ANKARA YILDIRIM BEYAZIT UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES



COMPARATIVE STUDY OF NOVEL DEEP NEURAL NETWORK ARCHITECTURE DESIGNS FOR FASTER-THAN-NYQUIST SIGNALING DETECTION

Ph.D. Thesis by

Osman TOKLUOĞLU

Department of Electrical and Electronics Engineering

December, 2025

ANKARA

COMPARATIVE STUDY OF NOVEL DEEP NEURAL NETWORK ARCHITECTURE DESIGNS FOR FASTER-THAN-NYQUIST SIGNALING DETECTION

A Thesis Submitted to the

Graduate School of Natural And Applied Sciences of

Ankara Yildirim Beyazit University

In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Electrical and Electronics Engineering, Department of Electrical and Electronics Engineering

by

Osman TOKLUOĞLU

December, 2025

ANKARA

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled "**COMPARATIVE STUDY OF NOVEL DEEP NEURAL NETWORK ARCHITECTURE DESIGNS FOR FASTER-THAN-NYQUIST SIGNALING DETECTION**" completed by **OSMAN TOKLUOĞLU** under supervision of **PROF. DR. ENVER ÇAVUŞ** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Enver ÇAVUŞ

Supervisor

Prof. Dr. Asaf Behzat ŞAHİN

Thesis Committee Member

Prof. Dr. Bülent TAVLI

Thesis Committee Member

Prof. Dr. Cengiz ÖZZAİM

Examining Committee Member

Prof. Dr. Ali KARA

Examining Committee Member

Prof. Dr. İlyas ÇANKAYA

Director

Graduate School of Natural and Applied Sciences

ETHICAL DECLARATION

I hereby declare that, in this thesis which has been prepared in accordance with the Thesis Writing Manual of Graduate School of Natural and Applied Sciences,

- All data, information and documents are obtained in the framework of academic and ethical rules,
- All information, documents and assessments are presented in accordance with scientific ethics and morals,
- All the materials that have been utilized are fully cited and referenced,
- No change has been made on the utilized materials,
- All the works presented are original,

and in any contrary case of above statements, I accept to renounce all my legal rights.

Date: 2025, 20 December **Signature:** _____

Name & Surname: Osman TOKLUOĞLU

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisor, Prof. Enver Çavuş, for his invaluable guidance, encouragement, and continuous support throughout the course of this research.

I would also like to extend my sincere thanks to my co-advisors, Prof. Asaf Behzat Şahin and Prof. Bülent Tavlı, for their insightful suggestions and constructive feedback that greatly improved the quality of this work.

I am especially grateful to Prof. Ebrahim Bedeer from the University of Saskatchewan and Prof. Halim Yanıkömeroğlu from Carleton University for their academic collaboration and mental support during the research process.

Finally, I would like to express my heartfelt appreciation to my family and, in particular, to my beloved wife, Merve, for her unwavering patience, understanding, and continuous support throughout this journey.

December, 2025 **Osman TOKLUOĞLU**

COMPARATIVE STUDY OF NOVEL DEEP NEURAL NETWORK ARCHITECTURE DESIGNS FOR FASTER-THAN-NYQUIST SIGNALING DETECTION

ABSTRACT

Faster-than-Nyquist (FTN) signaling has emerged as a promising technique to enhance spectral efficiency beyond the conventional Nyquist limit. However, the intentional introduction of inter-symbol interference (ISI) in FTN systems makes optimal detection computationally demanding. This thesis investigates a series of novel deep learning-based detector architectures developed to effectively address this challenge. Distinct neural network models, including convolutional and recurrent structures, are designed with domain-informed principles to exploit the inherent temporal dependencies within FTN signaling. Each proposed model is trained and evaluated over various compression factors and modulation schemes to assess its robustness, scalability, and complexity. A comprehensive comparison is performed against classical detection algorithms to quantify the trade-off between performance and computational cost. The results demonstrate that the proposed deep neural network architectures can achieve near-optimal detection accuracy while substantially reducing implementation complexity, thereby offering a practical and efficient alternative to model-based approaches. This work contributes to the growing research interest in AI-driven FTN detection and provides valuable insights into architecture selection and design strategies for future communication systems.

Keywords: Faster-than-Nyquist signaling, deep learning, convolutional neural network, gated recurrent unit, attention mechanism, inter-symbol interference, signal detection.

YENİ DERİN SİNİR AĞI MİMARİ TASARIMLARININ FASTER-THAN-NYQUIST İŞARET ALGILAMA İÇİN KARŞILAŞTIRMALI İNCELENMESİ

ÖZ

Faster-than-Nyquist (FTN) işaretleme, geleneksel Nyquist sınırının ötesine geçerek spektral verimliliği artırmayı amaçlayan umut verici bir yöntem olarak öne çıkmaktadır. Ancak, FTN sistemlerinde kasıtlı olarak oluşturulan semboller arası girişim (ISI), algılama sürecini oldukça karmaşık hale getirmektedir. Bu tez çalışmasında, söz konusu zorluğu etkin bir biçimde ele almak amacıyla geliştirilen yeni derin öğrenme tabanlı algılayıcı mimariler sunulmuştur. Çalışma kapsamında, FTN işaretlerinin zamansal bağımlılıklarını en verimli şekilde kullanmak üzere tasarlanmış evrişimsel ve tekrarlayan sinir ağı yapıları dâhil olmak üzere çeşitli derin sinir ağı modelleri önerilmiştir. Önerilen her bir model, farklı sıkıştırma faktörleri ve modülasyon türleri altında eğitilmiş ve değerlendirilmiştir. Bu sayede modellerin dayanıklılığı, ölçeklenebilirliği ve hesaplama karmaşıklıkları karşılaştırmalı olarak incelenmiştir. Klasik algılama algoritmalarıyla yapılan kapsamlı karşılaştırmalar, önerilen mimarilerin benzer algılama doğruluğunu çok daha düşük hesaplama maliyetiyle elde edebildiğini göstermiştir. Elde edilen sonuçlar, yapay zekâ destekli FTN algılama yöntemlerine yönelik artan araştırma ilgisine katkı sağlamakta ve gelecekteki haberleşme sistemleri için mimari tasarım stratejileri konusunda önemli çıkarımlar sunmaktadır.

Anahtar kelimeler: Faster-than-Nyquist işaretleme, derin öğrenme, evrişimsel sinir ağı, kapılı tekrarlayan birim, dikkat mekanizması, semboller arası girişim, işaret algılama.

CONTENTS

Ph.D. THESIS EXAMINATION RESULT FORM ii
ETHICAL DECLARATION iii
ACKNOWLEDGEMENTS iv
ABSTRACT v
ÖZ vi
NOMENCLATURE ix
LIST OF TABLES xii
LIST OF FIGURES xvi

CHAPTER 1 – INTRODUCTION 1

1.1 Background and Motivation 1
1.2 Problem Definition 4
1.3 Contributions of This Thesis 5
1.4 Thesis Organization 6

CHAPTER 2 – SYSTEM MODEL FOR FASTER-THAN-NYQUIST SIGNALING 7

2.1 FTN Transmitter 7
2.2 Matched Filter Receiver 8
2.3 Mathematical Representation of ISI 8
2.4 Channel Models 11
2.5 Performance Metrics 12

CHAPTER 3 – CNN-BASED FTN SIGNALING DETECTION WITH FIXED KERNEL LAYERS 14

3.1 Dataset Preparation 15
3.2 Data Arrangement 16
3.3 CNN-Based FTN Detector System Model 17
3.4 Complexity Analysis for Structured Fixed Kernel Layers 22
3.5 Learning Rate Strategy for Training 24

3.6 Simulation Results and Complexity Comparison 26
3.6.1 BER Results 27
3.6.2 Computational Complexity Comparison 36
3.7 Summary of the Chapter 43

CHAPTER 4 – GRU-BASED FTN DETECTOR 44

4.1 Dataset Preparation 44
4.2 Data Arrangement 45
4.3 GRU-Based FTN Detector System Model 47
4.4 Comparative Analysis of Training Optimization Techniques 52
4.5 Simulation Results and Complexity Comprasion 55
4.5.1 BER Results 56
4.5.2 Computational Complexity Comparison 64
4.6 Summary of the Chapter 68

CHAPTER 5 – BI-GRU-BASED FTN SIGNALING DETECTION WITH POSITION-AWARE ATTENTION MECHANISM 69

5.1 Dataset Generation and Training Hyperparameters 70
5.2 Data Preprocessing and Input Formatting 71
5.3 Bi-GRU-Based Sequence Modeling 72
5.4 Position-Aware Attention Mechanism 73
5.5 Simulation Results and Computational Complexity Comparison 76
5.6 Summary of the Chapter 80

CHAPTER 6 – COMPARATIVE ANALYSIS AND DISCUSSION 81

6.1 Overview of Developed FTN Detection Architectures 81
6.2 Unified Performance Comparison Across All Proposed Detectors 82

CHAPTER 7 – CONCLUSION AND FUTURE RESEARCH DIRECTIONS .. 88

REFERENCES 91

NOMENCLATURE

Roman Letter Symbols

$\hat{a}_k$	Detected (estimated) symbol for index k
$\mathbf{c}$	Context vector from PAA mechanism
$\mathbf{H}$	Bi-GRU hidden state matrix
$\mathbf{h}_t$	GRU hidden state at time step t
$\mathbf{W}, \mathbf{b}$	Weights and bias of fully connected layers
$\mathbf{Y}_k$	Input vector of length $2N+1$ centered at y_k
a_k	Transmitted symbol at time index k
B	Training batch size
K	Total number of transmitted symbols
L	Sequence length ($L = 2N+1$)
T_{att}	Attention temperature parameter
y_k	Matched-filter output sampled at $k\tau T$

Greek Letter Symbols

α_i	Attention coefficient for symbol position i
β	RRC roll-off factor
$\phi(\cdot)$	Output activation function (sigmoid or softmax)
$\sigma(\cdot)$	Sigmoid activation function
τ	FTN acceleration (time-compression) factor
θ	Neural network trainable parameter set

Subscripts

AWGN	Additive White Gaussian Noise
BCJR	Bahl–Cocke–Jelinek–Raviv algorithm
Bi-GRU	Bidirectional Gated Recurrent Unit
CNN	Convolutional Neural Network
GRU	Gated Recurrent Unit
ISI	Inter-Symbol Interference
LDPC	Low-Density Parity-Check
M-BCJR	Reduced-state BCJR (M-BCJR) algorithm
PAA	Position-Aware Attention

RRC Root-Raised Cosine

SNR Signal-to-Noise Ratio

Acronyms

$\mathbf{X}$ Toeplitz matrix of ISI coefficients

L Total number of symbols in the ISI window

N One-sided ISI span

T Original Nyquist symbol interval

$T_{\text{eff}} = \tau T$ Compressed FTN symbol interval

$w_{p,i}$ Trainable PAA weight for position i

x_i One-sided ISI coefficient at offset i

LIST OF TABLES

Table 2.1 One-sided ISI coefficients for $\tau = 0.7$, 0.8, and 0.9. 11

Table 3.1 Data structure for $N = 6$. Each row represents one CNN input vector; the bold element indicates the center sample corresponding to the target symbol.. 17

Table 3.2 Number of filters for each fixed kernel layer under different τ values in the proposed CNN detector. 21

Table 3.3 Training hyperparameters of the proposed CNN-based FTN detector for different τ values. 25

Table 3.4 Comparison of Operation Numbers for M-BCJR and CNN-Based FTN Detector (100 Symbols) 40

Table 3.5 LUT Utilizations for Different Operations (10-bit) 40

Table 3.6 LUT-Weighted Computational Cost Metrics for M-BCJR and CNN-Based FTN Detector (BPSK and QPSK Modulations) 41

Table 3.7 Percentage Improvement of CNN over M-BCJR in LUT-Weighted Computational Cost Metric 42

Table 3.8 Comparison of Operation Numbers for Go-Back-K and CNN-Based FTN Detector (100 Symbols, BPSK, $\tau = 0.8$) 42

Table 4.1 Data Structure for $N = 6$ 47

Table 4.2 Convergence Metrics for GRU-Based FTN Detectors Optimized with NADAM and ADAM Optimizers 54

Table 4.3 Summary of Network and Training Hyperparameters 57

Table 4.4 Comparison of Operation Numbers for M-BCJR and GRU-Based FTN Detector (100 Symbols) for BPSK Modulation 65

Table 4.5 Comparison of Operation Numbers for M-BCJR and GRU-Based FTN Detector (100 Symbols) for QPSK Modulation 65

Table 4.6 LUT Utilizations for Different Operations (10-bit) 66

Table 4.7 LUT-Weighted Computational Cost Metrics for M-BCJR and GRU-Based FTN Detector (BPSK and QPSK Modulations) 67

Table 4.8 Percentage Improvement of GRU over M-BCJR in LUT-Weighted Computational Cost Metric 67

Table 5.1 Training Parameters for the Proposed Bi-GRU with PAA Detector 71

Table 5.2 Unified Operation Counts and LUT-Weighted Cost for M-BCJR vs. Bi-GRU with PAA at $\tau = 0.5$ 79

Table 6.1 Summary of Proposed FTN Detection Architectures 82

LIST OF FIGURES

Figure 2.1 Faster-than-Nyquist signaling structure and interference coefficients for a sequence of five BPSK symbols with one-sided ISI length $N = 2$. 9

Figure 3.1 FTN signaling-based communication system with the proposed CNN-based detector. 15

Figure 3.2 Illustration of conventional kernelization techniques with varying kernel sizes ($K = 5$, $N = 2$): (a) 5, (b) 4, (c) 3, and (d) 2. 19

Figure 3.3 Illustration of structured fixed kernel layers with domain-informed masking: (a) Layer 1 captures symbols one unit away from the center, (b) Layer 2 captures those two units away, and so on up to (f) Layer 6.....20

Figure 3.4 Illustration of fixed kernel layers applied with two different filters for $N = 2$, demonstrating diverse feature extraction at each distance.....21

Figure 3.5 Convolutional neural network (CNN)-based faster-than-Nyquist (FTN) signaling detector architecture for BPSK and QPSK modulations. 22

Figure 3.6 Structured fixed kernel layers for a sequence length of 5 ($N = 2$), designed to capture ISI effects at specific distances from the central symbol.....23

Figure 3.7 BER comparison between the proposed structured fixed-kernel CNN detector (CNN-FK3) and conventional convolutional CNNs (CNN-K2, CNN-K3, CNN-K4, CNN-K5) for $\tau = 0.9$ and $\beta = 0.35$ over an AWGN channel. 28

Figure 3.8 BER comparison between the proposed CNN-based FTN detector and the optimal BCJR algorithm for $\tau = 0.8$ and $\tau = 0.7$ with $\beta = 0.35$ over an AWGN channel. 29

Figure 3.9 BER comparison of the proposed CNN-based FTN detector and the Go-Back-K equalizer [1] for $\tau = 0.9$, $\tau = 0.8$, and $\tau = 0.7$ under an AWGN channel with $\beta = 0.35$. 30

Figure 3.10 Comparison of the proposed CNN detector with the FTN detection techniques presented in [2], [3], and [4] for $\tau = 0.7$ and $\beta = 0.5$ over an AWGN channel. 31

Figure 3.11 BER performance of the proposed CNN-based FTN detector for QPSK, 16-QAM, and 64-QAM with $\tau = 0.8$ and $\beta = 0.5$ under an AWGN channel. ...32

Figure 3.12 BER performance of the proposed CNN-FK3 detector over a quasi-static multipath Rayleigh fading channel for $\tau = 0.7$, $\tau = 0.8$, and $\tau = 0.9$ using QPSK modulation. The corresponding BCJR benchmark curves are included for reference. 33

Figure 3.13 BER performance comparison of LDPC-coded FTN systems employing BCJR-based and CNN-FK3-based detectors for $\tau = 0.7$, $\tau = 0.8$, and $\tau = 0.9$. A rate-1/2 LDPC code of length 1056 bits is used with BPSK modulation over an AWGN channel. 35

Figure 3.14 Spectral efficiency for BPSK, QPSK, 16-QAM, and 64-QAM under FTN signaling with $\tau = 0.7$, $\tau = 0.8$, $\tau = 0.9$, and the Nyquist case ($\tau = 1$). Roll-off factor $\beta = 0.35$ is used for BPSK/QPSK and $\beta = 0.5$ for 16-QAM/64-QAM. .. 37

Figure 4.1 FTN signaling-based communication system incorporating the proposed GRU-based detector. 46

Figure 4.2 Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures..... 48

Figure 4.3 Gated Recurrent Unit (GRU)-based faster-than-Nyquist (FTN) signaling detector architecture for binary phase shift keying (BPSK) and quadrature phase shift keying (QPSK) modulations. 51

Figure 4.4 BER performance comparison between the proposed NADAM optimized GRU-based FTN detector and BCJR algorithm for different compression factors ($\tau = 0.9$, $\tau = 0.8$, and $\tau = 0.7$), with a roll-off factor of $\beta = 0.35$..... 58

Figure 4.5 BER performance comparison between the proposed NADAM optimized GRU-based FTN detector and LSTM-based FTN detector for different compression factors ($\tau = 0.9$, $\tau = 0.8$, and $\tau = 0.7$), with a roll-off factor of $\beta = 0.35$..... 59

Figure 4.6 Comparison of the BER performance of the proposed NADAM Optimized GRU-based FTN detector with the methods in [2], [3], [4], [5], and [6] for $\tau = 0.7$ and $\beta = 0.5$. 60

Figure 4.7 BER performance analysis of the GRU-based FTN detector for varying values of N at different τ values..... 61

Figure 4.8 BER performance of the proposed GRU-based detector under quasi-static multipath Rayleigh fading channel for $\tau = 0.7$, $\tau = 0.8$, and $\tau = 0.9$ with QPSK modulation. The benchmark BCJR results are also included for comparison. . 62

Figure 4.9 BER performance comparison of LDPC-coded FTN systems using BCJR-based and GRU-based detectors for $\tau = 0.7$, $\tau = 0.8$, and $\tau = 0.9$. All configurations use a rate-1/2 LDPC code of length 1056 bits under BPSK modulation over an AWGN channel. 63

Figure 5.1 FTN signaling-based communication system with proposed signal detector. 70

Figure 5.2 Bi-GRU-based faster-than-Nyquist (FTN) signaling detector architecture with Position-Aware Attention mechanism. 70

Figure 5.3 BER performance comparison of the Bi-GRU detector with and without the proposed Position-Aware Attention (PAA) mechanism, evaluated for $\tau = 0.5$, 0.6, and 0.7. The benchmark BCJR algorithm and no-ISI condition are included for reference. 77

Figure 5.4 BER performance comparison between the proposed Bi-GRU with PAA detector and existing FTN detection methods from the literature, evaluated for $\tau = 0.6$. The benchmark BCJR algorithm and no-ISI condition are included for reference. 78

Figure 5.5 BER performance comparison of the proposed Bi-GRU with PAA detector and the BCJR algorithm over a quasi-static Rayleigh fading channel for $\tau \geq 0.6$. 79

Figure 6.1 Unified BER comparison of the proposed CNN-FK, GRU-NADAM, and Bi-GRU+PAA detectors for $\tau = 0.7$ under an AWGN channel with roll-off factor $\beta = 0.35$. All models achieve near-optimal performance, closely matching the BCJR benchmark. 83

Figure 6.2 Unified BER comparison of the CNN-FK, GRU-NADAM, and Bi-GRU+PAA detectors for $\tau = 0.6$ under an AWGN channel with $\beta = 0.35$. As ISI severity increases, the CNN-based model deviates from the BCJR benchmark, while the Bi-GRU+PAA detector remains closely aligned across all SNR values. 84

Figure 6.3 BER comparison of the CNN-FK, GRU-NADAM, and Bi-GRU+PAA detectors for $\tau = 0.5$ under an AWGN channel with $\beta = 0.35$. Severe ISI causes significant degradation in CNN- and GRU-based detectors, whereas the proposed Bi-GRU+PAA model continues to match the optimal BCJR performance. 85

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

The rapid growth of wireless communication systems, driven by emerging applications such as massive Internet of Things (IoT), machine-type communications, and high-throughput mobile broadband services, has intensified the demand for high spectral efficiency. Since the frequency spectrum is a scarce and expensive resource, communication systems must employ waveform and receiver designs that maximize the utilization of available bandwidth.

One promising technique addressing this challenge is faster-than-Nyquist (FTN) signaling, introduced by Mazo [7]. Unlike conventional Nyquist signaling, which eliminates intersymbol interference (ISI) through orthogonal pulse design, FTN intentionally violates the Nyquist criterion by transmitting symbols at an accelerated rate using a compression factor $\tau < 1$. This deliberate acceleration increases the achievable data rate without expanding the occupied bandwidth, thereby improving spectral efficiency.

However, intentional ISI significantly complicates the symbol detection task. As τ decreases, the ISI span extends across many neighboring symbols, making optimal sequence detection computationally prohibitive. Classical approaches such as the BCJR algorithm deliver near-optimal performance but demand exponential complexity with respect to the ISI length and modulation order [8]. This complexity makes them impractical for real-time systems or hardware-constrained applications.

In response to the increasing demand for low-complexity FTN detection, several studies have introduced methods to mitigate the severe ISI inherent to faster-than-Nyquist signaling. Prlja and Anderson proposed a reduced-complexity M-BCJR algorithm tailored for turbo equalization in FTN systems [1]. Their technique effectively addressed the ISI produced by the time-compression factor τ across various modulation formats and AWGN channels, achieving BER performance

close to the full BCJR algorithm while lowering computational load through minimum-phase preprocessing. Nevertheless, the M-BCJR algorithm still suffers from exponential complexity growth with the modulation order, making it less suitable for higher-order modulations or real-time implementations. Sugiura et al. later introduced a frequency-domain equalization (FDE) scheme aimed at reducing time-domain detection complexity [2]. By incorporating cyclic prefixes and MMSE-based equalization, their method attained near-optimal BER performance for relatively high τ values under a roll-off factor of $\beta = 0.5$. However, this reliance on $\beta = 0.5$ restricts its general applicability across different roll-off factors, requires additional bandwidth, and reduces spectral efficiency due to the use of cyclic prefixes.

For low-complexity detection within the Mazo limit, Bedeer et al. proposed the successive symbol-by-symbol sequence estimation (SSSSE) method in [5], achieving competitive BER performance with further enhancements through a go-back-K strategy. Yet, its performance degrades at lower τ values (e.g., $\tau = 0.7$), where ISI becomes more dominant. Li et al. later presented a simplified M-BCJR detector based on the Ungerboeck observation model that incorporates future symbol information to improve spectral efficiency and SNR for $\tau = 0.8$ [9]. While effective for BPSK, the method inherits the exponential complexity increase associated with higher-order modulations, limiting its practicality for QPSK and beyond.

In addition to receiver-side algorithms, transmitter-side precoding has been widely investigated as an alternative means of mitigating ISI in FTN signaling. Early work by Wang and Lee [10] focused on maximizing the minimum Euclidean distance between FTN symbol sequences. Rusek and Anderson [11] later extended this concept to linear precoding for nonbinary FTN systems. Subsequent studies explored various precoding and pre-equalization methods, including Tomlinson–Harashima precoding, linear spectral precoding [12], and spatio-temporal symbol-level precoding for multiuser MISO systems [13]. SVD-based precoding with optimal and truncated power allocation was proposed in [14], followed by a low-complexity Cholesky-based linear precoding approach in [15]. More recent work includes joint precoding and pre-equalization for multipath fading channels [16] and studies on physical-layer

security in precoded FTN systems [17]. Despite their benefits, precoding techniques do not eliminate the need for efficient receiver-side detection, which remains a central challenge.

Motivated by this need, deep learning has recently emerged as a promising tool for FTN detection. Many studies integrate neural networks with conventional FTN algorithms to improve interference mitigation, parameter optimization, or overall detection performance, while fewer works propose standalone deep learning-based detectors. Baek et al. employed LSTM networks for FTN detection in emergency communication scenarios, achieving BCJR-comparable performance for $\tau = 0.8$ with reduced complexity [18]. Song et al. proposed two deep learning-based FTN receivers—one incorporating successive interference cancellation (SIC) and another combining detection and decoding—both demonstrating near-optimal BER for coded and uncoded FTN systems at $\tau = 0.8$ [3]. Pan et al. applied a fully connected DNN-based sliding-window detector to FTN–NOMA systems, achieving improved accuracy over MMSE-FDE while approaching maximum-likelihood performance with lower latency [19]. Additional advances include a deep learning-enhanced list sphere decoding (DL-LSD) algorithm that leverages RNNs to optimize radius selection, reducing complexity while maintaining strong BER performance for $\tau = 0.8$ and $\tau = 0.9$ [20]. Baek et al. further extended Bi-LSTM-based FTN detection to multipath fading environments, achieving BCJR-like performance for $\tau = 0.7$ and $\tau = 0.8$ without explicit channel estimation [21]. Another notable example is the deep learning-assisted sum-product detection algorithm (DL-SPDA) [22], which integrates neural networks with the sum-product algorithm to mitigate residual ISI and short-cycle effects, delivering near-optimal BER for $\tau = 0.6$ and $\tau = 0.5$ under reduced complexity. Overall, existing deep learning approaches predominantly rely on RNN-based architectures—especially LSTM and Bi-LSTM—which, while effective, impose high training and inference complexity.

Despite the potential of RNNs, convolutional neural networks (CNNs) remain underexplored for FTN detection, even though they offer inherent parallel processing and efficient feature extraction. Prior works have emphasized RNNs due to their

ability to capture temporal dependencies, but their computational intensity limits use in latency-sensitive systems. In contrast, 1D CNNs can process sequential signals with significant reductions in complexity, leveraging advancements demonstrated across various domains [23], including OFDM receivers where CNNs have replaced traditional modules and improved robustness under diverse channel conditions [24]. Hybrid CNN–RNN architectures have shown promise in handling ISI in FTN signaling [25]. A recent study by De Filippo et al. applied CNNs with skip connections for FTN detection, demonstrating strong improvements over classical equalizers [4]. Although their architecture effectively extracts features, it relies on standard convolutional layers without explicitly incorporating domain knowledge of ISI structure. Consequently, the deep stacked convolutions may introduce additional computational overhead and limit interpretability in ISI-dominated settings. Collectively, these works highlight the potential of 1D CNNs to efficiently learn temporal dependencies and motivate their application as standalone FTN detectors.

1.2 Problem Definition

Detecting FTN signals under severe ISI remains fundamentally challenging. The optimal BCJR algorithm performs maximum a posteriori sequence estimation but becomes computationally infeasible as the ISI span increases. Even reduced-state variants or minimum-phase preprocessing techniques exhibit exponential or near-exponential complexity for moderate or low τ values.

Existing deep learning–based FTN detection methods address some of these challenges but still exhibit several gaps:

- LSTM/Bi-LSTM networks offer strong BER performance but incur high computational complexity during both training and inference.
- CNN-based FTN detectors explored in the literature do not explicitly incorporate domain knowledge of the FTN ISI structure, relying instead on generic convolutional feature extraction.
- No existing work provides a unified investigation of how different neural architectures behave across FTN regimes ranging from mild ISI ($\tau \approx 0.8$) to

severe ISI ($\tau \leq 0.5$).

- There is limited exploration of low-complexity, domain-informed deep learning architectures suitable for real-time or hardware-constrained deployments.

Therefore, there is a clear need for near-optimal, low-complexity, and domain-informed FTN detection architectures that generalize across different compression factors and modulation formats.

1.3 Contributions of This Thesis

This thesis introduces three novel deep learning–based FTN detectors, each designed to operate in a different FTN regime and explicitly informed by the underlying ISI structure. The main contributions are summarized as follows:

- **Structured CNN-Based FTN Detector:** A standalone detector employing fixed-kernel, domain-informed convolution layers that explicitly capture ISI patterns at predefined symbol distances. This architecture achieves near-optimal BER for $\tau \geq 0.7$ while significantly reducing computational complexity compared to conventional CNNs and reduced-state BCJR algorithms.

- **GRU-Based FTN Sequence Detector Aligned with ISI Structure:** A lightweight recurrent neural network that leverages the FTN ISI structure by feeding the model with symbol triplets aligned to the one-sided ISI span. This approach reduces model size, improves learning stability, and achieves strong BER performance for $\tau \in [0.6, 0.7]$ with substantially lower complexity than LSTM-based methods.

- **Bi-GRU-Based FTN Detector with Position-Aware Attention (PAA):** A novel attention mechanism that assigns learned, position-indexed weights to input symbols based on their ISI relevance. Combined with bidirectional recurrent processing, this detector successfully operates under severe ISI conditions ($\tau = 0.5$), achieving BER performance close to that of the BCJR algorithm while maintaining significantly lower complexity.

Collectively, these contributions provide a comprehensive study of deep learning–based FTN detection across different ISI regimes and demonstrate that domain-informed neural architectures can effectively balance complexity and performance.

1.4 Thesis Organization

The remainder of this thesis is organized as follows:

- **Chapter 2** introduces the FTN system model, including waveform definitions, ISI formation, and the mathematical representation of FTN transmission.
- **Chapter 3** presents the structured CNN-based FTN detector, detailing its architecture, kernel design principles, training methodology, and performance evaluation.
- **Chapter 4** describes the GRU-based FTN sequence detector, including its ISI-aligned input structure and performance across moderate compression factors.
- **Chapter 5** introduces the Bi-GRU detector with Position-Aware Attention, explaining the attention formulation and demonstrating its effectiveness in severe ISI scenarios.
- **Chapter 6** provides simulation and complexity comparison for all proposed methods, including detailed BER comparisons.
- **Chapter 7** concludes the thesis and outlines potential directions for future research.

CHAPTER 2

SYSTEM MODEL FOR FASTER-THAN-NYQUIST SIGNALING

2.1 FTN Transmitter

In this thesis, faster-than-Nyquist (FTN) signaling is considered in a complex baseband framework. Let $g(t)$ denote the transmit pulse with a two-sided bandwidth of $(1+\beta)/2T$, where β is the roll-off factor and T is the Nyquist symbol interval. The transmitted FTN signal can be expressed as

$$s(t) = \sum_{k} a_k g(t - k\tau T), \tag{2.1}$$

where $s(t)$ denotes the transmitted baseband signal and a_k is the k-th complex data symbol. In this thesis, a_k is drawn from either BPSK or higher-order modulation constellations up to 64-QAM, depending on the scenario under consideration.

The parameter τ with $0 < \tau < 1$ is the *time acceleration factor* (or compression factor) that determines the signaling rate relative to the Nyquist rate. When $\tau = 1$, the system reduces to conventional Nyquist signaling. For $\tau < 1$, the symbol interval is compressed to τT, and the effective symbol rate is increased by a factor of $1/\tau$. This leads to an improvement in spectral efficiency by a factor of $1/\tau$ at the cost of deliberately introducing intersymbol interference (ISI). For instance, when $\tau = 0.9$, the corresponding spectral efficiency gain is approximately $1/0.9 \approx 1.11$, i.e., about 11%.

The pulse $g(t)$ is normalized to have unit energy, i.e.,

$$\int_{-\infty}^{+\infty} |g(t)|^2 \, dt = 1. \tag{2.2}$$

In the numerical evaluations throughout this thesis, $g(t)$ is chosen as a root-raised cosine (RRC) pulse with an excess bandwidth of $\beta = 0.35$, which is a widely adopted choice in the literature [26]. Since the occupied bandwidth of $g(t)$ is $(1+\beta)/2T$, compressing the

symbol spacing to τT with $\tau < 1$ makes the pulse bandwidth narrower than $(1+\beta)/2\tau T$. As a result, the transmitted pulses overlap in time, and the received samples are affected by ISI from neighboring symbols.

2.2 Matched Filter Receiver

At the receiver, a matched filter is employed to maximize the signal-to-noise ratio (SNR) at the sampling instants. The received continuous-time signal is first passed through the filter matched to $g(t)$, namely $g^*(-t)$, and subsequently sampled at integer multiples of the accelerated symbol interval τT.

Assuming an additive white Gaussian noise (AWGN) channel, the received signal after matched filtering and sampling at $t = n\tau T$ can be written as

$$y(n\tau T) = \sum_k a_k x((n-k)\tau T) + w(n\tau T), \tag{2.3}$$

where $y(n\tau T)$ denotes the n-th received sample and $w(n\tau T)$ is the corresponding noise sample. The noise is modeled as a zero-mean Gaussian random process with two-sided power spectral density $N_0/2$, and after matched filtering it becomes colored. The effective overall pulse $x(t)$ is defined as the convolution of the transmit filter $g(t)$ with its time-reversed version $g(-t)$:

$$x(t) = g(t) * g(-t), \tag{2.4}$$

where $*$ denotes the convolution operation. The function $x(t)$ characterizes the ISI structure of the FTN system, and its sampled values at multiples of τT directly determine the interference between neighboring symbols.

2.3 Mathematical Representation of ISI

The ISI introduced by FTN signaling can be conveniently described in terms of the sampled values of the overall pulse $x(t)$. Let

$$x_i \triangleq x(i\tau T), \quad i \in \mathbb{Z}, \tag{2.5}$$

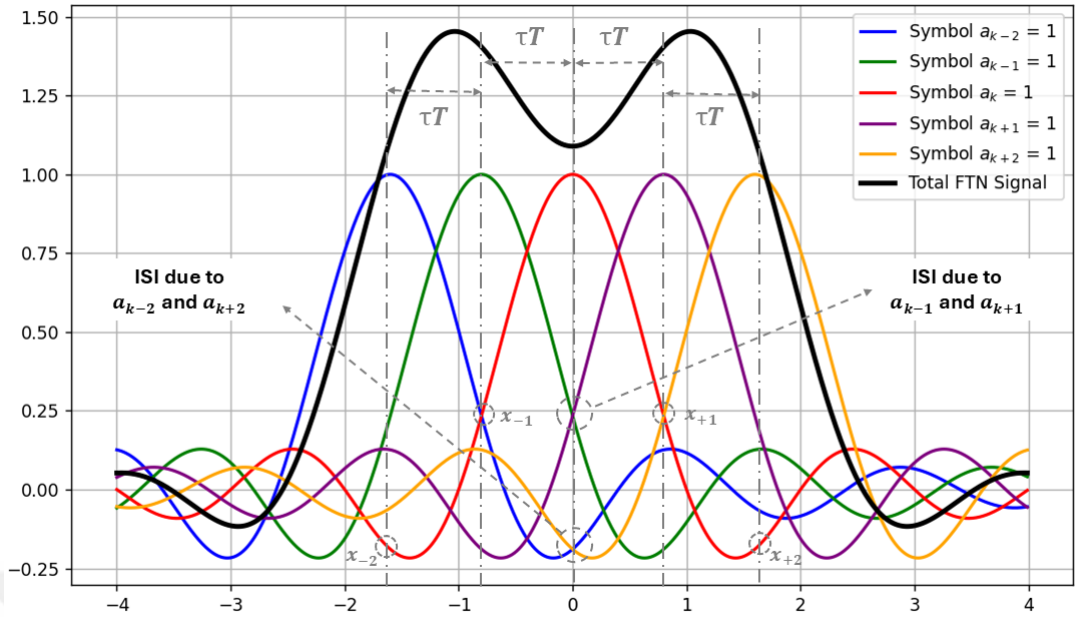


Figure 2.1 Faster-than-Nyquist signaling structure and interference coefficients for a sequence of five BPSK symbols with one-sided ISI length $N = 2$.

denote the discrete-time ISI coefficients at multiples of the accelerated symbol interval τT. The one-sided ISI length N is defined as the maximum integer such that x_i is non-negligible for $|i| \leq N$, and $x_i \approx 0$ for $|i| > N$. The parameter N depends on the time acceleration factor τ and the pulse shape.

To illustrate the ISI structure, consider a transmitted BPSK sequence [1 1 1 1 1] and let $N = 2$, as depicted in Fig. 2.1. When detecting the k-th symbol a_k, the received sample is affected not only by a_k itself but also by its neighbors a_{k-2}, a_{k-1}, a_{k+1}, and a_{k+2}. The corresponding ISI coefficients can be written inside a vector as $[x_{-2}\ x_{-1}\ x_0\ x_{+1}\ x_{+2}]$, where x_0 is the main tap, x_{-1} and x_{+1} represent the first-order interference coefficients, and x_{-2} and x_{+2} denote the second-order interference coefficients. Due to the symmetry of $x(t)$, we typically have $x_{-i} = x_{+i}$.

Using this representation, the received sample y_k corresponding to the detection of a_k

can be written as follows:

$$y_k = [x_N\, x_{N-1} \ldots x_1\, x_0\, x_1 \ldots x_{N-1}\, x_N] \begin{bmatrix} a_{k-N} \\ a_{k-N+1} \\ \vdots \\ a_{k-1} \\ a_k \\ a_{k+1} \\ \vdots \\ a_{k+N-1} \\ a_{k+N} \end{bmatrix} + w_k. \tag{2.6}$$

For a transmitted sequence of length K and one-sided ISI length N, the overall input–output relationship can be expressed in compact vector–matrix form. As an illustrative example, consider $K = 7$ and $N = 2$. The received vector can then be written as

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} x_0 & x_1 & x_2 & 0 & 0 & 0 & 0 \\ x_1 & x_0 & x_1 & x_2 & 0 & 0 & 0 \\ x_2 & x_1 & x_0 & x_1 & x_2 & 0 & 0 \\ 0 & x_2 & x_1 & x_0 & x_1 & x_2 & 0 \\ 0 & 0 & x_2 & x_1 & x_0 & x_1 & x_2 \\ 0 & 0 & 0 & x_2 & x_1 & x_0 & x_1 \\ 0 & 0 & 0 & 0 & x_2 & x_1 & x_0 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} + \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \end{bmatrix}. \tag{2.7}$$

In the general case, the received vector $\mathbf{y}$ of length K can be represented as

$$\mathbf{y} = \mathbf{X}\mathbf{a} + \mathbf{w} = \begin{bmatrix} x_0 & x_1 & x_2 & \cdots & x_N & 0 & \cdots & 0 \\ x_1 & x_0 & x_1 & \cdots & x_{N-1} & x_N & \cdots & 0 \\ x_2 & x_1 & x_0 & \cdots & x_{N-2} & x_{N-1} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ x_N & x_{N-1} & x_{N-2} & \cdots & x_0 & x_1 & \cdots & x_N \\ 0 & x_N & x_{N-1} & \cdots & x_1 & x_0 & \cdots & x_{N-1} \\ 0 & 0 & x_N & \cdots & x_2 & x_1 & \cdots & x_{N-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & x_N & x_{N-1} & \cdots & x_0 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{K-4} \\ a_{K-3} \\ a_{K-2} \\ a_{K-1} \end{bmatrix} + \begin{bmatrix} w_0 \\ \vdots \\ w_{K-1} \end{bmatrix}, \tag{2.8}$$

Table 2.1 One-sided ISI coefficients for $\tau = 0.7$, 0.8, and 0.9.

Coeff.	$\tau = 0.7$	$\tau = 0.8$	$\tau = 0.9$
x_0	9.99×10^{-1}	9.99×10^{-1}	9.99×10^{-1}
x_1	3.53×10^{-1}	2.22×10^{-1}	1.02×10^{-1}
x_2	-1.83×10^{-1}	-1.52×10^{-1}	-7.86×10^{-2}
x_3	3.24×10^{-2}	7.62×10^{-2}	4.87×10^{-2}
x_4	3.16×10^{-2}	-2.44×10^{-2}	-2.26×10^{-2}
x_5	-2.79×10^{-2}	5.93×10^{-3}	1.24×10^{-2}
x_6	1.37×10^{-2}	3.16×10^{-3}	-3.61×10^{-3}
x_7	3.31×10^{-4}	-1.73×10^{-3}	9.81×10^{-4}
x_8	-1.73×10^{-3}	6.64×10^{-4}	-1.69×10^{-4}

where $\mathbf{a}$ is the $K\times1$ transmitted symbol vector, $\mathbf{w}$ is the $K\times1$ noise vector, and $\mathbf{X}$ is the $K\times K$ Toeplitz ISI matrix whose entries are determined by the coefficients $\{x_i\}$. The covariance of $\mathbf{w}$ after matched filtering is proportional to $\mathbf{X}$ due to noise coloring.

The values of the one-sided ISI coefficients for selected values of τ used in this thesis are summarized in Table 2.1. As τ decreases, the magnitude and span of the ISI coefficients increase, reflecting stronger interference from neighboring symbols.

For a more detailed exposition of FTN signaling theory, including the derivation of ISI coefficients and the impact of the compression factor τ, the reader is referred to [7, 27–29].

2.4 Channel Models

In this thesis, two main channel models are considered: the additive white Gaussian noise (AWGN) channel and a multipath fading channel that captures more realistic wireless propagation conditions.

In the AWGN case, the channel introduces only additive noise with power spectral density $N_0/2$, and the received samples after matched filtering are modeled by Equation **2.3**. The ISI structure is fully characterized by the pulse shape and the time acceleration factor τ, as described in the previous sections.

In practical wireless environments, the transmitted signal often reaches the receiver

through multiple propagation paths, each associated with different attenuation and delay due to reflection, scattering, and diffraction. To account for such effects, the FTN system model is extended to a multipath fading channel with L taps. In this case, the received signal after matched filtering and sampling at $t = n\tau T$ is given by

$$y(n\tau T) = \sum_{l=0}^{L-1} \sum_{k} h_l a_k x\big((n-k-\delta_l)\tau T\big) + w(n\tau T), \tag{2.9}$$

where h_l denotes the complex channel coefficient of the l-th path, δ_l is the corresponding delay (in symbol intervals), and L is the number of multipath components. The overall channel impulse response can be written as

$$\mathbf{h} = [h_0, h_1, \ldots, h_{L-1}], \tag{2.10}$$

and is typically modeled as Rayleigh distributed for non-line-of-sight scenarios. In the numerical evaluations of this thesis, quasi-static Rayleigh fading is considered, where $\mathbf{h}$ remains constant over the duration of a frame and changes independently from frame to frame.

2.5 Performance Metrics

To evaluate and compare the proposed FTN detectors with conventional algorithms, two primary performance metrics are employed throughout this thesis: the bit error rate (BER) and computational complexity.

The BER is defined as the ratio of the number of erroneously detected bits to the total number of transmitted bits. Let N_{err} denote the number of bit errors observed at the detector output and N_{tot} the total number of transmitted bits in a simulation run. The empirical BER is then given by

$$\text{BER} = \frac{N_{\text{err}}}{N_{\text{tot}}}. \tag{2.11}$$

BER curves are presented as a function of the signal-to-noise ratio (SNR) and are used to assess the detection performance of the proposed and benchmark schemes under various values of τ, modulation formats, and channel conditions.

Computational complexity is assessed in both algorithmic and implementation-oriented terms.

- **Operation Count:** The number of required arithmetic operations (e.g., real-valued additions and multiplications) per detected symbol is used to compare algorithms such as BCJR, reduced-state detectors, and the proposed deep learning–based FTN detectors.
- **Hardware Complexity:** For hardware-oriented implementations, the complexity is quantified in terms of look-up table (LUT) usage, memory requirements, and the number of multiply–accumulate (MAC) operations. These metrics are particularly relevant when deploying the proposed architectures on FPGA or ASIC platforms.

By jointly analyzing BER performance and complexity measures (operation count, LUTs, and MACs), this thesis provides a comprehensive comparison between the proposed deep learning–based FTN detectors and existing model-based approaches under different FTN regimes.

CHAPTER 3

CNN-BASED FTN SIGNALING DETECTION WITH FIXED KERNEL LAYERS

In this chapter, a convolutional neural network (CNN)-based detector is developed for faster-than-Nyquist (FTN) signaling. Building on the system model presented in Chapter 2, the proposed detector is designed to efficiently learn and mitigate the strong intersymbol interference (ISI) introduced by the time-accelerated transmission. The approach combines structured convolutional kernels with domain-informed preprocessing to extract ISI patterns in a computationally efficient manner, enabling accurate symbol estimation across different values of the acceleration factor τ.

The complete FTN communication system incorporating the proposed CNN-based detector is illustrated in Fig. 3.1. As shown, the detector operates directly on matched-filtered and downsampled received sequences, learning the nonlinear mapping between ISI-corrupted inputs and transmitted symbols. In later sections of this chapter, a detailed complexity analysis is provided to demonstrate the computational advantages of the structured CNN architecture over conventional FTN detection techniques.

The remainder of this chapter is structured as follows. Section 3.1 describes the dataset preparation procedure used for training the CNN models, including the generation of modulation symbols, ISI-aware received samples, and SNR-dependent training sets. Section 3.2 introduces the structured CNN architecture and the design of fixed convolutional kernels tailored to FTN interference patterns. Section 3.3 discusses the training procedure, including loss functions, optimization strategies, and normalization techniques. Finally, Section 3.4 presents the computational complexity analysis and compares the proposed detector with existing FTN detection schemes.

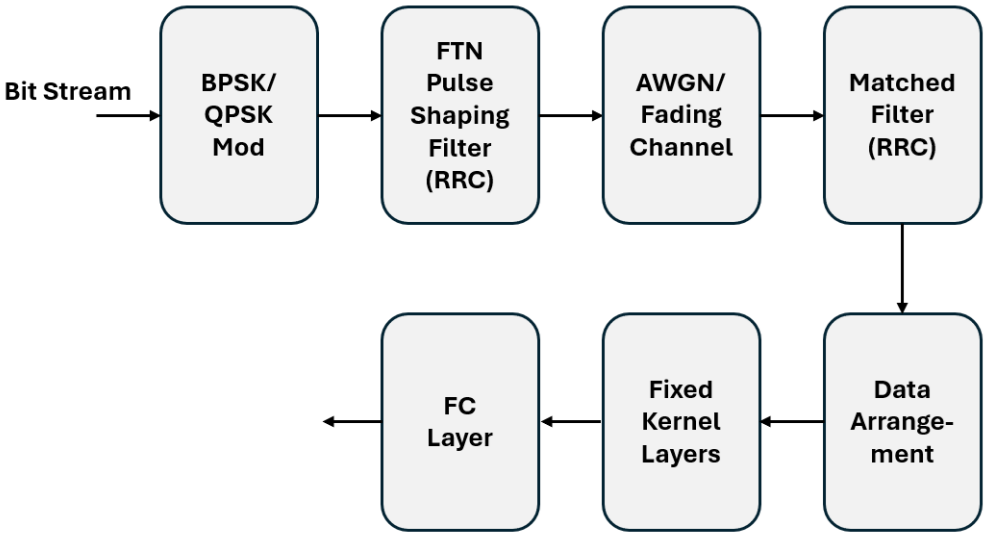


Figure 3.1 FTN signaling-based communication system with the proposed CNN-based detector.

3.1 Dataset Preparation

To train the proposed CNN-based FTN detector, a comprehensive dataset was generated spanning a wide range of τ values and signal-to-noise ratio (SNR) levels. The dataset includes BPSK as well as higher-order modulation schemes up to 64-QAM, simulated under an AWGN channel model. For each (τ, SNR) combination, corresponding input–output sample pairs were produced, ensuring that the model is exposed to diverse operating conditions and varying ISI intensities.

To enable the network to accurately learn the deterministic ISI structure inherent in FTN signaling, high-SNR datasets were emphasized during training for each τ value. High-SNR scenarios highlight the underlying interference patterns without excessive noise distortion, allowing the CNN to capture ISI dynamics more effectively. Training extensively at very low SNRs was avoided, as all FTN detectors—including conventional model-based ones—exhibit degraded BER performance in this region, and samples from such regimes contribute limited practical value. Nevertheless, the trained CNN models were evaluated across a full SNR range, including low-SNR conditions, to assess robustness and generalization.

Since the structure and magnitude of ISI vary significantly with the compression factor τ, a separate CNN model was trained for each τ value. This strategy allows the detector to learn τ-specific interference characteristics with higher precision and

avoids the performance deterioration commonly observed when a single network is trained across multiple τ regimes. By tailoring each network to its corresponding ISI structure, the proposed design achieves improved detection accuracy and prevents cross-regime generalization errors.

3.2 Data Arrangement

In FTN signaling, each received sample is influenced not only by the corresponding transmitted symbol but also by several preceding and succeeding symbols. The extent of this interference is determined by the compression factor τ, which dictates the one-sided ISI length N. To ensure that the neural network receives complete contextual information during both training and inference, the input data is reorganized to group each target symbol together with its N neighboring received samples on both sides.

Accordingly, the value of the k-th transmitted symbol is estimated from a window of $2N+1$ received samples centered at y_k, expressed as

$$\mathbf{y}_k = \begin{bmatrix} y_{k-N} & y_{k-N+1} & \cdots & y_k & \cdots & y_{k+N-1} & y_{k+N} \end{bmatrix}, \tag{3.1}$$

where y_k denotes the received sample associated with symbol a_k. The Data Arrangement block constructs input vectors in this format for all symbol indices. For the first and last N positions of each sequence, where insufficient neighboring samples exist, zero-padding is applied to maintain a consistent input length.

The CNN-based FTN detector is designed to process one input vector $\mathbf{y}_k$ per detection instance. The parameter N was determined based on extensive simulations and the analysis of one-sided ISI coefficients in Table 2.1. The selected values provide a balance between detection accuracy and computational cost. For example, $N = 2$ was used for $\tau = 0.9$, $N = 6$ for $\tau = 0.8$, and $N = 8$ for $\tau = 0.7$. While increasing N improves the detector's ability to capture wider ISI effects, it also increases the model size and computational burden, and therefore was chosen carefully for each τ.

Table 4.1 illustrates the data arrangement for $N = 6$. Each row corresponds to a single input vector supplied to the CNN during training or testing, with zero-padding applied

where necessary. The bold entry in each row marks the central received sample used to estimate the corresponding transmitted symbol.

Table 3.1 Data structure for $N = 6$. Each row represents one CNN input vector; the bold element indicates the center sample corresponding to the target symbol.

0th input	0	0	0	0	0	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6
1st input	0	0	0	0	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7
2nd input	0	0	0	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8
3rd input	0	0	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9
4th input	0	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
5th input	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}
6th input	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}	y_{12}
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
$(k-6)$th input	y_{k-12}	y_{k-11}	y_{k-10}	y_{k-9}	y_{k-8}	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k
$(k-5)$th input	y_{k-11}	y_{k-10}	y_{k-9}	y_{k-8}	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0
$(k-4)$th input	y_{k-10}	y_{k-9}	y_{k-8}	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0	0
$(k-3)$th input	y_{k-9}	y_{k-8}	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0	0	0
$(k-2)$th input	y_{k-8}	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0	0	0	0
$(k-1)$th input	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0	0	0	0	0
kth input	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0	0	0	0	0	0

3.3 CNN-Based FTN Detector System Model

The central challenge in designing an effective CNN-based FTN detector is the accurate characterization of the intersymbol interference (ISI) introduced by the compressed symbol intervals determined by the parameter τ. As described in Chapter 2, the discrete-time received sample at index k can be expressed as

$$y_k = \sum_{i=-N}^{N} x_i a_{k-i} + w_k, \tag{3.2}$$

where $a_{k-i} \in \mathcal{A}$ denotes the transmitted symbol drawn from the modulation alphabet $\mathcal{A}$, x_i is the ISI coefficient at lag i, $w_k \sim \mathcal{N}(0, \sigma^2)$ represents the colored Gaussian noise sample, and N is the one-sided ISI length determined by τ. This model indicates that each observation y_k is affected by a linear combination of $2N + 1$ transmitted symbols. Although Equation **3.2** is initially derived under the AWGN assumption, the proposed CNN-based architecture is later evaluated under both AWGN and multipath fading conditions, as discussed in Section 3.6.

To exploit this structure, the input to the CNN detector is formed as a vector $\mathbf{y}_k \in \mathbb{R}^{2N+1}$, defined as in Equation **3.1**. This representation provides the network with the full ISI span surrounding the symbol of interest.

Conventional convolutional design: As a first step, conventional convolutional kernels were applied to the input vector $\mathbf{y}_k$, which consists of the received sample y_k together with its N neighbors on both sides. In this setting, a learnable convolutional kernel $\boldsymbol{\theta} \in \mathbb{R}^r$, where $r \leq 2N+1$, is used to extract local features. The convolution operation at position m, corresponding to the m-th valid sliding window over the input sequence, is given by

$$z_m = \sum_{j=0}^{r-1} \theta_j y_{k-N+m+j}, \quad m \in [0, 2N-r+1], \tag{3.3}$$

where z_m denotes the pre-activation output at position m, θ_j is the j-th kernel coefficient, and $y_{k-N+m+j}$ is the corresponding input sample from the local context window. Here, r is the kernel size and m ranges over all positions where the kernel fits entirely within the length-$(2N+1)$ input window. The nonlinearly transformed output is then obtained as

$$o_m = \phi(z_m) = \phi\left(\sum_{j=0}^{r-1} \theta_j y_{k-N+m+j}\right), \tag{3.4}$$

where $\phi(\cdot)$ is a nonlinear activation function, such as `ReLU` or `tanh`.

Although this standard convolutional approach can capture local correlations, it exhibits two important drawbacks in the FTN setting. First, using large kernels ($r \approx 2N+1$) aggregates a broad context but tends to dilute the strong dependencies associated with the nearest interfering symbols around y_k. Second, smaller kernels ($r \ll 2N+1$) may fail to span the full ISI region or properly align with the central symbol, which can degrade detection performance. These limitations, illustrated in Fig. 3.2, motivated the development of a more structured architecture based on fixed kernel layers and domain-informed masking.

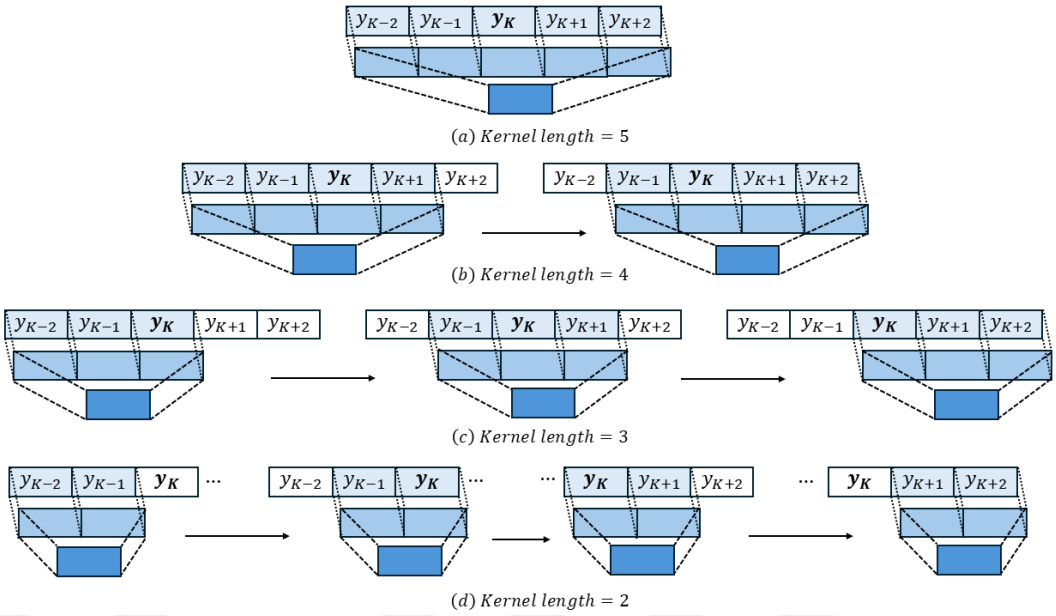


Figure 3.2 Illustration of conventional kernelization techniques with varying kernel sizes ($K = 5$, $N = 2$): (a) 5, (b) 4, (c) 3, and (d) 2.

Structured fixed kernel layers: To overcome the limitations of sliding kernels, we propose *structured fixed kernel layers*, in which each kernel is associated with a predefined positional relationship relative to the central symbol. For a given ISI span N, N fixed kernel layers are defined, each focusing on received samples at distance $i \in \{1, 2, \ldots, N\}$ from the central position y_k. The input to the i-th kernel layer is a triplet containing the central symbol and its symmetric neighbors:

$$\mathbf{y}_k^{(i)} = \begin{bmatrix} y_{k-i} & y_k & y_{k+i} \end{bmatrix}. \tag{3.5}$$

Each triplet is then processed by a learnable linear transformation followed by a nonlinearity:

$$o_k^{(i)} = \phi\left(\mathbf{y}_k^{(i)} \cdot \boldsymbol{\theta}^{(i)} + b^{(i)}\right), \tag{3.6}$$

where $\boldsymbol{\theta}^{(i)} \in \mathbb{R}^{3 \times F_i}$ denotes the weight matrix of the i-th fixed kernel layer, $b^{(i)}$ is the bias vector, F_i is the number of filters in that layer, and $\phi(\cdot)$ is the activation function, chosen as `tanh`. The choice of `tanh` is supported both empirically—no significant performance difference was observed among standard activation functions—and conceptually: because the ISI structure is primarily determined by the known pulse

shape and the compression factor τ, the network chiefly exploits spatial relationships, and `tanh` offers sufficient expressive power for stable learning.

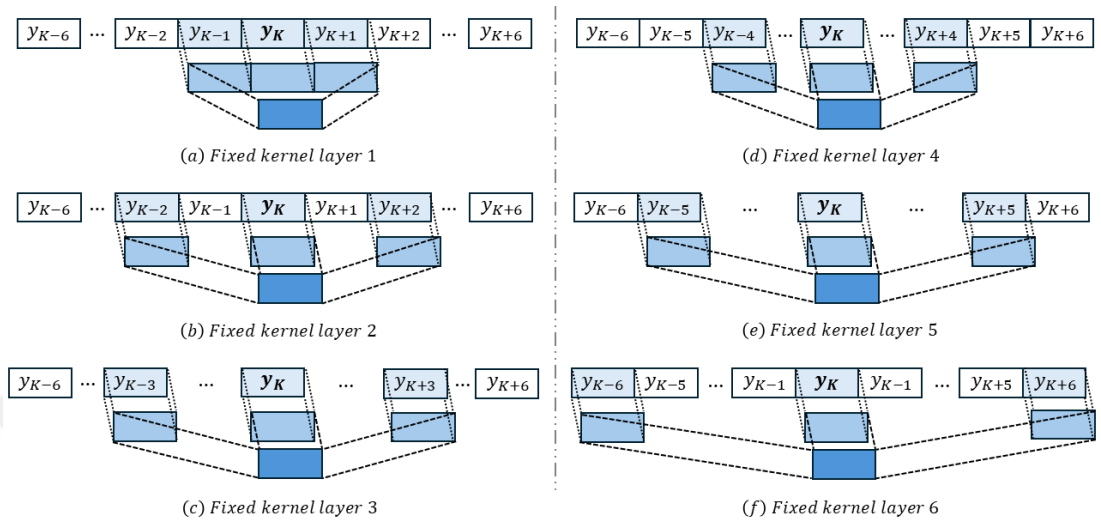


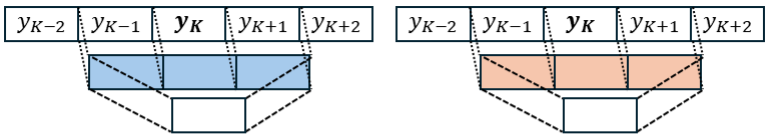
Figure 3.3 Illustration of structured fixed kernel layers with domain-informed masking: (a) Layer 1 captures symbols one unit away from the center, (b) Layer 2 captures those two units away, and so on up to (f) Layer 6.

The outputs $o_k^{(i)}$ from all N fixed kernel layers are concatenated to form a unified feature vector. This design removes the need for sliding kernels and enforces precise alignment between the learned features and the ISI contributions at each distance i. Domain-informed masking is employed to isolate the triplets $\mathbf{y}_k^{(i)}$, ensuring that each kernel focuses on its intended subset of received samples. For $N = 6$, the resulting architecture contains six fixed kernel layers, as depicted in Fig. 3.3. This modular structure mirrors the symmetric ISI behavior described in Chapter 2, where interference arises from both past and future symbols.

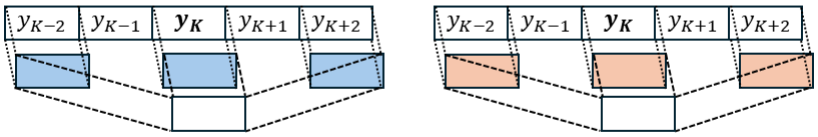
Beyond the structure of the fixed kernels, the number of filters assigned to each layer plays a crucial role in shaping the detector's performance. Increasing the number of filters enables the network to extract a richer set of features from each ISI-affected region, allowing it to better capture subtle interference patterns. As illustrated in Fig. 3.4, every fixed-kernel layer employs multiple filters, each learning a distinct feature representation from the same input window. In the figure, these filters are visualized using blue and orange overlays, highlighting how different filter responses

Table 3.2 Number of filters for each fixed kernel layer under different τ values in the proposed CNN detector.

	$N = 8$ ($\tau = 0.7$)	$N = 6$ ($\tau = 0.8$)	$N = 2$ ($\tau = 0.9$)
Layer 1	8	4	2
Layer 2	6	2	1
Layer 3	4	2	–
Layer 4	2	1	–
Layer 5	2	1	–
Layer 6	1	1	–
Layer 7	1	–	–
Layer 8	1	–	–



(a) First fixed kernel layer with two different filters



(b) Second fixed kernel layer with two different filters

Figure 3.4 Illustration of fixed kernel layers applied with two different filters for $N = 2$, demonstrating diverse feature extraction at each distance.

collectively enhance the feature extraction process. In the example shown for $N = 2$ (sequence length 5), two fixed-kernel layers are used, and each layer contains two filters to increase representational diversity.

To further improve efficiency, a hierarchical filter allocation strategy is adopted. Because ISI contributions are strongest near the central symbol, a larger number of filters is assigned to the early layers, where the most significant interference components are processed. For example, when $N = 6$ (corresponding to $\tau = 0.8$), as depicted in Fig. 3.3, the first layer employs 4 filters, the next two layers use 2 filters each, and the remaining layers utilize a single filter. This allocation ensures that the most influential ISI regions receive higher modeling capacity while preventing unnecessary growth in complexity. The filter configurations used for different τ values are listed in Table 3.2.

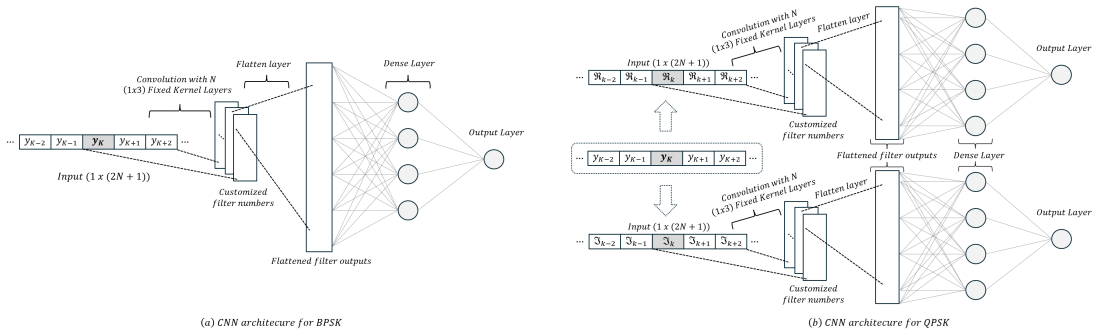


Figure 3.5 Convolutional neural network (CNN)-based faster-than-Nyquist (FTN) signaling detector architecture for BPSK and QPSK modulations.

The feature maps generated by all fixed kernel layers are concatenated and passed to a fully connected dense layer with 4 neurons. The pre-activation and activation of the l-th neuron are given by

$$z_l = \sum_{j=1}^{F_{\text{total}}} \theta_j^{(l)} v_j + b^{(l)}, \quad h_l = \phi(z_l), \tag{3.7}$$

where F_{total} is the total number of filters, v_j is the j-th input feature, and $\theta_j^{(l)}$ and $b^{(l)}$ are the weights and bias associated with neuron l. The activation function $\phi(\cdot)$ is again chosen as `tanh`. The final output layer consists of a single neuron with a sigmoid activation function, which yields the probability estimate of the transmitted symbol. The complete CNN-based FTN detector architecture is depicted in Fig. 3.5.

For complex modulation schemes such as QPSK, the real and imaginary parts of each received sample are separated and processed by two identical CNN branches. The outputs of these branches are then combined in the final layer to reconstruct the complex-valued symbol decision. This strategy preserves architectural modularity while enabling a unified treatment of both real-valued (e.g., BPSK) and complex-valued (e.g., QPSK) constellations.

3.4 Complexity Analysis for Structured Fixed Kernel Layers

In this section, the computational efficiency of the proposed structured fixed kernel approach is evaluated and compared with that of the conventional convolutional kernel technique. To provide a concrete and interpretable comparison, the analysis focuses on a

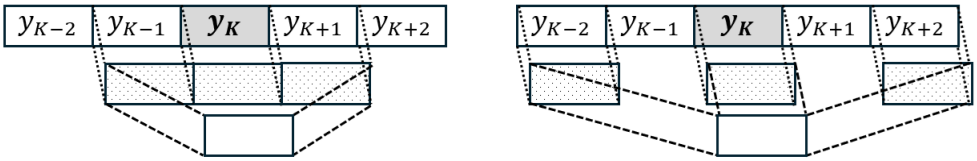


Figure 3.6 Structured fixed kernel layers for a sequence length of 5 ($N = 2$), designed to capture ISI effects at specific distances from the central symbol.

sequence length of five samples ($N = 2$), which corresponds to the ISI span encountered at $\tau = 0.9$. As discussed in Section 3.3, among the conventional convolutional designs, the configuration employing a kernel size of three (illustrated in Fig. 3.2(c)) provides the most accurate results; therefore, this kernel size is used as the reference in the following analysis.

For the same sequence length ($N = 2$), the structured fixed kernel architecture is illustrated in Fig. 3.6. Unlike conventional sliding kernels, this architecture explicitly incorporates domain knowledge regarding the underlying ISI structure. By assigning each fixed kernel to a specific positional relationship relative to the central symbol, the method eliminates redundant computations and concentrates its learning capacity on the most relevant ISI contributions.

In the conventional convolutional approach, a kernel of length three slides across the five-sample input sequence. At each valid position, the kernel carries out three multiplications and three additions (including the bias term). Since the kernel must operate over three valid positions when $N = 2$, this results in a total of $3 \times 3 = 9$ multiplications and $3 \times 3 = 9$ additions for a single filter. In practice, however, reliable feature extraction generally requires at least two filters, increasing the total computational burden to 18 multiplications and 18 additions. This illustrates the relatively high arithmetic cost associated with conventional sliding kernels, particularly when multiple filters are needed to achieve satisfactory detection accuracy.

In contrast, the structured fixed kernel method requires only two computational steps for $N = 2$, corresponding to the ISI contributions from symbols located one and two positions away from the central symbol. As summarized in Table 3.2, the first fixed

kernel layer employs two filters, each performing three multiplications and three additions, resulting in a total of $2 \times 3 = 6$ multiplications and 6 additions. The second fixed kernel layer uses a single filter, which produces an additional 3 multiplications and 3 additions. Consequently, the overall computational load of the proposed method is only 9 multiplications and 9 additions, representing a substantial reduction compared with the conventional sliding-kernel approach.

Under identical conditions, the structured fixed kernel architecture reduces the total number of arithmetic operations by 50% compared to the conventional convolutional approach. Since the conventional method requires 18 operations and the proposed method requires only 9, the relative reduction is calculated as $(18-9)/18 \times 100\% = 50\%$. This reduction is achieved without any loss in detection accuracy; in fact, by explicitly modeling ISI patterns through position-specific kernels, the structured method enhances representational efficiency while simultaneously lowering computational overhead. The resulting improvements in bit-error rate (BER) performance are discussed in detail in Section 3.6.1.

3.5 Learning Rate Strategy for Training

The learning rate plays a central role in the successful training of the proposed CNN-based FTN detector, as it directly influences the optimizer's ability to converge and reduces the risk of stagnation during the learning process. Although the adaptive moment estimation (Adam) optimizer is used as the primary optimization method due to its robustness against noisy gradients and its per-parameter learning rate adaptation, relying solely on its default behavior with a fixed global learning rate can lead to suboptimal convergence.

Adam is particularly suitable for the proposed network architecture, which employs structured fixed kernel layers to model ISI contributions at varying distances from the center symbol. Its adaptive nature helps stabilize training despite the heterogeneous feature scales across fixed kernel layers. However, when a constant learning rate is used, the optimizer may struggle to refine model parameters beyond an initial point of convergence. For example, when training the model for $\tau = 0.8$, the validation loss

Table 3.3 Training hyperparameters of the proposed CNN-based FTN detector for different τ values.

Parameter	$\tau = 0.9$	$\tau = 0.8$	$\tau = 0.7$
Number of training samples	4,000,000	4,000,000	4,000,000
Number of test samples	4,000,000	4,000,000	4,000,000
SNR values during training	7, 8, 9, 10 dB	7, 8, 9, 10 dB	7, 8, 9, 10 dB
Batch size	1000	1000	1000
Steps per epoch	4000	4000	4000
Number of epochs	20	20	20
Optimizer	Adam	Adam	Adam
Initial learning rate	0.001	0.001	0.001
Learning rate schedule	Exponential Decay	Exponential Decay	Exponential Decay
Decay steps	20,000	20,000	20,000
Decay interval (epochs)	Every 5 epochs	Every 5 epochs	Every 5 epochs
Decay rate	0.9	0.9	0.9
Decay staircase	True	True	True
Number of interference symbols (N)	2	6	8
Modulation formats	BPSK, QPSK, 16-QAM, 64-QAM	BPSK, QPSK, 16-QAM, 64-QAM	BPSK, QPSK
Loss function	Binary Cross-Entropy	Binary Cross-Entropy	Binary Cross-Entropy
Output activation	Sigmoid	Sigmoid	Sigmoid

stabilizes around 6×10^{-4} after the fourth epoch and shows negligible improvement even with extended training.

To address this limitation, an exponentially decaying learning rate schedule is incorporated alongside Adam. The learning rate is initialized at 0.001 and gradually reduced by a fixed decay factor at predetermined intervals during training. This scheduling strategy enables the optimizer to take larger exploratory steps during the early stages of learning and progressively finer adjustments in later epochs. By complementing Adam's local adaptivity, the decaying schedule prevents overshooting, improves stability near convergence, and encourages the model to escape plateau regions of the loss landscape.

The benefits of this combined approach are reflected in the training behavior. For instance, under $\tau = 0.8$, the model trained with exponential learning rate decay reduces its validation loss to approximately 2×10^{-4} by the tenth epoch, achieving substantially better convergence compared to the constant learning rate case. This improvement highlights the importance of dynamic learning rate scheduling for FTN detection, where fine-grained adjustments in later epochs are essential for accurately capturing subtle ISI structures.

A comprehensive summary of the training configuration—including learning rate schedule, batch size, optimizer settings, and other hyperparameters used for each τ value—is presented in Table 3.3.

3.6 Simulation Results and Complexity Comparison

This section presents the performance evaluation of the proposed CNN-based FTN detector, including bit-error rate (BER) results and computational complexity comparisons with existing FTN detection techniques.

In the simulation framework, the transmitted signal is first evaluated over an additive white Gaussian noise (AWGN) channel, where noise samples are drawn from a Gaussian distribution $\mathcal{N}(0, N_0/2)$. The signal-to-noise ratio (SNR) is defined as SNR = E_s/N_0, where E_s denotes the average symbol energy after pulse shaping. To examine the detector under more realistic propagation conditions, a quasi-static multipath Rayleigh fading channel is also considered. In this scenario, the received signal consists of three Rayleigh-faded components of equal average power, with relative delays of τT between successive paths, where τ denotes the FTN compression factor. The fading coefficients $\{h_0, h_1, h_2\}$ are independently drawn from a Rayleigh distribution with unit mean power and remain constant during both training and testing, reflecting a quasi-static environment without Doppler variations. AWGN is added after the multipath combination to maintain consistency with standard wireless channel modeling. It should be noted that this study focuses exclusively on wireless channel impairments; fiber-optic propagation effects such as chromatic dispersion, polarization-mode dispersion, and nonlinear distortions are not included in the scope of this work.

The BER analysis begins by comparing the proposed structured fixed-kernel CNN design with conventional CNN architectures under an AWGN channel for $\tau = 0.9$. The evaluation is then extended to more severe FTN configurations with $\tau = 0.8$ and $\tau = 0.7$, demonstrating the robustness of the proposed method under increasing levels of ISI. To ensure statistical reliability, 95% confidence intervals are calculated for all BER curves using binomial error statistics. Beyond AWGN-based experiments, the detector is further assessed over the quasi-static Rayleigh fading channel, where its performance is benchmarked against the optimal BCJR algorithm across multiple FTN compression factors.

In addition to BER evaluations, a spectral efficiency analysis is conducted to quantify the achievable throughput as a function of τ and the employed modulation scheme. Finally, a detailed computational complexity study is presented, including operation-level comparisons and hardware resource assessments. The proposed detector is compared with the M-BCJR algorithm and other existing FTN detection techniques to highlight its computational advantages and practical feasibility.

3.6.1 BER Results

To assess the effectiveness of the proposed CNN-based FTN detector with structured fixed kernel layers and domain-informed masking, we first compare it against a series of conventional convolutional CNN architectures for the case of $\tau = 0.9$, where the input sequence length is 5 ($N = 2$). This particular compression factor is chosen because conventional CNN kernels are unable to approach BCJR-level performance at lower τ values such as 0.8 and 0.7, where the severity of ISI increases substantially. The comparison for $\tau = 0.9$ is provided in Fig. 3.7.

Throughout the figures, the proposed detector is denoted by *CNN-FK3*, where "FK" refers to the use of fixed kernels and "3" indicates that each fixed kernel operates on a triplet consisting of the central sample and its two symmetric neighbors, as described in Section 3.3. For comparison, *CNN-K2*, *CNN-K3*, *CNN-K4*, and *CNN-K5* denote conventional CNN architectures employing sliding kernels of lengths 2, 3, 4, and 5, respectively, whose structures are illustrated in Fig. 3.2.

Figure 3.7 shows that the structured fixed-kernel approach consistently provides superior BER performance. The proposed CNN-FK3 achieves BER values nearly indistinguishable from those of the optimal BCJR algorithm for $\tau = 0.9$, demonstrating its ability to accurately capture the underlying ISI structure. In contrast, all conventional CNN architectures exhibit degraded performance. Among them, CNN-K3 provides the best results, as its kernel length achieves a balance between receptive field size and alignment with the central symbol. However, even CNN-K3 falls short of the proposed method.

These observations can be attributed to the inherent limitations of sliding

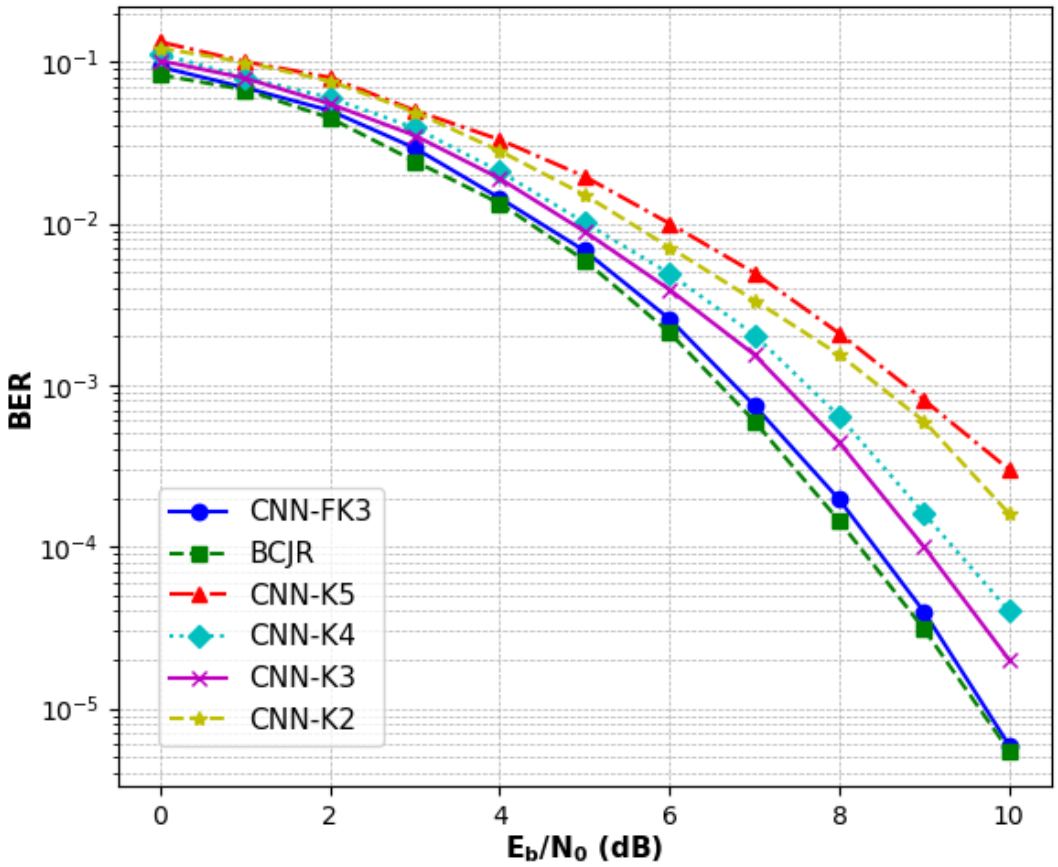


Figure 3.7 BER comparison between the proposed structured fixed-kernel CNN detector (CNN-FK3) and conventional convolutional CNNs (CNN-K2, CNN-K3, CNN-K4, CNN-K5) for $\tau = 0.9$ and $\beta = 0.35$ over an AWGN channel.

convolutional kernels when applied to FTN signaling. Larger kernels (e.g., size 5), shown in Fig. 3.2(a), dilute local dependencies by aggregating an overly broad context, whereas smaller kernels (e.g., size 2), shown in Fig. 3.2(d), fail to cover the full ISI span. Intermediate kernel sizes (3 and 4) offer improved performance, but they still lack the explicit structural alignment needed to model symmetric ISI contributions effectively. The structured fixed-kernel architecture eliminates these shortcomings by designing kernels that directly correspond to the known ISI pattern at each distance, which not only enhances detection accuracy but also improves computational efficiency, as analyzed in Section 3.4.

Overall, the results confirm that the proposed CNN-FK3 architecture delivers near-optimal detection performance while maintaining a significantly lower computational burden compared to conventional CNN approaches.

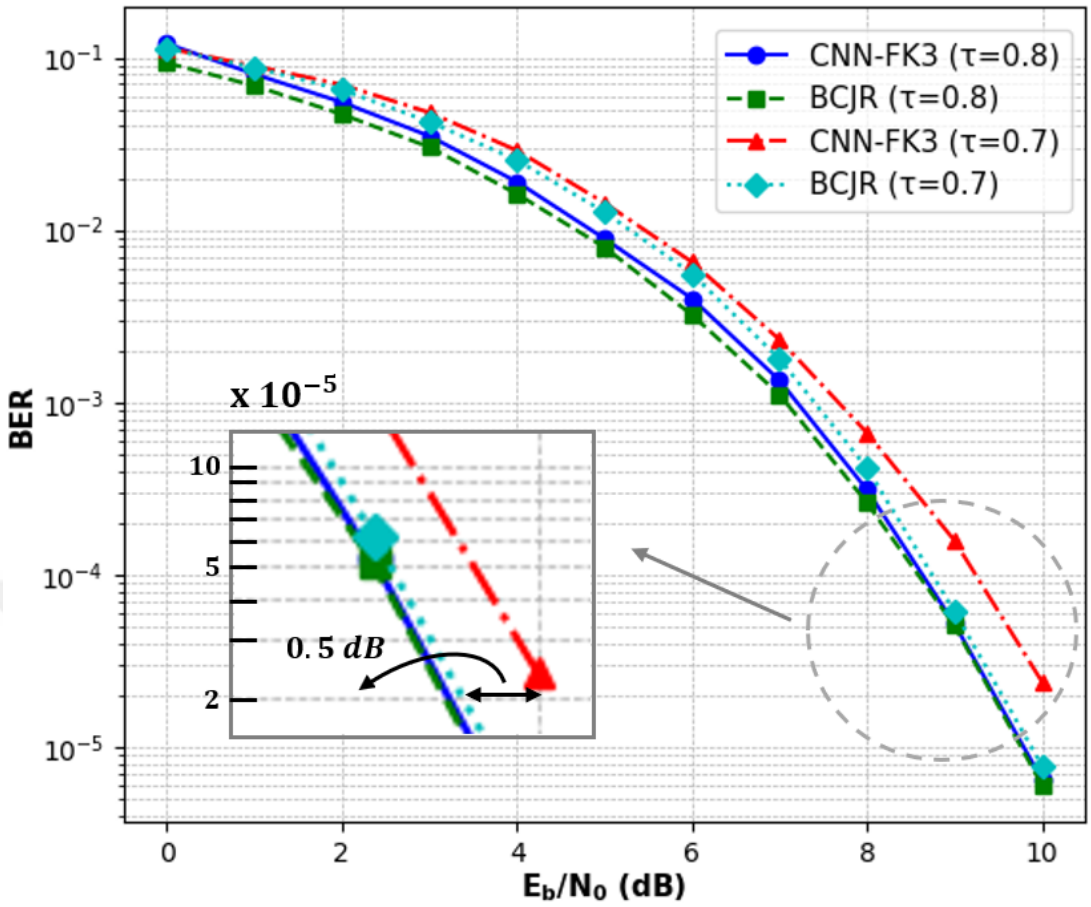


Figure 3.8 BER comparison between the proposed CNN-based FTN detector and the optimal BCJR algorithm for $\tau = 0.8$ and $\tau = 0.7$ with $\beta = 0.35$ over an AWGN channel.

To further assess the capability of the proposed CNN-based FTN detector, its performance was evaluated for smaller compression factors, namely $\tau = 0.8$ and $\tau = 0.7$. These configurations represent more challenging FTN scenarios due to the increased severity of ISI. The corresponding BER results are shown in Fig. 3.8. Since the BER curves for BPSK and QPSK are nearly identical across all operating points, a single plot is provided for clarity.

For $\tau = 0.8$, the proposed detector achieves BER performance that is virtually identical to that of the optimal BCJR algorithm across the full SNR range. This close alignment demonstrates the ability of the structured fixed-kernel architecture to accurately capture and mitigate ISI components even under moderately compressed signaling conditions.

At $\tau = 0.7$, where ISI becomes significantly more pronounced, the proposed detector continues to perform remarkably well. A minor performance loss—approximately 0.5

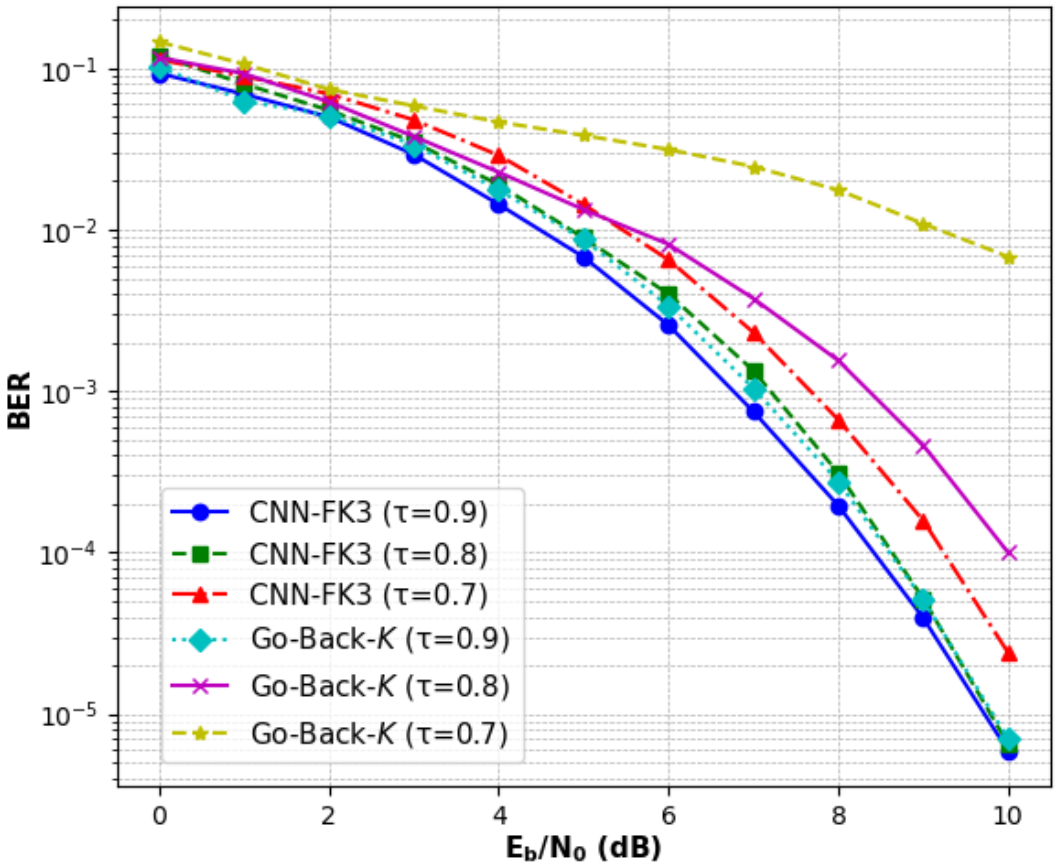


Figure 3.9 BER comparison of the proposed CNN-based FTN detector and the Go-Back-K equalizer [1] for $\tau = 0.9$, $\tau = 0.8$, and $\tau = 0.7$ under an AWGN channel with $\beta = 0.35$.

dB at a BER of 2×10^{-5}—is observed relative to the BCJR algorithm at high SNRs. Although this deviation might appear notable on first inspection, achieving such accuracy solely with a CNN-based architecture is a substantial result. Existing FTN works commonly rely on hybrid deep learning structures, such as CNN–BiLSTM combinations [25, 30], to handle severe ISI. In contrast, the proposed method demonstrates that a properly designed CNN with structured fixed kernels can function as a standalone detector while maintaining performance close to the optimal algorithm. These results highlight both the adaptability and robustness of the proposed architecture across a wide range of τ values.

To further examine the balance between performance and computational efficiency, the proposed method is compared with the low-complexity successive symbol-by-symbol sequence estimator introduced in [5], under a roll-off factor of $\beta = 0.35$. Since the detector in [5] is specifically designed for reduced complexity, it offers an appropriate

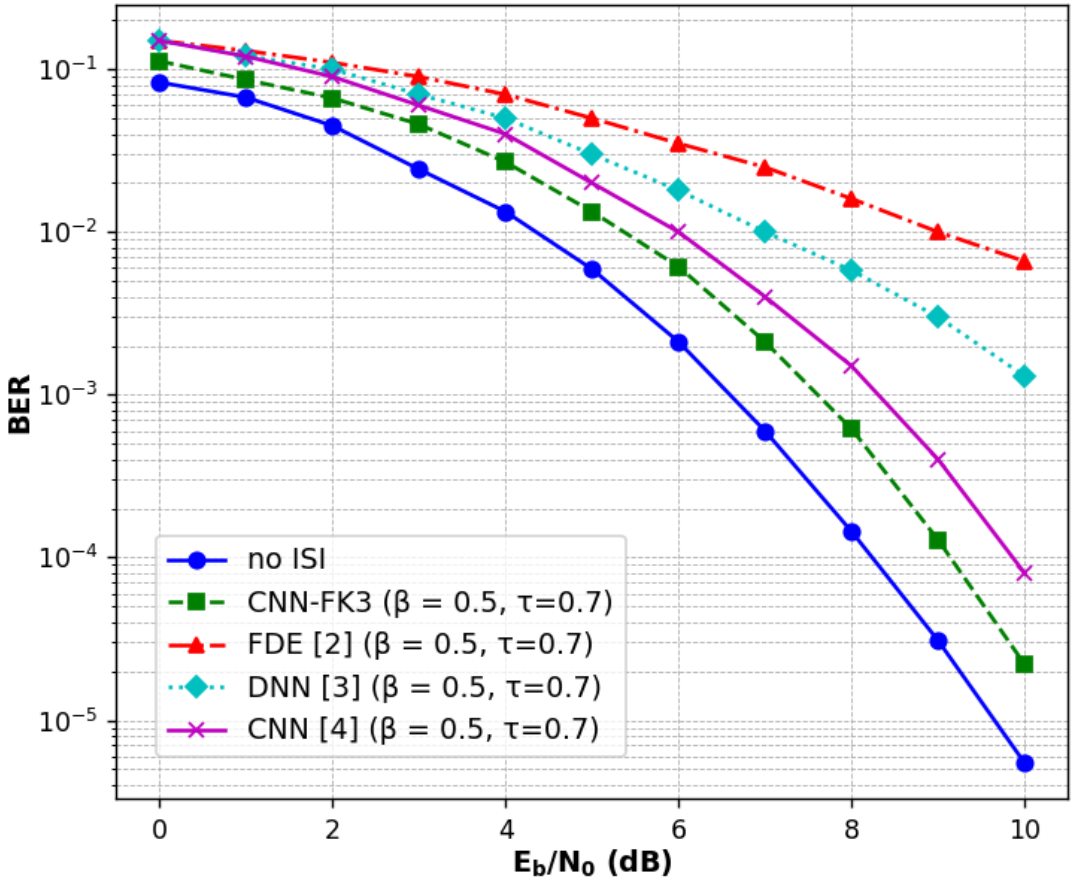


Figure 3.10 Comparison of the proposed CNN detector with the FTN detection techniques presented in [2], [3], and [4] for $\tau = 0.7$ and $\beta = 0.5$ over an AWGN channel.

benchmark for evaluating both accuracy and computational cost. The BER comparison, shown in Fig. 3.9, reveals that for $\tau = 0.9$, both approaches yield nearly identical performance, as expected for scenarios with relatively mild ISI. However, for $\tau = 0.8$ and $\tau = 0.7$, the proposed CNN-based detector clearly outperforms the method in [5], demonstrating its ability to effectively manage stronger interference levels.

In addition to BER evaluation, these two techniques are also compared in terms of computational complexity in Section 3.6.2. Presenting both accuracy and complexity analyses offers a comprehensive assessment of the proposed detector. By achieving superior detection performance while maintaining low computational cost, the proposed method establishes itself as a strong and practical alternative for FTN signaling detection.

To further benchmark the performance of the proposed CNN-based FTN detector, its

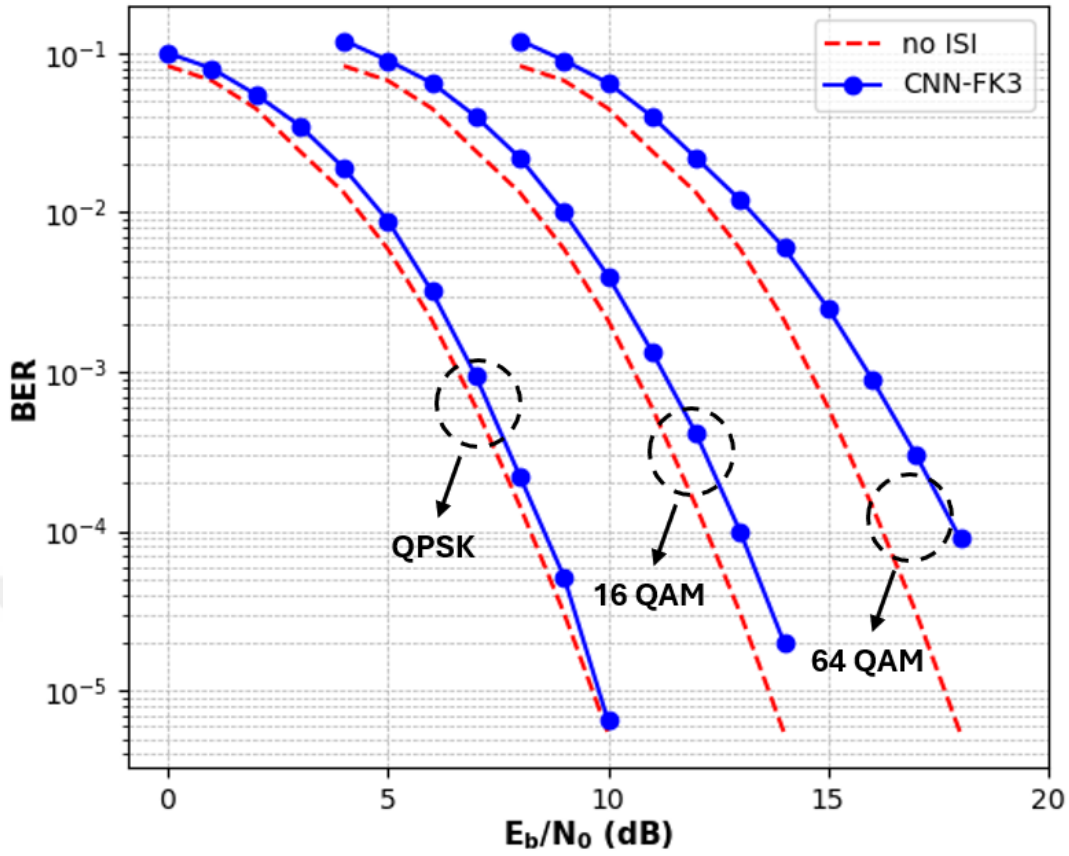


Figure 3.11 BER performance of the proposed CNN-based FTN detector for QPSK, 16-QAM, and 64-QAM with $\tau = 0.8$ and $\beta = 0.5$ under an AWGN channel.

BER characteristics are compared with the detection schemes presented in [2], [3], and [4]. For a fair comparison, all competing methods are evaluated under the same FTN configuration with $\tau = 0.7$ and $\beta = 0.5$. As illustrated in Fig. 3.10, the proposed detector consistently outperforms these existing approaches across a broad SNR range, achieving substantially lower BER values. In addition to its improved accuracy, the proposed method also achieves superior computational efficiency, as will be detailed in Section 4.5.2.

In addition to QPSK, the adaptability of the proposed detector to higher-order constellations is examined using 16-QAM and 64-QAM. These evaluations employ the same network architecture developed for QPSK, shown in Fig. 3.5(b). Simulation results for $\tau = 0.8$ and $\beta = 0.5$, presented in Fig. 3.11, demonstrate reliable performance across all examined modulation formats. This indicates that the structured fixed-kernel architecture generalizes effectively to more complex symbol

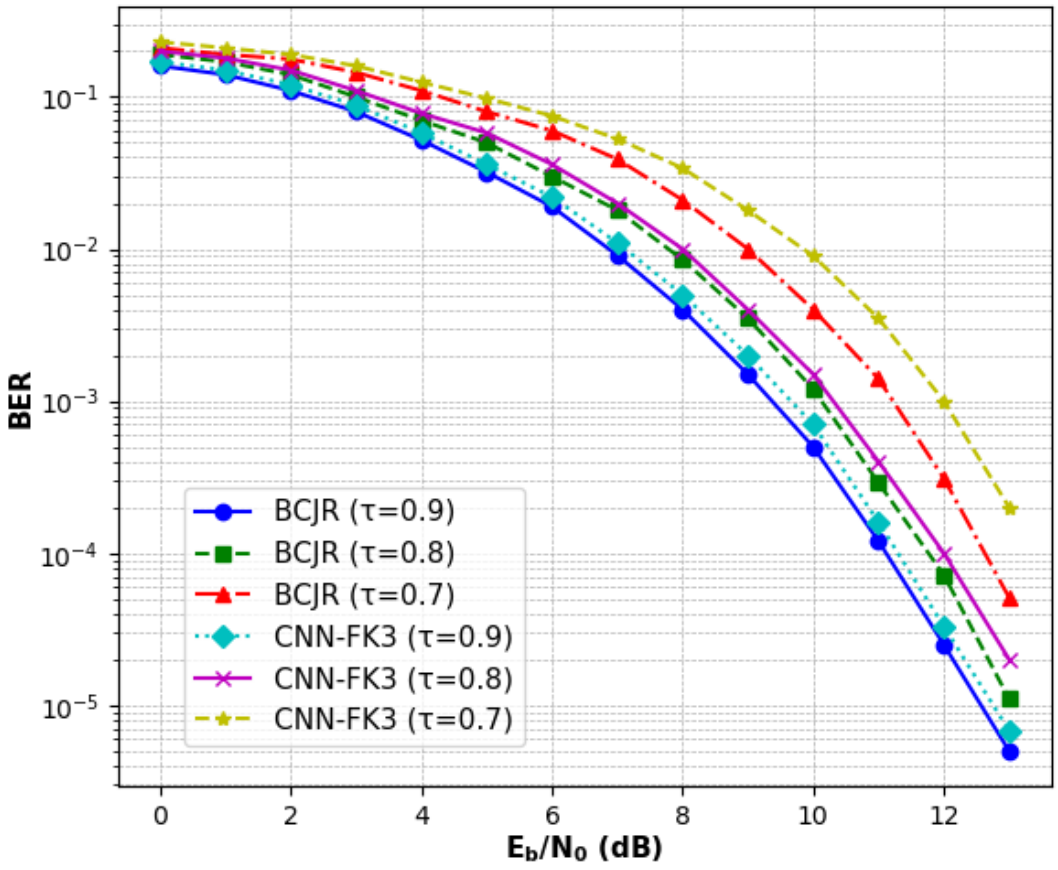


Figure 3.12 BER performance of the proposed CNN-FK3 detector over a quasi-static multipath Rayleigh fading channel for $\tau = 0.7$, $\tau = 0.8$, and $\tau = 0.9$ using QPSK modulation. The corresponding BCJR benchmark curves are included for reference.

alphabets without requiring modifications to the model structure.

To investigate the detector's robustness under realistic wireless propagation conditions, an additional set of simulations is conducted over a quasi-static multipath Rayleigh fading channel. The fading environment consists of three equal-power paths, with relative delays between consecutive paths set to τT. This configuration causes each delayed component to overlap with a subsequent FTN symbol, intensifying the ISI effect. The fading coefficients remain constant within each frame and vary independently across frames during both training and testing, consistent with a quasi-static fading assumption.

Figure 3.12 compares the BER performance of the proposed CNN-FK3 detector with that of the optimal BCJR algorithm for $\tau = 0.9$, $\tau = 0.8$, and $\tau = 0.7$ using QPSK modulation. The proposed method achieves BER curves closely matching those of

the BCJR detector for all compression factors. Even at $\tau = 0.7$, where ISI and fading effects are most severe, the deviation from the BCJR benchmark remains minimal. These results confirm the capability of the structured fixed-kernel design to learn and compensate for both ISI and multipath fading impairments. Consequently, the proposed detector demonstrates strong applicability to practical FTN systems operating under realistic wireless channel conditions, extending its effectiveness well beyond AWGN-only scenarios.

To further examine the practicality of the proposed CNN-based FTN detector in modern communication systems, we extend our analysis to LDPC-coded transmission. LDPC codes are widely adopted in current standards such as 5G NR and IEEE 802.11 due to their near-capacity performance and efficient iterative decoding. In this study, we employ a standardized LDPC code with a block length of $n = 1056$ and a coding rate of $R = 1/2$, corresponding to $k = 528$ information bits and 528 parity bits. The parity-check matrix used in the simulations is derived from the base graph structures defined in the 3GPP 5G NR specification (3GPP TS 38.212), ensuring realistic code behavior and compatibility with practical deployments.

At the transmitter, the input bitstream is first encoded using the LDPC encoder, and the resulting 1056-bit codeword is modulated using BPSK, where $\{0, 1\}$ map to $\{+1, -1\}$, respectively. These modulated symbols are pulse-shaped using an RRC filter and transmitted using FTN signaling with compression factors $\tau = 0.9$, $\tau = 0.8$, and $\tau = 0.7$. As discussed in Section 2, reducing τ generates increasingly severe ISI due to tighter symbol packing. After propagation through the AWGN channel, the received signal is matched-filtered and sampled at intervals of τT.

The proposed CNN-based detector processes the sampled sequence by exploiting the structured ISI patterns inherent to FTN signaling. For each target bit, a window of $(2N + 1)$ samples is fed into the CNN, which outputs a soft-valued estimate representing the likelihood of the central symbol. These soft outputs are then mapped to log-likelihood ratios (LLRs) and passed to the LDPC decoder. The decoder employs the belief propagation (BP) algorithm with a maximum of 50 decoding iterations.

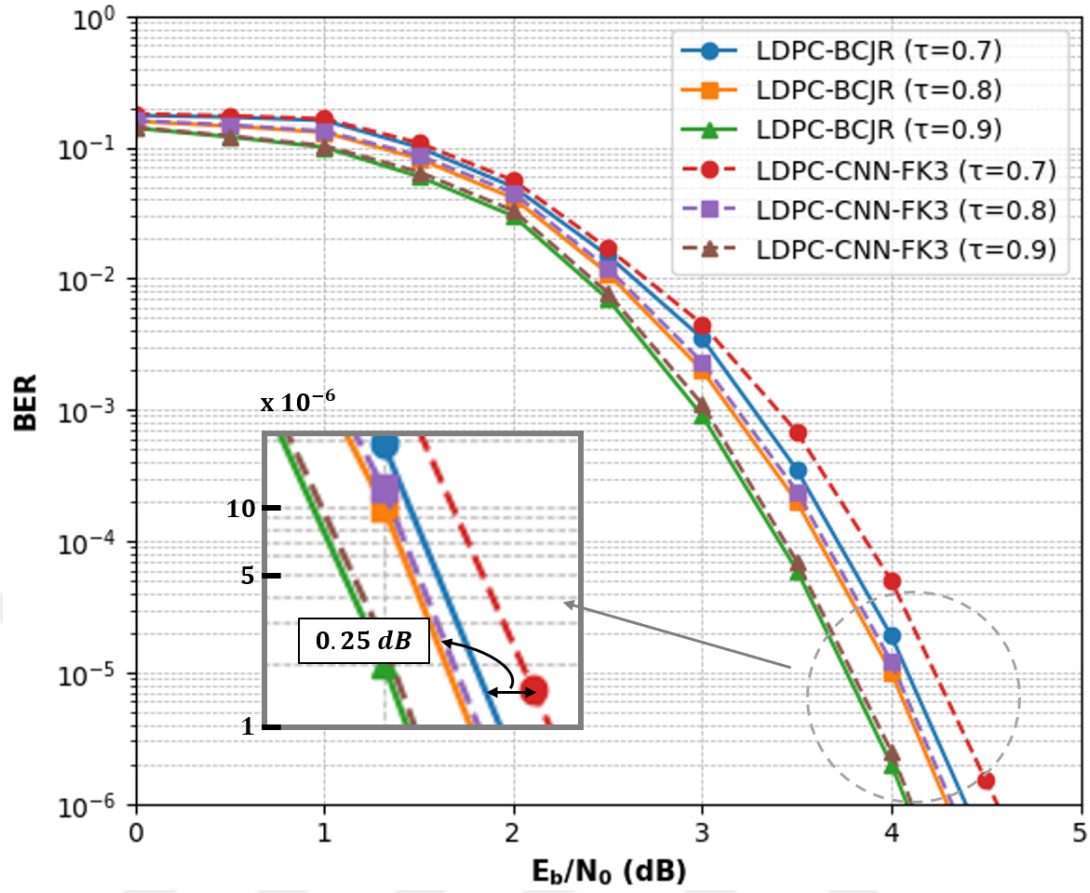


Figure 3.13 BER performance comparison of LDPC-coded FTN systems employing BCJR-based and CNN-FK3-based detectors for $\tau = 0.7$, $\tau = 0.8$, and $\tau = 0.9$. A rate-1/2 LDPC code of length 1056 bits is used with BPSK modulation over an AWGN channel.

To benchmark the LDPC-coded performance, the proposed method is compared with an LDPC-coded FTN system employing the optimal BCJR detector under identical signaling conditions. The resulting BER curves are shown in Fig. 3.13. As expected, LDPC coding significantly enhances error-rate performance, particularly for lower τ values where ISI is more severe. More importantly, the combined CNN+LDPC system achieves BER performance comparable to that of the BCJR+LDPC system, even at $\tau = 0.7$, while maintaining substantially reduced computational complexity. These findings confirm that the structured fixed-kernel CNN architecture can be effectively integrated with forward error correction and serves as a practical, low-complexity alternative to optimal detection in coded FTN systems.

To complement the BER-based evaluation, a spectral efficiency analysis is carried out to quantify the trade-off between detection reliability and bandwidth utilization for different modulation formats and FTN compression factors. Spectral efficiency (SE),

defined as the number of successfully transmitted information bits per second per unit bandwidth, is expressed as

$$\text{SE} = \frac{\log_2 M}{\tau(1+\beta)} \quad \text{[bps/Hz]}, \tag{3.8}$$

where M denotes the modulation order, τ represents the FTN time compression factor, and β is the roll-off factor associated with the RRC pulse-shaping filter. This formulation jointly incorporates the inherent efficiency of the modulation through $\log_2 M$ and the spectral broadening due to excess bandwidth β. In accordance with common practice, the roll-off factor is set to $\beta = 0.35$ for BPSK and QPSK, while a larger value of $\beta = 0.5$ is used for 16-QAM and 64-QAM.

The resulting spectral efficiency values for $\tau = 0.7$, $\tau = 0.8$, $\tau = 0.9$, and the Nyquist case ($\tau = 1$) are illustrated in Fig. 3.14. For higher-order modulations such as 16-QAM and 64-QAM, evaluations are restricted to $\tau \geq 0.8$, since excessively aggressive time packing (e.g., $\tau = 0.7$) leads to BER levels that render these constellations impractical under FTN signaling. As anticipated, spectral efficiency increases as τ decreases, underscoring the primary advantage of FTN signaling in achieving higher throughput through intentional time compression. These results provide a quantitative framework for balancing throughput, complexity, and detection performance, further highlighting the suitability of the proposed detector for spectrally efficient communication scenarios.

3.6.2 Computational Complexity Comparison

An essential factor in evaluating the practicality of any FTN detection scheme is its computational efficiency, particularly in resource-constrained environments. To this end, we compare the inference-time computational complexity of the proposed CNN-based FTN detector with that of the traditional M-BCJR algorithm. The analysis quantifies the number of arithmetic operations required to detect a block of 100 symbols for both BPSK and QPSK modulations, considering three FTN compression factors: $\tau = 0.7$, $\tau = 0.8$, and $\tau = 0.9$.

To provide insight into how the operation counts are derived for the CNN-based detector, we detail the computational steps for the representative case of BPSK detection at

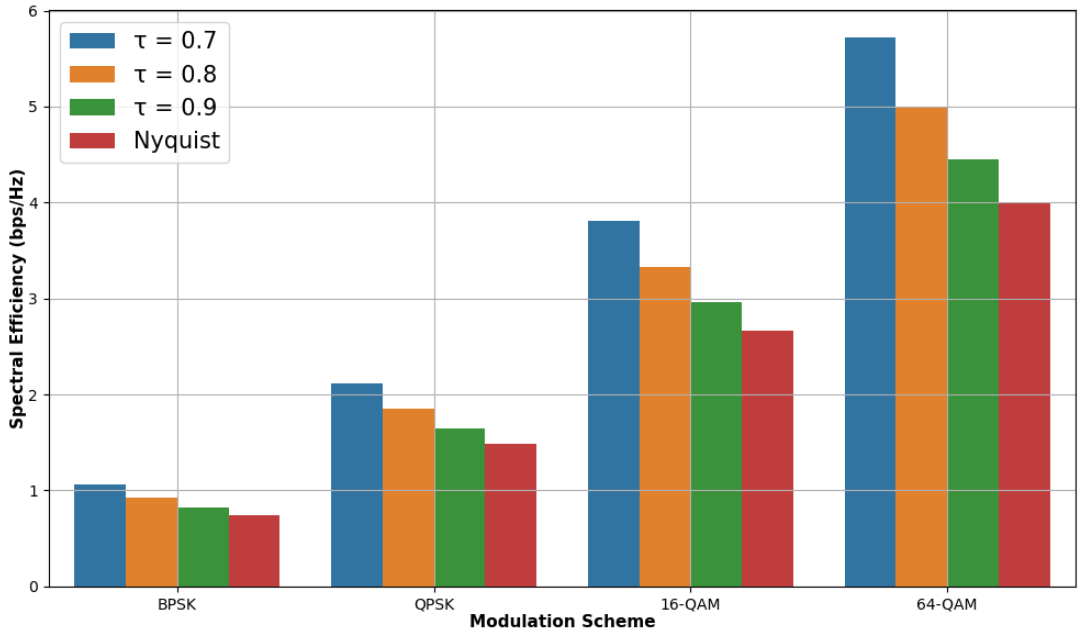


Figure 3.14 Spectral efficiency for BPSK, QPSK, 16-QAM, and 64-QAM under FTN signaling with $\tau = 0.7$, $\tau = 0.8$, $\tau = 0.9$, and the Nyquist case ($\tau = 1$). Roll-off factor $\beta = 0.35$ is used for BPSK/QPSK and $\beta = 0.5$ for 16-QAM/64-QAM.

$\tau = 0.8$. For brevity, the derivations for $\tau = 0.7$ and $\tau = 0.9$ follow the same procedure and are therefore omitted.

As described in Section 3.3, each fixed kernel layer processes a triplet of received samples centered around symbol index k, as defined by Equation **3.5**. For a given ISI distance $i \in \{1, 2, \ldots, N\}$, the corresponding triplet is multiplied by a weight matrix $\boldsymbol{\theta}^{(i)} \in \mathbb{R}^{3 \times F_i}$, followed by the addition of a bias vector $b^{(i)} \in \mathbb{R}^{F_i}$, and an element-wise nonlinear activation, as formalized in Equation **3.6**. The outputs of all N fixed kernel layers are then concatenated to form the feature vector associated with the current symbol.

For $\tau = 0.8$, the filter configuration specified in Table 3.2 assigns 4 filters to the first fixed kernel layer, 2 filters each to the second and third layers, and 1 filter to each of the remaining layers, resulting in a total of 11 filters. Each filter processes its corresponding triplet through 3 multiplications, 3 additions, and one `tanh` operation. Consequently, across all fixed kernel layers, the total computational load amounts to 33 multiplications, 33 additions, and 11 `tanh` evaluations.

The computational cost of the dense layer can be determined using the formulation introduced in Section 3.3. Let F_{total} denote the total number of filters across all fixed kernel layers, and let $\mathbf{v} \in \mathbb{R}^{F_{\text{total}}}$ represent the concatenated feature vector obtained after flattening their outputs. For the l-th neuron in the dense layer, the pre-activation value is computed as

$$z_l = \sum_{j=1}^{F_{\text{total}}} \theta_j^{(l)} v_j + b^{(l)}, \tag{3.9}$$

where $\theta_j^{(l)}$ and $b^{(l)}$ denote the weight and bias parameters of the l-th neuron. The activated output is obtained by applying the `tanh` nonlinearity:

$$h_l = \phi(z_l). \tag{3.10}$$

This operation is repeated for all four neurons in the dense layer. The final output of the network is produced by a single neuron with a sigmoid activation function. Let $F_{\text{dense}} = 4$ denote the number of dense-layer neurons. The output of the detector for symbol index k is given by

$$\hat{a}_k = \sigma\left(\sum_{j=1}^{F_{\text{dense}}} \theta_j^{(\text{out})} h_j + b^{(\text{out})} \right), \tag{3.11}$$

where $\theta_j^{(\text{out})}$ and $b^{(\text{out})}$ represent the output neuron's weights and bias, and $\sigma(\cdot)$ is the sigmoid function.

For $\tau = 0.8$, the dense layer receives $F_{\text{total}} = 11$ features. Each of the four dense-layer neurons therefore performs 11 multiplications and 11 additions for the weight and bias terms, along with 10 additional additions required for accumulating the weighted inputs, followed by one `tanh` activation. In total, the dense layer carries out 44 multiplications, 44 bias additions, 40 accumulation additions, and 4 `tanh` operations.

The final output layer consists of a single neuron and processes the four dense-layer outputs. As a result, it requires 4 multiplications, 4 bias additions, 3 accumulation additions, and one `sigmoid` activation to produce the final probability estimate.

Based on the above calculations, the proposed CNN-based FTN detector requires a total of 81 multiplications, 124 additions, 15 `tanh` activations, and a single `sigmoid` operation to detect one BPSK-modulated symbol when $\tau = 0.8$. As illustrated in the CNN architecture in Fig. 3.5(b), QPSK detection processes the in-phase and quadrature components independently through identical branches. Consequently, all operation counts simply double for QPSK. Thus, for the detection of one QPSK-modulated symbol, the proposed method performs 162 multiplications, 248 additions, 30 `tanh` activations, and 2 `sigmoid` operations at $\tau = 0.8$.

A broader comparison between the proposed detector and the M-BCJR algorithm [31] is provided in Table 3.4. The table summarizes the total number of operations required to detect 100 symbols under BPSK and QPSK modulations for $\tau = 0.7$, $\tau = 0.8$, and $\tau = 0.9$. Unlike M-BCJR, which involves costly operations such as divisions, exponentials, and numerous comparisons, the CNN-based detector relies solely on multiplications, additions, and nonlinear activations, resulting in a substantially more hardware-friendly profile. These results highlight the significant computational savings offered by the proposed method, especially for lower τ values where M-BCJR complexity grows rapidly.

Because the CNN-based detector and the M-BCJR algorithm rely on fundamentally different classes of operations, a direct comparison based solely on raw operation counts would not yield a fair assessment of computational efficiency. To provide a more meaningful evaluation, we adopt a commonly used FPGA resource estimation framework to quantify the hardware cost associated with each operation type in terms of lookup tables (LUTs). Expressing complexity in LUTs provides a normalized metric that enables objective comparison across heterogeneous operations such as multiplications, additions, divisions, and nonlinear functions.

Moreover, LUT utilization is not only indicative of hardware resource consumption but also correlates strongly with dynamic energy usage in FPGA platforms. Since LUTs constitute a significant portion of the switching fabric, lower LUT occupancy typically leads to reduced switching activity and, consequently, lower dynamic power consumption [32]. Thus, the LUT-weighted computational cost used in this analysis

Table 3.4 Comparison of Operation Numbers for M-BCJR and CNN-Based FTN Detector (100 Symbols)

Operation	**Algorithm**	**BPSK Modulation**			**QPSK Modulation**		
		$\tau = 0.7$	$\tau = 0.8$	$\tau = 0.9$	$\tau = 0.7$	$\tau = 0.8$	$\tau = 0.9$
Multiplication	M-BCJR	4631	2783	1877	33048	19964	13916
	CNN	17900	8100	2500	35800	16200	5000
Addition	M-BCJR	9982	5129	2584	70857	36137	18671
	CNN	27800	12400	3600	55600	24800	7200
Comparison	M-BCJR	4160	1056	272	29283	7439	1976
	CNN	–	–	–	–	–	–
Division	M-BCJR	3982	2981	1436	28819	21857	9488
	CNN	–	–	–	–	–	–
Exponential	M-BCJR	205	204	203	410	408	406
	CNN	–	–	–	–	–	–
Tanh	M-BCJR	104	103	102	208	206	204
	CNN	2900	1500	700	5800	3000	1400
Sigmoid	M-BCJR	–	–	–	–	–	–
	CNN	100	100	100	200	200	200

offers insight into both hardware efficiency and expected power behavior of the compared detection methods.

All LUT estimations in this study assume 10-bit numerical precision, consistent with typical fixed-point implementations. The resulting hardware resource requirements for each operation class are summarized in Table 3.5.

Table 3.5 LUT Utilizations for Different Operations (10-bit)

Operation	**Multiplication**	**Addition**	**Comparison**	**Division**	**Exponential**	**tanh**	**sigmoid**
LUT Utilization	113	10	11	236	73	1	1

Using the operation counts summarized in Table 3.4, together with the LUT-based hardware requirements listed in Table 3.5, we construct a composite metric to quantitatively assess the total computational complexity of both detection methods. This metric is obtained by multiplying the number of each operation type by its corresponding LUT cost and summing the results. In doing so, the analysis incorporates not only the frequency of operations but also their relative hardware

burdens, thereby yielding an integrated measure of computational cost. This LUT-weighted evaluation provides a holistic perspective on efficiency by jointly capturing algorithmic workload, hardware resource usage, and the implied energy consumption characteristics of the two detection schemes.

Table 3.6 LUT-Weighted Computational Cost Metrics for M-BCJR and CNN-Based FTN Detector (BPSK and QPSK Modulations)

Modulation	**Algorithm**	$\tau = 0.7$	$\tau = 0.8$	$\tau = 0.9$
BPSK	M-BCJR	1623704	1095896	594750
	CNN	2303700	1040900	319300
QPSK	M-BCJR	11596529	7887373	4049964
	CNN	4607400	2081800	638600

The results of the computational efficiency analysis, summarized in Table 3.6, highlight the substantial advantages of the proposed CNN-based FTN detector, particularly for higher-order modulation formats such as QPSK. For BPSK modulation, the CNN-based method achieves notable reductions in overall computational cost, with improvements of 46% and 5% for $\tau = 0.9$ and $\tau = 0.8$, respectively, as reported in Table 3.7. The benefits become even more pronounced for QPSK modulation, where the proposed detector yields reductions of 84% for $\tau = 0.9$, 74% for $\tau = 0.8$, and 60% for $\tau = 0.7$ compared to the M-BCJR algorithm.

These trends stem from fundamental differences in how the computational complexity of the two approaches scales with modulation order. Whereas the M-BCJR algorithm exhibits exponential growth in complexity as the constellation size increases, the CNN-based detector grows only linearly with the number of parallel branches, making it particularly well-suited for modulation schemes beyond BPSK. This efficiency gain is attributed to the architectural characteristics of the CNN—namely, its structured fixed kernel layers, parallelizable computation, and reliance on learned parameters—rather than any implementation-specific optimizations. Indeed, careful inspection of our implementation confirms algorithmic fidelity to the original M-BCJR formulation, ensuring that the reported reductions reflect intrinsic theoretical advantages. Consequently, the CNN-based detector achieves up to an 84% reduction in LUT-weighted computational complexity over the M-BCJR algorithm,

independent of low-level code refinements or hardware optimizations.

Table 3.7 Percentage Improvement of CNN over M-BCJR in LUT-Weighted Computational Cost Metric

Modulation	$\tau = 0.7$	$\tau = 0.8$	$\tau = 0.9$
BPSK	–	5%	46%
QPSK	60%	74%	84%

In addition to the comparative study against the M-BCJR algorithm, we further assessed the computational efficiency of the proposed CNN-based FTN detector relative to the detection methods reported in [2], [5], [3], and [4]. The algorithms in [2], [3], and [4] require 8194 additions and 8196 multiplications, 9720 additions and 9720 multiplications, and 53,500 MAC operations per detected symbol, respectively. These values clearly exceed the computational requirements of the proposed CNN-FK detector, indicating that these methods are significantly more demanding in terms of per-symbol complexity.

Table 3.8 Comparison of Operation Numbers for Go-Back-K and CNN-Based FTN Detector (100 Symbols, BPSK, $\tau = 0.8$)

Algorithm	**Multiplication**	**Addition**	**Comparison**	**Tanh**	**Sigmoid**
Go-Back-K	14850	15444	300	–	–
CNN	8100	12400	–	1500	100

Among the considered approaches, the Go-Back-K algorithm proposed in [5] emerges as the most suitable benchmark for a detailed comparison due to its emphasis on low-complexity FTN detection. Accordingly, we conducted an operation-level analysis under $\tau = 0.8$ with BPSK modulation, with the corresponding operation counts summarized in Table 3.8. Using the LUT-weighted complexity metric described earlier, the total computational cost of the Go-Back-K algorithm is calculated as 1,696,790 LUT-equivalent units. In contrast, the proposed CNN-FK detector incurs a total LUT-weighted cost of 1,040,900 under the same conditions. This corresponds to an approximate 39% reduction in overall complexity.

This substantial reduction underscores the resource efficiency of the proposed method, even when evaluated against another algorithm specifically designed for low-complexity FTN detection. Moreover, as demonstrated in Section 3.6.1, the BER performance of

the proposed detector also surpasses that of [5] for $\tau = 0.8$ and $\tau = 0.7$, highlighting its superior detection capability in scenarios with moderate to severe ISI. Collectively, these findings establish the proposed CNN-FK detector as a compelling alternative for FTN signaling applications requiring both high detection accuracy and reduced computational cost.

3.7 Summary of the Chapter

This chapter introduced a novel CNN-based FTN detection framework built upon structured fixed kernel layers and domain-informed masking. In contrast to conventional sliding-kernel CNN architectures, the proposed method utilizes position-specific fixed kernels that explicitly capture ISI contributions at different symbol distances. A hierarchical filter allocation strategy further enhances feature extraction while maintaining computational efficiency.

The proposed detector demonstrated near-optimal BER performance for $\tau \geq 0.7$, closely matching the optimal BCJR algorithm under both AWGN and quasi-static multipath Rayleigh fading channels. In addition, significant reductions in LUT-weighted computational complexity were achieved, including up to 46% for BPSK and 84% for QPSK when compared with the M-BCJR algorithm. The method was also evaluated against several state-of-the-art FTN detection techniques and consistently demonstrated superior efficiency with competitive or improved accuracy. Its applicability to higher-order modulations (including 16-QAM and 64-QAM) confirms the scalability and flexibility of the proposed design.

The results presented in this chapter highlight the effectiveness of integrating structured domain knowledge with deep learning architectures for FTN signaling detection. Building on these findings, the next chapter explores a recurrent neural network–based FTN detector using GRUs, enabling the modeling of long-range temporal dependencies and providing an alternative perspective on deep learning–based FTN detection.

CHAPTER 4

GRU-BASED FTN DETECTOR

This chapter presents a GRU-based detection framework designed specifically for faster-than-Nyquist (FTN) signaling over an AWGN channel. The development of the proposed method follows a structured workflow comprising dataset generation, data preprocessing and arrangement, and the formulation of the GRU-based detection model. As illustrated in Fig. 5.1, the FTN communication system incorporates a signal detection block that processes the matched-filtered samples through dedicated data transformation and arrangement stages, producing inputs tailored for the GRU network.

The data arrangement procedure plays a central role in shaping the GRU input representation, determining both the number of recurrent units and the temporal structure provided to the network. This step ensures that the sequential dependencies induced by FTN signaling—particularly those arising from inter-symbol interference (ISI)—are effectively modeled by the GRU architecture.

Following the data arrangement discussion, the GRU-based system model is introduced in detail. The architectural motivations for employing GRUs over other recurrent neural network (RNN) variants, such as LSTMs, are examined, emphasizing advantages in complexity, convergence, and suitability for FTN detection. A comparative analysis between GRU and LSTM architectures is provided, highlighting computational savings and performance considerations. Additionally, the chapter evaluates key training optimization strategies, with particular emphasis on the choice between NADAM and ADAM optimizers. Both theoretical insights and empirical validation using training loss profiles are presented to justify the adopted optimization approach.

4.1 Dataset Preparation

To train the proposed GRU-based FTN detector, a comprehensive dataset was generated covering multiple values of the FTN compression factor τ and a range of

SNR levels under BPSK- and QPSK-modulated FTN signaling over an AWGN channel. For each combination of modulation type and τ, a total of 4×10^6 input–output symbol pairs were produced. The dataset was randomly partitioned such that 90% was allocated for training and the remaining 10% for validation. An additional, independently generated test set consisting of 10^6 pairs was used to assess the detector's generalization performance across different SNR conditions.

The training subset was employed to optimize the network parameters, while the validation subset was used to monitor potential overfitting and to support early stopping during the optimization process. For each τ value, all subsets were drawn from the same underlying statistical distribution to ensure consistency in performance evaluation. To facilitate the learning of ISI characteristics inherent to FTN signaling, high-SNR samples were emphasized during training, enabling the GRU model to more accurately capture the deterministic ISI structure before being tested under noisier conditions. A separate training phase was conducted for each τ value to allow the GRU-based detector to learn τ-specific ISI patterns and network weights, thereby enhancing detection accuracy for each operating scenario.

4.2 Data Arrangement

In FTN signaling, both past and future symbols contribute to inter-symbol interference (ISI), with the number of interfering neighbors determined by the chosen compression factor τ. To accommodate this effect, the FTN receiver architecture is designed such that, during both training and inference, the N preceding and N succeeding received samples—representing the one-sided ISI length—are included when estimating the central symbol. Instead of processing each received sample independently, a preprocessing stage groups each target symbol with its adjacent neighbors, forming a structured input window of length $L = 2N + 1$. The resulting input vector is expressed as

$$\mathbf{Y}_k = \begin{bmatrix} y_{k-N} & y_{k-N+1} & \cdots & y_k & \cdots & y_{k+N-1} & y_{k+N} \end{bmatrix}, \tag{4.1}$$

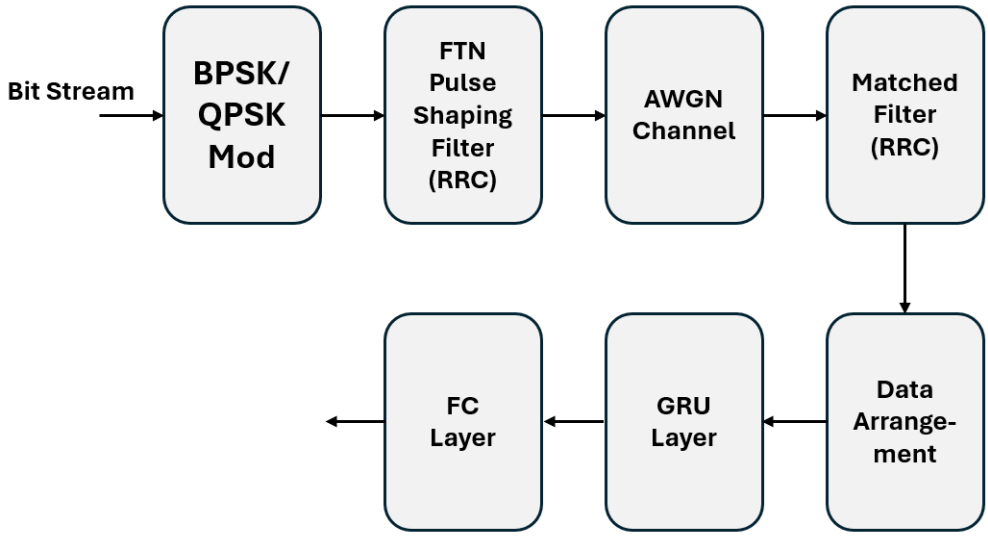


Figure 4.1 FTN signaling-based communication system incorporating the proposed GRU-based detector.

where y_k denotes the k-th received sample corresponding to the transmitted symbol a_k. This arrangement aligns the input window with the ISI structure of FTN signaling, enabling the GRU to learn temporal dependencies associated with dominant interference terms.

A boundary-handling mechanism is applied for the first N and last N symbols in the dataset. Since these positions lack sufficient neighbors to construct a full window of size L, the missing entries are padded with zeros. This zero-padding approach preserves structural consistency across all input samples while preventing index underflow or overflow.

The GRU-based FTN detector developed in this study employs an adaptive configuration in which the one-sided ISI length N is selected according to the chosen compression factor τ. This selection is guided by a joint evaluation of empirical BER performance and the corresponding ISI coefficient magnitudes provided in Table 2.1. Based on this analysis, the detector uses $N = 2$ for $\tau = 0.9$, $N = 4$ for $\tau = 0.8$, and $N = 6$ for $\tau = 0.7$, ensuring that the model captures the dominant ISI contributions without incurring unnecessary computational overhead.

These values provide a balanced trade-off between modeling accuracy and computational cost, ensuring that the GRU captures the most significant ISI

contributions without unnecessary architectural overhead. A detailed justification is provided in Section 4.5, where the impact of N on BER is examined. For illustration, the structured input format corresponding to $N = 6$ is presented in Table 4.1, where each row represents the GRU input associated with a single transmitted symbol.

Table 4.1 Data Structure for $N = 6$

0th input	0	0	0	0	0	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6
1st input	0	0	0	0	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7
2nd input	0	0	0	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8
3rd input	0	0	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9
4th input	0	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
5th input	0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}
6th input	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}	y_{11}	y_{12}
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
$(k-6)$th input	y_{k-12}	y_{k-11}	y_{k-10}	y_{k-9}	y_{k-8}	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k
$(k-5)$th input	y_{k-11}	y_{k-10}	y_{k-9}	y_{k-8}	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0
$(k-4)$th input	y_{k-10}	y_{k-9}	y_{k-8}	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0	0
$(k-3)$th input	y_{k-9}	y_{k-8}	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0	0	0
$(k-2)$th input	y_{k-8}	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0	0	0	0
$(k-1)$th input	y_{k-7}	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0	0	0	0	0
kth input	y_{k-6}	y_{k-5}	y_{k-4}	y_{k-3}	y_{k-2}	y_{k-1}	y_k	0	0	0	0	0	0

4.3 GRU-Based FTN Detector System Model

In the proposed FTN signaling detector, a Gated Recurrent Unit (GRU) network is employed in place of the more commonly used Long Short-Term Memory (LSTM) architecture. This design choice is motivated by the need to balance temporal modeling capability with computational efficiency. While both architectures are well-suited for sequence processing and can effectively capture the temporal dependencies introduced by ISI, GRUs feature a simpler internal structure with fewer parameters. Specifically, GRUs merge the input and forget gates of LSTMs into a single update gate, as illustrated in Fig. 4.2, which reduces memory usage and computational overhead. This streamlined gating mechanism leads to faster training and inference, making GRUs particularly appropriate for real-time FTN detection scenarios where both accuracy and efficiency are essential [33].

The internal operations of a GRU cell at time index t are governed by the following

equations:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z), \tag{4.2}$$

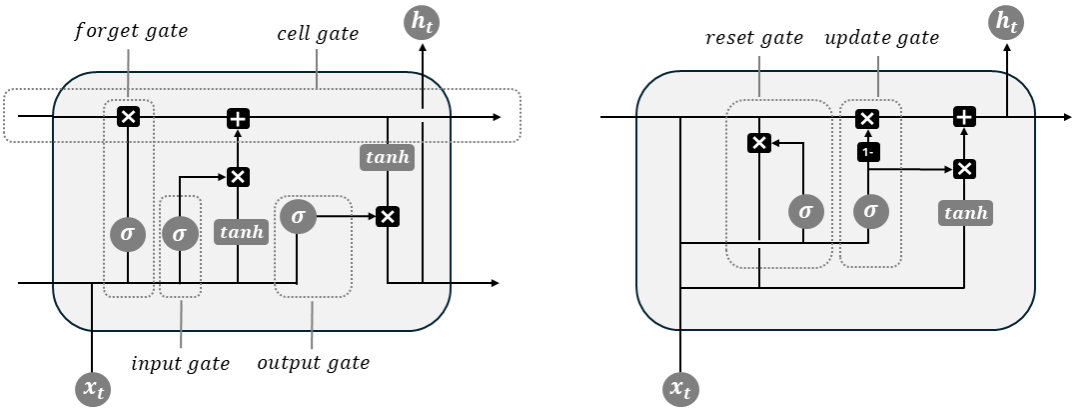
$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r), \tag{4.3}$$

$$\tilde{h}_t = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h), \tag{4.4}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t, \tag{4.5}$$

where $W_{\{\cdot\}}$, $U_{\{\cdot\}}$, and $b_{\{\cdot\}}$ denote the learnable weight matrices and bias vectors associated with the update gate (z_t), reset gate (r_t), and candidate hidden state ($\tilde{h}_t$), respectively. The update gate z_t regulates how much of the previous hidden state h_{t-1} is carried forward, allowing the GRU to retain long-term dependencies when necessary. The reset gate r_t controls the degree to which past information is forgotten before computing the candidate state, thereby enabling the model to selectively emphasize more recent observations. The candidate hidden state $\tilde{h}_t$ blends the current input x_t with a gated version of the previous hidden state, and the final hidden state h_t is obtained through a convex combination of h_{t-1} and $\tilde{h}_t$ weighted by the update gate.

This gating mechanism allows the GRU to dynamically adapt its memory over time, balancing the influence of new and past information. As a result, the GRU architecture offers a compelling trade-off between modeling capability and computational simplicity, making it well-suited for FTN detection where efficient handling of sequential ISI-induced dependencies is crucial.



(a) *LSTM architecure* (b) *GRU architecure*

Figure 4.2 Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures.

The internal operations of the LSTM network are governed by the following set of

equations:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f), \tag{4.6}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i), \tag{4.7}$$

$$\tilde{C}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c), \tag{4.8}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t, \tag{4.9}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o), \tag{4.10}$$

$$h_t = o_t \odot \tanh(C_t), \tag{4.11}$$

where f_t, i_t, and o_t denote the forget, input, and output gates, respectively, and $\tilde{C}_t$ represents the candidate cell state. The cell state C_t acts as an explicit memory pathway, combining retained information from C_{t-1} with newly selected content gated by i_t. The hidden state h_t is then generated by modulating the activated cell state with the output gate o_t. This architecture enables the LSTM to store, update, and expose information selectively over long sequences, making it particularly effective for modeling long-term dependencies.

Although both GRUs and LSTMs are designed to process sequential data and capture temporal correlations, they differ in architectural complexity. GRUs achieve this functionality using a reduced gating mechanism, combining the input and forget gates into a single update gate and eliminating the explicit memory cell. This results in fewer parameters and lower computational overhead, which is advantageous for real-time FTN detection. LSTMs, on the other hand, offer a more expressive memory structure through their dedicated cell state and additional gate, potentially providing advantages in tasks requiring deep long-term dependency tracking. In the context of the proposed FTN signaling detector, the following analysis evaluates the computational efficiency trade-offs associated with each architecture.

The overall architecture of the proposed GRU-based FTN detector for BPSK and QPSK modulation is presented in Fig. 4.3, where the one-sided ISI length N is selected according to the corresponding value of τ. Unlike a generic GRU application, the proposed detector explicitly exploits the structured ISI characteristics of FTN signaling

by aligning its input window with the ISI span. This design enables the network to learn interference patterns efficiently while avoiding unnecessary computational overhead.

The selection of an appropriate ISI span N was performed by jointly examining the BER behavior of the GRU-based detector across different N values for $\tau = 0.7$, $\tau = 0.8$, and $\tau = 0.9$, and by analyzing the one-sided ISI coefficients listed in Table 2.1. For example, the third column of Table 2.1 (corresponding to $\tau = 0.7$) shows a sharp drop in coefficient magnitude—from 1.37×10^{-2} to 3.31×10^{-4}—indicating that ISI contributions beyond the sixth symbol offset are negligible. This suggests that $N = 6$ sufficiently captures the dominant ISI components for $\tau = 0.7$. This choice was further validated through BER simulations conducted for different ISI spans, as reported in Section 4.5.1.

For BPSK modulation, the received samples are directly processed by a single GRU layer. For QPSK modulation, each complex-valued input symbol is decomposed into its real ($\Re$) and imaginary ($\Im$) components, which are processed in parallel by two independent GRU branches, as illustrated in Fig. 4.3(b). The outputs of these branches are concatenated and passed through a fully connected (FC) layer followed by a softmax activation. The resulting probability vector represents the likelihood of each symbol class, and the final hard decision is obtained by selecting the class with the highest softmax output.

To maintain computational efficiency and ensure suitability for real-time applications, a single-layer GRU architecture is adopted in this work. Although deeper or stacked GRU networks may offer additional performance gains under severe ISI conditions, our objective is to demonstrate that even a shallow GRU architecture—when structured to leverage FTN-specific ISI patterns—can achieve near-optimal detection performance with significantly reduced complexity for $\tau \geq 0.7$.

The GRU-based FTN detection process can be expressed mathematically as follows. Let the received sample vector be given by $\mathbf{y} = [y_0, y_1, \ldots, y_{K-1}]^T$, as defined in Equation **2.8**. The objective of the detector is to estimate the transmitted symbol a_k using a window of $L = 2N + 1$ received samples centered at y_k.

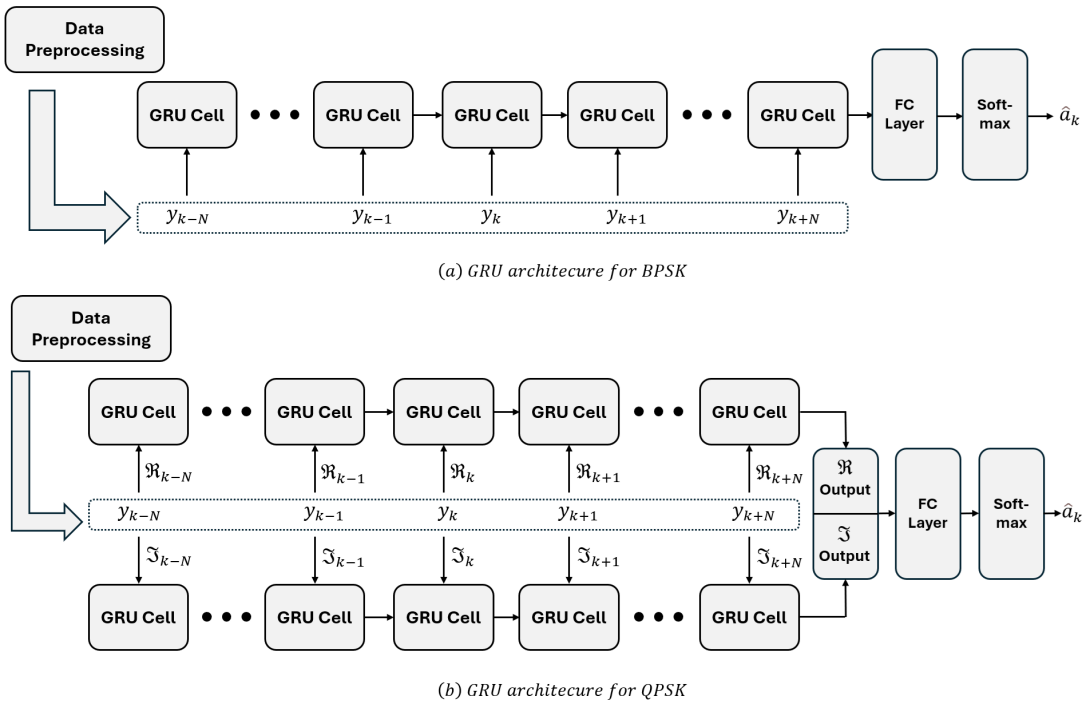


Figure 4.3 Gated Recurrent Unit (GRU)-based faster-than-Nyquist (FTN) signaling detector architecture for binary phase shift keying (BPSK) and quadrature phase shift keying (QPSK) modulations.

For each valid symbol index $k \in \{N, N+1, \ldots, K-N-1\}$, the input vector is constructed as in Equation 4.1. This windowed representation is then fed into the GRU network, which extracts temporal features associated with the ISI structure. The GRU maps the input sequence $\{\mathbf{Y}_k\}$ to a hidden representation according to

$$\mathbf{h}_k = \mathrm{GRU}(\mathbf{Y}_k; \theta), \tag{4.12}$$

where θ denotes the set of trainable parameters, including the weights and biases of the update gate, reset gate, and candidate hidden state.

The hidden state $\mathbf{h}_k$ is then passed through a fully connected layer followed by an appropriate output activation function—sigmoid for BPSK or softmax for QPSK—to generate the predicted symbol probabilities:

$$\hat{\mathbf{p}}_k = \phi(\mathbf{W}\mathbf{h}_k + \mathbf{b}), \tag{4.13}$$

where $\phi(\cdot)$ is the activation function and $\mathbf{W}$, $\mathbf{b}$ denote the weight matrix and bias vector of the output layer.

A hard decision is obtained by selecting the symbol class corresponding to the maximum predicted probability:

$$\hat{a}_k = \arg\max_i \hat{\mathbf{p}}_k[i], \tag{4.14}$$

where $\hat{\mathbf{p}}_k[i]$ denotes the i-th entry of the output probability vector.

This procedure is repeated for all indices k for which a complete input window is available, ultimately producing the estimated transmitted sequence $\hat{\mathbf{a}} = [\hat{a}_N, \hat{a}_{N+1}, \ldots, \hat{a}_{K-N-1}]^T$.

4.4 Comparative Analysis of Training Optimization Techniques

Training deep neural networks for FTN signaling detection requires navigating highly non-convex loss surfaces, where optimization methods play a critical role in achieving stable convergence, reducing training time, and avoiding suboptimal local minima. In this study, we evaluate and compare two widely used optimizers—ADAM [34] and NADAM [35]—to determine the most effective training strategy for the proposed GRU-based FTN detector. NADAM extends the standard ADAM formulation by incorporating Nesterov momentum [36], providing more informed gradient updates and often improving convergence behavior in practice.

ADAM updates network parameters by computing exponentially decaying averages of both the gradients (first moment) and squared gradients (second moment). Specifically,

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \qquad v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2, \tag{4.15}$$

where $g_t = \nabla_\theta f(\theta_{t-1})$ denotes the gradient of the loss function $f(\theta)$ at iteration t, while m_t and v_t represent first- and second-moment estimates. The decay parameters $\beta_1, \beta_2 \in [0, 1)$ control how heavily past gradients influence the current estimates.

Since both moment estimates are initialized at zero, bias correction is applied to

compensate for the initialization effect:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \qquad \hat{v}_t = \frac{v_t}{1-\beta_2^t}. \tag{4.16}$$

The parameter update step uses an adaptive learning rate for each parameter dimension:

$$\theta_t = \theta_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}, \tag{4.17}$$

where α is the global learning rate and ϵ is a small constant ensuring numerical stability. Although ADAM provides adaptive step sizes and demonstrates strong performance in many applications, it does not account for anticipated gradient directions. As a result, ADAM may exhibit oscillatory behavior or slow convergence, particularly in noisy and highly non-convex optimization landscapes such as those encountered in FTN detection.

NADAM addresses this limitation by incorporating Nesterov momentum into ADAM's update formulation. Instead of applying momentum based solely on past gradients, NADAM introduces a "look-ahead" mechanism that anticipates the next gradient position before applying the update. In this framework, the effective first-moment estimate becomes

$$\hat{m}_t = \frac{\beta_1 m_{t-1} + (1-\beta_1) g_t}{1-\beta_1^t}, \tag{4.18}$$

where the term $\beta_1 m_{t-1}$ reflects the anticipated gradient direction. This modification enables smoother, more stable updates and often leads to improved convergence, especially when the optimization surface contains sharp curvature or multiple local minima.

In the subsequent subsection, the practical impact of NADAM versus ADAM is analyzed using training loss behavior, demonstrating that NADAM yields consistently faster and more stable convergence for the GRU-based FTN detector.

This adjustment reflects the combined influence of the current gradient g_t and the accumulated momentum from previous gradients m_{t-1}. Consequently, the parameter

update incorporates this anticipatory correction through the adaptive learning-rate formulation in (**4.17**). By explicitly accounting for the expected future gradient direction, NADAM refines the update trajectory, reducing overshooting and mitigating oscillatory behavior—effects commonly observed in noisy or highly non-convex optimization landscapes.

This anticipatory mechanism yields smoother parameter transitions and accelerates convergence, particularly in deep neural network training where the loss surface contains numerous local minima and saddle points, as is characteristic of FTN signaling detection problems. Within the proposed architecture, NADAM enables more stable and adaptive parameter evolution, leading to improved convergence behavior relative to ADAM. These benefits are especially relevant for FTN models, where strong ISI patterns and high-dimensional optimization challenges necessitate robust and stable learning dynamics. By integrating NADAM into the training procedure, the proposed detector achieves enhanced reliability and efficiency, offering a substantive contribution to optimization methodologies applied in FTN detection.

From a computational perspective, we compare the GRU-based FTN detector trained with NADAM against the same architecture trained using ADAM, focusing on convergence speed under identical experimental conditions. The comparison is carried out by evaluating the number of iterations required for each optimizer to reach a predefined validation loss threshold, a standard metric for assessing generalization performance during training. The validation loss (`val_loss`) is computed over a dedicated validation set that is separate from the training data, serving as an indicator of the model's ability to generalize to unseen examples. Monitoring `val_loss` provides insight into whether the model is improving consistently during training, and allows us to determine the iteration or epoch at which convergence is achieved without overfitting.

Table 4.2 Convergence Metrics for GRU-Based FTN Detectors Optimized with NADAM and ADAM Optimizers

Optimizer	**Validation Loss Target**	**Number of Iterations**
NADAM	3×10^{-4}	84,375
ADAM	3×10^{-4}	102,093

In this study, a validation loss threshold of 3×10^{-4} was selected as the stopping criterion to evaluate the convergence speed of the two optimization methods. This threshold reflects a sufficiently low error level for the FTN detection task and provides a rigorous yet practical benchmark for assessing optimizer performance. By fixing the target val_loss, the analysis isolates the intrinsic behavior of each optimizer, ensuring that the comparison focuses solely on convergence efficiency rather than absolute performance differences. Convergence speed was quantified by counting the number of training iterations required for each model to reach the specified validation loss. This approach offers a hardware-agnostic and implementation-independent measure of computational effort, making the comparison robust and fair.

As reported in Table 4.2, the GRU-based FTN detector trained with NADAM reached the target validation loss of 3×10^{-4} in 84,375 iterations, whereas the version trained with ADAM required 102,093 iterations to attain the same performance level. These results indicate that NADAM achieves convergence approximately 20% faster than ADAM. This improvement is primarily attributed to NADAM's effective integration of Nesterov momentum with adaptive moment estimation, enabling more anticipatory and stable parameter updates. In the context of FTN signaling detection—where strong ISI and highly non-convex optimization landscapes are prevalent—the characteristics of NADAM align well with the underlying network topology, resulting in more efficient convergence behavior within the proposed GRU-based framework.

The results of this comparative analysis highlight the computational advantages of NADAM, especially for complex tasks such as FTN signaling detection. Its faster convergence substantially shortens the overall training process and improves the efficiency of computational resource usage. These findings position NADAM as a highly effective optimization method for deep learning models in time-sensitive scenarios, where rapid and stable convergence is essential.

4.5 Simulation Results and Complexity Comprasion

This section provides a comprehensive evaluation of the proposed FTN signaling detection framework, examining both bit error rate (BER) performance and

computational complexity. The BER results are presented under various network configurations and optimization strategies to demonstrate the effectiveness of the NADAM-optimized GRU-based detector. In parallel, a detailed complexity comparison is carried out between the proposed detector and the conventional M-BCJR algorithm. To ensure a fair assessment, the computational analysis incorporates FPGA-based hardware resource estimation using LUT utilization as a standardized metric, enabling a meaningful comparison of overall computational cost. Together, these analyses highlight the balance between accuracy and efficiency offered by the proposed approach and underscore its suitability for practical FTN communication systems.

All training procedures were implemented in TensorFlow using a single GPU, with early stopping applied based on validation loss to prevent overfitting. A fixed random seed (42) was used during dataset generation and model initialization to ensure reproducibility across different runs. Independent models were trained for each compression factor τ, and training for a given τ was performed jointly across an SNR range of 7–10 dB to enhance generalization. The number of GRU hidden units was selected according to the severity of ISI, using 5, 9, and 13 units for $\tau = 0.9$, 0.8, and 0.7, respectively. The primary network and training hyperparameters—including batch size, optimizer parameters, learning rate schedule, and weight initialization (Glorot uniform)—are summarized in Table 4.3.

4.5.1 BER Results

This subsection presents the BER performance of the proposed FTN signaling detection framework under varying network configurations and optimization strategies.

Figure 4.4 summarizes two key comparisons. First, the GRU-based FTN detector optimized with NADAM is benchmarked against the optimal BCJR algorithm for both BPSK and QPSK modulations. Since the BER curves for BPSK and QPSK are nearly identical, they are plotted jointly without separate labels for improved readability. The results indicate that the NADAM-optimized GRU detector achieves BER performance remarkably close to that of the BCJR algorithm for all tested SNR values when $\tau \geq 0.7$,

Table 4.3 Summary of Network and Training Hyperparameters

Hyperparameter	**Value**
Number of GRU layers	1
GRU hidden units	5 ($\tau = 0.9$), 9 ($\tau = 0.8$), 13 ($\tau = 0.7$)
Activation function (output)	Softmax (QPSK), Sigmoid (BPSK)
Output type	Soft probabilities
Batch size	128
Learning rate (initial)	0.001
Optimizer	NADAM
Loss function	Binary cross-entropy (BPSK), Categorical cross-entropy (QPSK)
Epochs (maximum)	50
Early stopping criteria	Validation loss ($\Delta < 10^{-4}$ over 5 epochs)
Dropout	Not used
Gradient clipping	Enabled, clip norm = 1.0
Training/Validation split	90% / 10% of 4×10^6 samples
Test set	Independently generated, 10^6 samples
Random seed	Fixed at 42
SNR training strategy	Joint training over 7–10 dB SNR range (per τ)
Model weight initialization	Default Xavier/Glorot uniform

demonstrating its effectiveness in capturing the dominant ISI characteristics inherent in FTN signaling.

The second comparison in Fig. 4.4 evaluates the impact of the NADAM and ADAM optimizers on the GRU-based detector across different τ values. Although the performance of both optimizers is similar in the low-SNR regime, clear performance differences emerge at higher SNRs, particularly under stronger ISI conditions. For instance, at $\tau = 0.8$ and a BER of 2.9×10^{-5}, the NADAM-trained model outperforms the ADAM-trained model by approximately 0.55 dB. This performance gap widens as ISI becomes more dominant; at $\tau = 0.7$ and a BER of 7×10^{-5}, the SNR difference increases to nearly 0.95 dB. These observations reflect the advantages of NADAM in handling highly non-convex optimization landscapes. By integrating Nesterov accelerated gradients into adaptive moment estimation, NADAM provides more stable and anticipatory parameter updates, enabling superior learning of complex ISI patterns compared to ADAM. As a result, the NADAM-optimized GRU detector consistently delivers improved high-SNR performance in FTN signaling scenarios.

Figure 4.5 compares the BER performance of the GRU-based FTN detector with that of an LSTM-based detector, where both models are optimized using the NADAM algorithm. The performance of the two architectures is nearly identical; for instance, at

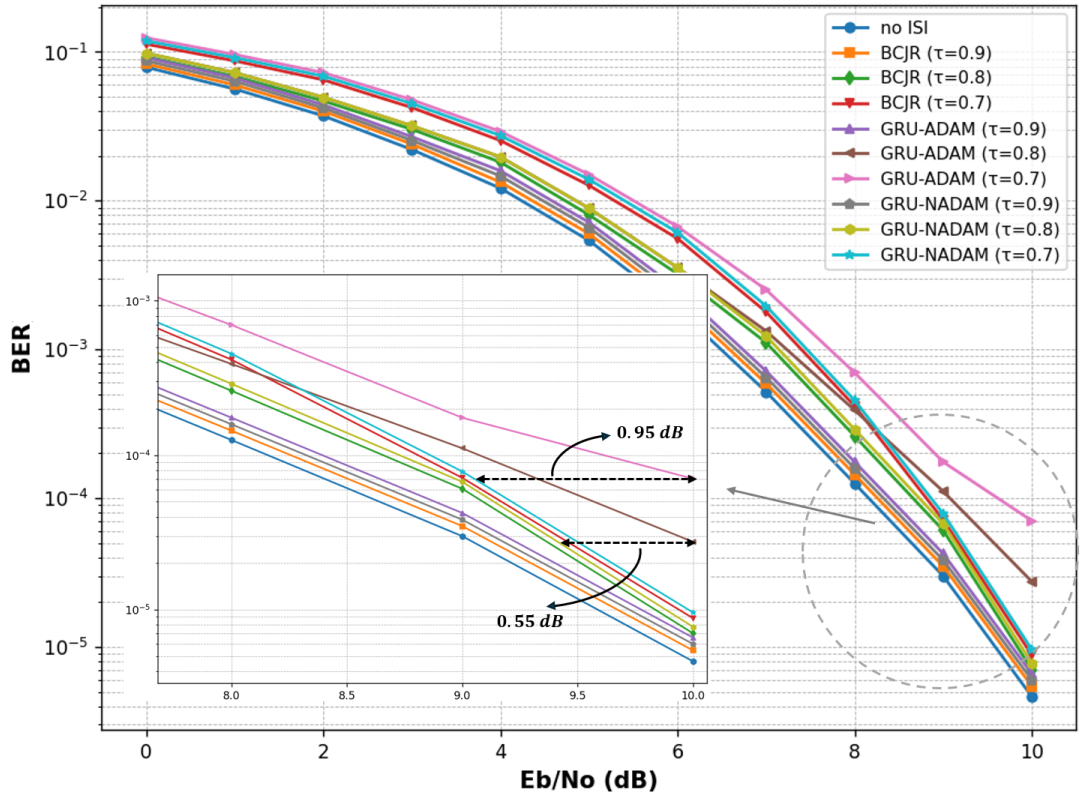


Figure 4.4 BER performance comparison between the proposed NADAM optimized GRU-based FTN detector and BCJR algorithm for different compression factors ($\tau = 0.9$, $\tau = 0.8$, and $\tau = 0.7$), with a roll-off factor of $\beta = 0.35$.

a BER of 10^{-5}, the SNR difference between the GRU-based and LSTM-based detectors is less than 0.1 dB. This result confirms that the GRU architecture achieves the same level of detection accuracy as LSTM, despite having a simpler internal structure and reduced parameter count. Consequently, GRUs offer a more computationally efficient solution for FTN signaling detection without compromising BER performance.

To further assess the effectiveness of the proposed GRU-based detector, we compare its BER performance with the approaches presented in [2], [3], [4], [5], and [6]. For fairness and consistency, all methods are evaluated under the same FTN configuration: $\tau = 0.7$ and $\beta = 0.5$. The choice of $\beta = 0.5$ provides a balanced benchmark scenario where ISI is moderate. This setting is especially important when comparing against low-complexity algorithms such as those in [5] and [3], which may experience noticeable degradation under more severe ISI conditions (e.g., smaller roll-off factors). Using a moderate roll-off ensures that the comparison reflects realistic operating conditions without disproportionately favoring more

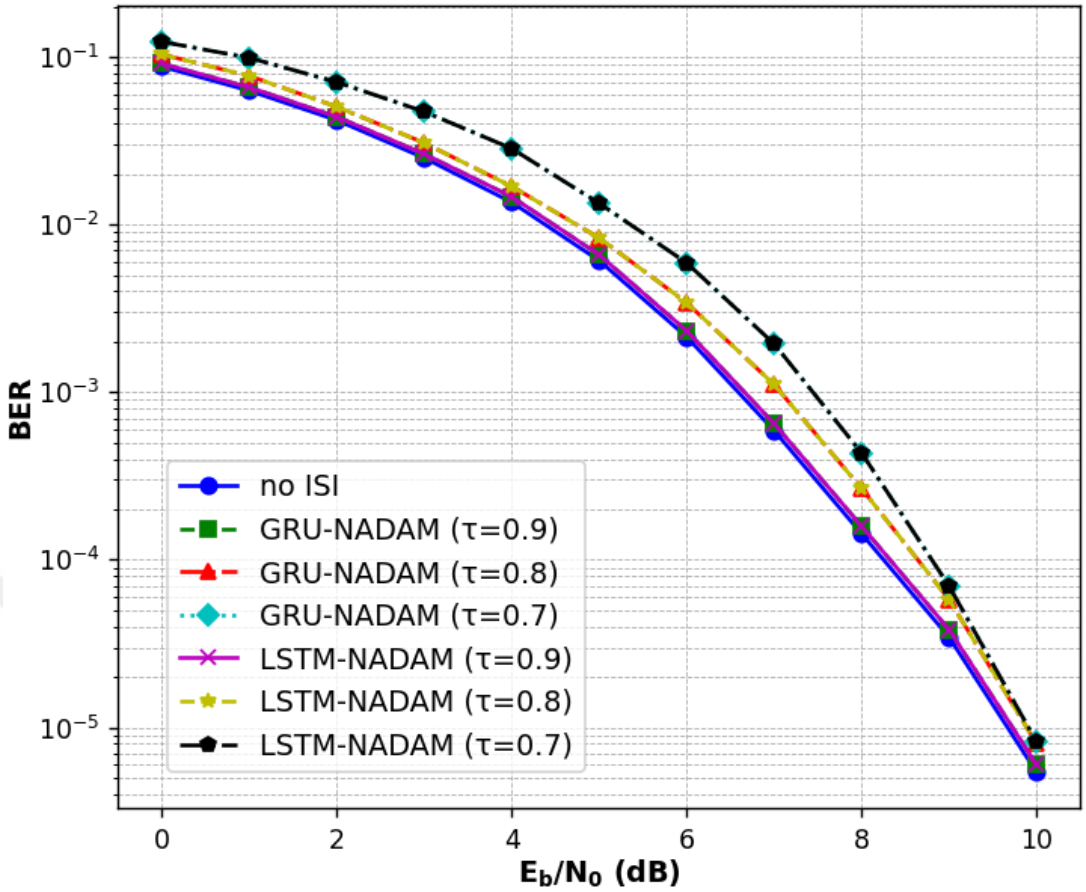


Figure 4.5 BER performance comparison between the proposed NADAM optimized GRU-based FTN detector and LSTM-based FTN detector for different compression factors ($\tau = 0.9$, $\tau = 0.8$, and $\tau = 0.7$), with a roll-off factor of $\beta = 0.35$.

computationally intensive detection schemes.

As shown in Fig. 4.6, the proposed GRU-based detector consistently outperforms the methods in [2], [3], [4], [5], and [6] across the entire evaluated SNR range. Notably, when compared with the closest-performing deep learning approach in [4], the proposed detector achieves an SNR gain of approximately 1.2 dB at a BER of 7×10^{-5}. Moreover, while the proposed detector attains the same near-optimal BER performance as the Cholesky and GTMH precoding schemes, it does so with substantially lower computational complexity, as detailed in Section 4.5.2. These results reaffirm the robustness, efficiency, and practical viability of the proposed GRU-based FTN detector in challenging ISI-dominated scenarios.

To determine the optimal value of N (the one-sided ISI length) for the proposed system, the BER performance was evaluated for different N values across $\tau = 0.7$, $\tau = 0.8$, and

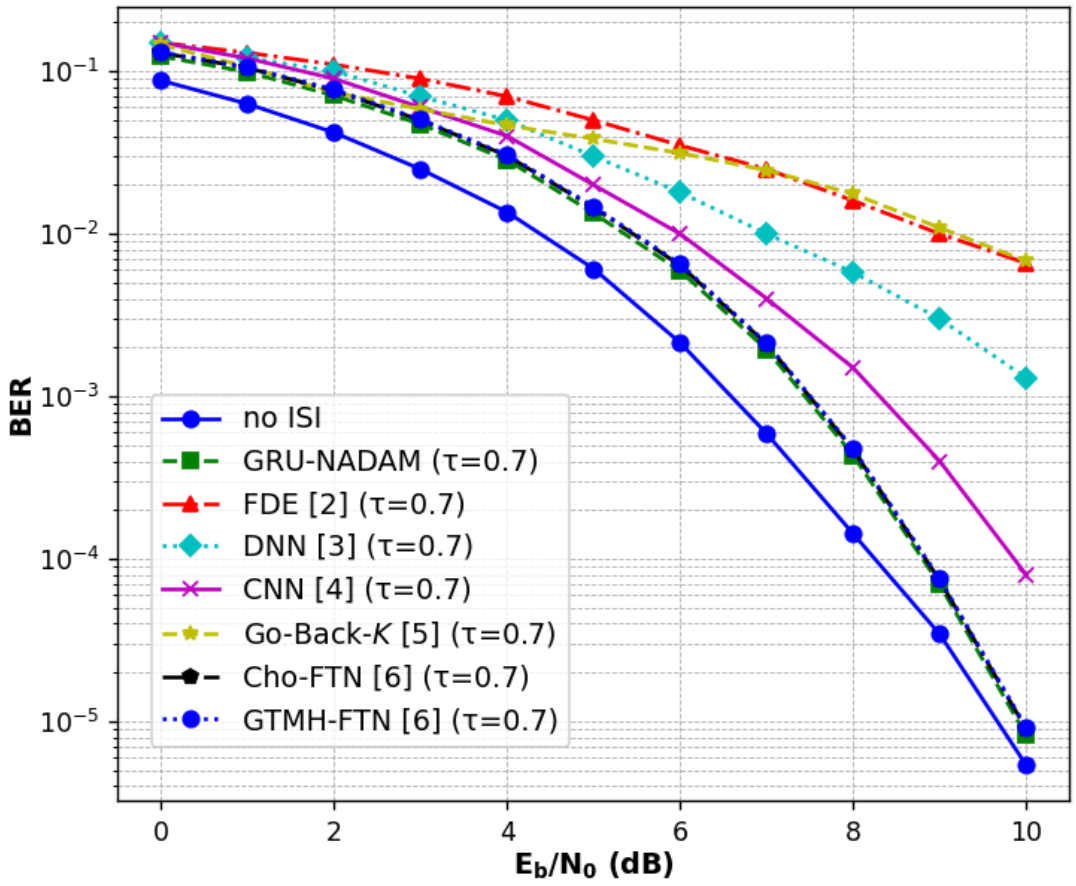


Figure 4.6 Comparison of the BER performance of the proposed NADAM Optimized GRU-based FTN detector with the methods in [2], [3], [4], [5], and [6] for $\tau = 0.7$ and $\beta = 0.5$.

$\tau = 0.9$. The selection process began by examining the one-sided ISI coefficients listed in Table 2.1 to assess the relative contribution of interference terms at various symbol offsets. For example, for $\tau = 0.7$, the third column of Table 2.1 exhibits a sharp reduction in coefficient magnitude, dropping from 1.37×10^{-2} to 3.31×10^{-4}. This abrupt decline suggests that an N value of approximately 6 is sufficient to capture the dominant ISI contributions. To validate this analytically motivated choice, BER simulations were conducted for multiple N values, as illustrated in Fig. 4.7. The results show that the BER curves for $N = 6$ and $N = 7$ are nearly identical, whereas configurations with $N < 6$ exhibit marked performance degradation. This confirms that $N = 6$ adequately models the significant ISI terms for $\tau = 0.7$ without introducing unnecessary computational overhead.

A similar procedure was applied to determine appropriate N values for the remaining τ configurations. For $\tau = 0.8$, the ISI coefficient magnitudes and BER evaluations jointly

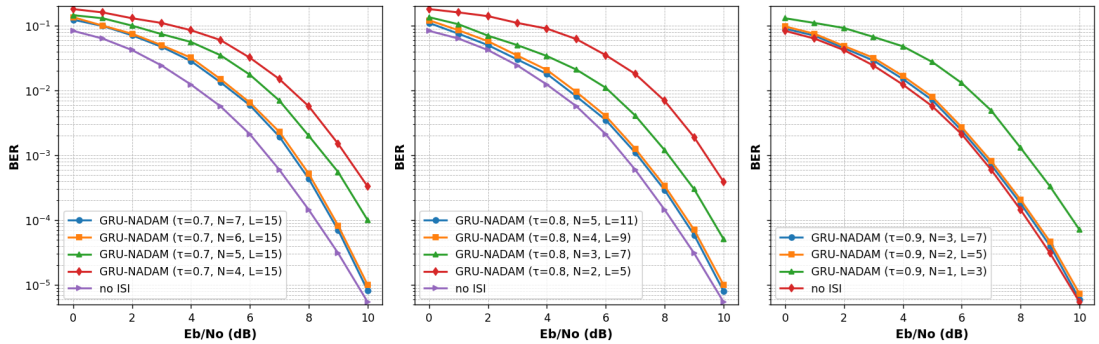


Figure 4.7 BER performance analysis of the GRU-based FTN detector for varying values of N at different τ values.

indicate that $N = 4$ provides an effective balance between accuracy and complexity. For $\tau = 0.9$, the ISI contributions decay more rapidly, and the performance analysis shows that $N = 2$ is sufficient. These findings highlight the importance of selecting N adaptively with respect to τ to ensure accurate ISI modeling while maintaining computational efficiency.

To further assess the robustness of the proposed GRU-based FTN detector in realistic wireless environments, additional experiments were conducted over a quasi-static multipath Rayleigh fading channel. A three-tap equal-power Rayleigh fading model was adopted, with tap delays separated by τT, causing each multipath component to align with a subsequent symbol position and thereby intensifying the ISI. Under the quasi-static assumption, the fading coefficients remain constant within each frame but change independently from frame to frame during both training and testing, representing slowly varying mobile channels.

The GRU-based detector was evaluated under this fading model for $\tau = 0.9$, $\tau = 0.8$, and $\tau = 0.7$, using QPSK modulation, and its performance was benchmarked against the optimal BCJR algorithm under identical conditions. As shown in Fig. 4.8, the GRU-based detector exhibits strong robustness to multipath fading, with BER curves closely following those of the BCJR algorithm across all τ values. Only minor deviations are observed for $\tau = 0.7$, where ISI is most severe; for example, at a BER of 10^{-4}, the SNR difference is approximately 0.2 dB. These results demonstrate that the proposed GRU-based architecture effectively learns the combined effects of FTN-induced ISI and multipath propagation. Its ability to generalize beyond AWGN

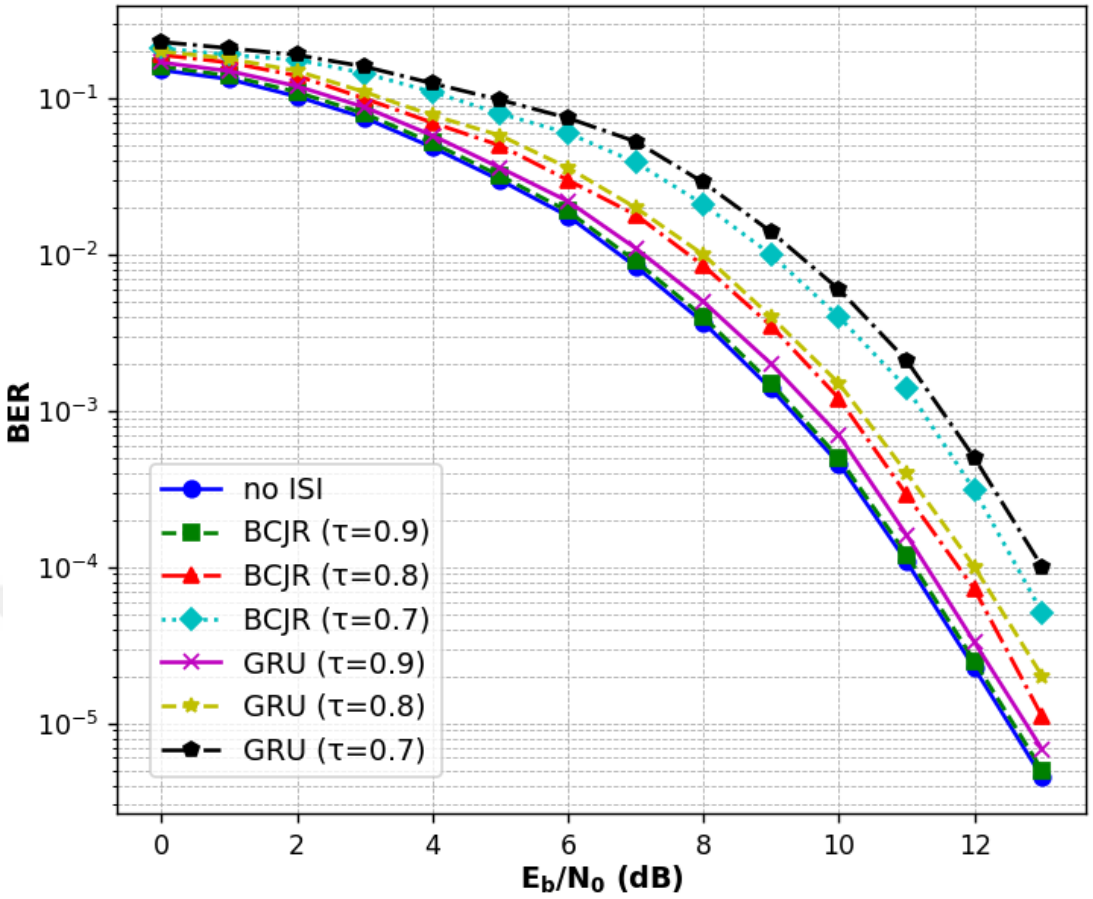


Figure 4.8 BER performance of the proposed GRU-based detector under quasi-static multipath Rayleigh fading channel for $\tau = 0.7$, $\tau = 0.8$, and $\tau = 0.9$ with QPSK modulation. The benchmark BCJR results are also included for comparison.

conditions confirms its suitability for deployment in practical FTN systems operating in realistic wireless channels.

To assess the performance of the proposed GRU-based FTN detector under practical communication settings, we incorporate forward error correction using LDPC codes. LDPC coding is widely adopted in modern communication standards such as 5G NR and IEEE 802.11, owing to its capacity-approaching performance achieved through iterative decoding. In this work, we employ a standardized LDPC code with block length $n = 1056$ bits and code rate $R = 1/2$, corresponding to $k = 528$ information bits and 528 parity bits. The parity-check matrix is derived from the base graph structures defined in the 3GPP 5G NR specifications (3GPP TS 38.212), ensuring compatibility with real-world implementation scenarios and robust decoding performance.

At the transmitter, the input bitstream is encoded using the LDPC encoder, and the

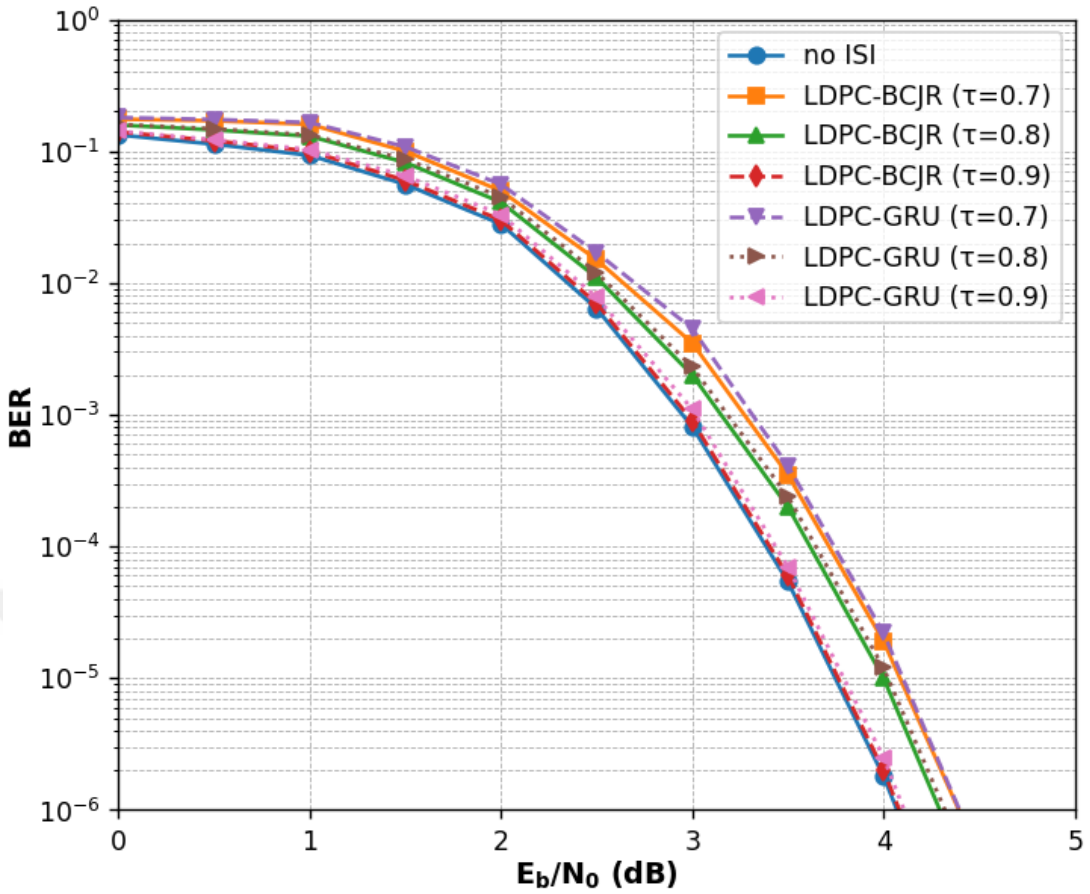


Figure 4.9 BER performance comparison of LDPC-coded FTN systems using BCJR-based and GRU-based detectors for $\tau = 0.7$, $\tau = 0.8$, and $\tau = 0.9$. All configurations use a rate-1/2 LDPC code of length 1056 bits under BPSK modulation over an AWGN channel.

resulting 1056-bit codeword is modulated using BPSK, where bits $\{0, 1\}$ are mapped to symbols $\{+1, -1\}$. The modulated symbols are pulse-shaped with an RRC filter and transmitted using FTN signaling with time compression factors $\tau = 0.9$, 0.8, and 0.7. Due to intentional time packing, the received signal is affected by both ISI and AWGN. At the receiver, the signal is match-filtered and sampled at intervals of τT. The sampled sequence is then processed by the proposed GRU-based FTN detector, which is trained to capture and mitigate the structured ISI patterns characteristic of FTN signaling. For each target bit, a window of $(2N + 1)$ adjacent received samples is fed into the GRU network, producing a real-valued soft estimate representing the likelihood of the central symbol. These soft estimates are converted into log-likelihood ratios (LLRs) and passed to the LDPC decoder, which operates using the belief propagation (BP) algorithm with a maximum of 50 iterations. By exploiting the redundancy of the LDPC code and the soft information provided by the GRU-based detector, the decoder refines the symbol

estimates and enhances overall detection reliability.

To benchmark coded performance, we compare the GRU-based detector with the optimal BCJR algorithm under equivalent FTN signaling conditions. The BER curves for both systems across a wide SNR range are presented in Fig. 4.9. The incorporation of LDPC coding significantly improves robustness and enables the proposed GRU-based detector to achieve performance levels nearly identical to those of the BCJR algorithm, while still offering substantial reductions in computational complexity. These results demonstrate the practical viability of the proposed deep learning-based detection framework in coded FTN communication systems.

4.5.2 Computational Complexity Comparison

A critical factor in assessing the practicality of any FTN detection algorithm is its computational efficiency, especially in scenarios where hardware resources or processing time are limited. To this end, we conduct a detailed comparison between the inference complexity of the proposed GRU-based FTN detector and the conventional M-BCJR algorithm. The comparison focuses on quantifying the number of operations required to detect 100 symbols under both BPSK and QPSK modulation schemes, evaluated across different compression factors $\tau \in \{0.7, 0.8, 0.9\}$.

By evaluating the total number of additions, multiplications, and nonlinear operations, this analysis provides a clear perspective on the computational burden imposed by each method during symbol detection. The operation counts for BPSK- and QPSK-modulated signals are summarized in Table 4.4 and Table 4.5, respectively, with the M-BCJR values referenced from [31].

Since the GRU-based detector and the M-BCJR algorithm rely on fundamentally different types of operations, a comparison based solely on raw operation counts would not yield an accurate assessment of their computational demands. To establish a fair and meaningful basis for comparison, we employed a widely used FPGA resource estimation tool to determine the hardware cost of each operation in terms of lookup tables (LUTs). This approach captures the heterogeneous computational nature of the two algorithms and translates their operation sets into a unified

Table 4.4 Comparison of Operation Numbers for M-BCJR and GRU-Based FTN Detector (100 Symbols) for BPSK Modulation

Operation	**Algorithm**	$\tau = 0.7$	$\tau = 0.8$	$\tau = 0.9$
Multiplication	M-BCJR	4631	2783	1877
	GRU	11700	8100	4500
Addition	M-BCJR	9982	5129	2584
	GRU	10400	7200	4000
Comparison	M-BCJR	4160	1056	272
	GRU	–	–	–
Division	M-BCJR	3982	2981	1436
	GRU	–	–	–
Exponential	M-BCJR	205	204	203
	GRU	–	–	–
Tanh	M-BCJR	104	103	102
	GRU	1300	900	500
Sigmoid	M-BCJR	–	–	–
	GRU	2600	1800	1000

Table 4.5 Comparison of Operation Numbers for M-BCJR and GRU-Based FTN Detector (100 Symbols) for QPSK Modulation

Operation	**Algorithm**	$\tau = 0.7$	$\tau = 0.8$	$\tau = 0.9$
Multiplication	M-BCJR	33048	19964	13916
	GRU	23400	16200	9000
Addition	M-BCJR	70857	36137	18671
	GRU	20800	14400	8000
Comparison	M-BCJR	29283	7439	1976
	GRU	–	–	–
Division	M-BCJR	28819	21857	9488
	GRU	–	–	–
Exponential	M-BCJR	410	408	406
	GRU	–	–	–
Tanh	M-BCJR	208	206	204
	GRU	2600	1800	1000
Sigmoid	M-BCJR	–	–	–
	GRU	5200	3600	2000

hardware-oriented metric. All LUT estimates were obtained under the assumption of 10-bit numerical precision, and the corresponding resource utilizations for each operation type are summarized in Table 4.6.

Table 4.6 LUT Utilizations for Different Operations (10-bit)

Operation	**LUT Utilization**
Multiplication	113
Addition	10
Comparison	11
Division	236
Exponential	73
tanh	1
sigmoid	1

Using the operational data presented in Table 4.4 and Table 4.5, together with the LUT-based hardware requirements listed in Table 4.6, a unified comparative metric was developed to evaluate the overall computational complexity of the GRU-based FTN detector and the M-BCJR algorithm. This metric is obtained by weighting each operation count with its corresponding LUT utilization value, thereby reflecting not only the number of operations but also their hardware implementation cost. By jointly considering both the type and quantity of operations, the resulting measure provides a comprehensive and hardware-aware assessment of computational efficiency.

The outcomes of this analysis, summarized in Table 4.7, show that the proposed GRU-based FTN detector achieves substantial reductions in computational cost relative to the M-BCJR algorithm. Specifically, as detailed in Table 4.8, the GRU-based detector reduces LUT-weighted complexity by approximately 12% for BPSK and 76% for QPSK at $\tau = 0.7$. This performance gap becomes increasingly pronounced as the modulation order increases. While the computational complexity of the M-BCJR algorithm grows approximately quadratically with the modulation order, the GRU-based detector exhibits near-linear scaling. This behavior underscores the suitability of the GRU-based approach for higher-order modulation schemes such as QPSK, where its computational efficiency becomes particularly advantageous.

This improvement in computational efficiency underscores the advantages of the proposed GRU-based detector, particularly in scenarios where lower τ values are employed. The results confirm the practicality of the GRU-based approach as a resource-efficient alternative to the M-BCJR algorithm for FTN signaling detection when $\tau \geq 0.7$.

Table 4.7 LUT-Weighted Computational Cost Metrics for M-BCJR and GRU-Based FTN Detector (BPSK and QPSK Modulations)

Modulation	**Algorithm**	$\tau = 0.7$	$\tau = 0.8$	$\tau = 0.9$
BPSK	M-BCJR	1623704	1095896	594750
	GRU	1430000	990000	550000
QPSK	M-BCJR	11596529	7887373	4049964
	GRU	2860000	1980000	1100000

Table 4.8 Percentage Improvement of GRU over M-BCJR in LUT-Weighted Computational Cost Metric

Modulation	$\tau = 0.7$	$\tau = 0.8$	$\tau = 0.9$
BPSK	12%	10%	8%
QPSK	76%	75%	73%

Beyond comparison with the optimal M-BCJR algorithm, we also evaluated the computational efficiency of the proposed GRU-based FTN detector relative to several notable existing FTN detection methods, including those presented in [2], [3], [4], [5], and [6]. The CNN- and DNN-based detectors proposed in [2], [3], and [4] require approximately 8194 additions and 8196 multiplications, 9720 additions and 9720 multiplications, and roughly 53,500 multiply-accumulate (MAC) operations per detected symbol, respectively. Likewise, the Cholesky- and GTMH-based FTN detectors in [6] require on the order of 5000 additions and 5000 multiplications for each symbol.

In contrast, the proposed GRU-based detector achieves competitive or superior detection accuracy while requiring substantially fewer operations than these approaches. Although the Go-Back-K algorithm in [5] offers lower computational complexity than the proposed GRU-based detector, this reduction comes at a notable performance cost. As demonstrated in Fig. 4.6, the proposed detector significantly outperforms [5] across the evaluated SNR range, particularly under moderate-to-severe ISI conditions.

Overall, these results highlight the favorable trade-off achieved by the proposed model: it not only reduces computational complexity relative to the majority of classical and deep-learning-based FTN detection schemes, but also maintains high detection accuracy across different operating conditions. This dual advantage

reinforces the suitability of the GRU-based detector as an efficient and reliable solution for practical FTN signaling systems.

4.6 Summary of the Chapter

This chapter presented a GRU-based deep learning framework for faster-than-Nyquist (FTN) signaling detection, together with an extensive performance and complexity analysis. The proposed approach introduced a structured GRU input design aligned with the one-sided ISI span, enabling the network to learn FTN-induced interference patterns efficiently. A comparative study of optimization methods demonstrated that NADAM offers faster and more stable convergence than ADAM, particularly under non-convex learning conditions typical of FTN detection.

Simulation results showed that the GRU-based detector achieves near-optimal BER performance for $\tau \geq 0.7$, closely matching the BCJR benchmark while significantly reducing computational complexity. Additional evaluations under quasi-static Rayleigh fading and LDPC-coded transmission confirmed the robustness and practical viability of the proposed approach. Furthermore, complexity analyses highlighted the advantages of GRUs over LSTMs and demonstrated notable efficiency gains relative to classical and deep-learning-based FTN detectors.

Despite these strengths, some limitations remain. Performance under more severe ISI conditions (e.g., $\tau < 0.7$) and under time-varying or imperfect channel estimation scenarios has not been fully explored. Moreover, the current architecture employs a single-layer GRU with fixed input size, suggesting opportunities for enhancements such as deeper recurrent models, bidirectional processing, or attention mechanisms.

Overall, the findings of this chapter illustrate the potential of GRU-based architectures as computationally efficient and high-performing detectors for FTN systems. The insights gained here lay the groundwork for the subsequent chapter, which introduces a more advanced detection framework incorporating bidirectional recurrent processing and position-aware attention mechanisms.

CHAPTER 5

BI-GRU-BASED FTN SIGNALING DETECTION WITH POSITION-AWARE ATTENTION MECHANISM

This chapter introduces a deep learning-based detection framework for faster-than-Nyquist (FTN) signaling that combines a bidirectional gated recurrent unit (Bi-GRU) architecture with a position-aware attention (PAA) mechanism. The goal of the proposed method is to more effectively mitigate inter-symbol interference (ISI) by (i) capturing both forward and backward temporal dependencies in the received signal and (ii) adaptively emphasizing input symbols according to their positional contribution to ISI.

As illustrated in Fig. 5.1, the detection system operates on the matched-filtered output and processes it through a set of structured data arrangement steps that prepare the input sequence for Bi-GRU-based sequential modeling. A key distinguishing feature of this framework is the incorporation of a position-aware attention mechanism, which assigns symbol-specific attention weights informed by the ISI structure of FTN signaling. This enables the model to selectively prioritize the most influential neighboring symbols, enhancing detection performance while avoiding unnecessary computational overhead.

The overall architecture of the proposed Bi-GRU-based FTN detector, augmented with PAA, is depicted in Fig. 5.2. The system integrates three core components: (i) synthetic dataset generation tailored to FTN-induced ISI patterns, (ii) ISI-aware preprocessing that aligns input windows with the one-sided ISI span, and (iii) a Bi-GRU model equipped with structured attention to exploit the position-dependent nature of ISI. The methodology for designing, training, and evaluating this architecture is presented in

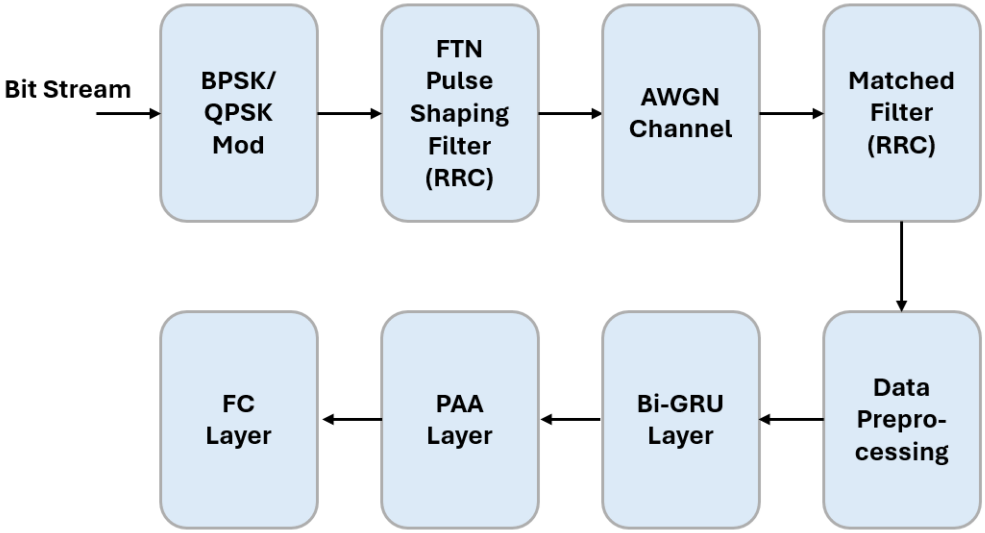


Figure 5.1 FTN signaling-based communication system with proposed signal detector.

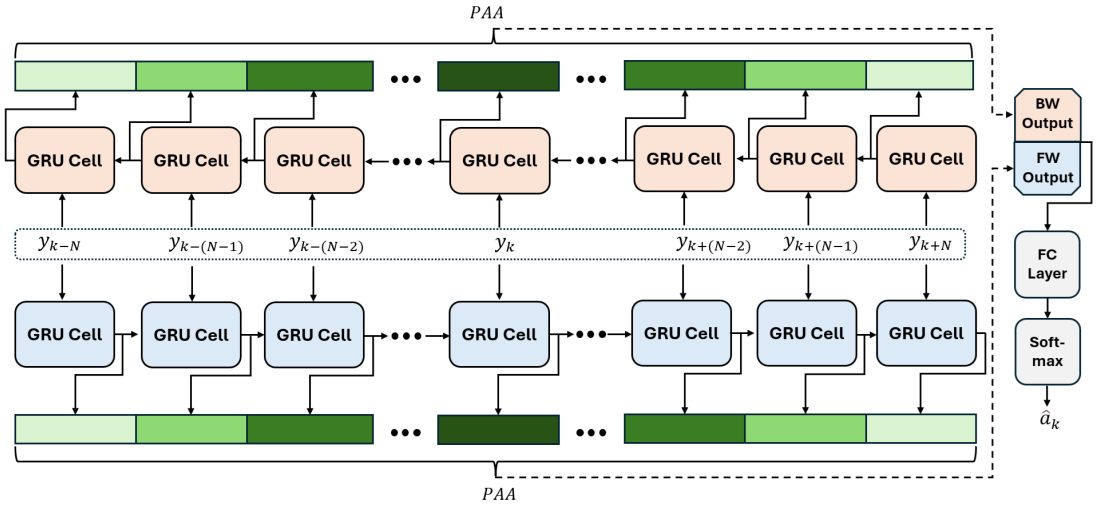


Figure 5.2 Bi-GRU-based faster-than-Nyquist (FTN) signaling detector architecture with Position-Aware Attention mechanism.

detail in the following subsections.

5.1 Dataset Generation and Training Hyperparameters

To train the proposed Bi-GRU-based FTN detector enhanced with the position-aware attention (PAA) mechanism, large-scale synthetic datasets were generated in accordance with the FTN system model described in Chapter 2. Since ISI becomes more dominant at moderate-to-high SNR levels, training samples were drawn from the 6–10 dB SNR range to enable the network to effectively learn the underlying interference structure.

Two independent datasets were constructed for the compression factors $\tau = 0.5$ and

Table 5.1 Training Parameters for the Proposed Bi-GRU with PAA Detector

Parameter	Value
Optimizer	Adam
Loss Function	Mean Squared Error (MSE)
Batch Size	256
Learning Rate	0.001
Number of Epochs	50
Early Stopping	Patience = 5 epochs
Validation Split	10%
GRU Hidden Units	21 (per direction)
PAA Attention Vector Length	21 (i.e., $2N + 1$)
Training Samples per τ	5 million

$\tau = 0.6$. For each value of τ, approximately 5 million labeled samples were produced. Each sample consists of a $(2N + 1)$-symbol window centered on the target symbol, where the window length is selected based on the one-sided ISI span corresponding to the chosen τ value. A root-raised cosine (RRC) pulse with roll-off factor $\beta = 0.35$ was employed for pulse shaping, ensuring consistency with conventional FTN system configurations.

A separate Bi-GRU+PAA model was trained for each τ value using supervised learning. The mean squared error (MSE) loss function was adopted due to its stability and suitability for symbol-wise regression-based detection. Optimization was performed using the Adam optimizer, while hyperparameters such as batch size, learning rate, and number of training epochs were tuned via validation to ensure stable convergence and optimal performance. The complete set of training hyperparameters used in this study is summarized in Table 5.1.

5.2 Data Preprocessing and Input Formatting

In faster-than-Nyquist (FTN) signaling, each received symbol is affected by contributions from multiple neighboring symbols due to the intentionally accelerated transmission rate. The severity of this inter-symbol interference (ISI) is governed by the acceleration factor τ. To ensure that the Bi-GRU-based detector fully captures the temporal dependencies introduced by ISI, the input to the network was constructed using a symmetric window around each target symbol.

For every transmitted symbol, the preprocessing stage collects N preceding and N following received samples, yielding a structured input vector of length $L = 2N + 1$. This design ensures that the detector has access to the complete one-sided ISI span relevant to the target symbol. Formally, the input representation for symbol index k is expressed as

$$\mathbf{Y}_k = \left[\underbrace{y_{k-N} \quad \cdots \quad y_{k-1}}_{N} \quad y_k \quad \underbrace{y_{k+1} \quad \cdots \quad y_{k+N}}_{N} \right], \tag{5.1}$$

where y_k denotes the matched-filtered received sample corresponding to the transmitted symbol a_k. This structured representation enables the Bi-GRU to simultaneously exploit forward and backward temporal dependencies, aligning naturally with the bidirectional processing capability of the model.

In this study, the one-sided ISI length was set to $N = 10$ for both $\tau = 0.5$ and $\tau = 0.6$. This choice was guided by empirical analysis of the ISI coefficients and BER sensitivity to input window length. The selected value of N was found to sufficiently encompass the dominant ISI contributions for both acceleration factors, ensuring robust detection performance without incurring unnecessary computational overhead.

5.3 Bi-GRU-Based Sequence Modeling

To effectively capture the temporal dependencies introduced by inter-symbol interference (ISI) in FTN signaling, the proposed detector employs a Bidirectional Gated Recurrent Unit (Bi-GRU) architecture as its primary sequential modeling component. Unlike unidirectional recurrent networks, the Bi-GRU processes each input sequence in both the forward and backward temporal directions, making it well-suited for FTN scenarios where both past and future symbols exert significant influence on the central symbol due to intentional time packing.

Let $\mathbf{x}_k = [a_{k-N}, \ldots, a_k, \ldots, a_{k+N}] \in \mathbb{C}^{2N+1}$ denote the structured input window for the k-th detection task, where each entry corresponds to a matched-filtered received symbol. For QPSK modulation, the complex-valued samples in this window are separated into their real and imaginary components before being passed to the Bi-GRU layer.

A GRU maintains a hidden state by regulating memory updates using two gating mechanisms. At each time step t, the GRU computes:

$$\mathbf{z}_t = \sigma(\mathbf{W}_z \mathbf{x}_t + \mathbf{U}_z \mathbf{h}_{t-1} + \mathbf{b}_z), \tag{5.2}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r \mathbf{x}_t + \mathbf{U}_r \mathbf{h}_{t-1} + \mathbf{b}_r), \tag{5.3}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_h \mathbf{x}_t + \mathbf{U}_h(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_h), \tag{5.4}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t, \tag{5.5}$$

where $\mathbf{z}_t$ and $\mathbf{r}_t$ denote the update and reset gates, respectively; $\sigma(\cdot)$ is the sigmoid activation function; $\odot$ represents element-wise multiplication; and $\mathbf{W}_*, \mathbf{U}_*, \mathbf{b}_*$ are trainable parameters.

In the bidirectional configuration, two GRUs operate in parallel: the forward GRU processes the sequence from left to right, while the backward GRU processes it from right to left. Their outputs at each time step are concatenated as

$$\mathbf{h}_t^{\text{bi}} = [\overrightarrow{\mathbf{h}}_t ; \overleftarrow{\mathbf{h}}_t], \tag{5.6}$$

yielding a sequence representation $\mathbf{H} \in \mathbb{R}^{(2N+1) \times D}$, where D denotes the combined dimensionality of the forward and backward hidden states. This bidirectional representation captures the full temporal context around the target symbol and serves as the input to the Position-Aware Attention (PAA) mechanism described in Section 5.4.

5.4 Position-Aware Attention Mechanism

Following the Bi-GRU layer—which captures bidirectional temporal dependencies among the input symbols—we introduce a Position-Aware Attention (PAA) mechanism designed to further refine detection by emphasizing symbol positions that contribute most strongly to ISI. Unlike standard content-based attention mechanisms that compute dynamic relevance scores for each sample, the proposed PAA incorporates domain knowledge intrinsic to FTN signaling by learning a globally shared, position-indexed attention distribution. As such, the resulting attention

weights are interpretable, structured, and specific to the FTN detection task, rather than a generic form of attention.

Let $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_L] \in \mathbb{R}^{L \times D}$ denote the Bi-GRU output sequence, where $L = 2N + 1$ is the input window length and D is the concatenated hidden-state dimension. The PAA mechanism introduces a trainable parameter vector $\mathbf{w}_p \in \mathbb{R}^{1 \times L \times 1}$, where each element corresponds to a unique positional index relative to the center of the window. Importantly, this weight vector is shared across all training samples and does not depend on the instantaneous content of the input, reflecting its role as a position-dependent prior rather than a content-driven attention process.

To obtain valid attention coefficients, the elements of $\mathbf{w}_p$ are normalized using a softmax function:

$$\alpha_i = \frac{\exp\left(w_{p,i}/T_{\text{att}}\right)}{\sum_{j=1}^{L} \exp\left(w_{p,j}/T_{\text{att}}\right)}, \qquad i = 1, \ldots, L, \tag{5.7}$$

where $T_{\text{att}} > 0$ is a temperature parameter controlling the sharpness of the attention distribution. This produces an attention vector $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_L]$, which forms a learned probability distribution over symbol positions.

Although the underlying ISI pattern is deterministic for a fixed pulse shape and compression factor τ, learning the attention weights $\mathbf{w}_p$—rather than imposing fixed heuristics—offers several advantages. Because PAA is trained jointly with the Bi-GRU and does not introduce content-dependent computations, it adds negligible inference overhead. At the same time, the learned, position-dependent weighting adapts to the statistical structure captured by the Bi-GRU, enabling more robust and data-aligned emphasis of the most influential ISI positions.

Given these attention weights, the Bi-GRU representations are aggregated into a single context vector by computing a weighted sum across symbol positions:

$$\mathbf{c}^{(b)} = \sum_{i=1}^{L} \alpha_i \mathbf{h}_i^{(b)} = \mathbf{H}^{(b)\top} \boldsymbol{\alpha}, \tag{5.8}$$

where $\mathbf{h}_i^{(b)} \in \mathbb{R}^D$ denotes the Bi-GRU hidden state at position i for the b-th sample, and $\mathbf{H}^{(b)} \in \mathbb{R}^{L\times D}$ is the full hidden-state matrix corresponding to that sample. The resulting vector $\mathbf{c}^{(b)} \in \mathbb{R}^D$ provides a compact, position-weighted summary of the entire input window, where positions with higher learned attention weights contribute more strongly to the final representation.

The context vector $\mathbf{c}^{(b)}$ is subsequently passed through a lightweight output layer to estimate the central transmitted symbol. This is implemented as a linear transformation followed by an activation function:

$$\hat{a}^{(b)} = f\Big(\mathbf{W}_o\mathbf{c}^{(b)} + \mathbf{b}_o\Big), \tag{5.9}$$

where $\mathbf{W}_o$ and $\mathbf{b}_o$ denote the trainable weights and bias of the output layer, respectively, and $f(\cdot)$ is the activation function appropriate for the task (e.g., identity for regression-based detection, or a sigmoid/tanh for symbol classification). This final stage yields a refined, PAA-guided estimate of the target symbol, completing the FTN detection process.

To train the detector, the estimated output $\hat{a}^{(b)}$ is compared with the true central symbol $a^{(b)}$. The discrepancy is measured using a loss function $\ell(\cdot)$, such as mean-squared error (MSE) for regression-based detection or cross-entropy for classification. The total training loss over a mini-batch of size B is defined as

$$\mathcal{L}_{\text{task}} = \frac{1}{B}\sum_{b=1}^{B} \ell\Big(\hat{a}^{(b)}, a^{(b)}\Big), \tag{5.10}$$

ensuring joint optimization of both the Bi-GRU parameters and the PAA weights. In effect, the network learns not only to model temporal dependencies but also to assign appropriate importance to each symbol position.

A key aspect of the proposed framework is that the PAA weights are fully learnable and not predetermined heuristics. Since the normalized attention coefficients α_i depend on the underlying trainable vector $\mathbf{w}_p$, the gradients of the task loss naturally propagate back to each position weight through backpropagation. Specifically, the gradient of the

loss with respect to a position parameter $w_{p,j}$ is given by

$$\frac{\partial \mathcal{L}_{\text{task}}}{\partial w_{p,j}} = \frac{1}{BT_{\text{att}}} \sum_{b=1}^{B} \sum_{i=1}^{L} \alpha_i (\delta_{ij} - \alpha_j) (\nabla_{\mathbf{c}^{(b)}} \ell(\hat{a}^{(b)}, a^{(b)}))^{\top} \mathbf{h}_i^{(b)}. \tag{5.11}$$

where δ_{ij} is the Kronecker delta and $\nabla_{\mathbf{c}^{(b)}} \ell(\hat{a}^{(b)}, a^{(b)})$ is the gradient of the sample-wise loss with respect to the context vector. This relation shows that symbol positions whose hidden states align strongly with the gradient direction receive positive reinforcement, thereby increasing their corresponding attention weights, whereas less relevant positions are downweighted.

Over the course of training, this mechanism drives $\mathbf{w}_p$ to converge toward an empirically learned importance pattern that mirrors the ISI profile of FTN signaling—typically assigning higher weights to symbols closer to the target and diminishing weights to more distant neighbors. The complete architecture of the proposed Bi-GRU with PAA detector is depicted in Fig. 5.2. As illustrated in the figure, the learned PAA weights are visualized using green color gradients, where darker shades denote higher attention and lighter shades indicate lower importance. This representation provides an intuitive interpretation of the learned ISI structure while keeping the mechanism computationally lightweight. Because the attention weights are shared across samples and independent of the instantaneous input, the PAA incurs negligible inference overhead yet significantly enhances detection accuracy.

5.5 Simulation Results and Computational Complexity Comparison

This section presents the simulation results and computational complexity analyses of the proposed Bi-GRU with PAA detector. We begin by examining its BER performance relative to a baseline Bi-GRU model and the optimal BCJR algorithm, highlighting the benefits introduced by the structured attention mechanism. We then benchmark the detector against several state-of-the-art FTN approaches from the literature. Finally, we perform a LUT-based complexity evaluation to quantify the computational advantages of the proposed method.

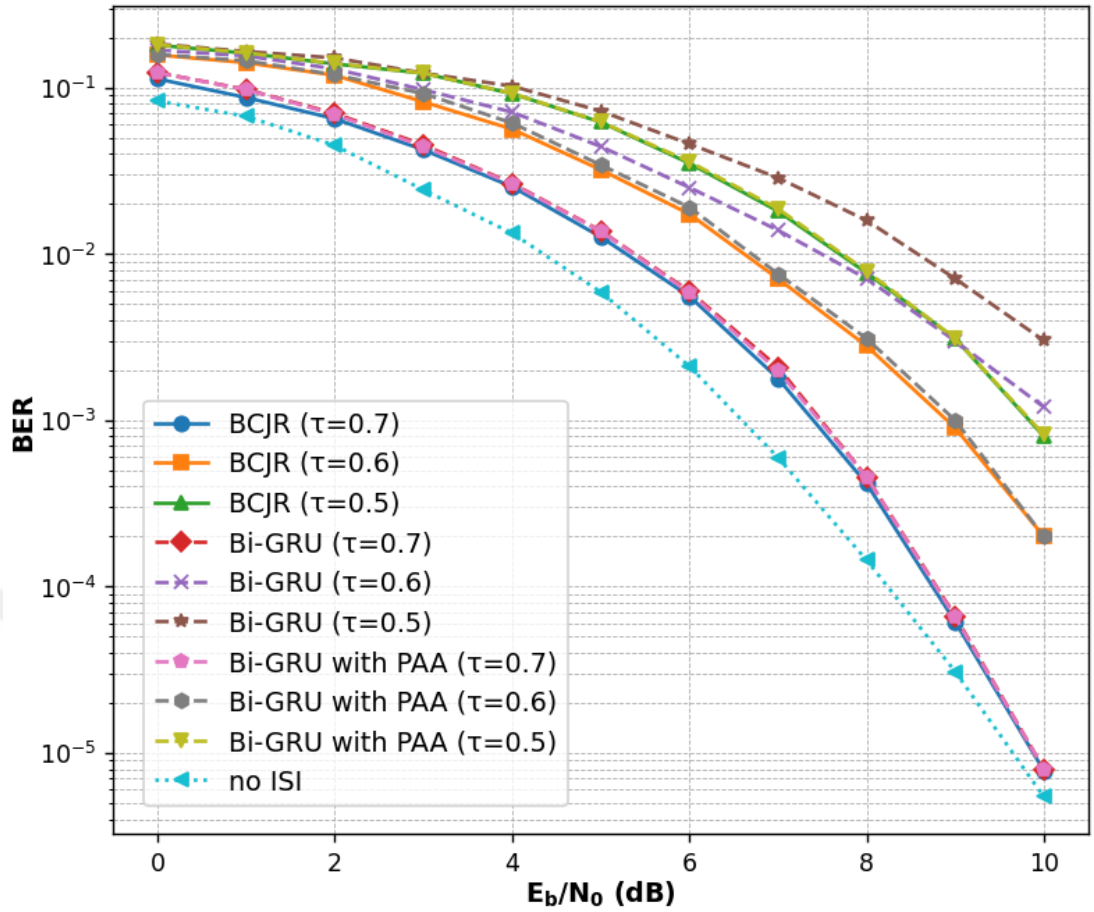


Figure 5.3 BER performance comparison of the Bi-GRU detector with and without the proposed Position-Aware Attention (PAA) mechanism, evaluated for $\tau = 0.5$, 0.6, and 0.7. The benchmark BCJR algorithm and no-ISI condition are included for reference.

Fig. 5.3 compares the BER performance of the proposed Bi-GRU-based detectors—with and without the PAA mechanism—alongside the optimal Maximum A Posteriori (MAP) BCJR algorithm for compression factors $\tau = 0.5$, 0.6, and 0.7. As shown in the figure, the baseline Bi-GRU model achieves near-optimal BER performance for $\tau = 0.7$, where ISI remains relatively mild. However, as the signaling rate becomes more accelerated and the severity of ISI increases at $\tau = 0.6$ and $\tau = 0.5$, the baseline Bi-GRU detector experiences noticeable performance degradation. In contrast, the proposed Bi-GRU with PAA consistently matches the optimal BER performance achieved by the BCJR algorithm across all tested τ values. This improvement underscores the effectiveness of the PAA mechanism in guiding the network to focus on symbol positions that contribute more significantly to ISI, thereby preserving detection accuracy even under aggressive time-packing conditions.

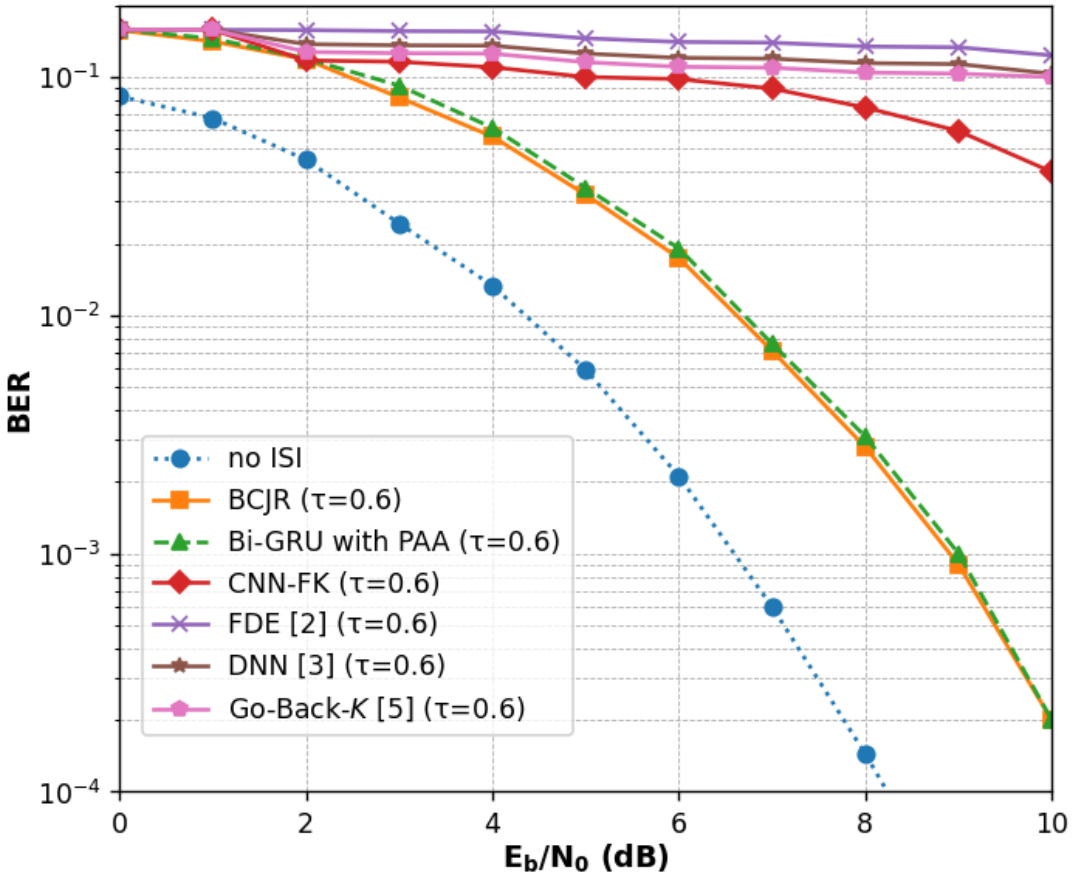


Figure 5.4 BER performance comparison between the proposed Bi-GRU with PAA detector and existing FTN detection methods from the literature, evaluated for $\tau = 0.6$. The benchmark BCJR algorithm and no-ISI condition are included for reference.

Furthermore, Fig. 5.4 provides a comparative performance analysis between the proposed detector and several existing FTN detection techniques from the literature. Across the evaluated SNR range, the Bi-GRU with PAA achieves superior performance, confirming its effectiveness relative to both classical and deep learning-based FTN detection methods.

To assess the robustness of the proposed Bi-GRU with PAA detector in more realistic wireless conditions, additional simulations were conducted over a quasi-static multipath Rayleigh fading channel with three equal-power taps. The relative delay between adjacent taps was set to the symbol interval τT, thereby intensifying the inter-symbol interference introduced by FTN signaling. The channel coefficients were assumed to remain constant within each frame and to vary independently across frames during both training and testing. As illustrated in Fig. 5.5, the proposed detector maintains near-optimal BER performance closely matching that of the BCJR algorithm for $\tau \geq 0.6$,

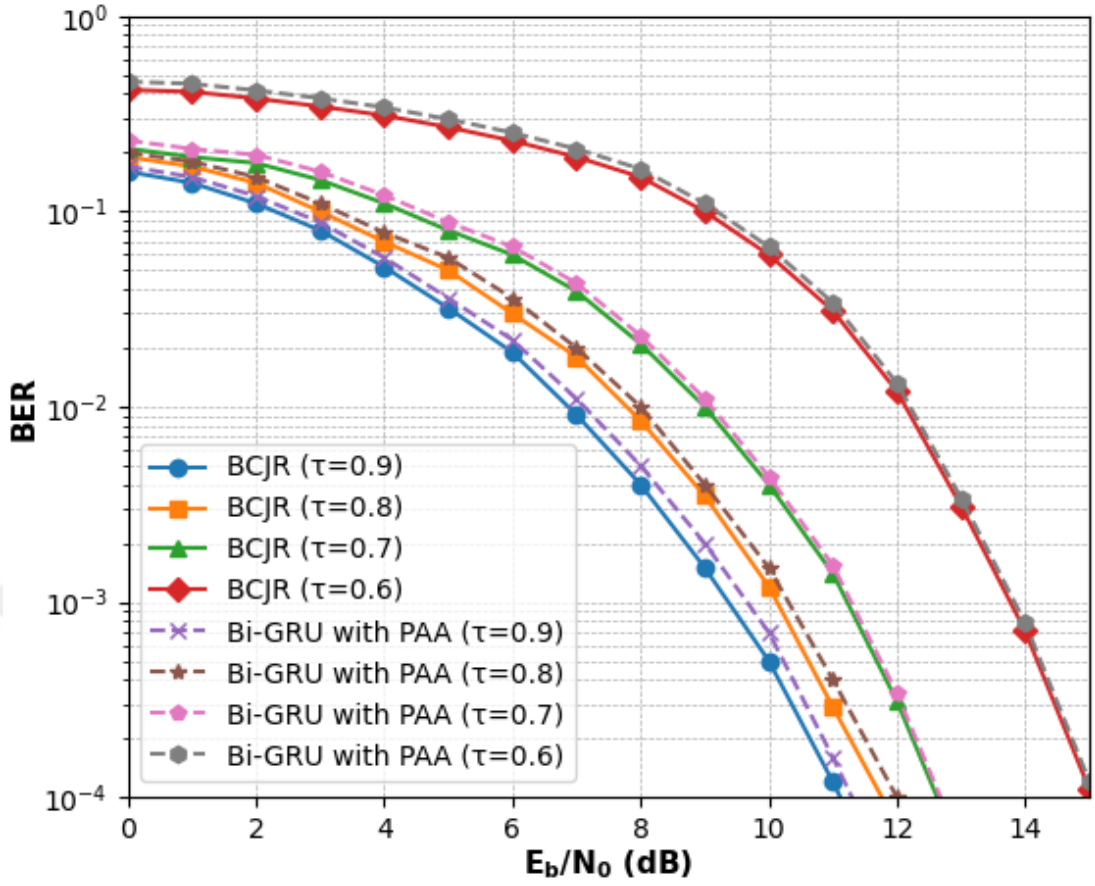


Figure 5.5 BER performance comparison of the proposed Bi-GRU with PAA detector and the BCJR algorithm over a quasi-static Rayleigh fading channel for $\tau \geq 0.6$.

demonstrating strong resilience to multipath fading and confirming its effectiveness in wireless environments beyond the AWGN channel.

Table 5.2 Unified Operation Counts and LUT-Weighted Cost for M-BCJR vs. Bi-GRU with PAA at $\tau = 0.5$

Operation	**LUT**	**BPSK M-BCJR**	**BPSK Bi-GRU**	**QPSK M-BCJR**	**QPSK Bi-GRU**
Multiplication	113	132	756	844	1512
Addition	10	367	714	2231	1428
Comparison	11	4196	0	24756	0
Division	236	113	0	745	0
Exponential	73	210	0	1428	0
tanh	1	109	42	218	84
sigmoid	1	0	84	0	168
Total LUT-Weighted Cost		**106,849**	**92,694**	**670,280**	**185,388**

Evaluating computational efficiency is essential in resource-constrained environments. Therefore, a unified comparison between the proposed Bi-GRU with PAA detector and the conventional M-BCJR algorithm [1] was performed, as summarized in Table 5.2. The table reports both the number of operations required to detect a single symbol

under BPSK and QPSK modulations for $\tau = 0.5$, as well as the corresponding hardware costs expressed in terms of LUT utilization. The LUT-weighted cost for each detector was obtained by multiplying the number of each operation type by its associated LUT requirement, providing an aggregate metric that reflects the overall hardware effort. The proposed detector achieves approximately 13.2% and 72.3% reductions in total LUT-weighted cost for BPSK and QPSK, respectively, demonstrating substantial improvements in computational efficiency over the M-BCJR algorithm.

5.6 Summary of the Chapter

This chapter presented a Bi-GRU-based FTN detection framework enhanced with a Position-Aware Attention (PAA) mechanism. By combining bidirectional temporal modeling with a domain-informed attention structure, the proposed detector effectively captures the ISI characteristics inherent to FTN signaling. Simulation results demonstrated that the Bi-GRU with PAA achieves near-optimal BER performance for $\tau \geq 0.5$, closely matching the performance of the BCJR algorithm even under severe ISI and fading conditions. Furthermore, the LUT-weighted complexity analysis confirmed that the proposed approach offers substantial hardware savings, achieving reductions of up to 72.3% compared to the M-BCJR algorithm. Overall, the findings of this chapter highlight the advantages of integrating structured attention with recurrent neural architectures for efficient and accurate FTN detection.

CHAPTER 6

COMPARATIVE ANALYSIS AND DISCUSSION

This chapter provides a unified comparison and high-level discussion of the three deep learning-based FTN detection architectures developed throughout this thesis: (i) the CNN-based detector with structured fixed kernels, (ii) the GRU-based detector optimized with NADAM, and (iii) the Bi-GRU-based detector enhanced with the Position-Aware Attention (PAA) mechanism.

6.1 Overview of Developed FTN Detection Architectures

Each architecture was designed with different strengths and objectives in mind, addressing the challenges posed by Faster-than-Nyquist signaling from complementary perspectives.

The CNN-based detector adopts a structured, domain-informed convolutional design that leverages fixed kernels to explicitly model ISI patterns at predetermined symbol offsets. Its primary advantages include low arithmetic complexity, inherent parallelism, and strong performance for moderate ISI scenarios (e.g., $\tau \geq 0.7$). The structured kernel allocation—combined with a fixed, non-sliding convolutional approach—enables efficient feature extraction while avoiding the computational burden associated with conventional moving-kernel CNNs.

The GRU-NADAM detector employs sequential modeling to learn temporal dependencies in the received symbol sequence. By aligning the GRU input with the one-sided ISI span, the model effectively captures interference originating from both preceding and succeeding symbols. The adoption of the NADAM optimizer provides faster and more stable convergence compared to ADAM, particularly in highly non-convex optimization landscapes. This architecture offers a favorable balance between complexity and accuracy, achieving near-optimal BER performance for $\tau \geq 0.7$ while maintaining lower parameter count than LSTM-based alternatives.

The third architecture, based on Bi-GRU with PAA, further enhances temporal

Table 6.1 Summary of Proposed FTN Detection Architectures

Detector	Key Characteristics	Performance Regime and Advantages
CNN-Based Fixed Kernel	Structured kernels aligned with ISI; non-sliding convolutions; low arithmetic cost.	Near-optimal BER for $\tau \geq 0.7$; highly parallelizable; lowest hardware complexity among the three.
GRU-NADAM	Sequence modeling with update/reset gates; optimized with NADAM for faster convergence.	Strong performance for $\tau \geq 0.7$; robust to ISI variations; lower parameter count than LSTM.
Bi-GRU + PAA	Bidirectional GRU with trainable position-aware attention; captures full temporal context.	Near-optimal BER for $\tau \geq 0.5$; best performance under severe ISI; significant LUT savings compared to BCJR.

modeling capabilities by incorporating both forward and backward dependencies as well as a learned positional weighting mechanism. The PAA component assigns importance scores to different symbol positions, reflecting their ISI contributions, and enables the detector to focus on the most influential inputs. This design proves especially effective under more aggressive FTN compression (e.g., $\tau \leq 0.6$), where ISI is highly asymmetric and substantial. As demonstrated in Chapter 5, the Bi-GRU with PAA achieves near-optimal BER performance for $\tau \geq 0.5$ while significantly reducing LUT-weighted hardware complexity relative to classical detection methods.

Table 6.1 provides a consolidated comparison of the three architectures, highlighting their key characteristics and performance regimes. This high-level overview serves as a foundation for the subsequent sections of this chapter, which present a unified BER analysis, computational complexity comparison, and practical insights drawn from the collective results of the thesis.

6.2 Unified Performance Comparison Across All Proposed Detectors

This section presents a unified BER analysis comparing the three deep learning-based FTN detectors developed throughout this thesis: (i) the CNN-based detector with structured fixed kernels, (ii) the GRU-based detector optimized with NADAM, and (iii) the Bi-GRU-based detector equipped with the Position-Aware Attention (PAA) mechanism. By examining their performance under a consistent experimental framework, we highlight the relative strengths of each architecture and identify their effective operating regimes.

Fig. 6.1 summarizes the BER performance of all detectors for $\tau = 0.7$. In this

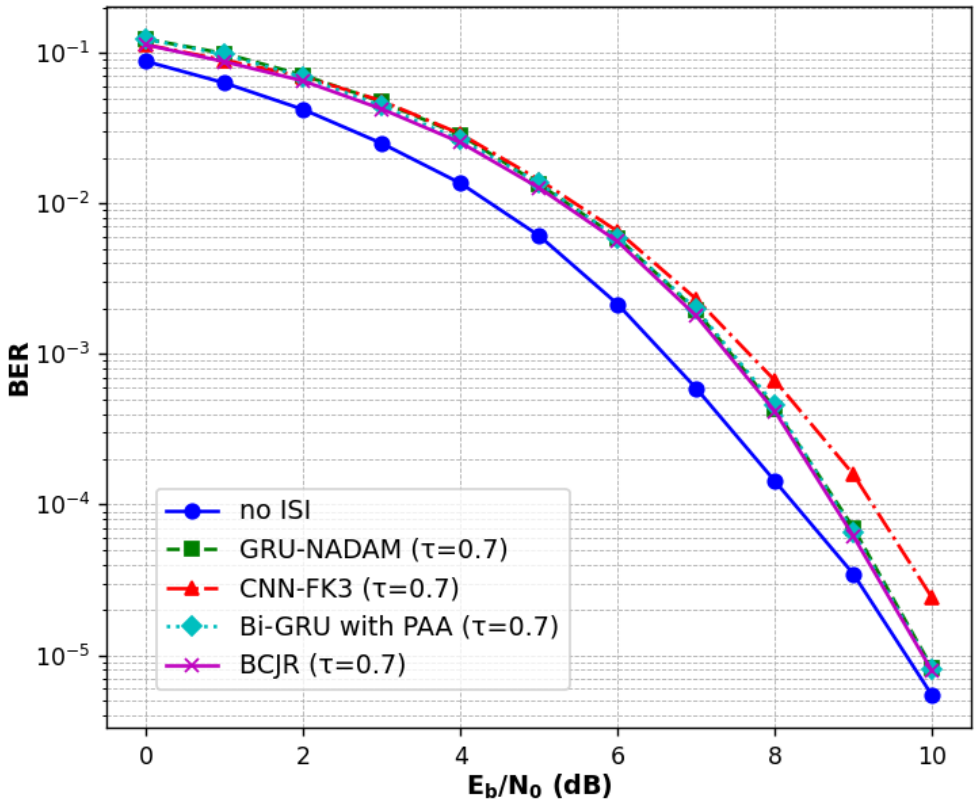


Figure 6.1 Unified BER comparison of the proposed CNN-FK, GRU-NADAM, and Bi-GRU+PAA detectors for $\tau = 0.7$ under an AWGN channel with roll-off factor $\beta = 0.35$. All models achieve near-optimal performance, closely matching the BCJR benchmark.

moderate ISI regime, all three models achieve BER curves closely aligned with the optimal BCJR algorithm, with the Bi-GRU+PAA model providing the closest match. The CNN-based detector performs particularly well in this region, demonstrating that structured convolutional kernels are sufficient to capture the relatively limited ISI span when $\tau \geq 0.7$. The GRU-NADAM detector yields similarly strong results, confirming its ability to learn short-range temporal dependencies efficiently.

A more challenging scenario is presented in Fig. 6.2, which shows the BER curves for $\tau = 0.6$, where ISI becomes significantly more severe. In this setting, the performance gaps between the detectors become more apparent. The CNN-based detector begins to deviate from the BCJR benchmark due to its limited capacity to model long-range ISI via fixed-depth convolution. The GRU-NADAM detector performs notably better, benefiting from its recurrent memory structure, but still shows a visible performance gap at higher SNRs. In contrast, the Bi-GRU+PAA model remains closely matched to the BCJR performance across the entire SNR range. This can be attributed to its

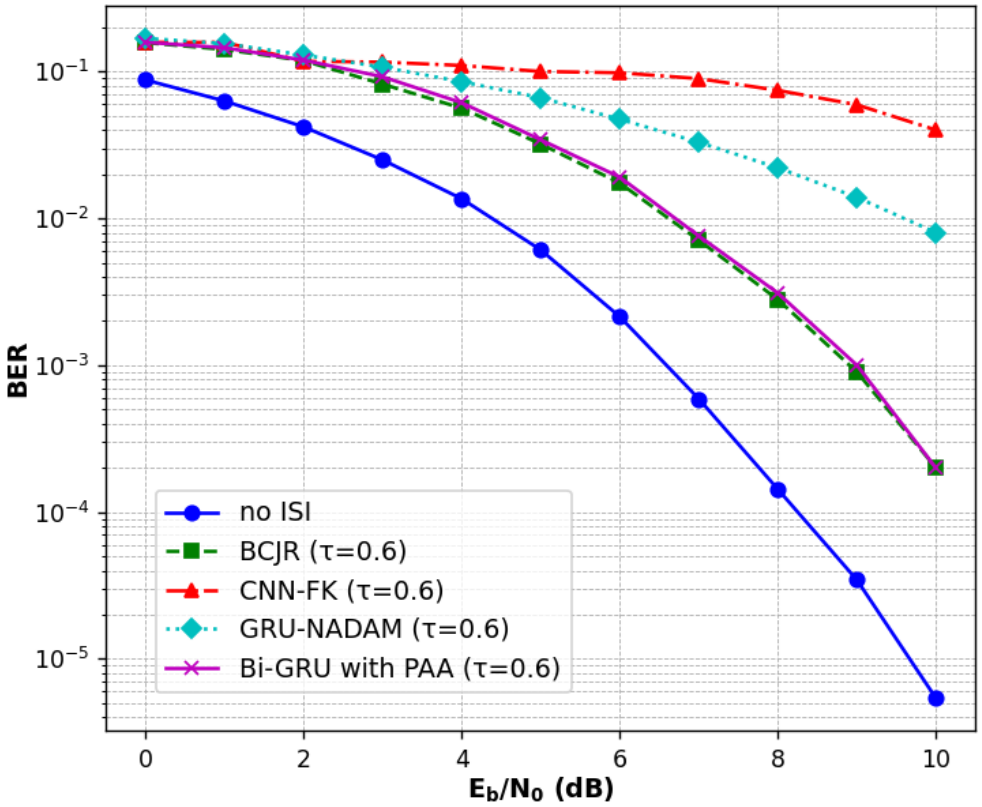


Figure 6.2 Unified BER comparison of the CNN-FK, GRU-NADAM, and Bi-GRU+PAA detectors for $\tau = 0.6$ under an AWGN channel with $\beta = 0.35$. As ISI severity increases, the CNN-based model deviates from the BCJR benchmark, while the Bi-GRU+PAA detector remains closely aligned across all SNR values.

bidirectional processing and structured attention mechanism, which together enable effective modeling of asymmetric and long-range ISI patterns.

The most demanding case is shown in Fig. 6.3, corresponding to $\tau = 0.5$. At this compression level, the ISI extends deeply across many neighboring symbols, making the detection problem substantially more difficult. Both the CNN-based and GRU-based detectors exhibit notable performance degradation, as neither architecture incorporates an explicit mechanism to differentially weigh symbol positions based on their ISI contribution. In contrast, the Bi-GRU+PAA detector continues to match BCJR performance even at $\tau = 0.5$, underscoring the importance of the learned positional weighting introduced by the PAA mechanism. This demonstrates that PAA plays a crucial role in addressing severe ISI scenarios by guiding the network to prioritize more influential positions.

These unified BER results highlight clear performance trends:

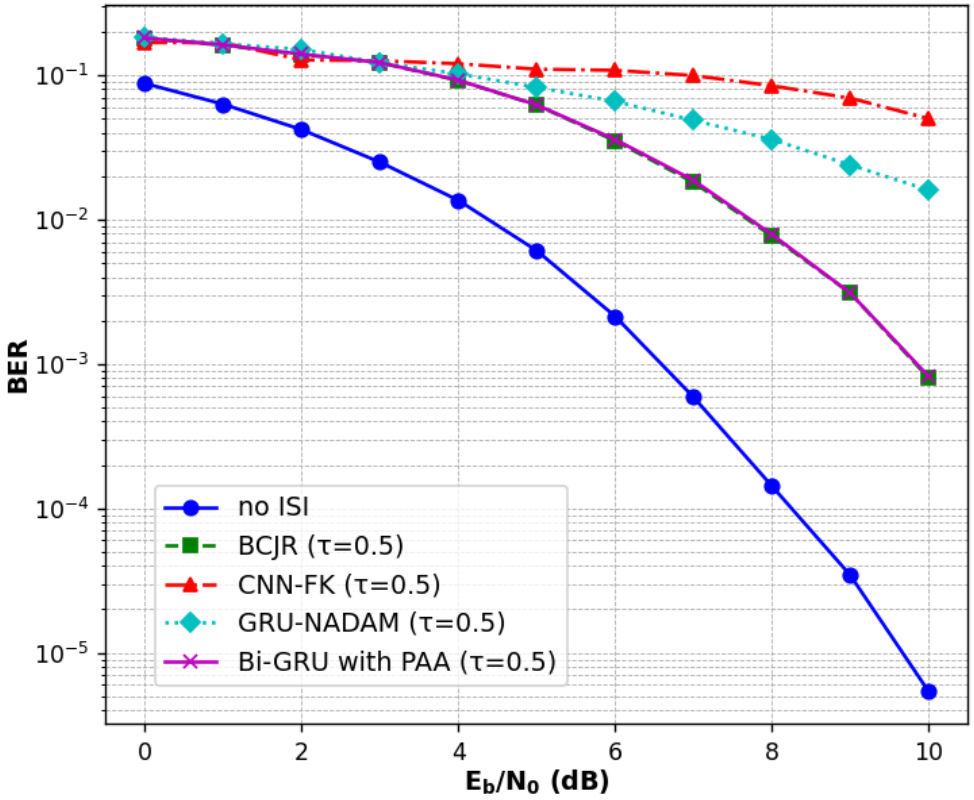


Figure 6.3 BER comparison of the CNN-FK, GRU-NADAM, and Bi-GRU+PAA detectors for $\tau = 0.5$ under an AWGN channel with $\beta = 0.35$. Severe ISI causes significant degradation in CNN- and GRU-based detectors, whereas the proposed Bi-GRU+PAA model continues to match the optimal BCJR performance.

- For mild ISI ($\tau \geq 0.7$), **all three detectors perform comparably**, offering flexibility in selecting architectures based on hardware constraints.
- For moderate ISI ($0.6 \leq \tau < 0.7$), the **GRU-NADAM detector outperforms the CNN-based design**, while the **Bi-GRU+PAA achieves BCJR-level performance**.
- For strong ISI ($\tau \leq 0.5$), the **Bi-GRU+PAA is the only viable deep learning-based detector** among the three, consistently achieving near-optimal BER.

Overall, this unified BER comparison demonstrates that while each detector provides advantages in its respective operating regime, the Bi-GRU+PAA model offers the most robust and scalable performance across a wide range of FTN compression factors. The next section provides a parallel comparison in terms of computational complexity,

offering a holistic view of the trade-offs between accuracy and efficiency.

In addition to BER performance, it is important to consider the computational efficiency of the three detection architectures. Although the detailed numerical comparisons were presented earlier in this thesis, several qualitative observations can be summarized here to contextualize the unified BER results.

The CNN-based detector, owing to its structured fixed-kernel design, provides the lowest inference complexity among the three models. Its feed-forward convolutional structure requires no recurrent operations, making it particularly attractive in hardware-constrained environments or latency-critical systems. However, its limited receptive field and lack of temporal modeling restrict its applicability to scenarios where ISI remains relatively shallow, as observed for $\tau \geq 0.7$.

The GRU-based detector offers a more balanced trade-off between performance and computational cost. Its recurrent architecture significantly improves modeling capacity for moderate ISI conditions ($0.6 \leq \tau < 0.7$), while maintaining lower complexity compared to the Bi-GRU+PAA detector. Prior complexity analyses in Chapter 4 demonstrate that GRUs remain more efficient than LSTMs and scale gracefully with increasing ISI depth.

Finally, the Bi-GRU+PAA detector—although the most computationally demanding of the three due to its bidirectional recurrence and attention mechanism—delivers unmatched robustness across all FTN operating regimes. Its PAA layer adds only marginal overhead while providing a substantial gain in modeling long-range and asymmetric ISI patterns. As shown in the complexity evaluations in Chapter 5, this additional cost remains practical, especially considering that the Bi-GRU+PAA model is the only deep learning-based architecture that consistently maintains BCJR-level performance down to $\tau = 0.5$.

In summary, while each detector presents distinct computational and performance trade-offs, the unified analysis indicates the following:

- The **CNN-based detector** is the most efficient option when ISI is mild.
- The **GRU-based detector** offers the best balance of complexity and performance for moderate ISI.
- The **Bi-GRU+PAA detector** provides the most robust performance across all τ values and remains the only viable choice under severe ISI, with its complexity justified by its consistent near-optimal detection accuracy.

CHAPTER 7

CONCLUSION AND FUTURE RESEARCH DIRECTIONS

This thesis presented a comprehensive investigation into deep learning–based detection methods for faster-than-Nyquist (FTN) signaling, focusing on the development and evaluation of three novel receiver architectures: (i) a CNN-based detector with structured fixed kernels, (ii) a GRU-based detector optimized using the NADAM optimizer, and (iii) a Bi-GRU-based detector enhanced with a Position-Aware Attention (PAA) mechanism. The overarching goal was to address the severe inter-symbol interference (ISI) inherent to FTN signaling and to explore efficient alternatives to traditional detection methods such as BCJR and M-BCJR.

The first contribution introduced a CNN-based FTN detector that leverages domain-informed fixed convolutional kernels aligned with the Toeplitz ISI structure of FTN channels. Simulation results demonstrated that the CNN-FK detector achieves near-optimal bit error rate (BER) performance for mild ISI scenarios ($\tau \geq 0.7$) with reduced computational complexity, validating the effectiveness of structured convolution in ISI-limited communication environments.

The second major contribution was the GRU-NADAM detector, designed by aligning the GRU input with the one-sided ISI span and employing the NADAM optimizer to enhance gradient-based training stability. This architecture performed particularly well in moderate ISI conditions, achieving BER performance close to the optimal BCJR algorithm for $\tau \geq 0.7$ while offering faster convergence and reduced parameter complexity compared to LSTM-based alternatives.

The most advanced contribution of this thesis was the Bi-GRU detector with a structured PAA mechanism. By combining bidirectional temporal modeling with globally learned positional weighting, this architecture effectively captured the asymmetric and long-range ISI patterns encountered under severe FTN compression.

The Bi-GRU+PAA detector achieved near-optimal BER even for $\tau = 0.5$, outperforming both CNN- and GRU-based models as well as all existing deep learning–based FTN detectors in the literature. Additional evaluations under quasi-static Rayleigh fading and LDPC-coded transmission further confirmed the robustness and practical viability of the proposed approach.

Beyond BER analysis, the thesis provided a unified comparison of computational complexity across all three detectors and benchmarked their efficiency relative to M-BCJR and other classical methods. While detailed numerical results were provided in earlier chapters, the overall conclusion is that deep learning–based FTN detectors, when designed with domain knowledge and structured architectural enhancements, can achieve substantial reductions in computational complexity while retaining near-optimal detection performance across a wide operating range.

Collectively, the results of this thesis demonstrate that deep learning architectures—when tightly integrated with communication-theoretic principles—offer powerful, flexible, and computationally efficient alternatives for FTN detection. The three detectors proposed here form a progressive suite of solutions, each suited to different ISI regimes, and together they provide a strong foundation for future research in intelligent FTN receiver design.

Although the proposed models show strong performance and generalization, several important avenues for future research remain:

- **Extension to lower τ values and higher-order modulations:** Exploring FTN detection for $\tau < 0.5$ and higher-order QAM constellations may require deeper or hybrid architectures capable of modeling more severe ISI.
- **Joint detection and decoding:** End-to-end training of FTN detectors integrated with LDPC, Polar, or Turbo decoders could further improve robustness under practical channel conditions.
- **Adaptive and online learning:** Developing lightweight models capable of online parameter adaptation would enhance the applicability of FTN detection in dynamic or rapidly varying wireless environments.

- **Transformer-based and advanced attention architectures:** Extending the position-aware attention concept into efficient transformer-like mechanisms may improve long-range temporal modeling while maintaining manageable complexity.
- **Hardware implementation and FPGA prototyping:** Implementing the proposed detectors on FPGA or ASIC platforms would provide real-time performance validation and insights into power/area trade-offs.
- **Integration with transmitter-side precoding:** Combining learning-based receiver architectures with matrix-decomposition precoding (e.g., Cholesky, GTMH) could yield highly optimized FTN transceiver designs.

In summary, this thesis demonstrated that deep learning–based models, when guided by system-level insights, provide substantial gains in both detection accuracy and computational efficiency for FTN signaling. The presented CNN-FK, GRU-NADAM, and Bi-GRU+PAA architectures establish a solid and scalable foundation for future advancements in FTN communication systems.

REFERENCES

[1] Prlja A. and Anderson J. B., "Reduced-complexity receivers for strongly narrowband intersymbol interference introduced by faster-than-nyquist signaling," *IEEE Transactions on Communications*, vol. 60, no. 9, pp. 2591–2601, 09 2012.

[2] Sugiura S., "Frequency-domain equalization of faster-than-nyquist signaling," *IEEE Wireless Communications Letters*, vol. 2, no. 5, pp. 555–558, 10 2013.

[3] Song P., Pan J., Xie Y., and Tang W., "Receiver design for faster-than-nyquist signaling: Deep-learning-based architectures," *IEEE Access*, vol. 8, pp. 68 866–68 873, 2020.

[4] De Filippo B., Amatetti C., and Vanelli-Coralli A., "Faster-than-nyquist equalization with convolutional neural networks," 2025, arXiv:2501.11594. [Online]. Available: https://arxiv.org/abs/2501.11594

[5] Bedeer E., Ahmed M. H., and Yanikomeroglu H., "A very low complexity successive symbol-by-symbol sequence estimator for faster-than-nyquist signaling," *IEEE Access*, vol. 5, pp. 7414–7422, 2017.

[6] Wang H., Liu A., Liang X., Peng S., and Wang K., "Linear precoding for faster-than-nyquist signaling," in *Proceedings of the IEEE International Conference on Computer and Communications (ICCC)*, Chengdu, China, 2017, pp. 52–56.

[7] Mazo J. E., "Faster-than-nyquist signalling," *Bell System Technical Journal*, vol. 54, no. 8, pp. 1451–1462, 10 1975.

[8] Bahl L. R., Cocke J., Jelinek F., and Raviv J., "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284–287, 03 1974.

[9] Li S., Bai B., Zhou J., Chen P., and Yu Z., "Reduced-complexity equalization for faster-than-nyquist signaling: New methods based on ungerboeck observation

model," *IEEE Transactions on Communications*, vol. 66, no. 3, pp. 1190–1204, 03 2018.

[10] Wang C.-K. and Lee L.-S., "Practically realizable digital transmission significantly below the nyquist bandwidth," *IEEE Transactions on Communications*, vol. 43, no. 2/3/4, pp. 215–223, 04 1995.

[11] Rusek F. and Anderson J. B., "Nonbinary and precoded faster-than-nyquist signaling," *IEEE Transactions on Communications*, vol. 56, no. 5, pp. 808–817, 05 2008.

[12] Jana M., Medra A., Lampe L., and Mitra J., "Pre-equalized faster-than-nyquist transmission," *IEEE Transactions on Communications*, vol. 65, no. 10, pp. 4334–4346, 10 2017.

[13] Spano D., Alodeh M., Chatzinotas S., and Ottersten B., "Faster-than-nyquist signaling through spatio-temporal symbol-level precoding for the multiuser miso downlink channel," *IEEE Transactions on Wireless Communications*, vol. 17, no. 9, pp. 5994–6007, 09 2018.

[14] Ishihara T. and Sugiura S., "Svd-precoded faster-than-nyquist signaling with optimal and truncated power allocation," *IEEE Transactions on Wireless Communications*, vol. 18, no. 12, pp. 5904–5917, 12 2019.

[15] Li Y., Xiao S., Wang J., and Tang W., "Cholesky-decomposition aided linear precoding and decoding for ftn signaling," *IEEE Wireless Communications Letters*, vol. 10, no. 6, pp. 1163–1167, 06 2021.

[16] Liu C., Imran M. A., Wen S., and Liu G., "Joint precoding and pre-equalization for faster-than-nyquist transmission over multipath fading channels," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 5, pp. 6381–6394, 05 2023.

[17] Sugiura S., "Secrecy performance of evd-based ftn signaling and nofdm in quasi-static fading channels," *IEEE Transactions on Wireless Communications*, vol. 20, no. 7, pp. 4172–4184, 07 2021.

[18] Baek M.-S., Park W., and Lee Y.-T., “Deep learning-based signal detection technique for ftn signaling-based emergency alert communication system,” in *Proceedings of the IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, Chengdu, China, 2021, pp. 1–3.

[19] Pan J., Luo X., Chen Y., and Liu S., “A deep learning-aided detection method for ftn-based noma,” *Wireless Communications and Mobile Computing*, pp. 1–11, 01 2020.

[20] Abbasi S. and Bedeer E., “Deep learning-based list sphere decoding for faster-than-nyquist (ftn) signaling detection,” in *Proceedings of the IEEE Vehicular Technology Conference (VTC Spring)*, Helsinki, Finland, 2022, pp. 1–6.

[21] Baek M.-S., Park W., and Lee Y.-T., “Ftn-based non-orthogonal signal detection technique with machine learning in quasi-static multipath channel,” *IEEE Transactions on Broadcasting*, vol. 70, no. 1, pp. 78–86, 03 2024.

[22] Liu B., Li S., Xie Y., and Yuan J., “A novel sum-product detection algorithm for faster-than-nyquist signaling: A deep learning approach,” *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 5975–5987, 09 2021.

[23] Kiranyaz S., Avci O., Abdeljaber O., Ince T., Gabbouj M., and Inman D. J., “1d convolutional neural networks and applications: A survey,” *Mechanical Systems and Signal Processing*, vol. 151, p. 107398, 2021.

[24] Wang B., Dai H., Xu K., Sun Y., Zhang Y., and Li P., “A signal processing method of ofdm communication receiver based on cnn,” *Physical Communication*, vol. 59, p. 102055, 2023.

[25] Yang Q., Cao M., Zhou G., Zhang Y., Yao R., and Wang H., “Mhsa-cnn-bi-lstm based interference cancellation algorithm for mimo-ftn-owc system,” in *Proceedings of the IEEE International Conference on Advanced Infocomm Technology (ICAIT)*, 2024, pp. 1–6.

[26] Benedetto S. and Montorsi G., “Design of pulse shaping filters for efficient data transmission,” *IEEE Transactions on Communications*, vol. 41, no. 5, pp. 722–732, 05 1993.

[27] Anderson J. B., Rusek F., and Öwall V., “Faster-than-nyquist signaling,” *Proceedings of the IEEE*, vol. 101, no. 8, pp. 1817–1830, 08 2013.

[28] Dasalukunte D., Öwall V., Rusek F., and Anderson J. B., “Faster than nyquist signaling,” in *Algorithms to Silicon*. Switzerland: Springer, 2014.

[29] Ishihara T., Sugiura S., and Hanzo L., “The evolution of faster-than-nyquist signaling,” *IEEE Access*, vol. 9, pp. 86 535–86 564, 2021.

[30] Liu X., Wu Y., and Lin Z., “Data-driven and model-driven joint detection algorithm for faster-than-nyquist signaling in multipath channels,” *IEEE Transactions on Communications*, vol. 70, no. 2, pp. 1215–1227, 2022.

[31] Kokshoorn M., “M-bcjr algorithm,” https://github.com/mkokshoorn/M-BCJR-Algorithm, 2016, accessed: Feb. 12, 2025.

[32] Shalan M. and Bertacco V., “A study of power consumption on fpga-based designs,” in *Proceedings of the ACM/IEEE Design Automation Conference (DAC)*, Anaheim, CA, USA, 2003, pp. 234–239.

[33] Cho K., Van Merriënboer B., Bahdanau D., and Bengio Y., “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014, pp. 1724–1734.

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

[35] Dozat T., “Incorporating nesterov momentum into adam,” in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[36] Nesterov Y., “A method of solving a convex programming problem with convergence rate $o(1/k^2)$,” *Soviet Mathematics Doklady*, vol. 27, pp. 372–376, 1983.