

The following shows the representation of a single attention function for each $\text{head}_i, i = 1, \cdots, h$:

$$\text{head}_i = \text{attention}(QW_i{}^Q, KW_i{}^K, VW_i{}^V) \tag{4.7}$$

The following is the step-by-step computation of **Multi-Head Attention** [60]:

1. For each attention head_i, compute the linear projections of the queries, keys, and values by multiplying them with their corresponding weight matrices $W_i{}^Q, W_i{}^K, \text{and} W_i{}^V$ respectively.

2. For each head_i, execute the attention function by: (a) performing matrix multiplication between the projected queries and keys, (b) scaling the resultant matrix and applying the softmax function to normalize the values, and (c) multiplying the normalized

matrix with the projected values to produce the output for each attention head_i.

3. Perform concatenation of the outputs generated by each attention $\text{head}_i, i = 1, \cdots, h$.
4. Perform multiplication of the concatenated output by the weight matrix W^O to linearly project and generate the final result.

Vaswani et al. [1] employ multi-head attention in three distinct configurations:

1. **Encoder-Decoder Attention:** Queries from the prior decoder layer interact with memory keys and values from the encoder output, facilitating attention across all input sequence positions. This mechanism aligns with conventional encoder-decoder attention structures observed in sequence-to-sequence models.
2. **Encoder Self-Attention:** Each encoder position attentively relates to all positions in its preceding layer, with queries, keys, and values all emanating from the previous encoder layer's output.
3. **Decoder Self-Attention:** Implementing attention to all positions inclusively to the current one in the decoder while meticulously preserving the auto-regressive property by inhibiting leftward information flow, a technique actualized within the scaled dot-product attention framework by adeptly masking unacceptable connections during softmax computation.

The position-wise feed-forward networks within each encoder and decoder layer, in addition to attention sub-layers, consist of two linear transformations separated by a Rectified Linear Unit (ReLU) activation and are applied identically yet independently to each position:

$$FFN(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2 \tag{4.8}$$

The ReLU [64, 65] activation is mathematically represented as:

$$f(x) = \max(0, x) \tag{4.9}$$

The transformer encoder layers apply similar transformations to all the elements of the input sequence, but each layer employs different weights and bias values. Each sub-layer has a residual connection and is succeeded by a normalization layer, $LayerNorm(.)$, which normalizes the sum computed between input x and the output of the sub-layer:

$$LayerNorm(x + SubLayer(x)) \tag{4.10}$$

Layer Normalization [66] is a statistical technique applied in deep learning to enhance the training stability and convergence of neural networks. It operates by standardizing the activations within each network layer independently, making the data more amenable to training. This process involves subtracting the mean and dividing by the standard deviation of the activations, effectively normalizing them. Layer Normalization contributes to the overall effectiveness of deep learning models, particularly in scenarios with complex data distributions, by ensuring that gradients flow more consistently during training. The layer normalization statistics over all hidden units are computed as follows:

$$\mu^l = \frac{1}{H} \sum_{i=1}^{H} a_i^l \tag{4.11}$$

$$\sigma^l = \sqrt{\frac{1}{H} \sum_{i=1}^{H} (a_i^l - \mu^l)^2} \tag{4.12}$$

In Equations 4.11 and 4.12, H denotes the number of hidden units. All hidden units share the same normalization terms μ^l & σ^l, although these terms differ across different training cases.

The transformer model utilizes learned embeddings to convert input and output tokens into vectors of dimension d_{model}, with shared weights between embedding layers and pre-softmax linear transformations. To leverage sequence order due to the absence of recurrence and convolution in the model, positional encodings, summed with input embeddings, are incorporated. The sinusoidal functions chosen for positional encoding facilitate attention to relative positions and allow extrapolation to sequences of varied lengths, yielding results comparable to learned positional embeddings.

The transformer decoder has a similar implementation as the encoder, and interested readers can refer to the literature in [1, 60] for more details.

Chapter 5

Methodology: Transformer-Based Architecture

5.1 Dataset: Evaluation & Analysis

In the field of radio frequency (RF) signal classification, two datasets have garnered considerable attention: (i) RADIOML 2016.04C [12, 67, 68] & RADIOML 2016.10A (much cleaner and more normalized version of RADIOML 2016.04C) [68], and (ii) RADIOML 2018.01A [2, 3]. Each dataset exhibits unique characteristics that make them viable for distinct experimental paradigms discussed in the following sub-sections.

5.1.1 RADIOML 2016.04C & RADIOML 2016.10A: Overview

RADIOML 2016.04C [68] dataset represents a synthetic collection of RF signals, primarily generated through GNU Radio [69], a free software toolkit for building Software-Defined Radios (SDRs) [70]. It encompasses 11 distinct modulation schemes, eight of which are digital, and the remaining three are analog. The digital modulation schemes are BPSK, QPSK, 8PSK, 16QAM, 64QAM, BFSK, CPFSK, and PAM4. The analog modulation schemes are WB-FM, AM-SSB, and AM-DSB. These signals span a range of Signal-to-Noise Ratios (SNRs), providing a versatile but somewhat simplistic playground for machine learning algorithms. The transmit power is normalized at an average value of 0 dB and the data is modulated at 8 samples per symbol [12]. RADIOML 2016.10A [68] dataset is a cleaner and normalized version of RADIOML 2016.04C dataset and supersedes the latter.

5.1.2 RADIOML 2018.01A: Overview

The RADIOML 2018.01A [2] Dataset serves as a more comprehensive corpus. It includes 24 modulation types, both digital and analog. The digital modulation types are OOK, 4ASK, 8ASK, BPSK, QPSK, 8PSK, 16PSK, 32PSK, 16APSK, 32APSK, 64APSK, 128APSK, 16QAM, 32QAM, 64QAM, 128QAM, 256QAM, GMSK, OQPSK. The analog modulation types are AM-SSB-WC, AM-SSB-SC, AM-DSB-WC, AM-DSB-SC, and FM. The RF signal samples also incorporate simulated channel effects. The reliability and robustness of this dataset have been corroborated through extensive validation processes, which adds an additional layer of complexity and realism compared to its 2016 predecessor. The authors have claimed it to be a challenging dataset [3].

5.1.3 Justification for Dataset Selection

For the current experimental work in this thesis, we have opted for RADIOML 2018.01A [2] Dataset. This dataset offers a challenging yet versatile environment, comprising more than 2.5 million labeled signal instances. Transformers are known not to generalize well on insufficient amounts of data [7], one of the primary factors for selecting this considerably large dataset. Also, there are 24 classes of RF signal modulation, which would provide a wide variety of classes for evaluation. Each signal instance has two orthogonally related channels: In-Phase(I) and Quadrature (Q), each with 1024 features. The dataset's inherent diversity in terms of SNR levels and modulation types renders it a quintessential choice for validating the efficacy of our proposed transformer-based architecture. The subsequent segments will provide a meticulous dissection of the dataset's structure, including but not limited to feature distribution, class imbalances, if any, and underlying statistical properties.

5.1.4 RADIOML 2018.01A: Dataset Analysis

The RADIOML 2018.01A dataset encompasses a total of 2,555,904 individual samples. Each respective sample within this dataset manifests as a 2-dimensional array with a shape of (1024, 2). This configuration implies that every sample is characterized by 2048 distinctive features derived from the 1024 pairs of values contained within the array. Intricately, these pairs represent In-Phase and Quadrature (IQ) data points fundamental to the telecommunications field for modulating and demodulating signals. Each pair of IQ data effectively forms a complex number or a point in the complex plane, thereby encapsulating the amplitude and phase information of the signal. These features collectively encapsulate the pertinent information and attributes associated with each sample in the dataset, providing a comprehensive foundation for conducting further analysis and research using this extensive collection of data.

This dataset is characterized by a multitude of modulation classes, each accompanied by signal-to-noise ratio (SNR) values extending from -20 dB to +30 dB, incrementing in consistent steps of 2 dB. This configuration yields 26 distinctive SNR values, representing the various conditions under which the communication signals are typically received. In this structured dataset, each individual SNR value is represented across a substantial sample size of 98,304, thereby providing a robust and representative substrate for the analysis and evaluation of signal modulations under diverse and specified noise conditions. The distribution of samples corresponding to each SNR value within this dataset is tabulated in the accompanying Table 5.1, facilitating an insightful and systematic exploration of the dataset's inherent characteristics and structural composition.

Table 5.1: RADIOML 2018.01A [2] Dataset: No. of Samples for each SNR Value

Serial No.	SNR Value	No. of Samples	Serial No.	SNR Value	No. of Samples
1.	-20 dB	98304	14.	6 dB	98304
2.	-18 dB	98304	15.	8 dB	98304
3.	-16 dB	98304	16.	10 dB	98304
4.	-14 dB	98304	17.	12 dB	98304
5.	-12 dB	98304	18.	14 dB	98304
6.	-10 dB	98304	19.	16 dB	98304
7.	-8 dB	98304	20.	18 dB	98304
8.	-6 dB	98304	21.	20 dB	98304
9.	-4 dB	98304	22.	22 dB	98304
10.	-2 dB	98304	23.	24 dB	98304
11.	0 dB	98304	24.	26 dB	98304
12.	2 dB	98304	25.	28 dB	98304
13.	4 dB	98304	26.	30 dB	98304

Figure 5.6 offers a concise yet comprehensive visual representation, illustrating the distribution of signal-to-noise ratio (SNR) values, each correlated with their specific counts of samples within the dataset. This graphical delineation is crucial, providing immediate insights into the spread and frequency of SNR values and facilitating a clearer understanding of the data's inherent structure and distribution characteristics.

In a complementary manner, Figure 5.2 employs a box plot to provide additional insight into the data distribution. This plot effectively showcases the uniform distribution exhibited by the dataset across the 26 distinct SNR values. Through this visual device, it is evident that the dataset maintains a balanced distribution, with both the mean and median SNR values precisely situated at 5 dB. This symmetry and consistency in distribution not only validate the dataset's uniformity but also highlight its well-structured and evenly distributed nature.

It is pertinent to note that in the analysis of this dataset, the mode, a statistical metric frequently utilized to pinpoint the most recurrent value in a distribution, holds negligible

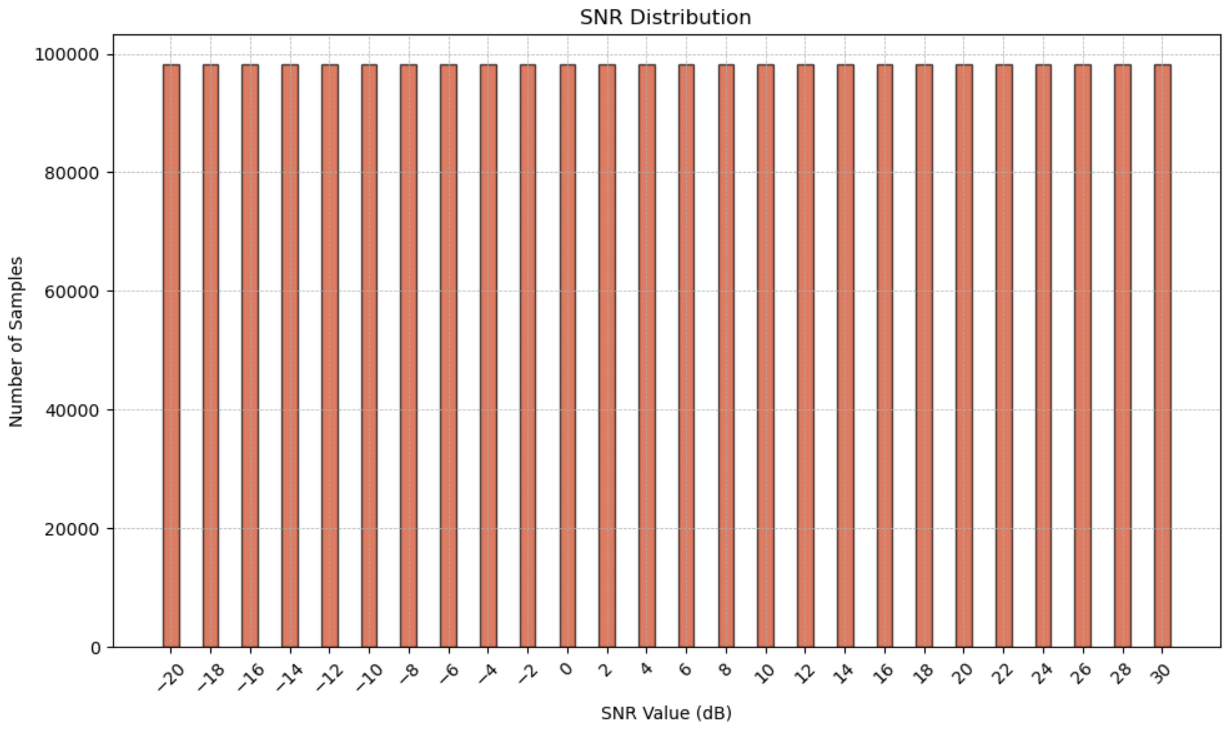


Figure 5.1: RADIOML 2018.01A [2] Dataset: SNR Distribution

relevance. This is primarily due to the dataset's unique structure, where each SNR value is equivalently represented with a consistent count of samples (98304 samples for each SNR). This uniformity across different SNR values renders the statistical central tendency mode ineffective as a distinguishing or insightful measure for this dataset.

As previously delineated, the RADIOML 2018.01A dataset comprises 24 unique modulation classes, each meticulously documented and characterized. Each of these distinct modulation classes encompasses a robust sample size of 106,496, ensuring a substantial data volume for each class critical for conducting rigorous and comprehensive analytical investigations. The generous allocation of samples for each modulation class enhances the dataset's utility and value for researchers by providing a rich and diverse substrate for the exploration and examination of various signal modulations and their intricate characteristics. A detailed breakdown and enumeration of the samples associated with each modulation class are pro-

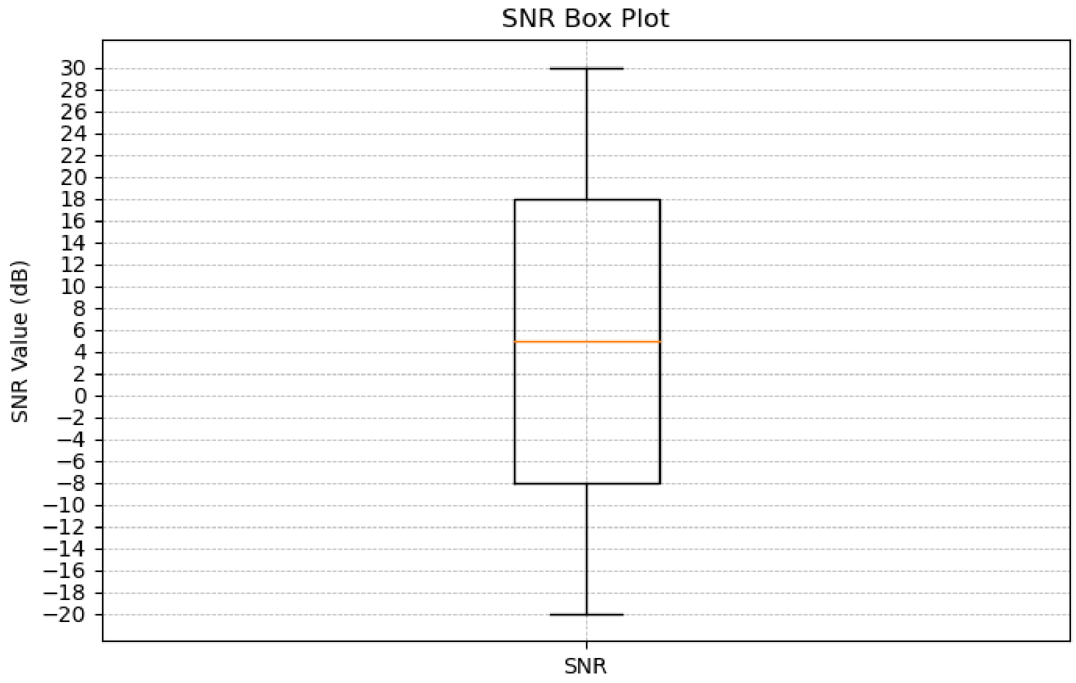


Figure 5.2: RADIOML 2018.01A [2] Dataset: SNR Box Plot

vided in the referenced Table 5.2, serving as a visual and systematic guide for readers and researchers seeking to familiarize themselves with the dataset's structure and composition.

Figure 5.3 provides a clear depiction of the modulation class distribution throughout the dataset indices, highlighting the careful assembly to ensure uniformity among the various classes. This balanced distribution is crucial, intentionally avoiding class dominance within the dataset. Such intentional organization is foundational for unbiased, accurate analysis and model training, preventing skewness towards a specific modulation class. The visual representation in the figure serves as a confirmation of the systematic data compilation process, promising a reliable basis for future analyses and modeling exercises that rely on this dataset's impartial and balanced structure.

Table 5.2: RADIOML 2018.01A [2] Dataset: No. of Samples for each Modulation Class

Serial No.	Modulation Class	No. of Samples	Serial No.	Modulation Class	No. of Samples
1.	OOK	106496	13.	16QAM	106496
2.	4ASK	106496	14.	32QAM	106496
3.	8ASK	106496	15.	64QAM	106496
4.	BPSK	106496	16.	128QAM	106496
5.	QPSK	106496	17.	256QAM	106496
6.	8PSK	106496	18.	AM-SSB-WC	106496
7.	16PSK	106496	19.	AM-SSB-SC	106496
8.	32PSK	106496	20.	AM-DSB-WC	106496
9.	16APSK	106496	21.	AM-DSB-SC	106496
10.	32APSK	106496	22.	FM	106496
11.	64APSK	106496	23.	GMSK	106496
12.	128APSK	106496	24.	OQPSK	106496

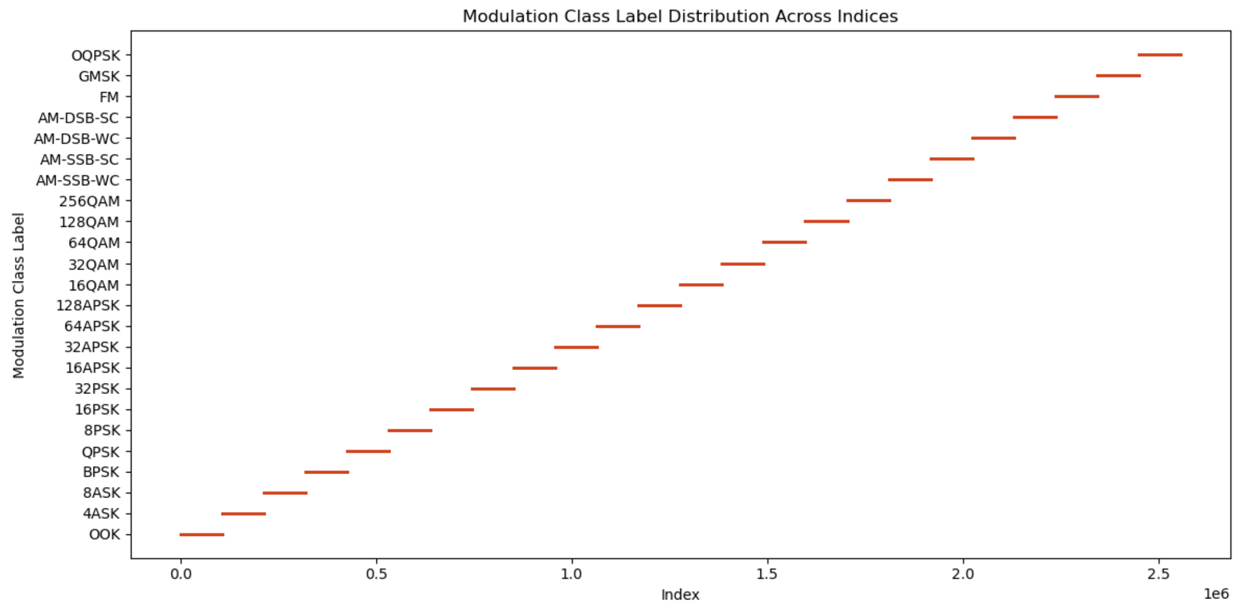


Figure 5.3: RADIOML 2018.01A [2] Dataset: Modulation Class Label Distribution Across Indices

In Figure 5.4, a careful arrangement of signal-to-noise ratio (SNR) distributions across different modulation classes is presented. Each class exhibits a uniform and incremental distribution of SNR values, with a consistent allocation of 4096 samples per discrete SNR level.

This systematic organization is pivotal, facilitating a more nuanced and precise analysis by ensuring a well-balanced representation of SNR values within each modulation class. Such deliberate structuring enhances the analytical depth, allowing for comprehensive exploration and yielding more insightful observations about each modulation class. The figure effectively visualizes this methodical assembly of data, establishing a solid basis that will be instrumental for in-depth studies and explorations in future research endeavors requiring SNR variations within the dataset and SNR-based dataset partitioning.

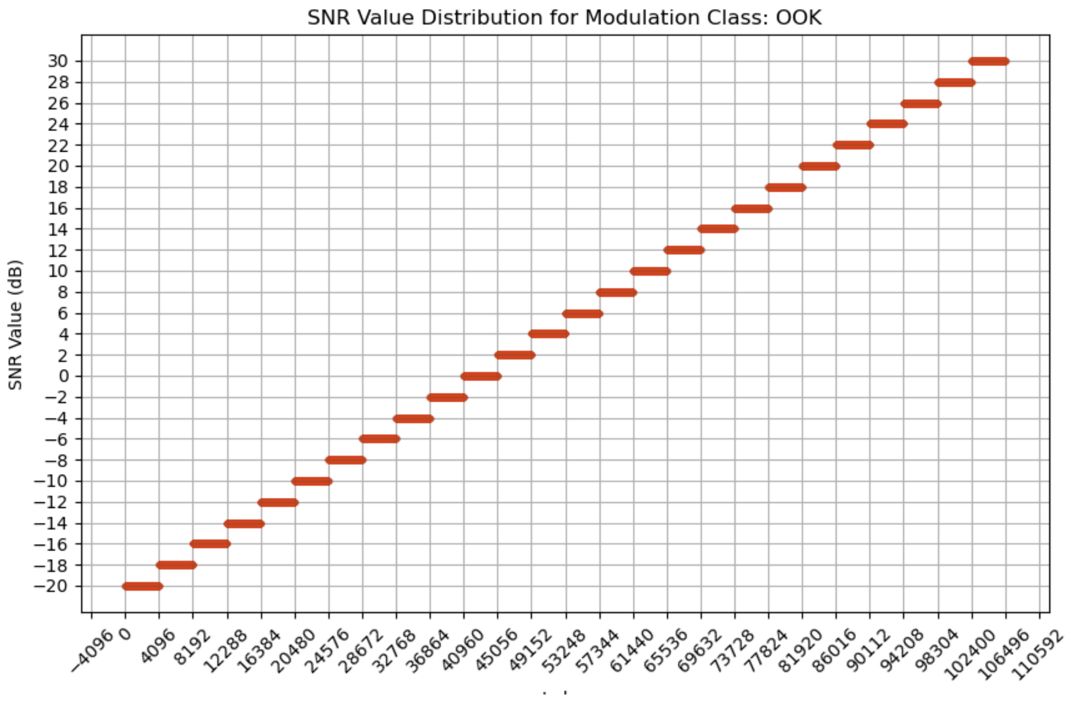


Figure 5.4: RADIOML 2018.01A [2] Dataset: SNR Value Distribution for Modulation Class: OOK

Concluding our initial examination, we have a dataset meticulously curated and notably free of class imbalances, making it highly suitable for intricate experiments and evaluations within the domains of Machine Learning (ML) and Deep Learning (DL). A crucial preprocessing step, data randomization, is fundamental in laying the groundwork for future objectives and is essential in algorithm development and performance evaluation.

Within Appendix A, a curated selection of signal samples is provided for visual inspection. This visual assortment aims to illuminate the inherent characteristics and intricate complexities associated with the dataset currently under examination. For each distinct modulation type present in the dataset, representative samples are depicted at both -20 dB and +30 dB signal-to-noise ratio (SNR) values. This deliberate presentation facilitates a comparative analysis between the pristine, unadulterated signals (showcased at +30 dB SNR) and their noisy counterparts (showcased at -20 dB SNR). The contrast between the two SNR levels underscores the complexities introduced when a signal is subjected to the influence of noise, engendering a formidable challenge in accurately identifying the modulation class of a given noisy signal. The visual representation and comparison of these samples at different SNR levels provide invaluable insights into the challenges and nuances entailed in modulation classification under noisy conditions.

5.1.5 Dataset Preprocessing

We discuss the preprocessing steps necessary for training and testing our transformer-based architecture. These steps are pivotal for the successful implementation and optimization of the model. RADIOML 2018.01A dataset is presented with the following constituents:

- X: Represents the features of the RF signal as IQ data points in their raw format.
- Y: Represents the labels in one-hot encoded format, encompassing 24 classes. The one-hot encoded labels can be mapped to their respective modulation scheme based on information in a separate reference file as part of the dataset, mapping the index of one-hot encoded label to a corresponding modulation scheme.

- Z: Represents the SNR values of the corresponding IQ data points.

The randomization process is meticulously executed to ensure that the constituents X, Y, and Z are coherently mapped to one another, maintaining the integrity and correspondence of the features, labels, and SNR values throughout the dataset. Figure 5.5 illustrates the label distribution and Figure 5.6 illustrates the SNR distribution across the 1000 samples after randomization. The raw IQ data is used for training the models in two distinctive training approaches, as discussed below:

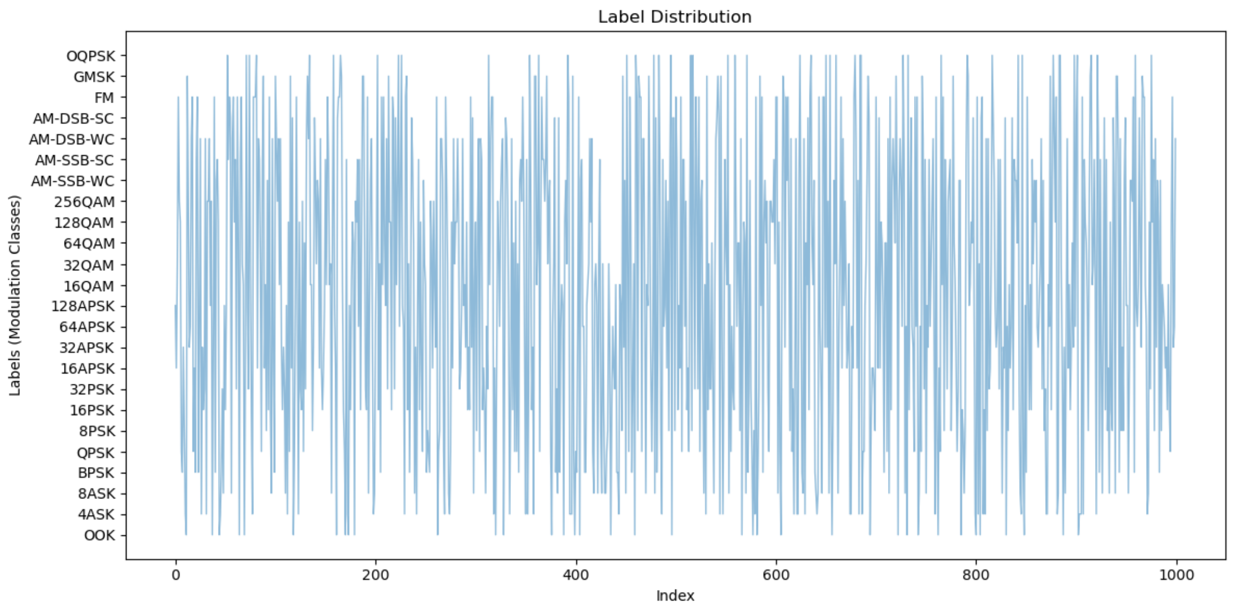


Figure 5.5: Label distribution after randomization of the RADIOML 2018.01A [2] dataset for 1000 samples

1. **Holistic Dataset Training:** Initially, the model is trained comprehensively using the entire dataset. This approach ensures that the model has a vast and diverse set of data to learn from, which is instrumental in achieving a generalized understanding of the modulation schemes.

   The dataset is randomized and split into 80% training and 20% testing data. In the model training phase, the training dataset is split into 90% training and 10% validation

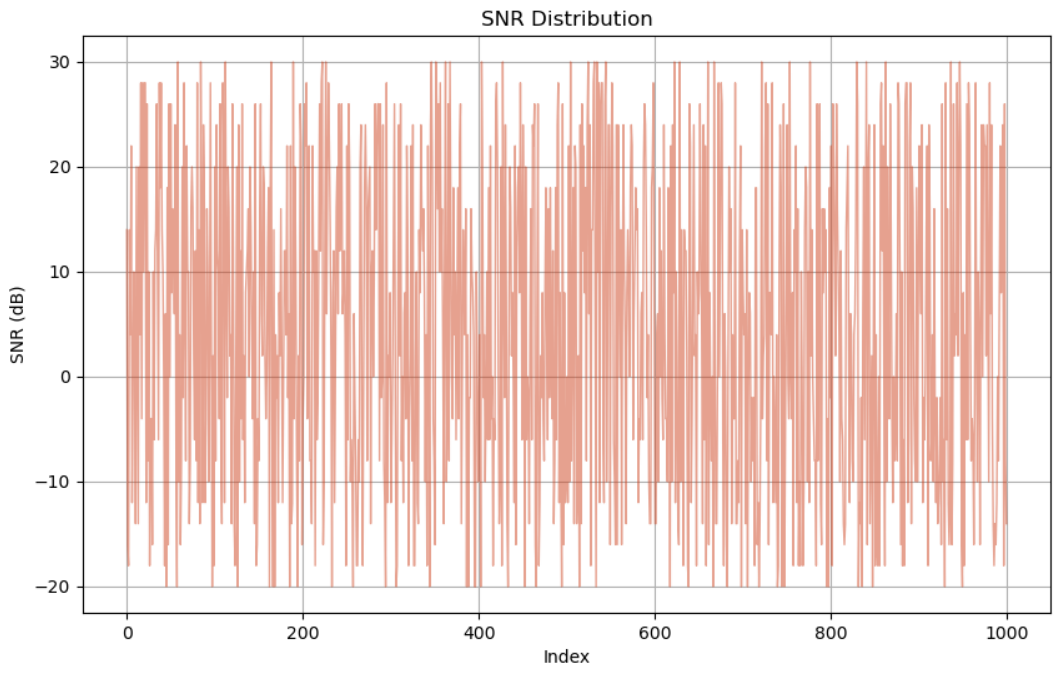


Figure 5.6: SNR distribution after randomization of the RADIOML 2018.01A [2] dataset for 1000 samples

data by the deep learning framework of Keras [71]. Figure 5.7 and Figure 5.8 showcase dataset sample signals with low SNR value and high SNR value signals, respectively, from the RADIOML 2018.01A [2] dataset.

2. **SNR-Based Dataset Segmentation & Training:**

   This strategic approach divides the dataset into groups based on SNR values before randomization. This is possible because the RADIOML 2018.01A Dataset is presented systematically, as discussed previously. The dataset is aggregated into five groups (Group A through Group E) as mentioned in Table 5.3. Figure 5.9 showcases a sample signal from **Group A**, Figure 5.10 from **Group B**, Figure 5.11 from **Group C**, Figure 5.12 from **Group D**, and Figure 5.13 from **Group E**, all from the RADIOML 2018.01A [2] dataset.

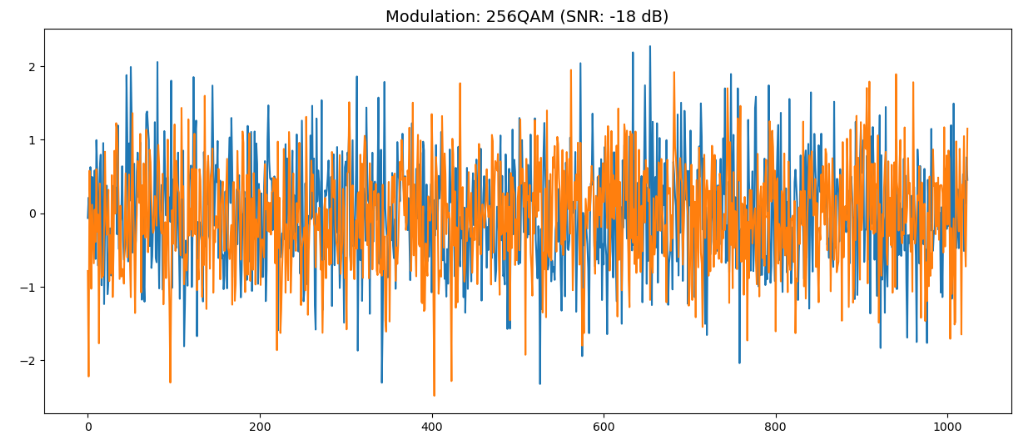


Figure 5.7: A low SNR signal sample from the RADIOML 2018.01A [2] dataset

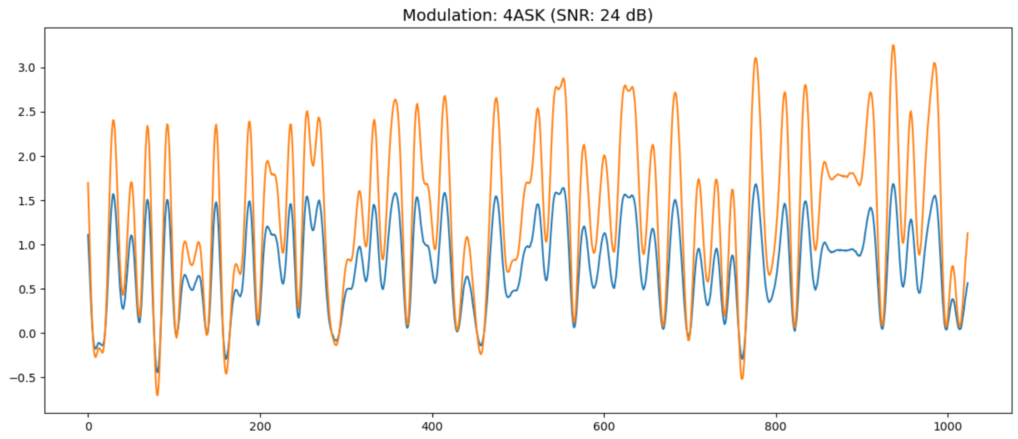


Figure 5.8: A high SNR signal sample from the RADIOML 2018.01A [2] dataset

Group Name	**SNR Range (dB)**
Group A	-20 to -10
Group B	-8 to 0
Group C	2 to 10
Group D	12 to 20
Group E	22 to 30

Table 5.3: SNR ranges for different groups

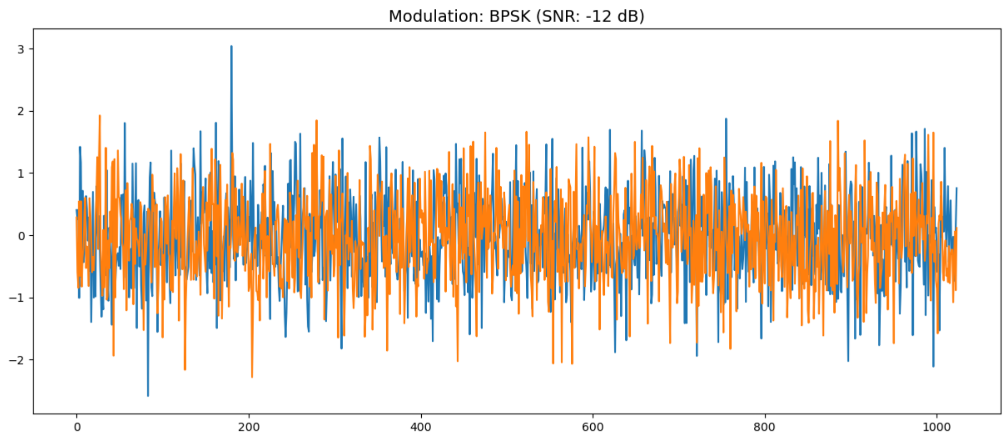


Figure 5.9: Group A signal sample from the RADIOML 2018.01A [2] dataset

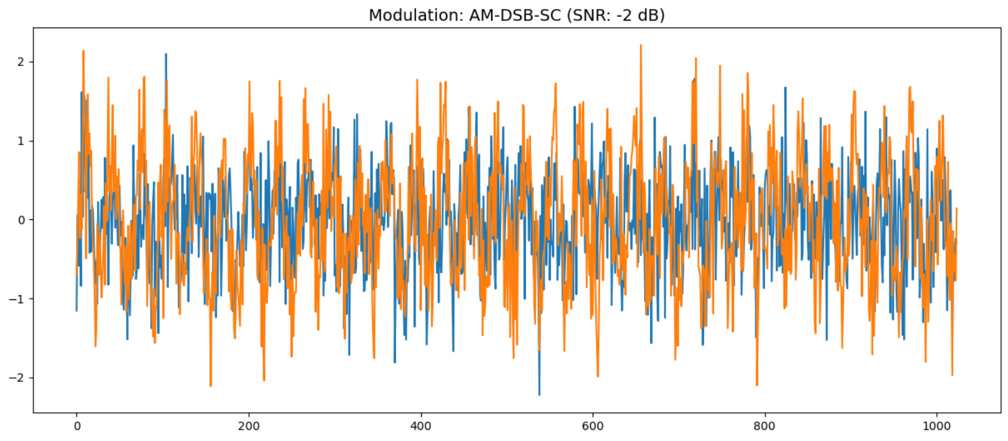


Figure 5.10: Group B signal sample from the RADIOML 2018.01A [2] dataset

The randomization preprocessing step is followed right after the segmentation step. The data in each group is split into 80% training and 20% testing data. Each group is individually trained, allowing the model to specialize and adapt to various SNR-level features. In the training phase for each group, the training dataset is further split into 90% training and 10% validation data by the deep learning framework of Keras [71].

Overall, this approach facilitates a more tailored and precise classification of the mod-

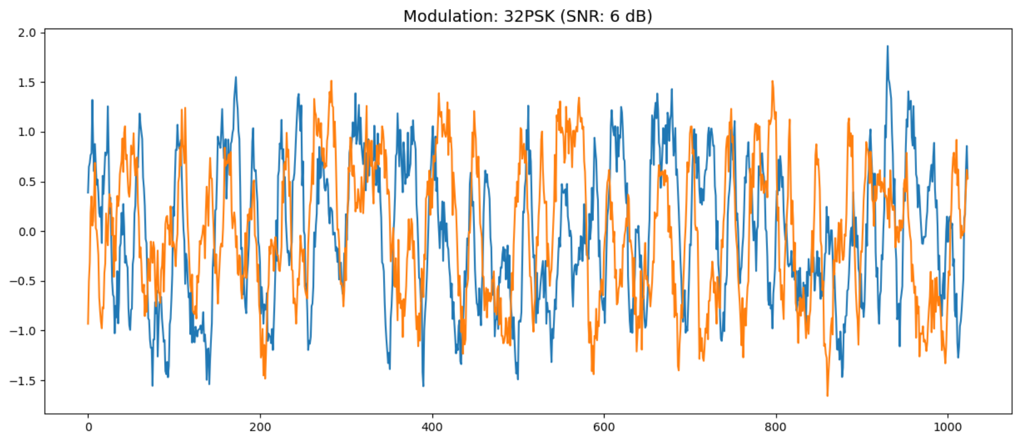


Figure 5.11: Group C signal sample from the RADIOML 2018.01A [2] dataset

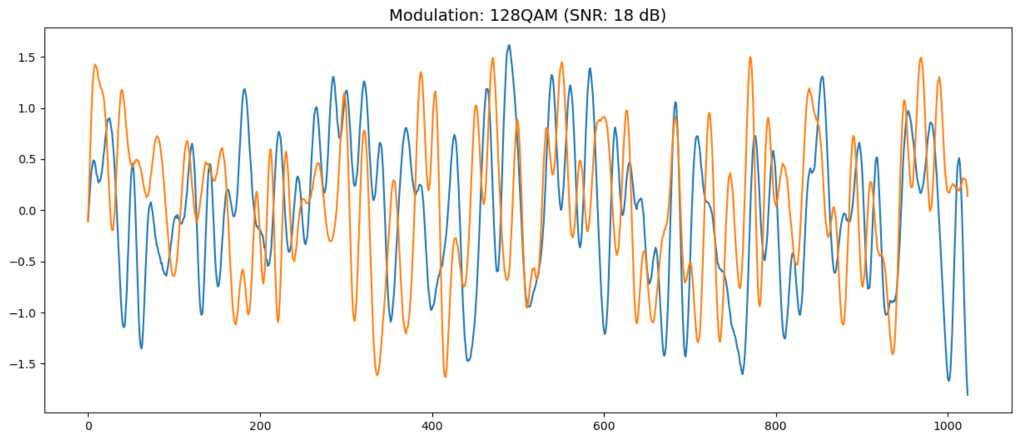


Figure 5.12: Group D signal sample from the RADIOML 2018.01A [2] dataset

ulation schemes. A selection criterion based on SNR values is employed to classify the modulation schemes effectively. This criterion ensures that the SNR influences the model's decision, allowing for a more robust and accurate classification.

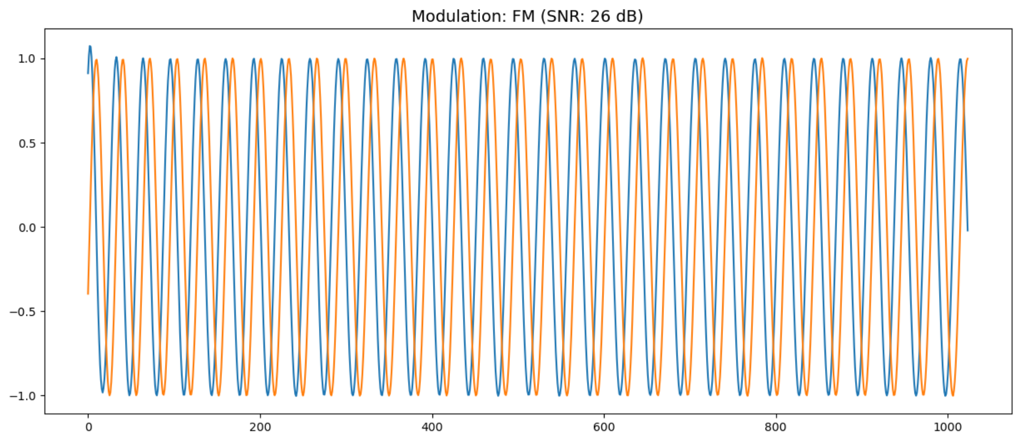


Figure 5.13: Group E signal sample from the RADIOML 2018.01A [2] dataset

5.2 Implementation

In the field of RF signal modulation classification, the application of machine learning or deep learning techniques is an artificial intelligence problem and is often referred to as automatic modulation classification (AMC) [46] or automatic modulation recognition (AMR) [22] in literature. In this thesis, we are evaluating our proposed deep learning model based on transformer encoder architecture for multi-class RF signal modulation classification on the RADIOML 2018.01A [2, 3] dataset. As already discussed, this dataset has 24 distinct RF signal modulation classes: OOK, 4ASK, 8ASK, BPSK, QPSK, 8PSK, 16PSK, 32PSK, 16APSK, 32APSK, 64APSK, 128APSK, 16QAM, 32QAM, 64QAM, 128QAM, 256QAM, AM-SSB-WC, AM-SSB-SC, AM-DSB-WC, AM-DSB-SC, FM, GMSK, and OQPSK.

Mathematically, the multi-class classification problem can be formulated as follows:

$$f(X) \rightarrow y_i, \text{ where } i \in \{1, 2, \ldots, n\} \tag{5.1}$$

In Equation 5.1, X represents the dataset, which is the RADIOML 2018.01A dataset, the function $f(X)$ maps the signal samples to their corresponding labels y_i where $i \in \{1, 2, \ldots, n\}$, which are the 24 modulation class labels. The following mathematical formulation represents the multi-class classification problem in terms of a neural network implementation:

$$f(X) = \text{argmax}\left(\text{softmax}(W \cdot X + b)\right) \tag{5.2}$$

In Equation 5.2, W is a weight matrix, b is the bias vector, X is the input feature vector (from the dataset). The function $softmax()$ is the activation function, which converts the function's raw output into class probabilities. The function $argmax()$ selects the class with the highest probability as the predicted class label.

In this thesis, we present an innovative transformer-based neural network architecture designed for the task of multi-class classification on a challenging dataset. The architecture is particularly suited for scenarios with high-dimensional input data. We leverage the transformer encoder architecture in our model to capture complex patterns in sequential data, enabling it to make precise predictions across multiple RF signal modulation classes. This demonstrates the versatility of the transformer architecture for domains beyond natural language processing having high-dimensional sequential data.

The core of our model consists of the following essential components, and their architecture and implementation are discussed subsequently:

1. **Multi-Head Self-Attention**, *refer sub-section 5.2.1*
2. **Transformer Block**, *refer sub-section 5.2.2*
3. **The Transformer-Based Neural Network**, *refer sub-section 5.2.3*

Appendix B discusses the additional mathematical foundations that are utilized to build the transformer-based architecture.

5.2.1 Multi-Head Self-Attention: Architecture & Implementation

Multi-Head Self-Attention: Architecture

Multi-Head Self-Attention is inspired by the transformer's self-attention mechanism. Multi-head self-attention is a specific type of multi-head attention where the input sequence is the same as the output sequence. In other words, it focuses on capturing relationships and dependencies within the same sequence, allowing each element to attend to other elements within the sequence itself. This layer enables our model to weigh the importance of different parts of the input data when making predictions. In essence, it learns to focus on relevant information while disregarding noise, a capability of this mechanism particularly valuable when dealing with intricate datasets. The mathematical formulation of this layer is illustrated in Equation 4.5. The architecture diagram of the Multi-Head Self-Attention Mechanism is illustrated in Figure 5.14.

Multi-head self-attention consists of multiple attention heads that operate in parallel, and each head computes attention scores for each element in the input sequence with respect to all other elements in the same sequence, without looking at external context. In transformers, multi-head attention allows concurrent processing of context-aware representations, streamlining sequence handling. In contrast, LSTMs operate sequentially, maintaining hidden states over timesteps, making them less time-efficient. The transformer's architecture thus offers substantial improvements in training and inference speeds.

Multi-Head Self-Attention: Implementation

The implementation of the Multi-Head Self-Attention Mechanism is described in Algorithm 1. The parameters that can be varied in this layer are the *embedding dimension* and *number of heads*. The *projection dimension* is computed by dividing *embedding dimension* with *number of heads*, and the division has to be exact with no remainders for the attention mechanism to work, which is one of the constraints we have to observe while setting the parameters.

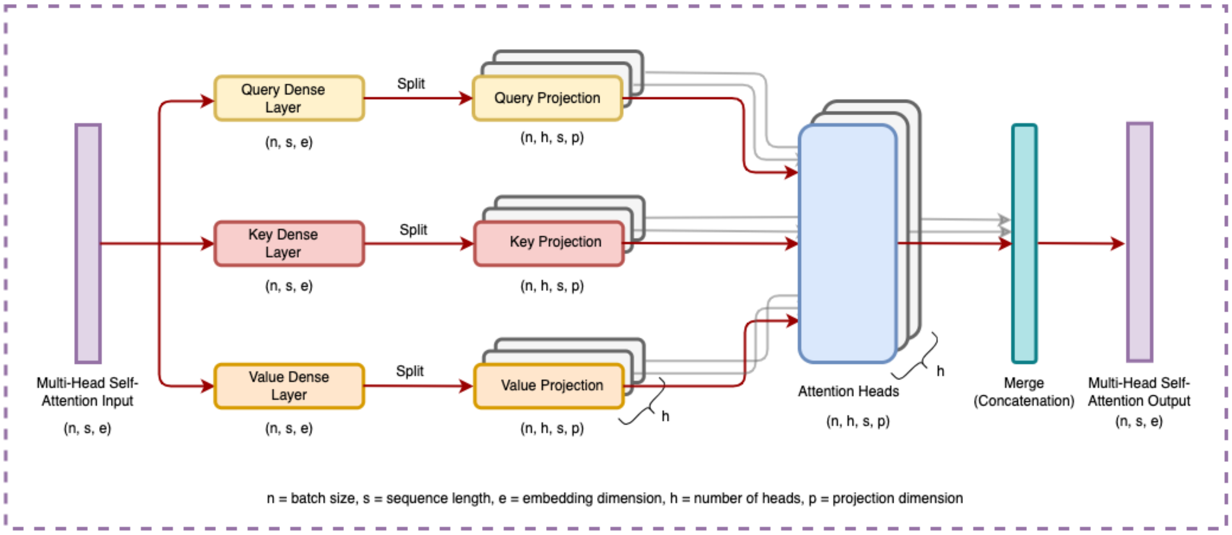


Figure 5.14: The architecture diagram of the Multi-Head Self-Attention Mechanism

5.2.2 Transformer Block: Architecture & Implementation

Transformer Block: Architecture

Building upon the self-attention mechanism, the Transformer Block further enhances the model's capabilities. It integrates feed-forward neural networks and layer normalization, allowing the model to capture higher-level abstractions within the data. Layer Normalization

Algorithm 1 Multi-Head Self-Attention

```
Input: $input[batch\_size][seq\_len][embed\_dim]$
Output: $output[batch\_size][seq\_len][embed\_dim]$
Data: $embed\_dim, num\_heads, projection\_dim$
Initialize: $query\_dense, key\_dense, value\_dense, combine\_heads$
  $projection\_dim \leftarrow embed\_dim \div num\_heads$
  function ATTENTION($query, key, value$)
      $score \leftarrow query \times key^\top$
      $dim\_key \leftarrow dim(key)$
      $scaled\_score \leftarrow score \div \sqrt{dim\_key}$
      $weights \leftarrow$ SOFTMAX($scaled\_score$)
      $output \leftarrow weights \times value$
      return $output, weights$
  end function
  function SEPARATE_HEADS($x, batch\_size$)
      $x \leftarrow$ reshape($x$)
      return $x^\top$
  end function
  $batch\_size \leftarrow dim(input)$
  $query \leftarrow query\_dense(input)$
  $key \leftarrow key\_dense(input)$
  $value \leftarrow value\_dense(input)$
  $query \leftarrow$ SEPARATE_HEADS($query, batch\_size$)
  $key \leftarrow$ SEPARATE_HEADS($key, batch\_size$)
  $value \leftarrow$ SEPARATE_HEADS($value, batch\_size$)
  $attention, weights \leftarrow$ ATTENTION($query, key, value$)
  $attention \leftarrow attention^\top$
  $concat\_attention \leftarrow$ reshape($attention$)
  $output \leftarrow combine\_heads(concat\_attention)$
  return $output$
```

and Dropout layers ensure that the model does not overfit or make errors due to irregularities within the data. By applying these components, our model becomes proficient at recognizing subtle features and relationships that might be challenging to discern with traditional machine learning approaches. The architecture diagram of the Transformer Block is illustrated in Figure 5.15.

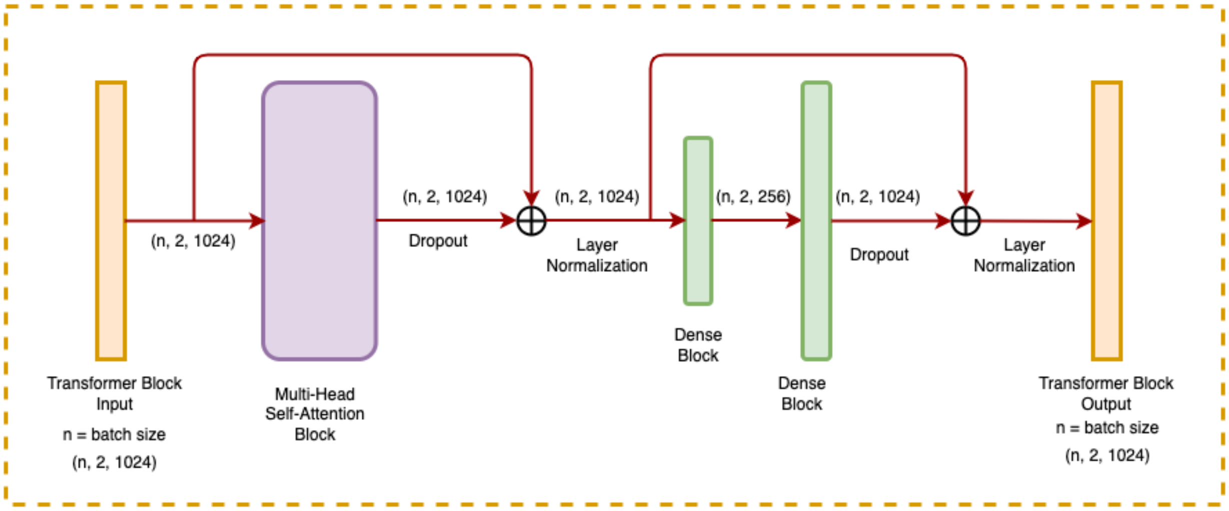


Figure 5.15: The architecture diagram of the Transformer Block

Transformer Block: Implementation

The input sequence to the Transformer Block, once processed by the Multi-Head Self-Attention Mechanism, is added to itself before going through a layer normalization block. This data is subsequently passed into the feed-forward network and added back to itself before going through another layer normalization block to generate the output of the Transformer Block. The implementation of the Transformer Block is described in Algorithm 2.

We can vary the *batch size*, the *dropout rates*, and the dimension of the first dense layer of the feed-forward network to tune the model for improved performance. The input data dimension and the output data dimension of the Transformer Block are to be kept the same.

Algorithm 2 Transformer Block

```
Input: $input[batch\_size][2][1024]$
Output: $output[batch\_size][2][1024]$
Data: $embed\_dim, num\_heads, ff\_dim$
  function TransformerBlock($embed\_dim, num\_heads, ff\_dim$)
      Initialize:
      $att \leftarrow$ MultiHeadSelfAttention($embed\_dim, num\_heads$)
      $ffn \leftarrow$ Sequential([$Dense(ff\_dim), Dense(embed\_dim)$])
      $layernorm1 \leftarrow$ LayerNormalization()
      $layernorm2 \leftarrow$ LayerNormalization()
      $dropout1 \leftarrow$ Dropout()
      $dropout2 \leftarrow$ Dropout()
  end function
  function call($input$)
      $attn\_output \leftarrow att(input)$
      $attn\_output \leftarrow dropout1(attn\_output)$
      $output \leftarrow layernorm1(input + attn\_output)$
      $ffn\_output \leftarrow ffn(output)$
      $ffn\_output \leftarrow dropout2(ffn\_output)$
      return $layernorm2(output + ffn\_output)$
  end function
```

5.2.3 The Transformer-Based Neural Network: Architecture, Implementation & Training

The Transformer-Based Neural Network: Architecture

Building the Complete Model through Dense Layers, the Neural Network Architecture: The complete neural network architecture is built by adding dense layers to the output of the Transformer Block. After the information is processed through the Transformer Block, the data flows through several neural networks to refine and prepare the data for final decision-making. The architecture diagram of the Neural Network Architecture is illustrated in Figure 5.16.

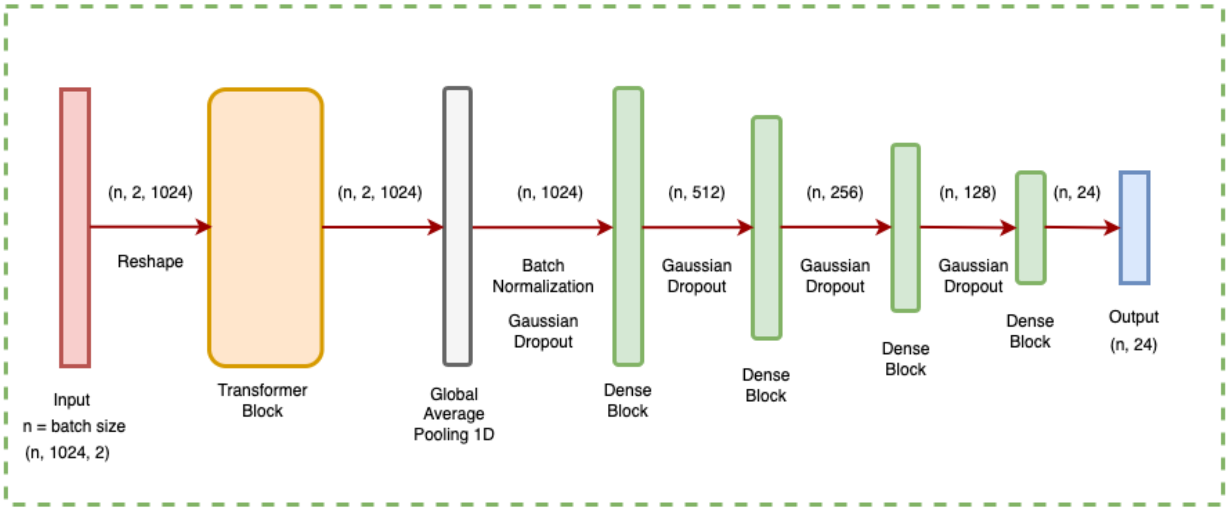


Figure 5.16: The Neural Network architecture diagram with the Transformer Block

The Transformer-Based Neural Network: Implementation

The implementation of the Transformer-Based Neural Network architecture is described in Algorithm 3. After receiving the output from the Transformer Block, the data is processed through Global Average Pooling 1D layer to reduce any spatial dimension for the feature map to reduce overfitting. Then, the output is processed through a Batch Normalization layer to accelerate training through quicker convergence. This layer also helps mitigate internal covariate shift and mildly introduces a form of regularization. Gaussian Dropout helps the model to be more resilient so that it doesn't overemphasize certain features and further adds regularization. The dense neural networks with Gaussian Error Linear Unit (GELU) activation, with diminishing dimension, analyze the refined data, and the raw output values from the final dense layer, i.e., logits, are processed by the softmax layer as output probabilities for the final categorization.

Algorithm 3 Transformer-based Neural Network Model & Training

```
Input: $X_{\text{train}}[batch\_size][1024][2]$, classes
Output: $output[batch\_size][24]$
  function TRANSFORMER_MODEL($X_{\text{train}}$, classes)
      Initialize: $embed\_dim, num\_heads, ff\_dim$
      $X_{\text{input}} \leftarrow$ INPUT($dim(X_{\text{train}}[0])$)        ▷ initiate input data placeholder tensor
      $X \leftarrow$ reshape($X_{\text{input}}$)
      $transformer\_block \leftarrow$ TRANSFORMERBLOCK($embed\_dim, num\_heads, ff\_dim$)
      $x \leftarrow transformer\_block(X)$
      $x \leftarrow$ GLOBALAVERAGEPOOLING1D($x$)
      $x \leftarrow$ BATCHNORMALIZATION($x$)
      $x \leftarrow$ GAUSSIANDROPOUT($x$)
      $x \leftarrow$ DENSE(512)($x$)
      $x \leftarrow$ GAUSSIANDROPOUT($x$)
      $x \leftarrow$ DENSE(256)($x$)
      $x \leftarrow$ GAUSSIANDROPOUT($x$)
      $x \leftarrow$ DENSE(128)($x$)
      $x \leftarrow$ GAUSSIANDROPOUT($x$)
      $x \leftarrow$ DENSE($24, activation = "softmax"$)($x$)
      return MODEL($input = X_{\text{input}}, output = x$)
  end function
  function LR_SCHEDULE(epoch, lr)
      DEFINE custom learning rate scheduler
      return lr
  end function
  model $\leftarrow$ TRANSFORMER_MODEL($X_{\text{train}}$, classes)
  COMPILE model WITH:
       loss = 'categorical_crossentropy'
       optimizer = Adam Optimizer
  lr_scheduler $\leftarrow$ LearningRateScheduler(schedule=LR_SCHEDULE)
  for epoch = 1 to $k$ do
      for all batch in batches of size $n$ from $X_{\text{train}}$ do
          FORWARD PASS on batch to compute predictions
          COMPUTE loss and gradients using 'categorical_crossentropy'
          UPDATE model parameters using Adam Optimizer
      end for
      UPDATE learning rate with lr_scheduler
  end for
```

The Transformer-Based Neural Network: Training

The model is implemented in Keras Framework [71] with Tensorflow backend. The model is compiled with categorical cross-entropy as the loss function, Adam Optimizer, an algorithm for first-order gradient-based optimization, for adjusting weights of the network for adaptive learning, and accuracy as the principal metric for monitoring the performance and convergence throughout the learning phase. The model has a total of **5,424,280** parameters, out of which **5,422,232** are trainable parameters, and **2,048** are non-trainable parameters.

The categorical cross-entropy loss function for a multi-class classification task is mathematically expressed as:

$$\mathcal{L}_{CCE} = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic} \cdot \log(p_{ic}) \tag{5.3}$$

In Equation 5.3, N denotes the total number of samples, C denotes the total number of classes, y_{ic} indicates the true probability of sample i belonging to class c, and p_{ic} indicates the predicted probability of sample i belonging to class c.

A **Batch Learning** strategy is applied in training this model. In batch learning, the model is trained on a batch of samples simultaneously, which allows for a more generalized and robust learning process by reducing variance in the model's weights during updates. The size of the batch is a hyperparameter that can be varied. In the **Forward Propagation** process, for each sample in the batch, the model predicts by assigning probability for each class, the true labels are provided in one-hot encoded representation, and the **categorical cross-entropy loss** is calculated for each sample in the batch based on the prediction and true labels. The **average categorical cross-entropy loss** is computed for all samples in the batch to get a single loss for all the samples in the batch. In the **Backward Propagation** process, the gradients are calculated based on the average loss, and the model's weights are updated accordingly. At this point, the **Adam Optimizer** uses the average loss to adjust the model parameters to minimize loss in subsequent iterations. The **Adam Optimizer** is capable of handling sparse gradients on noisy problems, which makes it particularly suited to this problem.

We implement specified callbacks at each epoch during the execution of the training phase for (i) saving the best model based on validation loss, (ii) early stopping if validation accuracy does not improve for a certain number of epochs, and (iii) adjust the learning rate. The custom learning rate scheduler is defined for training this model, in which the learning rate is reduced by 10% after a certain number of epochs, which is decided based on the training criteria. This way, we save the best model, avoid unnecessary epochs, and adaptively ensure better convergence during training.

This concludes the training process of the model. In the next Chapter, we discuss our experiments, findings, and results.

Chapter 6

Experiments & Results

In the previous chapter, we discussed the implementation of the model, covering the dataset and the essential preprocessing steps required. This discussion included details on both the holistic dataset training and the SNR-based dataset segmentation and training. We also walked through the transformer-based neural network architecture we implemented, providing insights into its internal workings and the overall training process.

In this chapter, we focus on hyperparameter tuning, documenting key observations from our experiments and findings. We then proceed to discuss the results derived from these experimental processes. We also propose a re-configurable framework for automatic modulation classification.

6.1 Hyperparameters

The hyperparameters within the transformer-based architecture are the *embedding dimension* for each token, the *number of heads* for Multi-Head Self-Attention Layer, & the *feed forward dense layer dimension* within the Transformer Block. We set the *embedding dimension* to 1024, based on the sequence length of the signal sample. We empirically set the *number of heads* to 128 and the *feed forward dense layer dimension* to 256. We do not vary these parameters so that our model is consistent across all experiments. As already discussed, our model compiles with a total of **5,424,280** parameters, out of which **5,422,232** are trainable parameters, and **2,048** are non-trainable parameters.

We have already discussed the categorial cross-entropy loss function and Adam Optimizer in the previous chapter that is implemented at the time of model compilation.

For model training, we have the *batch size*, *number of epochs*, and callbacks depending on the *learning rate*, validation accuracy, and validation loss. We vary them for the necessary hyperparameter tuning to get our results.

6.2 Holistic Dataset Training & Results

We train our architecture on the RADIOML 2018.01A Dataset holistically to understand how well our model generalizes over the range of SNRs. We conduct training in batch sizes of 512, 1024, and 2048. In subsequent discussions, we will refer to the architecture of these models as TR-AMC (*batch size*), denoting the **Tr**ansformer-Based architecture for **A**utomatic **M**odulation **C**lassification.

We present our findings in terms of accuracy on the test data in Figure 6.1. We observe that with an increase in batch size, the accuracy per SNR and the overall accuracy increase. We also present the confusion matrix for each batch size and observe that with larger batch sizes, the model generalizes well across the various modulation schemes, as shown in Figure 6.2.

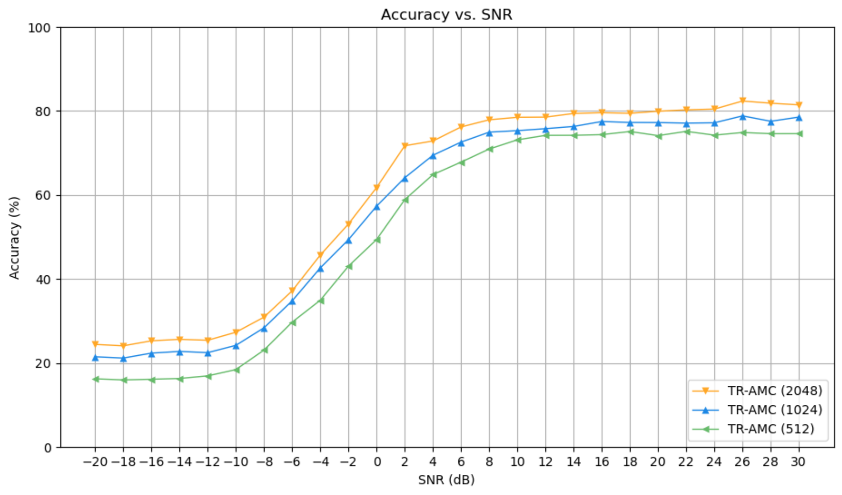
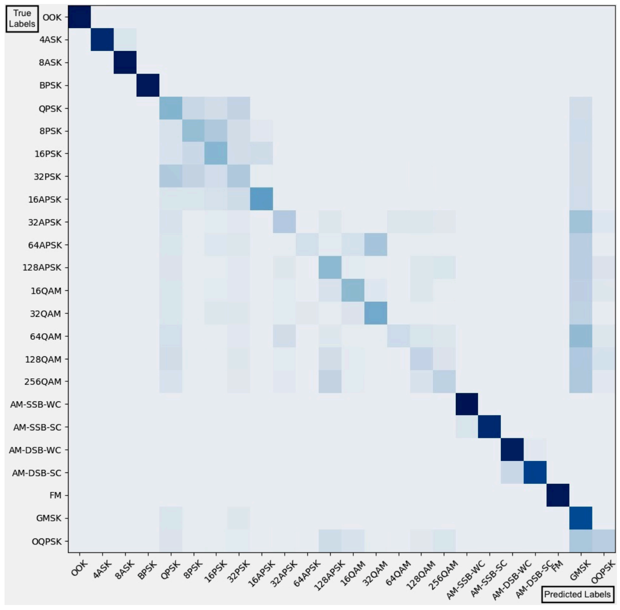


Figure 6.1: Holistic Training & Evaluation Accuracy per SNR

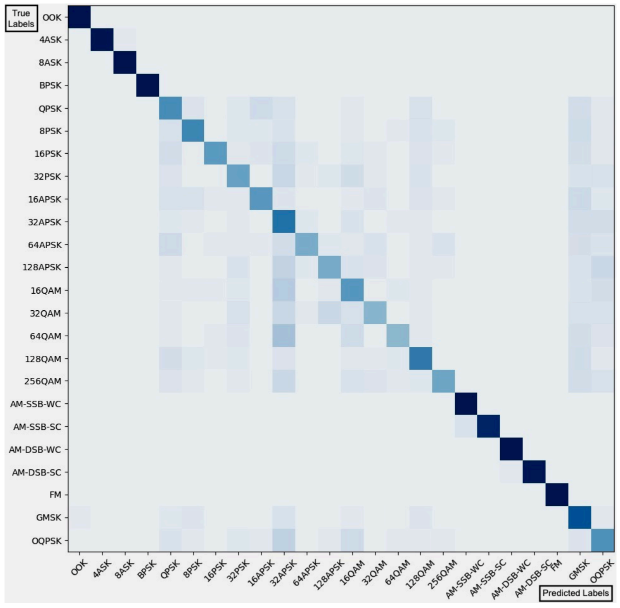
6.3 SNR-Based Dataset Segmentation, Training & Results

In the preprocessing phase, as previously described, dataset segmentation was conducted based on SNR values, resulting in five distinct groups, each representing a specific range of SNR values. Each group was trained using a unified architecture structure utilizing two distinct batch sizes, 1024 and 2048. Training with a batch size of 512 was infeasible due to the limitations of the architecture, poor convergence, and training instability.

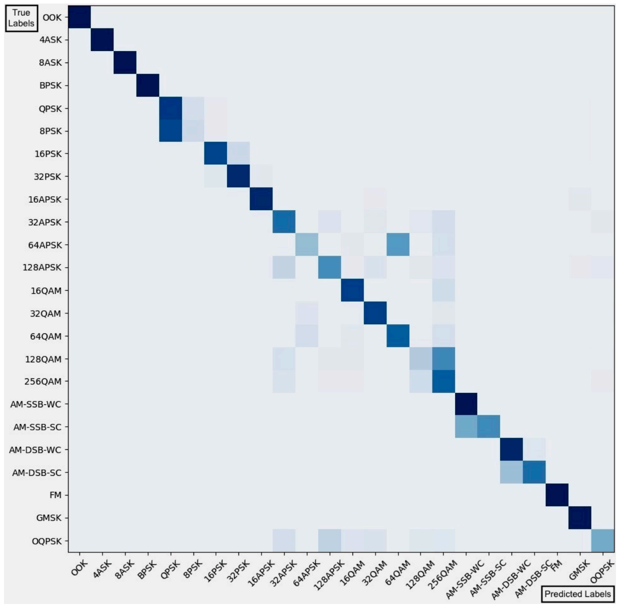
The models are systematically named according to the SNR groups they were trained on as per Table 6.1. The weights of the different models were stored in respective files and subsequently used to configure a **gating mechanism** based on SNRs to classify the RF signal modulation. In subsequent discussions, we will refer to the architecture of these models as GTR-AMC (*batch size*), denoting the **G**ated **Tr**ansformer-Based architecture for **A**utomatic **M**odulation **C**lassification. We discuss more about the gating mechanism in Section 6.6.



(a) Confusion Matrix: Holistic Training & Evaluation (batch size of 512)



(b) Confusion Matrix: Holistic Training & Evaluation (batch size of 1024)



(c) Confusion Matrix: Holistic Training & Evaluation (batch size of 2048)

Figure 6.2: Confusion Matrix: Holistic Training & Evaluation

We present the Evaluation Accuracy per SNR Graph for GTR-AMC (1024) and GTR-AMC (2048) in Figure 6.3. Also, the confusion matrix for GTR-AMC (1024) and GTR-AMC

Group Name	SNR Range (dB)	Model Name
Group A	-20 to -10	Model A
Group B	-8 to 0	Model B
Group C	2 to 10	Model C
Group D	12 to 20	Model D
Group E	22 to 30	Model E

Table 6.1: Model Names for various groups of SNR ranges

(2048) is illustrated in Figure 6.4. In comparison with the evaluation performance of TR-AMC, GTR-AMC has better evaluation performance when trained with a larger batch size.

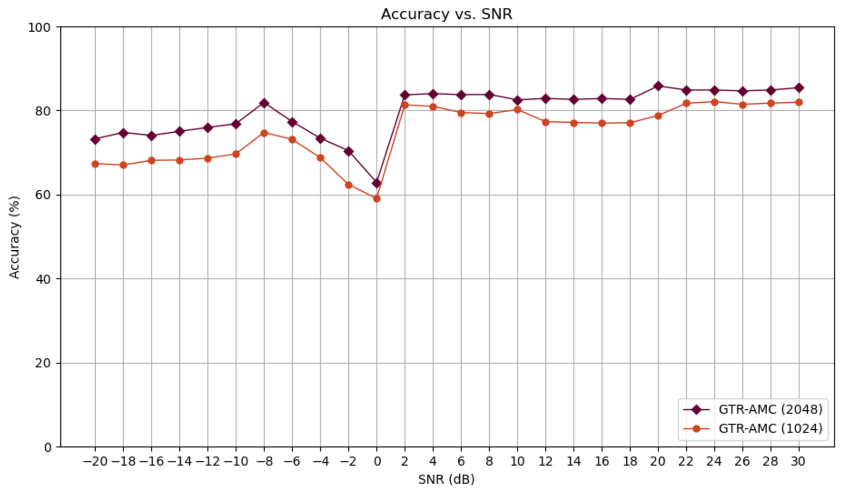
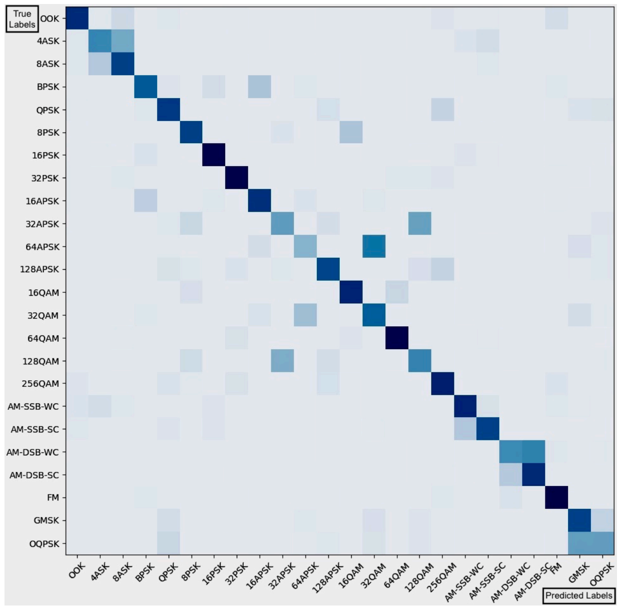


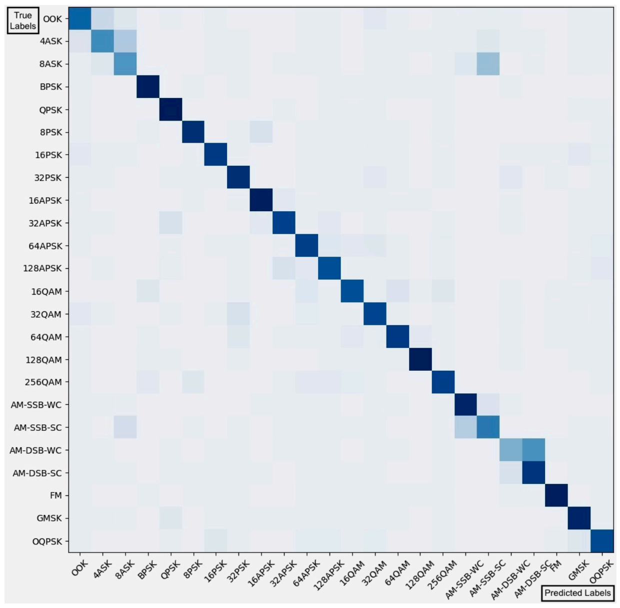
Figure 6.3: Group-wise Training & Evaluation Accuracy per SNR

6.4 Performance Comparison: Holistic Dataset vs. SNR-based Dataset Segmentation

In Figure 6.5, we observe the performance of both the architectures, viz., TR-AMC and GTR-AMC, with various batch sizes. We observe a substantial increase in performance below 0 dB SNR for the GTR-AMC architecture. Overall, GTR-AMC outperforms the TR-



(a) Confusion Matrix: Group-wise Training & Evaluation (batch size of 1024)



(b) Confusion Matrix: Group-wise Training & Evaluation (batch size of 2048)

Figure 6.4: Confusion Matrix: Group-wise Training & Evaluation

AMC for most SNRs. The GTR-AMC (2048) performed better than GTR-AMC (1024) at all SNR levels and for most modulation schemes. The overall accuracy of GTR-AMC (2048) is 79.81%, and the low SNR (-20 dB to 0 dB) accuracy is approximately 74.16% with the highest low SNR accuracy of 81.90% at -8 dB.

This observation is instrumental in identifying the capability of transformers to generalize well in a given context. In our use case, the context was the SNR value ranges to split the dataset and have different modulation types, but similar SNRs. The loss curves and the accuracy curves for TR-AMC and GTR-AMC for all batch sizes and groups are presented in Appendix C.

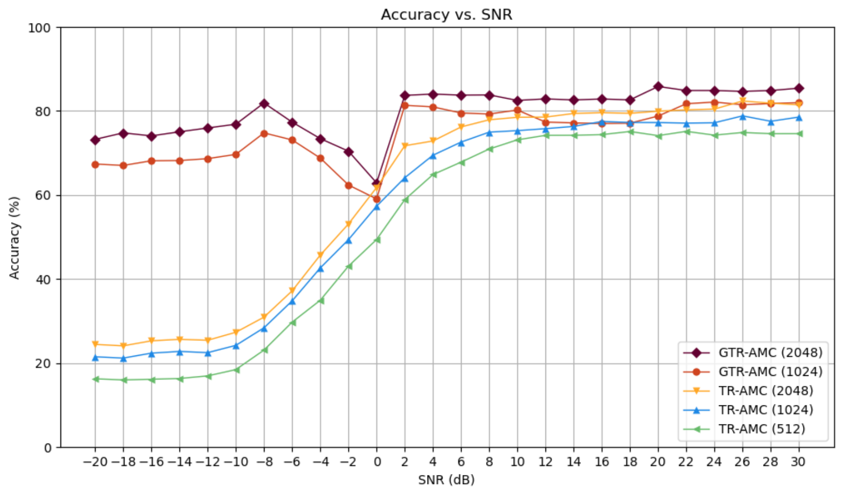


Figure 6.5: Holistic vs. Group-wise Training & Evaluation Accuracy per SNR

6.5 Performance Improvement: GTR-AMC (Hybrid)

We observe from Figure 6.5 that there is scope for improvement in the evaluation accuracy of modulation classification of RF signals in Group B. The transformer-based architecture was trained with Group B data with a batch size of 3072 & 4096. In another approach, we included data at -10 dB and 2 dB to extend the range of Group B data and used a batch size of 2048 to train the model.

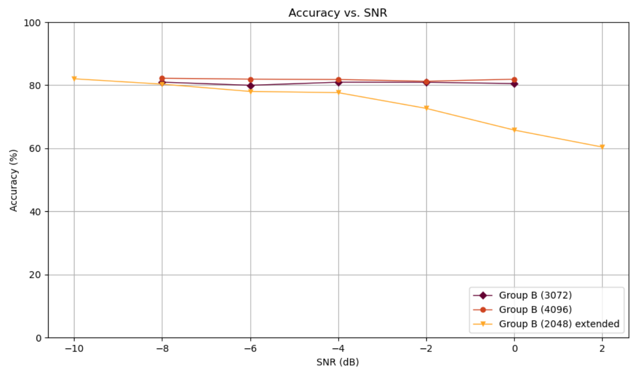


Figure 6.6: GTR-AMC Group B Additional Training

We observe a noticeable evaluation accuracy with the model trained with a batch size of 3072 & 4096. The model trained with a batch size of 4096 has marginally better performance. The model trained on the extended Group B data showed no substantial improvement. The comparative performance of the three training approaches is observed in Figure 6.6.

We create our final transformer-based model GTR-AMC (Hybrid) using the Group B model trained with a batch size of 4096 and the remaining groups (A, C, D, and E) from GTR-AMC (2048) using the gating mechanism discussed in Section 6.6. The confusion matrix for GTR-AMC (Hybrid) is presented in Figure 6.7.

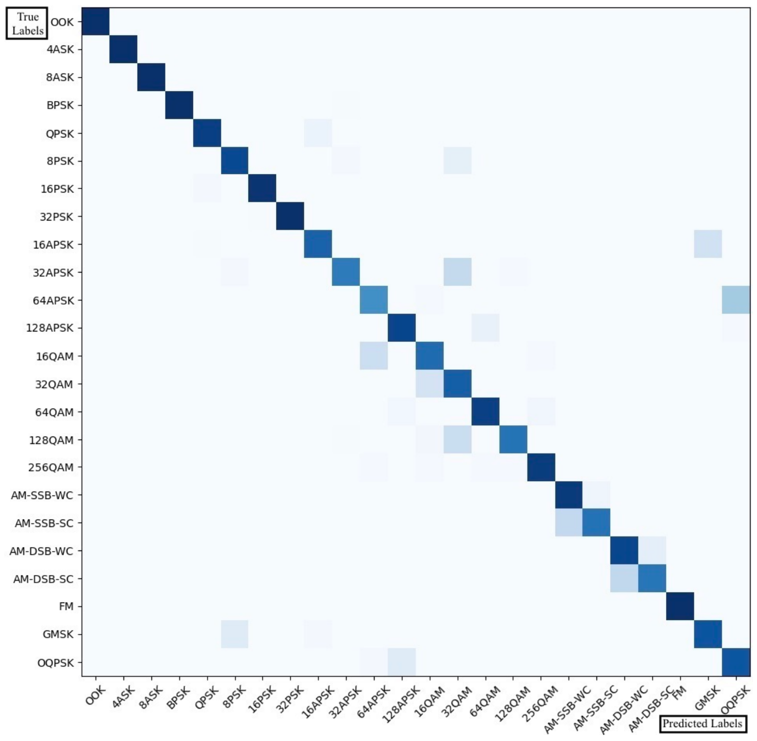


Figure 6.7: Confusion Matrix: GTR-AMC (Hybrid)

In Appendix D, we present the training loss curves for Group B additional training with batch sizes of 3072 and 4096 in Figure D.1 and Figure D.2, respectively. We also present the training loss curve of Group B with an extended range of SNRs in Figure D.3. Table E.1 and Table E.2 in Appendix E present the evaluation performance of TR-AMC and GTR-

AMC for various batch sizes both in terms of SNRs and modulation types. We observe the classification accuracy based on SNR for all the models evaluated in Figure 6.8.

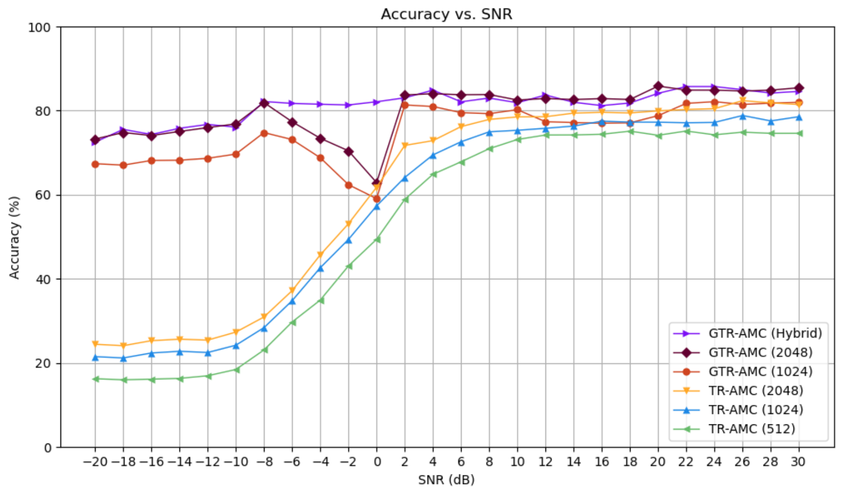


Figure 6.8: Holistic vs. Group-wise Evaluation Accuracy per SNR with GTR-AMC (Hybrid)

6.6 Proposed Model: Runtime Re-configurable Automatic Modulation Classification Framework

In our research, a model selection strategy based on signal-to-noise ratio (SNR) values is utilized. This strategy, known as a **gating mechanism**, systematically directs data to specific models optimized for distinct SNR ranges. The method ensures that each model functions within its optimized SNR boundaries, thus improving overall prediction accuracy and model robustness in varied SNR conditions. We have observed the performance of GTR-AMC to be superior over TR-AMC, which supports our proposed model. We observed in Section 6.5 that by changing the Group B model weights, we achieved higher classification accuracy. This flexibility is offered by applying a modular approach introduced with the gating mechanism.

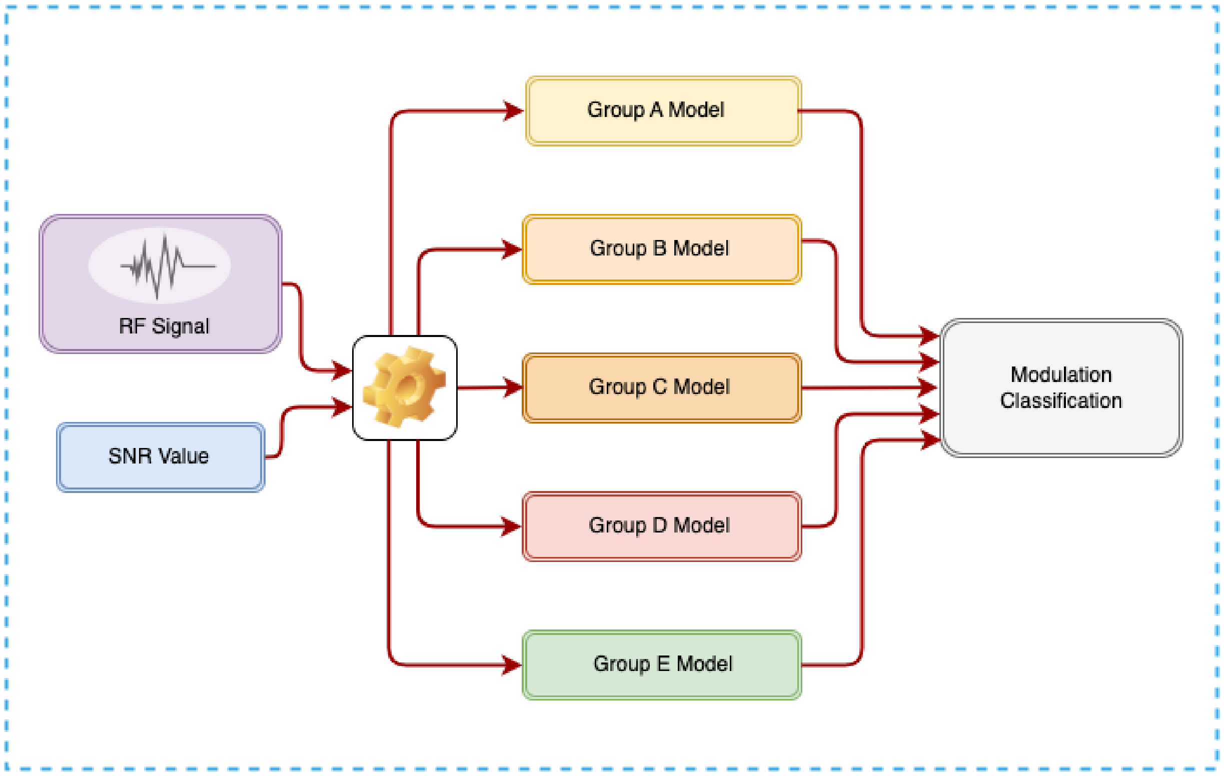


Figure 6.9: Runtime Re-configurable Automatic Modulation Classification Framework

We refer to our proposed model as **Runtime Re-configurable Automatic Modulation Classification Framework** as illustrated in Figure 6.9. In this architecture, we implement an optimized GTR-AMC model and, on a real-time basis, use the SNR value of an RF signal to provide the specific model weights to classify the modulation scheme. We must note that the base architecture will have a fixed number of parameters; only the weights will change based on context. This makes it economically viable to implement as a neural engine on hardware (chip).

Chapter 7

Conclusions

In this chapter, we discuss the conclusions drawn from the experiments conducted as a part of this thesis. Additionally, we provide valuable insights and recommendations for future work.

7.1 Conclusions

In this thesis, we have explored the initial development phase of PORT5+ and a specific critical use case at the radio receiver of automatic modulation classification. We implemented a transformer-based encoder architecture to train our models TR-AMC and GTR-AMC with various batch sizes and observed the evaluation accuracy of the models.

GTR-AMC incorporated a gating mechanism based on SNRs to upload context-specific weights for more accurate modulation classification. GTR-AMC (2048) had an overall accuracy of 79.81% for the entire range of SNRs, and approximately 74.16% accuracy at low SNR levels (-20 dB to 0 dB), which is notably higher than previous research. We achieve the highest accuracy of -8 dB within the low SNR range with this model. With further experiments to improve Group B evaluation accuracy, we developed GTR-AMC (Hybrid) using the modularization technique of the gating mechanism to improve performance. GTR-AMC (Hybrid) has a more uniform SNR classification with overall accuracy 81.25% and 78.15% at low SNR levels (-20 dB to 0 dB).

As a result, we propose a Runtime Re-configurable Automatic Modulation Classification Framework based on the GTR-AMC model developed using transformer-based architecture. This can use one architecture and multiple sets of weights that are fine-tuned for SNR-specific RF signals for accurate classification. Overall, the work conducted in this thesis is promising for integrating AI-driven solutions for cognitive radios and smart communication systems.

7.2 Future Work

The promising outcomes of employing gated transformer-based architecture for automatic modulation classification, as demonstrated in this thesis, lay the groundwork for several future research directions. We recommend implementing diverse data augmentation techniques, coupled with comprehensive data preprocessing, model training, and fine-tuning, to significantly enhance the model's generalizability. Some of the possible data augmentation techniques on RF signals are:

1. **Rotation of Signal in IQ Plane:** A possible solution is to use a preprocessing step to rotate the signal to an orientation that the model can accurately classify.

2. **Signal Fusion:** Combining two or more signals to create a new signal and cluster the signals.

3. **Noise Injection:** Adding various noises to the original signal to help the model learn to ignore variations.

4. **Time Shifting:** To simulate the effect of different propagation delays.

5. **Frequency Shifting:** To simulate the effect of Doppler shifts.

For future research, exploring ablation studies could prove invaluable in dissecting the components of transformer-based architectures for automatic modulation classification. Through a systematic process of selectively removing components and assessing the resulting impact on the model's efficacy, we can isolate and identify the fundamental elements vital to the model's performance. This analytical approach helps quantify the contribution of each component and provides an understanding of the architecture's operational dynamics. The various ablation studies that can be conducted for the transformer-based architecture are:

1. **Attention Heads Ablation:** Analyze how the number and configuration of attention heads affect the classification performance.

2. **Dropout Rate Alteration:** Adjust the dropout rates in different parts of the network to understand their impact on the model's ability to generalize and avoid overfitting.

3. **Learning Rate and Optimizer Changes:** Experiment with different learning rates and optimizers to evaluate their effect on the convergence and performance of the model.

4. **Activation Function Modification:** Replace activation functions in the network to assess how they affect the model's learning capability.

5. **Loss Function Modification:** Experiment with different loss functions to see how they affect the learning dynamics and classification accuracy.

6. **Attention Mechanism Modifications:** Modify the attention mechanism (like incorporating local or sparse attention) and observe the effects on the model's performance.

The transformer's encoder-decoder framework, equipped with an attention mechanism, presents a promising yet complex avenue for research in denoising signals for signal reconstruction, potentially establishing transformative applications within generative AI for signal processing. Future studies in this field present a wide range of opportunities across various types of signals, diversifying applications beyond smart communication systems.

Bibliography

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[2] T. J. O'Shea, T. Roy, and T. C. Clancy, "RADIOML 2018.01A," 2017. [Online]. Available: https://www.deepsig.ai/datasets.

[3] T. J. O'Shea, T. Roy, and T. C. Clancy, "Over-the-Air Deep Learning Based Radio Signal Classification," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 168–179, 2018.

[4] "5G eMBB, URLLC, and mMTC: Service Categories for a Smarter Tomorrow," 2023. [Online]. Available: https://resources.pcb.cadence.com/blog/2023-5g-embb-urllc-and-mmtc-service-categories-for-a-smarter-tomorrow.

[5] "Ultra Reliable and Low Latency Communications," 2023. [Online]. Available: https://www.3gpp.org/technologies/urlcc-2022.

[6] "5G Technology and Networks," 2022. [Online]. Available: https://www.thalesgroup.com/en/markets/digital-identity-and-security/mobile/inspired/5G.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An

Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," 2021. [Online]. Available: https://doi.org/10.48550/arXiv.2010.11929.

[8] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu, "Relational inductive biases, deep learning, and graph networks," 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1806.01261.

[9] A. Chadha, "Transformers," *Distilled AI*, 2020. https://aman.ai.

[10] "OpenAirInterface 5G Radio Access Network Project," 2023. [Online]. Available: https://openairinterface.org/oai-5g-ran-project/.

[11] S. Peng, S. Sun, and Y.-D. Yao, "A Survey of Modulation Classification Using Deep Learning: Signal Representation and Data Preprocessing," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 12, pp. 7020–7038, 2022. [Online]. Available: https://doi.org/10.1109/TNNLS.2021.3085433.

[12] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional Radio Modulation Recognition Networks," *arXiv.org*, 2016. [Online]. Available: https://doi.org/10.48550/arXiv.1602.04105.

[13] M. Kulin, T. Kazaz, I. Moerman, and E. De Poorter, "End-to-End Learning From Spectrum Data: A Deep Learning Approach for Wireless Signal Identification in Spectrum Monitoring Applications," *IEEE Access*, vol. 6, pp. 18484–18501, 2018. [Online]. Available: https://doi.org/10.1109/ACCESS.2018.2818794.

[14] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale

Image Recognition," 2015. [Online]. Available: https://doi.org/10.48550/arXiv.1409.1556.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," 2015. [Online]. Available: https://doi.org/10.48550/arXiv.1512.03385.

[16] L. Huang, W. Pan, Y. Zhang, L. Qian, N. Gao, and Y. Wu, "Data Augmentation for Deep Learning-Based Radio Modulation Classification," *IEEE Access*, vol. 8, pp. 1498–1506, 2020. [Online]. Available: https://doi.org/10.1109/ACCESS.2019.2960775.

[17] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269, 2017. [Online]. Available: https://doi.org/10.1109/CVPR.2017.243.

[18] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735.

[19] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A Search Space Odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2017. [Online]. Available: https://doi.org/10.1109/TNNLS.2016.2582924.

[20] X. Liu, D. Yang, and A. E. Gamal, "Deep neural network architectures for modulation classification," in *2017 51st Asilomar Conference on Signals, Systems, and Computers*, pp. 915–919, 2017. [Online]. Available: https://doi.org/10.1109/ACSSC.2017.8335483.

[21] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, Long Short-Term

Memory, fully connected Deep Neural Networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4580–4584, 2015. [Online]. Available: https://doi.org/10.1109/ICASSP.2015.7178838.

[22] M. Zhang, Y. Zeng, Z. Han, and Y. Gong, "Automatic Modulation Recognition Using Deep Learning Architectures," in *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–5, 2018. [Online]. Available: https://doi.org/10.1109/SPAWC.2018.8446021.

[23] H. Ma, G. Xu, H. Meng, M. Wang, S. Yang, R. Wu, and W. Wang, "Cross Model Deep Learning Scheme for Automatic Modulation Classification," *IEEE Access*, vol. 8, pp. 78923–78931, 2020. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.2988727.

[24] Z. Ke and H. Vikalo, "Real-Time Radio Modulation Classification With An LSTM Auto-Encoder," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4935–4939, 2021. [Online]. Available: 10.1109/ICASSP39728.2021.9414351.

[25] Z. Ke and H. Vikalo, "Real-Time Radio Technology and Modulation Classification via an LSTM Auto-Encoder," *IEEE Transactions on Wireless Communications*, vol. 21, no. 1, pp. 370–382, 2022. [Online]. Available: 10.1109/TWC.2021.3095855.

[26] A. P. Hermawan, R. R. Ginanjar, D.-S. Kim, and J.-M. Lee, "CNN-Based Automatic Modulation Classification for Beyond 5G Communications," *IEEE Communications Letters*, vol. 24, no. 5, pp. 1038–1041, 2020. [Online]. Available: https://doi.org/10.1109/LCOMM.2020.2970922.

[27] K. Yashashwi, A. Sethi, and P. Chaporkar, "A Learnable Distortion Correction Module

for Modulation Recognition," *IEEE Wireless Communications Letters*, vol. 8, no. 1, pp. 77–80, 2019. [Online]. Available: https://doi.org/10.1109/LWC.2018.2855749.

[28] S. Hong, Y. Zhang, Y. Wang, H. Gu, G. Gui, and H. Sari, "Deep Learning-Based Signal Modulation Identification in OFDM Systems," *IEEE Access*, vol. 7, pp. 114631–114638, 2019. [Online]. Available: https://doi.org/10.1109/ACCESS.2019.2934976.

[29] S. Hong, Y. Wang, Y. Pan, H. Gu, M. Liu, J. Yang, and G. Gui, "Convolutional Neural Network Aided Signal Modulation Recognition in OFDM Systems," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pp. 1–5, 2020. [Online]. Available: https://doi.org/10.1109/VTC2020-Spring48590.2020.9128455.

[30] S. Jaiswal, P. Paritosh, and P. Kumar, "Deep-Learning Based Modulation Identification in Wireless Communication System," in *2021 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, pp. 96–101, 2021. [Online]. Available: https://doi.org/10.1109/ANTS52808.2021.9936905.

[31] J. Shi, S. Hong, C. Cai, Y. Wang, H. Huang, and G. Gui, "Deep Learning-Based Automatic Modulation Recognition Method in the Presence of Phase Offset," *IEEE Access*, vol. 8, pp. 42841–42847, 2020. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.2978094.

[32] A. Ali and F. Yangyu, "k -Sparse Autoencoder-Based Automatic Modulation Classification With Low Complexity," *IEEE Communications Letters*, vol. 21, no. 10, pp. 2162–2165, 2017. [Online]. Available: https://doi.org/10.1109/LCOMM.2017.2717821.

[33] F. Meng, P. Chen, L. Wu, and X. Wang, "Automatic Modulation Classification: A Deep Learning Enabled Approach," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 10760–10772, 2018. [Online]. Available: https://doi.org/10.1109/TVT.2018.2868698.

[34] S. Zheng, P. Qi, S. Chen, and X. Yang, "Fusion Methods for CNN-Based Automatic Modulation Classification," *IEEE Access*, vol. 7, pp. 66496–66504, 2019. [Online]. Available: https://doi.org/10.1109/ACCESS.2019.2918136.

[35] H. Gu, Y. Wang, S. Hong, and G. Gui, "Blind Channel Identification Aided Generalized Automatic Modulation Recognition Based on Deep Learning," *IEEE Access*, vol. 7, pp. 110722–110729, 2019. [Online]. Available: https://doi.org/10.1109/ACCESS.2019.2934354.

[36] S. Hu, Y. Pei, P. P. Liang, and Y.-C. Liang, "Deep Neural Network for Robust Modulation Classification Under Uncertain Noise Conditions," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 564–577, 2020. [Online]. Available: https://doi.org/10.1109/TVT.2019.2951594.

[37] S. Hu, Y. Pei, P. P. Liang, and Y.-C. Liang, "Robust Modulation Classification under Uncertain Noise Condition Using Recurrent Neural Network," in *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7, 2018. [Online]. Available: https://doi.org/10.1109/GLOCOM.2018.8647582.

[38] Y. Wang, J. Wang, W. Zhang, J. Yang, and G. Gui, "Deep Learning-Based Cooperative Automatic Modulation Classification Method for MIMO Systems," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4575–4579, 2020. [Online]. Available: https://doi.org/10.1109/TVT.2020.2976942.

[39] S. Rajendran, W. Meert, D. Giustiniano, V. Lenders, and S. Pollin, "Deep Learning Models for Wireless Signal Classification With Distributed Low-Cost Spectrum Sensors," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 3, pp. 433–445, 2018. [Online]. Available: https://doi.org/10.1109/TCCN.2018.2835460.

[40] S. Rajendran, R. Calvo-Palomino, M. Fuchs, B. Van den Bergh, H. Cordobes, D. Giustiniano, S. Pollin, and V. Lenders, "Electrosense: Open and Big Spectrum Data," *IEEE Communications Magazine*, vol. 56, no. 1, pp. 210–217, 2018. [Online]. Available: https://doi.org/10.1109/MCOM.2017.1700200.

[41] O. S. Mossad, M. ElNainay, and M. Torki, "Deep Convolutional Neural Network with Multi-Task Learning Scheme for Modulations Recognition," in *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, pp. 1644–1649, 2019. [Online]. Available: https://doi.org/10.1109/IWCMC.2019.8766665.

[42] T. O'Shea and N. West, "Radio Machine Learning Dataset Generation with GNU Radio," *Proceedings of the GNU Radio Conference*, vol. 1, no. 1, 2016. [Online]. Available: https://pubs.gnuradio.org/index.php/grcon/article/view/11.

[43] F. N. Khan, K. Zhong, W. H. Al-Arashi, C. Yu, C. Lu, and A. P. T. Lau, "Modulation Format Identification in Coherent Receivers Using Deep Machine Learning," *IEEE Photonics Technology Letters*, vol. 28, no. 17, pp. 1886–1889, 2016. [Online]. Available: https://doi.org/10.1109/LPT.2016.2574800.

[44] F. N. Khan, K. Zhong, X. Zhou, W. H. Al-Arashi, C. Yu, C. Lu, and A. P. T. Lau, "Joint OSNR monitoring and modulation format identification in digital coherent receivers using deep neural networks," *Opt. Express*, vol. 25, pp. 17767–17776, Jul 2017. [Online]. Available: https://doi.org/10.1364/OE.25.017767.

[45] Y. Zhao, C. Shi, D. Wang, X. Chen, L. Wang, T. Yang, and J. Du, "Low-Complexity and Nonlinearity-Tolerant Modulation Format Identification Using Random Forest," *IEEE Photonics Technology Letters*, vol. 31, no. 11, pp. 853–856, 2019. [Online]. Available: https://doi.org/10.1109/LPT.2019.2910288.

[46] Y. Wang, M. Liu, J. Yang, and G. Gui, "Data-Driven Deep Learning for Automatic Modulation Recognition in Cognitive Radios," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 4074–4077, 2019. [Online]. Available: https://doi.org/10.1109/TVT.2019.2900460.

[47] H. Wu, Y. Li, L. Zhou, and J. Meng, "Convolutional neural network and multi-feature fusion for automatic modulation classification," *Electronics Letters*, vol. 55, 08 2019. [Online]. Available: http://dx.doi.org/10.1049/el.2019.1789.

[48] Y. Gu and X. Zhou, "Exploiting ResNeXt with Convolutional Shortcut for Signal Modulation Classification at Low SNRs," in *2023 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2023. [Online]. Available: https://doi.org/10.1109/IJCNN54540.2023.10191354.

[49] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated Residual Transformations for Deep Neural Networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5987–5995, 2017. [Online]. Available: https://doi.org/10.1109/CVPR.2017.634.

[50] Z. Zhang, C. Wang, C. Gan, S. Sun, and M. Wang, "Automatic Modulation Classification Using Convolutional Neural Network With Features Fusion of SPWVD and BJD," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 5, no. 3, pp. 469–478, 2019. [Online]. Available: https://doi.org/10.1109/TSIPN.2019.2900201.

[51] L. Li, C. Qin, G. Li, S. Hu, Y. Xie, and Z. Lei, "Transformer-based radio modulation mode recognition," *Journal of Physics: Conference Series*, vol. 2384, p. 012017, 12 2022. [Online]. Available: https://www.researchgate.net/publication/366113474_Transformer-based_radio_modulation_mode_recognition.

[52] Z. Liang, L. Wang, M. Tao, J. Xie, and X. Yang, "Attention Mechanism Based ResNeXt Network for Automatic Modulation Classification," in *2021 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, 2021. [Online]. Available: https://doi.org/10.1109/GCWkshps52748.2021.9682126.

[53] V. Sathyanarayanan, A. Jolly, and P. Gerstoft, "Novel Training Methodology to Enhance Deep Learning Based Modulation Classification," in *2021 55th Asilomar Conference on Signals, Systems, and Computers*, pp. 356–360, 2021. [Online]. Available: 10.1109/IEEECONF53345.2021.9723186.

[54] X. Wu, S. Wei, Y. Zhou, and F. Liao, "TSN-A: An Efficient Deep Learning Model for Automatic Modulation Classification Based on Intra-Class Confusion Reduction of Modulation Families," *IEEE Communications Letters*, vol. 26, no. 12, pp. 2964–2968, 2022. [Online]. Available: 10.1109/LCOMM.2022.3210586.

[55] F. Zhang, C. Luo, J. Xu, and Y. Luo, "An Efficient Deep Learning Model for Automatic Modulation Recognition Based on Parameter Estimation and Transformation," *IEEE Communications Letters*, vol. 25, no. 10, pp. 3287–3290, 2021. [Online]. Available: 10.1109/LCOMM.2021.3102656.

[56] H. Chen, L. Guo, C. Dong, F. Cong, and X. Mu, "Automatic Modulation Classification Using Multi-Scale Convolutional Neural Network," in *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, pp. 1–6, 2020. [Online]. Available: 10.1109/PIMRC48278.2020.9217305.

[57] A. Doering and J. Gall, "A Gated Attention Transformer for Multi-Person Pose Tracking," 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2306.05807.

[58] Z. Geng, Z. Chen, Q. Meng, and Y. Han, "Novel Transformer Based on Gated Convolutional Neural Network for Dynamic Soft Sensor Modeling of Industrial Processes," *IEEE*

Transactions on Industrial Informatics, vol. 18, no. 3, pp. 1521–1529, 2022. [Online]. Available: 10.1109/TII.2021.3086798.

[59] S. Jere, Y. Wang, I. Aryendu, S. Dayekh, and L. Liu, "Bayesian Inference-assisted Machine Learning for Near Real-Time Jamming Detection and Classification in 5G New Radio (NR)," 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2304.13660.

[60] S. Cristina and M. Saeed, "Building Transformer Models with Attention," 2022. [Online]. Available: https://machinelearningmastery.com/transformer-models-with-attention/.

[61] G. W. Lindsay, "Corrigendum: Attention in Psychology, Neuroscience, and Machine Learning," *Frontiers in Computational Neuroscience*, vol. 15, 2021. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fncom.2021.698574.

[62] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," 2014. [Online]. Available: https://doi.org/10.48550/arXiv.1409.0473.

[63] M.-T. Luong, H. Pham, and C. D. Manning, "Effective Approaches to Attention-based Neural Machine Translation," 2015. [Online]. Available: https://doi.org/10.48550/arXiv.1508.04025.

[64] V. Nair and G. E. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, (Madison, WI, USA), p. 807–814, Omnipress, 2010.

[65] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*,

vol. 61, pp. 85–117, jan 2015. [Online]. Available: https://doi.org/10.1016/j.neunet.2014.09.003.

[66] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer Normalization," 2016. [Online]. Available: https://doi.org/10.48550/arXiv.1607.06450.

[67] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Unsupervised representation learning of structured radio communication signals," in *2016 First International Workshop on Sensing, Processing and Learning for Intelligent Machines (SPLINE)*, pp. 1–5, 2016.

[68] T. J. O'Shea, J. Corgan, and T. C. Clancy, "RADIOML 2016.04C," 2016. [Online]. Available: https://www.deepsig.ai/datasets.

[69] E. Blossom, "GNU Radio: Tools for Exploring the Radio Frequency Spectrum," *Linux Journal*, 2004. [Online]. Available: https://www.linuxjournal.com/article/7319.

[70] A. Margulies, *The Software Defined Radio Forum*, ch. 3, pp. 73–92. John Wiley Sons, Ltd, 2002. [Online]. Available: https://doi.org/10.1002/0470846011.ch3.

[71] F. Chollet *et al.*, "Keras." https://keras.io, 2015.

[72] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," 2015. [Online]. Available: https://doi.org/10.48550/arXiv.1502.03167.

[73] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html.

[74] D. Hendrycks and K. Gimpel, "Gaussian Error Linear Units (GELUs)," 2023. [Online]. Available: https://doi.org/10.48550/arXiv.1606.08415.

[75] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*. Urbana: University of Illinois Press, 1949.

[76] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1412.6980.

[77] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of machine learning research*, vol. 12, no. 7, 2011. [Online]. Available: https://www.jmlr.org/papers/volume12/duchi11a/duchi11a.pdf.

Appendices

Appendix A

Comparative Visualization of Low and High SNR RF Signals Across Modulation Classes

In the field of RF communication, understanding the variations and implications of Signal-to-Noise Ratios (SNR) is crucial. To compare RF signals with different SNRs, the following sections provide samples showing low and high SNR values. These samples are extracted from various modulation classes in our comprehensive dataset, encompassing all 24 classes. The data used for this comparison is sourced from the RADIOML 2018.01A Dataset [3]. The side-by-side presentation aims to offer a clearer understanding of the nuances in how SNR varies across different modulation classes.

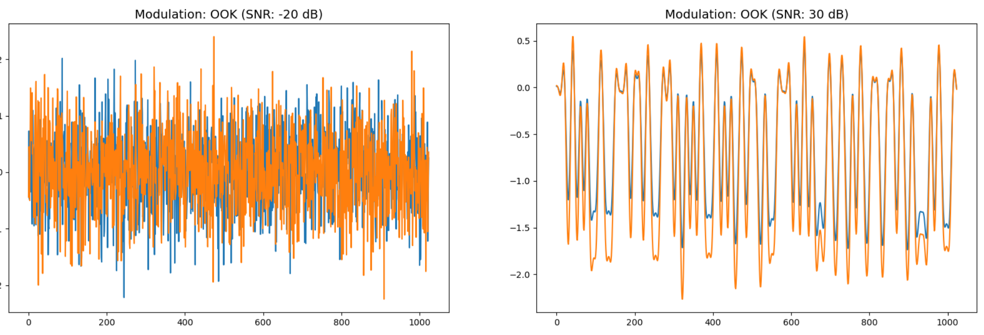


Figure A.1: RF Signal Modulation Class: OOK. Data Source: RADIOML 2018.01A [3]

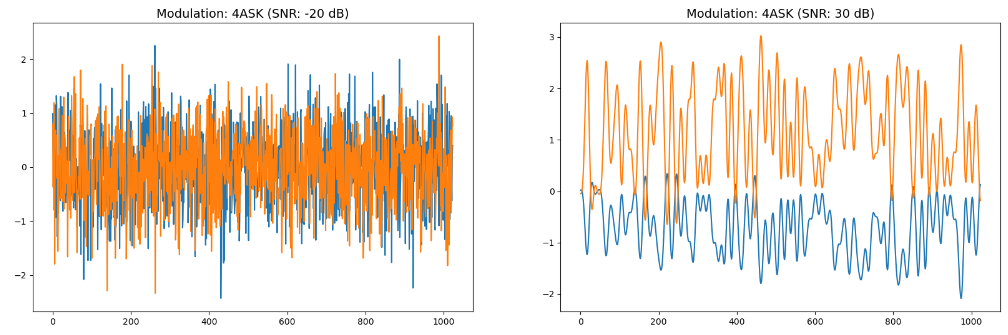


Figure A.2: RF Signal Modulation Class: 4ASK. Data Source: RADIOML 2018.01A [3]

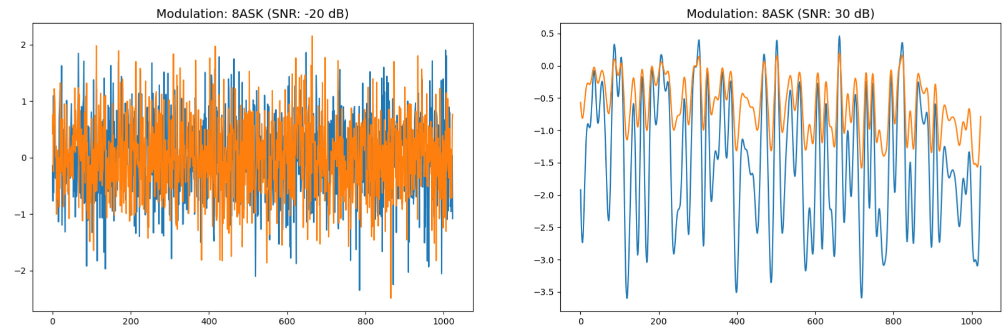


Figure A.3: RF Signal Modulation Class: 8ASK. Data Source: RADIOML 2018.01A [3]

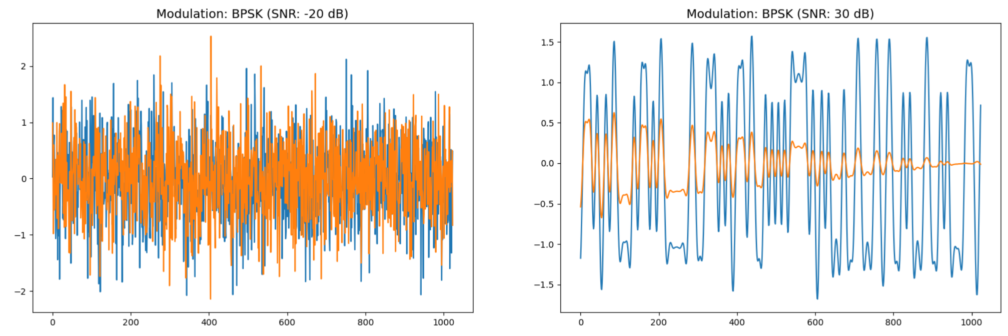


Figure A.4: RF Signal Modulation Class: BPSK. Data Source: RADIOML 2018.01A [3]

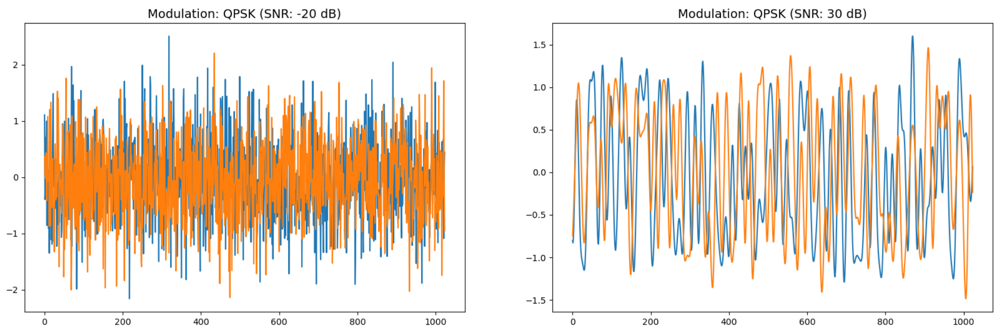


Figure A.5: RF Signal Modulation Class: QPSK. Data Source: RADIOML 2018.01A [3]

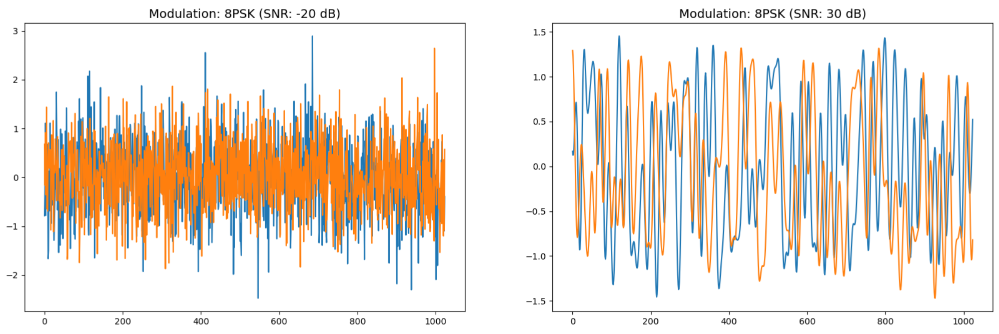


Figure A.6: RF Signal Modulation Class: 8PSK. Data Source: RADIOML 2018.01A [3]

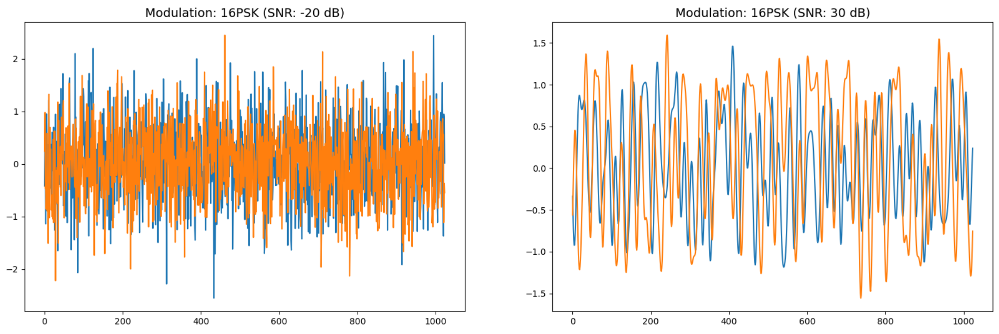


Figure A.7: RF Signal Modulation Class: 16PSK. Data Source: RADIOML 2018.01A [3]

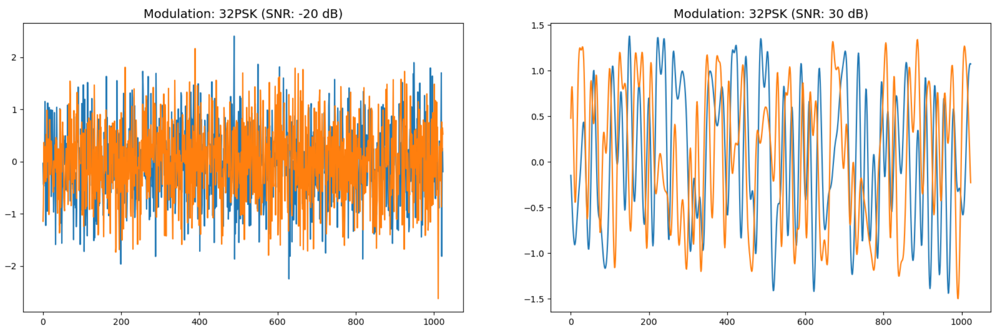


Figure A.8: RF Signal Modulation Class: 32PSK. Data Source: RADIOML 2018.01A [3]

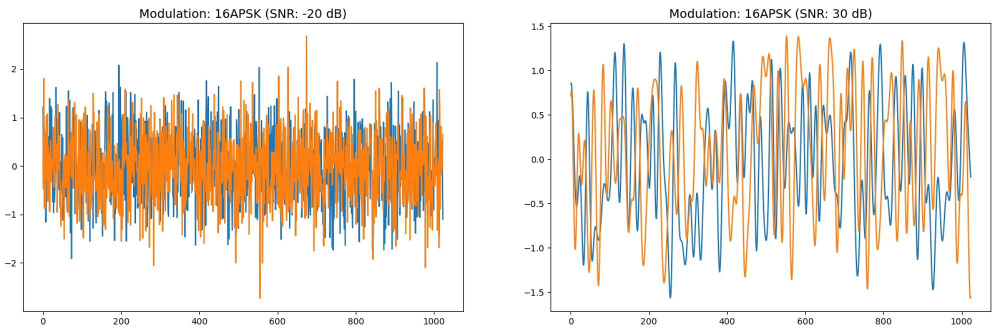


Figure A.9: RF Signal Modulation Class: 16APSK. Data Source: RADIOML 2018.01A [3]

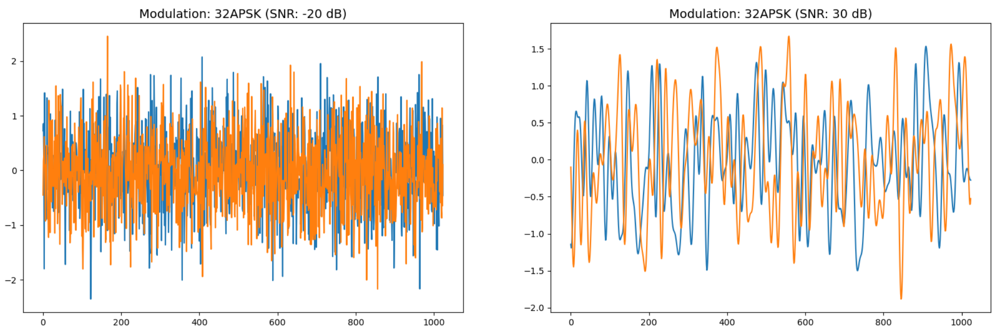


Figure A.10: RF Signal Modulation Class: 32APSK. Data Source: RADIOML 2018.01A [3]

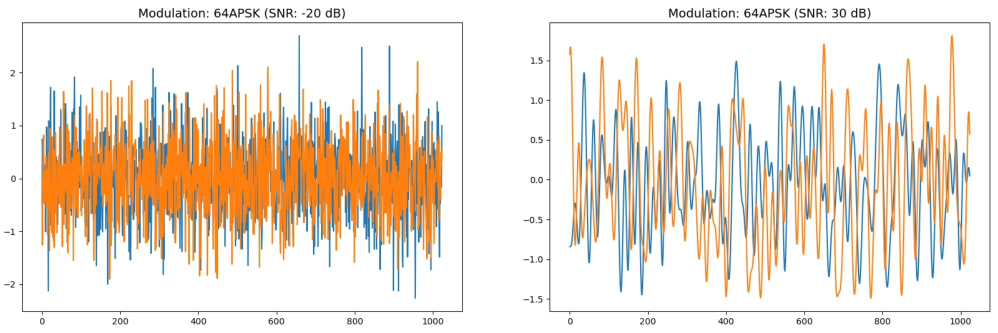


Figure A.11: RF Signal Modulation Class: 64APSK. Data Source: RADIOML 2018.01A [3]

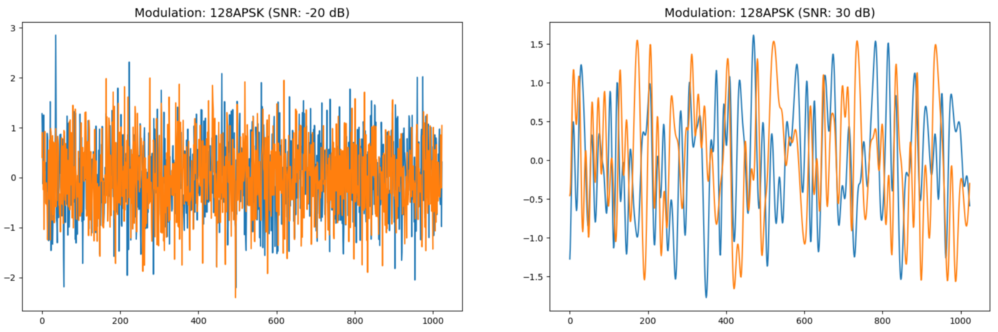


Figure A.12: RF Signal Modulation Class: 128APSK. Data Source: RADIOML 2018.01A [3]

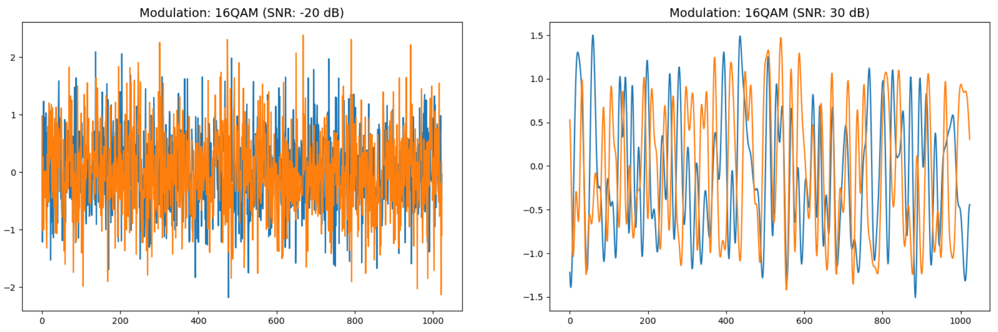


Figure A.13: RF Signal Modulation Class: 16QAM. Data Source: RADIOML 2018.01A [3]

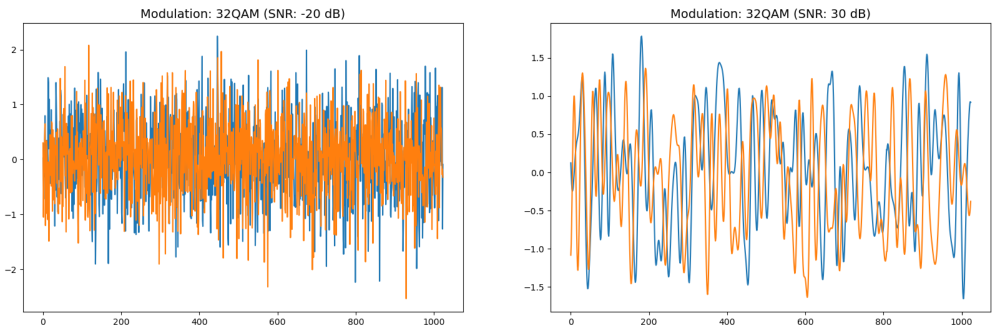


Figure A.14: RF Signal Modulation Class: 32QAM. Data Source: RADIOML 2018.01A [3]

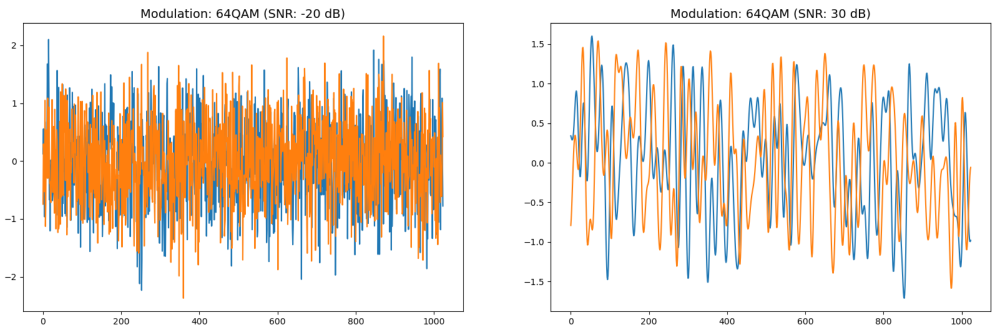


Figure A.15: RF Signal Modulation Class: 64QAM. Data Source: RADIOML 2018.01A [3]

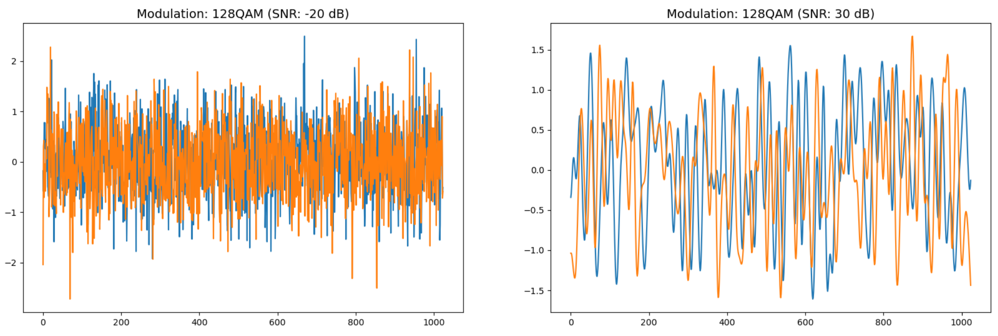


Figure A.16: RF Signal Modulation Class: 128QAM. Data Source: RADIOML 2018.01A [3]

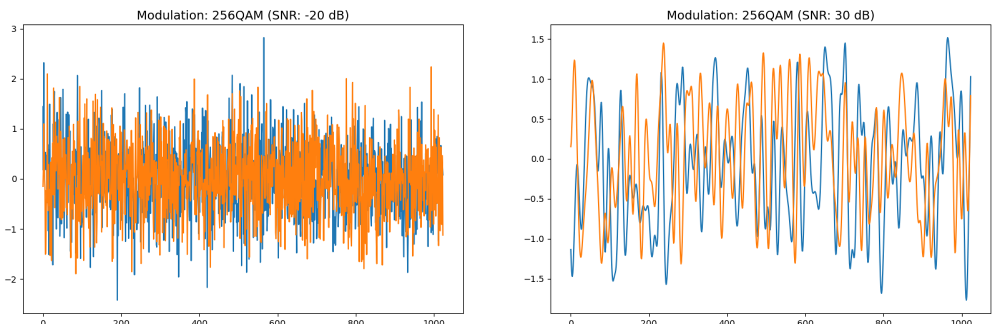


Figure A.17: RF Signal Modulation Class: 256QAM. Data Source: RADIOML 2018.01A [3]

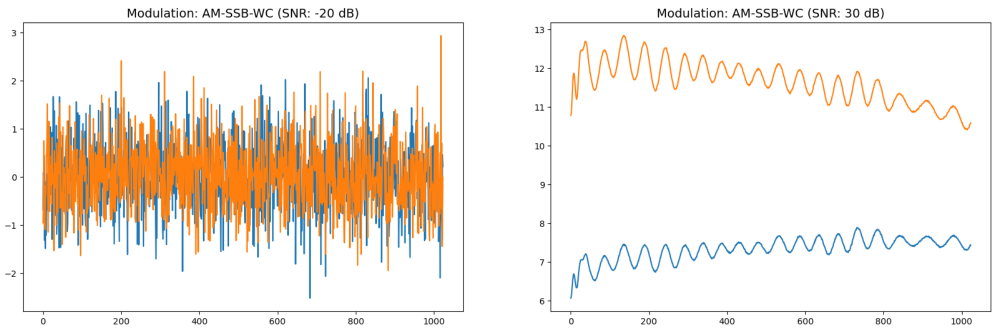


Figure A.18: RF Signal Modulation Class: AM-SSB-WC. Data Source: RADIOML 2018.01A [3]

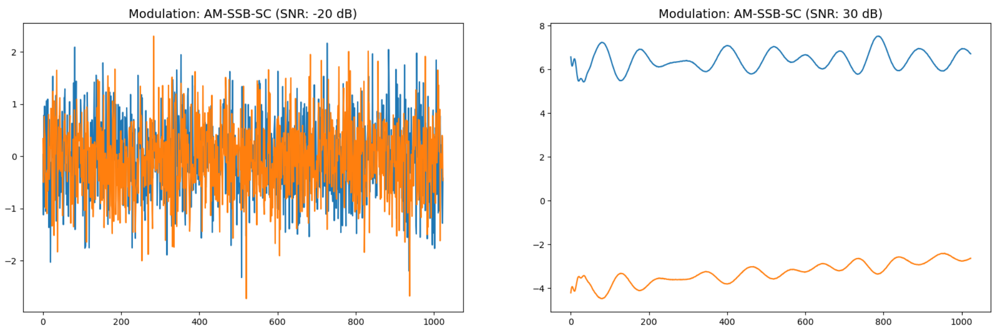


Figure A.19: RF Signal Modulation Class: AM-SSB-SC. Data Source: RADIOML 2018.01A [3]

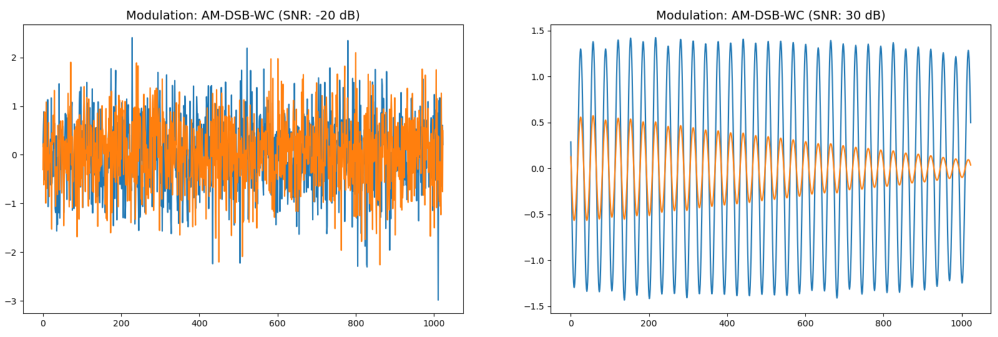


Figure A.20: RF Signal Modulation Class: AM-DSB-WC. Data Source: RADIOML 2018.01A [3]

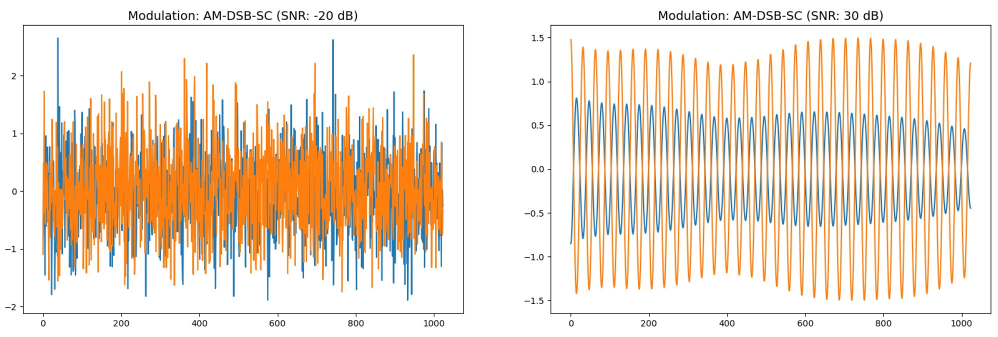


Figure A.21: RF Signal Modulation Class: AM-DSB-SC. Data Source: RADIOML 2018.01A [3]

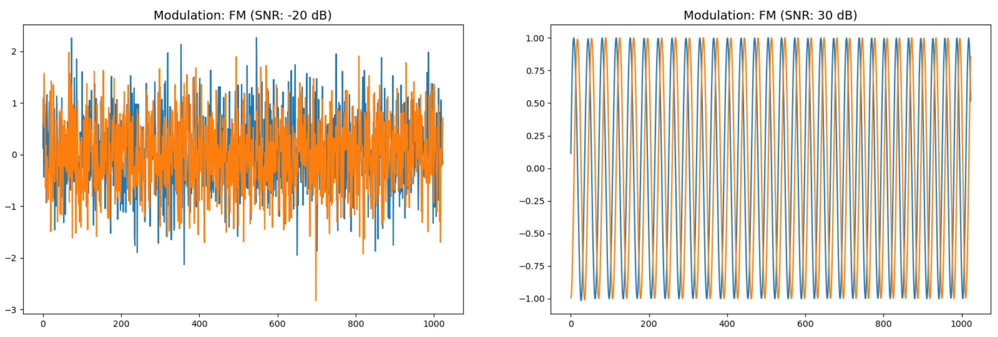


Figure A.22: RF Signal Modulation Class: FM. Data Source: RADIOML 2018.01A [3]

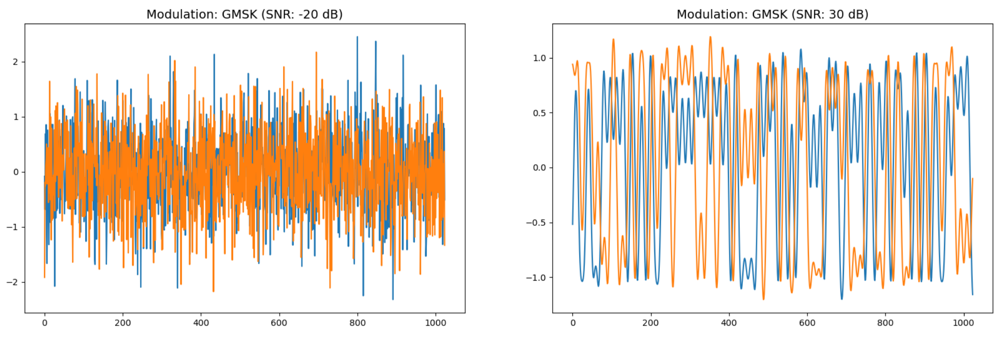


Figure A.23: RF Signal Modulation Class: GMSK. Data Source: RADIOML 2018.01A [3]

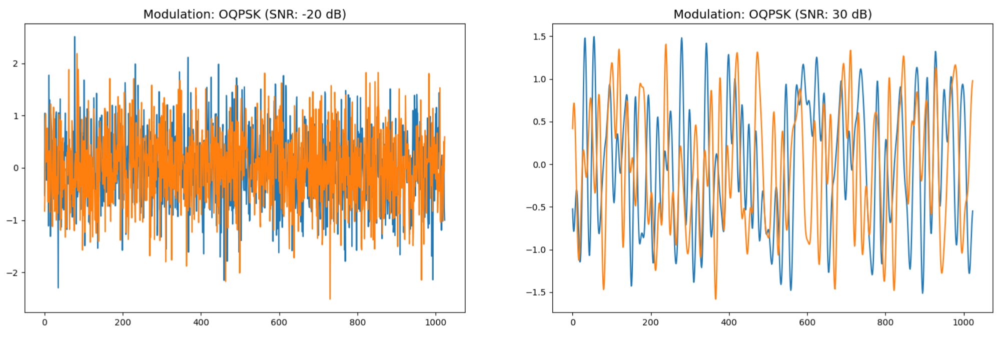


Figure A.24: RF Signal Modulation Class: OQPSK. Data Source: RADIOML 2018.01A [3]

Appendix B

Mathematical Formulae

This appendix contains essential mathematical formulae used to implement the transformer-based architecture discussed in this thesis.

1. **Batch Normalization** [72] is a technique designed to automatically standardize the inputs to a layer in a deep learning model, aiming to stabilize and accelerate training. Given a mini-batch of activations, the method computes the mean and variance of that mini-batch and then normalizes the activations.

$$\mu^B = \frac{1}{m}\sum_{i=1}^{m} x_i^B \tag{B.1}$$

$$\sigma^B = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(x_i^B - \mu^B)^2} \tag{B.2}$$

In Equations B.1 and B.2 μ^B and σ^B are the normalization for a mini-batch of size m. The normalized activation is as follows:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{B.3}$$

In Equation B.3, ϵ is a small constant for numerical stability. The final output of batch normalization is:

$$y_i = \gamma \hat{x}_i + \beta \tag{B.4}$$

In Equations B.4, γ and β are learnable parameters for scale and shift, respectively.

2. **Gaussian Dropout** [73] is a variant of the traditional dropout technique. Instead of setting a random subset of activations to zero (as in the traditional dropout), Gaussian Dropout multiplies the activations by random values sampled from a Gaussian distribution. Let W be the weights of a network and let W' be noisy weights after applying Gaussian Dropout. With a dropout rate of p, the noisy weights are represented as:

$$W' = W \odot \mathcal{N}\left(1, \sqrt{\frac{p}{1-p}}\right) \tag{B.5}$$

In Equation B.5, $\odot$ is element-wise multiplication, $\mathcal{N}\left(1, \sqrt{\frac{p}{1-p}}\right)$ represents a Gaussian distribution and mean of 1 and variance $\sqrt{\frac{p}{1-p}}$.

3. **Gaussian Error Linear Unit (GELU)** [74] activation function is a non-linear activation function, particularly in transformer-based models. GELU function can be viewed as a smooth version of ReLU. ReLU introduces non-linearlity by setting negative values to zero, GELU does this in a more gradual manner providing non-zero outputs for negative values. This helps in more stable training and faster convergence in some architectures. The downside is that it is computationally more intensive than ReLU. GELU is mathematically defined as:

$$\text{GELU}(x) = xP(X \leq x) = \frac{x}{2}\left[1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right] \tag{B.6}$$

An approximation of GELU using *tanh* is:

$$\approx 0.5x\left(1+\tanh\left(\sqrt{\frac{2}{\pi}}(x+0.044715x^3)\right)\right) \tag{B.7}$$

4. **Categorical Cross Entropy** [75] is a loss function used in multi-class classification tasks. The concept is deeply rooted in information theory, especially the definition of entropy. It quantifies the difference between two probability distributions: the true distribution p and the estimated distribution q. For a single data point, the formula for categorical cross entropy is:

$$H(p,q) = -\sum_i p_i \log(q_i) \tag{B.8}$$

In Euqation B.8 p_i is the true probability that the data point belongs to class i (usually 0 or 1 in one-hot encoded labels), q_i is the predicted probability that the data point belongs to class i, and the sum is taken over all classes.

5. **Adam Optimizer**, introduced by Kingma and Ba [76], is one of the most widely used optimization algorithms in DL. The Adam optimizer combines the power of two extensions of stochastic gradient descent: (a) AdaGrad [77], which adapts the learning rates of each parameter by squaring the gradient, and (b) RMSProp (Root Mean Square Propagation), which also adapts learning rates but uses a moving average of the squared gradient. The mathematical representation of Adam Optimizer is as follows:

   (a) Compute the gradient g_t:

$$g_t = \nabla J(\theta) \tag{B.9}$$

(b) Update the moving averages of the gradient m and the squared gradient v:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{B.10}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{B.11}$$

(c) Compute bias-corrected first and second moment estimates:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{B.12}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{B.13}$$

(d) Update the parameters:

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{B.14}$$

Where α is the learning rate. β_1, β_2 are hyperparameters representing the exponential decay rates for the moment estimates. ϵ is a small number to prevent any division by zero.

Appendix C

Transformer-Based Architecture: Training Curves

C.1 Training Curves: TR-AMC (512, 1024, 2048)

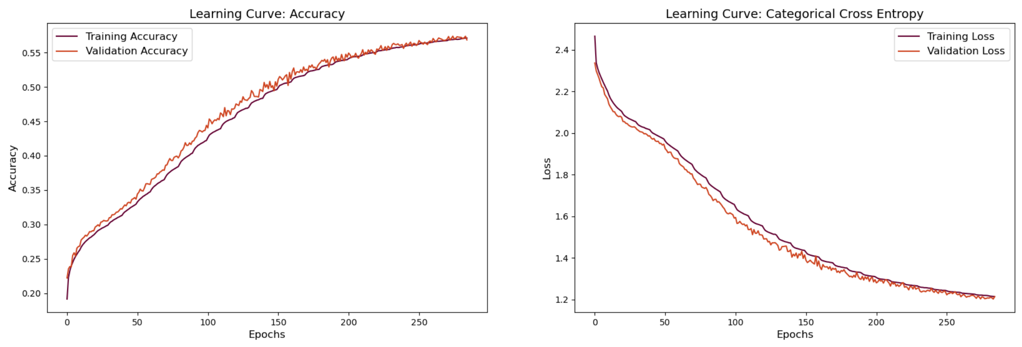


Figure C.1: Training Accuracy Curve and Loss Curve TR-AMC (512)

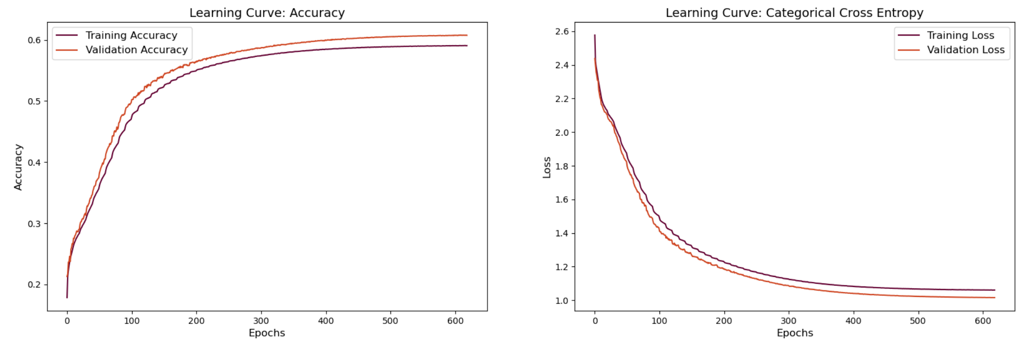


Figure C.2: Training Accuracy Curve and Loss Curve TR-AMC (1024)

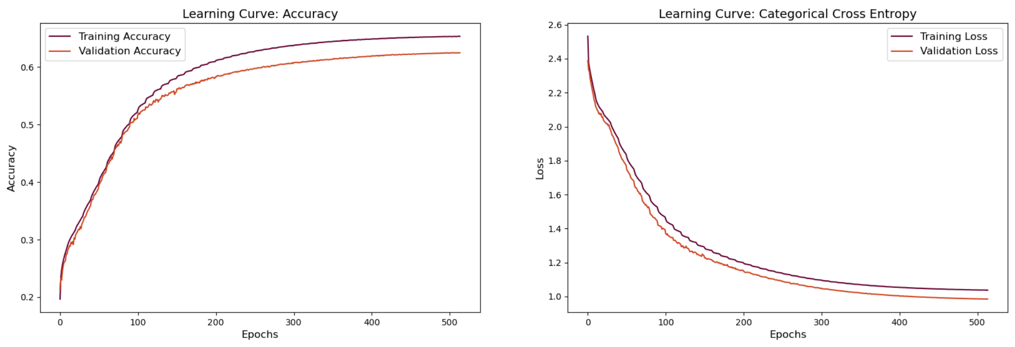


Figure C.3: Training Accuracy Curve and Loss Curve TR-AMC (2048)

C.2 Training Curves: GTR-AMC (1024)

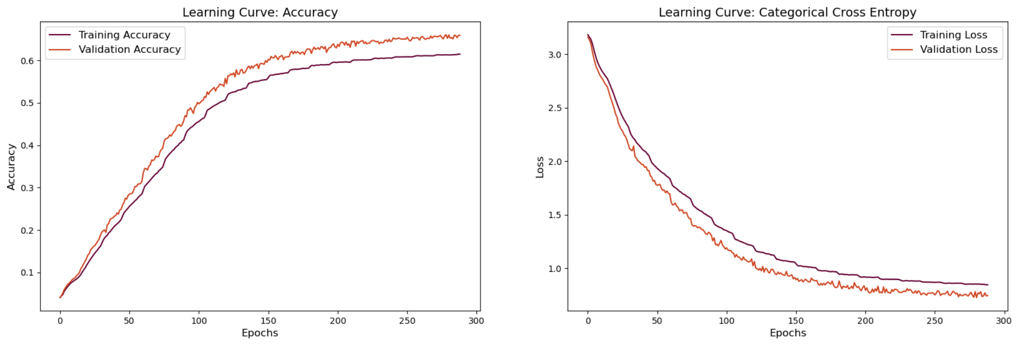


Figure C.4: Training Accuracy Curve and Loss Curve GTR-AMC (1024) Group A

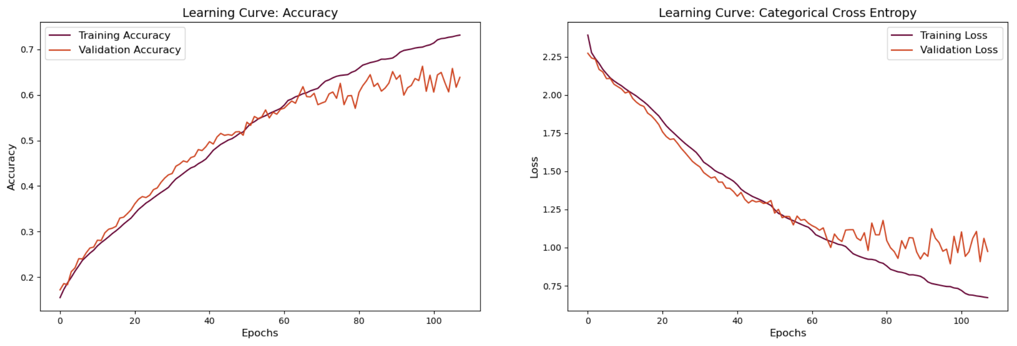


Figure C.5: Training Accuracy Curve and Loss Curve GTR-AMC (1024) Group B

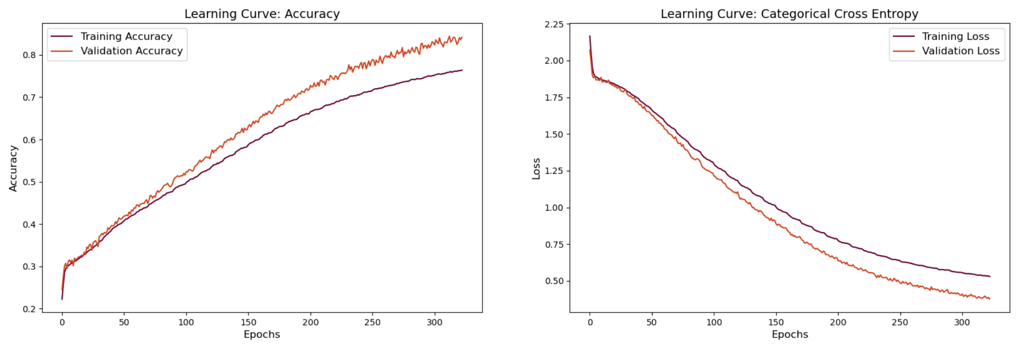


Figure C.6: Training Accuracy Curve and Loss Curve GTR-AMC (1024) Group C

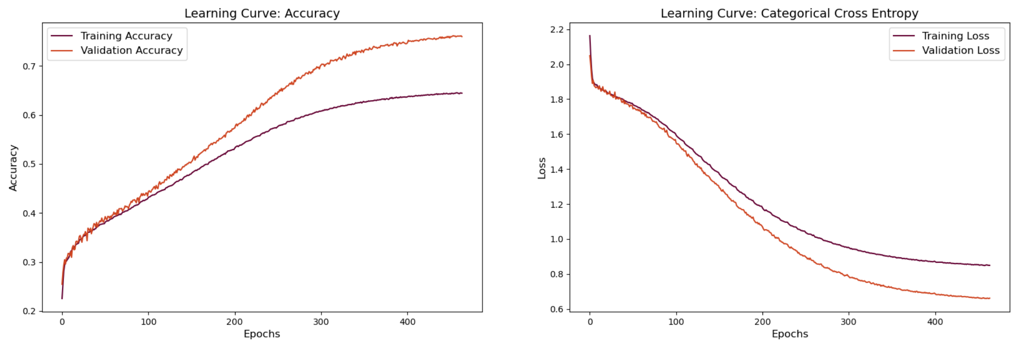


Figure C.7: Training Accuracy Curve and Loss Curve GTR-AMC (1024) Group D

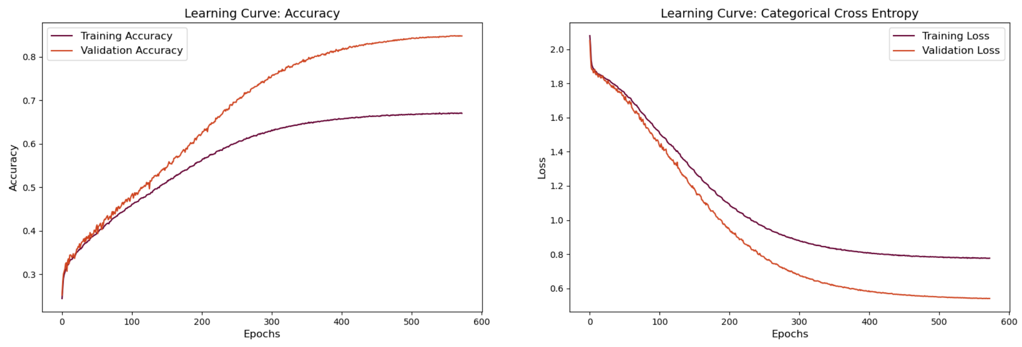


Figure C.8: Training Accuracy Curve and Loss Curve GTR-AMC (1024) Group E

C.3 Training Curves: GTR-AMC (2048)

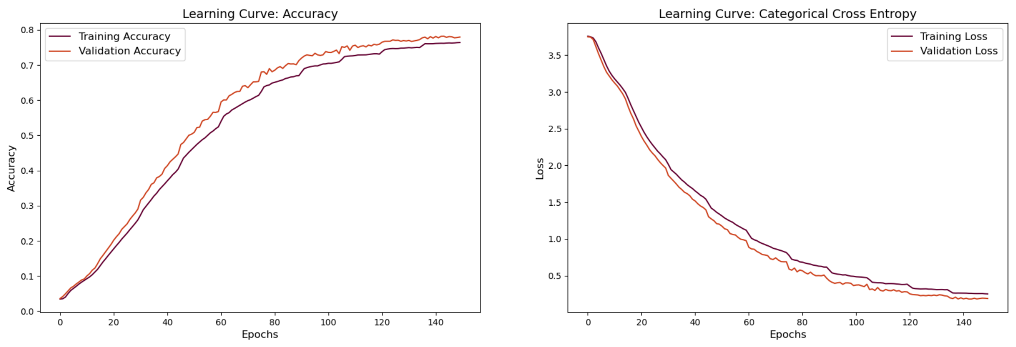


Figure C.9: Training Accuracy Curve and Loss Curve GTR-AMC (2048) Group A

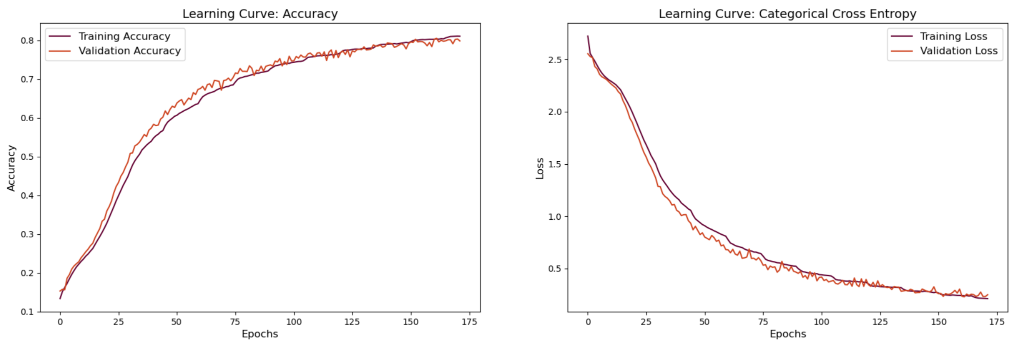


Figure C.10: Training Accuracy Curve and Loss Curve GTR-AMC (2048) Group B

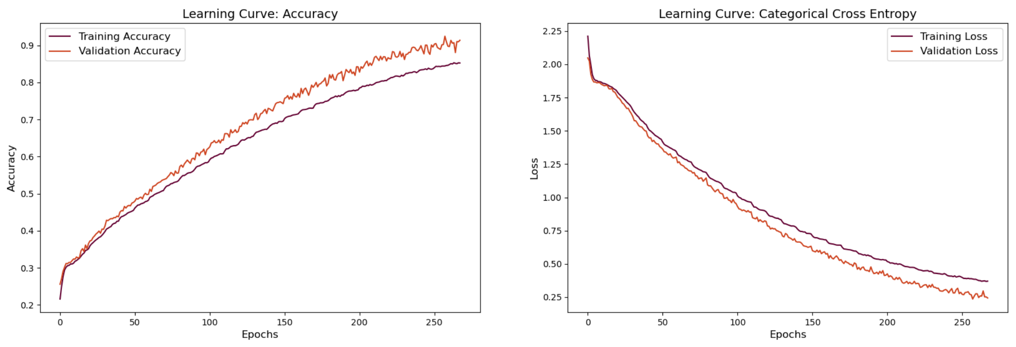


Figure C.11: Training Accuracy Curve and Loss Curve GTR-AMC (2048) Group C

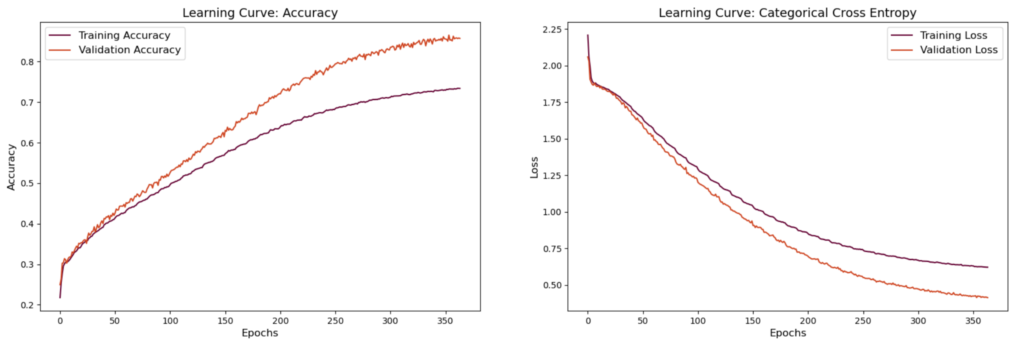


Figure C.12: Training Accuracy Curve and Loss Curve GTR-AMC (2048) Group D

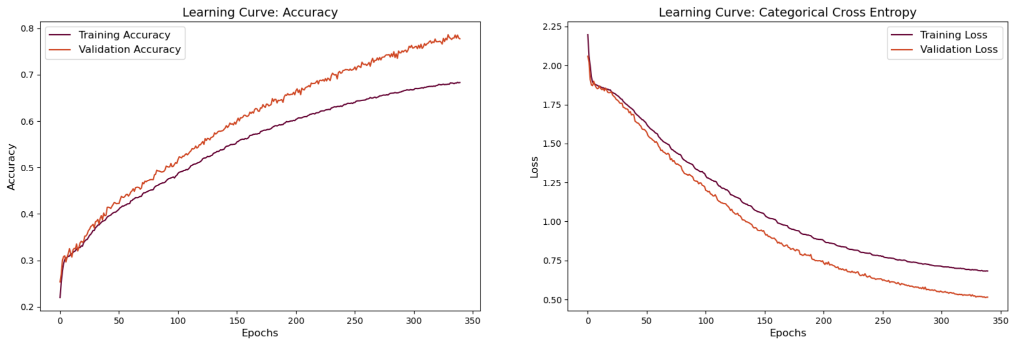


Figure C.13: Training Accuracy Curve and Loss Curve GTR-AMC (2048) Group E

Appendix D

Additional Group B Training Curves for GTR-AMC (Hybrid)

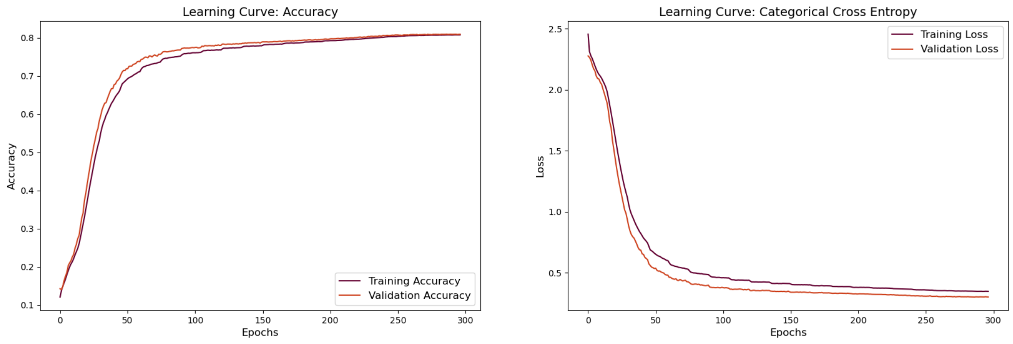


Figure D.1: Training Accuracy Curve and Loss Curve GTR-AMC Group B (3072) for GTR-AMC (Hybrid)

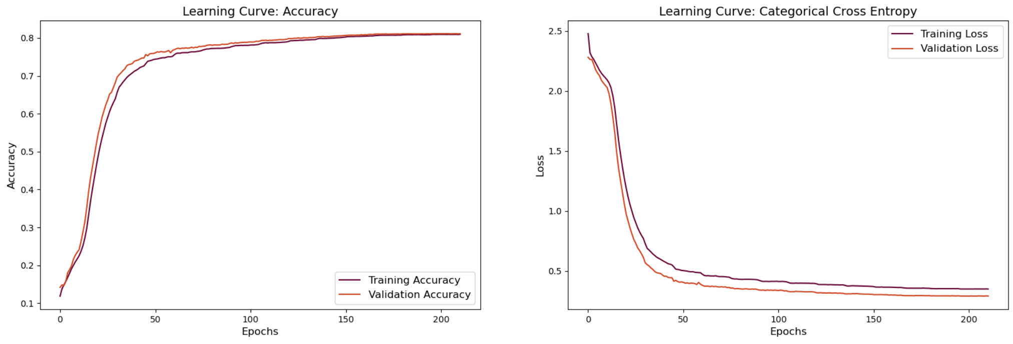


Figure D.2: Training Accuracy Curve and Loss Curve GTR-AMC Group B (4096) for GTR-AMC (Hybrid)

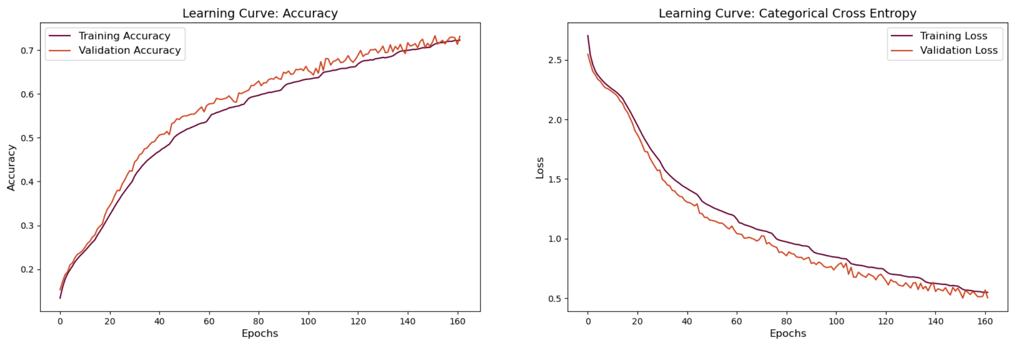


Figure D.3: Training Accuracy Curve and Loss Curve GTR-AMC Extended Group B (2048) for GTR-AMC (Hybrid)

Appendix E

Tables

Table E.1: Classification Accuracy (%) based on SNR: TR-AMC vs. GTR-AMC on RADIOML 2018.01A

SNR	TR-AMC (512)	TR-AMC (1024)	TR-AMC (2048)	GTR-AMC (1024)	GTR-AMC (2048)	GTR-AMC (Hybrid)
-20	16.23	21.47	24.42	67.36	73.23	72.49
-18	15.97	21.16	24.08	67.02	74.78	75.53
-16	16.12	22.34	25.27	68.13	74.06	74.32
-14	16.30	22.75	25.63	68.19	75.03	75.79
-12	16.92	22.46	25.40	68.64	75.95	76.71
-10	18.42	24.19	27.31	69.65	76.83	76.07
-8	23.06	28.37	30.91	74.77	81.90	82.15
-6	29.71	34.79	37.17	73.10	77.33	81.71
-4	34.93	42.63	45.69	68.78	73.41	81.51
-2	43.04	49.35	53.08	62.37	70.43	81.34
0	49.39	57.33	61.76	59.07	62.87	82.10
2	58.88	64.09	71.72	81.34	83.70	83.04
4	64.84	69.44	72.84	80.97	84.02	84.86
6	67.81	72.58	76.19	79.51	83.76	82.09
8	70.93	74.95	77.89	79.27	83.81	82.98
10	73.12	75.31	78.49	80.20	82.51	81.86
12	74.19	75.78	78.53	77.35	82.87	83.70
14	74.20	76.31	79.39	77.15	82.62	81.97
16	74.36	77.48	79.58	76.99	82.84	81.19
18	75.10	77.26	79.43	77.08	82.63	81.81
20	74.12	77.23	79.93	78.78	85.82	84.11
22	75.13	77.10	80.26	81.71	84.87	85.72
24	74.23	77.18	80.45	82.11	84.86	85.71
26	74.87	78.83	82.36	81.47	84.66	84.97
28	74.58	77.53	81.84	81.76	84.85	84.16
30	74.59	78.54	81.45	81.96	85.42	84.57
Overall	52.35	56.79	60.04	74.80	79.81	81.25

Table E.2: Classification Accuracy (%) based on Modulation: TR-AMC vs. GTR-AMC on RADIOML 2018.01A

Modulation Types	TR-AMC (512)	TR-AMC (1024)	TR-AMC (2048)	GTR-AMC (1024)	GTR-AMC (2048)	GTR-AMC (Hybrid)
OOK	80.38	81.03	81.95	77.62	70.02	80.27
4ASK	82.71	84.11	83.65	56.66	56.33	76.78
8ASK	82.99	85.29	85.58	72.90	52.79	77.39
BPSK	83.11	86.19	85.13	68.13	91.14	90.20
QPSK	42.72	45.65	57.39	73.31	92.83	89.27
8PSK	37.65	42.05	21.61	72.38	86.34	83.29
16PSK	34.73	40.14	54.05	88.71	83.83	84.52
32PSK	42.69	41.58	50.55	88.99	86.77	86.93
16APSK	36.73	41.41	52.97	76.11	91.60	82.49
32APSK	27.87	57.82	56.67	57.88	81.22	78.31
64APSK	24.73	38.78	38.01	46.58	81.93	75.81
128APSK	41.58	39.69	52.89	69.62	77.64	82.67
16QAM	34.27	45.22	54.12	79.87	76.25	72.63
32QAM	38.55	39.25	38.47	67.97	79.84	77.82
64QAM	22.33	40.88	53.01	87.19	84.06	85.21
128QAM	24.97	37.07	26.52	59.15	92.77	89.72
256QAM	28.47	42.20	60.96	80.54	82.02	85.23
AM-SSB-WC	81.31	84.84	83.14	79.40	89.13	87.43
AM-SSB-SC	80.07	81.15	79.53	73.72	62.33	82.27
AM-DSB-WC	81.65	84.27	85.52	60.74	42.73	75.83
AM-DSB-SC	75.41	76.06	67.28	78.53	84.80	68.55
FM	83.24	86.72	84.99	90.68	91.18	92.02
GMSK	68.27	71.28	69.92	74.48	89.45	90.49
OQPSK	38.08	42.89	47.76	50.23	78.26	80.28