AN RNN-BASED DECODING SCHEME FOR IM/DD VISIBLE LIGHT COMMUNICATION SYSTEMS

Dung Le Dinh*, Tho Nguyen Van, Phong Pham Thanh

The University of Danang - VN-UK Institute for Research and Executive Education, Vietnam

*Corresponding author: dung.ledinh@vnuk.udn.vn

(Received: October 24, 2025; Revised: January 22, 2026; Accepted: January 29, 2026)

DOI: 10.31130/ud-jst.2026.24(1).596E

Abstract - In Visible Light Communication (VLC) systems, the integration of Forward Error Correction (FEC) and Run-Length Limited (RLL) coding is essential for ensuring transmission reliability and flicker-free operation. Despite their efficacy, traditional joint decoding schemes often suffer from prohibitive computational overhead. This paper introduces a Deep Learning-based decoding framework designed to replace the conventional concatenated RLL-FEC decoding chain. The proposed architecture employs a single-layer Recurrent Neural Network (RNN) followed by a dense fully-connected layer. Furthermore, a confidence-bit demodulation strategy is implemented to refine the input features for the RNN. Empirical results demonstrate that for short-packet scenarios, the proposed RNN-centric decoder achieves error-correction performance on par with state-of-the-art benchmarks while delivering a significant leap in processing throughput.

Key words - VLC; RNN; LSTM; FEC decoding; visible light communication

1. Introduction

Driven by the rapid evolution of solid-state lighting, Light-Emitting Diodes (LEDs) have become the predominant choice in the commercial lighting sector due to their numerous technical benefits [1]. This widespread adoption has paved the way for VLC, a communication technology that utilizes LED infrastructures to provide simultaneous illumination and high-speed wireless data transmission. Over the past few decades, VLC has emerged as a focal point of intense research within the optical communication community.

In intensity-modulation and direct-detection (IM/DD) VLC systems, flicker mitigation is a vital requirement, as consecutive identical bits in the data stream can lead to unwanted fluctuations in light intensity. While RLL codes are widely adopted to address this by maintaining DC balance and stable brightness, they do not inherently offer coding gain for error recovery. To overcome this limitation and ensure communication reliability, FEC mechanisms are typically employed in conjunction with RLL schemes. This dual-coding approach balances the requirements of the lighting infrastructure with the demands of wireless data transfer.

The IEEE 802.15.7 standard [2] outlines the use of Manchester, 4B6B, and 8B10B as the primary RLL coding options for VLC applications. To achieve reliable communication, it is common practice to employ a serial concatenation of these RLL codes with FEC layers, such as Convolutional Codes (CC) or Reed-Solomon (RS) schemes, thereby leveraging the strengths of both synchronization and error recovery.

A primary drawback of the conventional RLL schemes is their reliance on hard FEC decoding, which inherently limits the error-correction potential of the entire system. To address this, recent studies have introduced soft-decision RLL (soft-RLL) architectures [3]-[5]. For instance, the soft-input soft-output (SISO) approach in [3] utilizes iterative posterior probability updates and inverse mapping to derive codeword probabilities. Similarly, bit-level soft-RLL designs [4] incorporate additional summation units to estimate bit-wise probabilities within each codeword. While these methods, when integrated with soft-decision FEC schemes, significantly enhance error performance, they introduce substantial computational overhead. Furthermore, the unidirectional nature of VLC and its dependence on direct detection make direct Log-Likelihood Ratio (LLR) estimation impractical. Although indirect LLR calculation has been proposed to mitigate this [6], the challenge of high algorithmic complexity remains an unresolved issue.

Over the last decade, deep learning (DL) has emerged as a transformative force, achieving groundbreaking milestones in fields such as computer vision and natural language processing [7]. These advancements have spurred a significant interest within the telecommunications community to adopt DL-driven methodologies for addressing persistent challenges in wireless networks. Key research efforts have successfully utilized neural architectures for signal classification [8] and error-correction decoding [9]-[13]. Furthermore, there is a growing trend toward re-envisioning the entire physical layer by substituting modular blocks with unified, end-to-end deep learning frameworks [8], [14], [15].

This paper proposes an integrated DL-based decoding method for concatenated FEC-RLL structures in VLC, specifically designed for low-bit ADC configurations. By employing a low-bit quantization, we extract more informative features from the received signal, which are then analyzed by the DL decoder. Recurrent Neural Networks (RNNs) are chosen as the primary architecture in this work, motivated by their success in sequence-related problems and robust performance [11]. The proposed system is evaluated against the high-performing SISO RLL and Polar concatenated decoder [5] to establish a performance baseline. Our key contributions are as follows:

• Proposal of a novel joint RLL-FEC decoding scheme specifically designed for VLC systems, which leverages low-bit quantization and a streamlined RNN model to

achieve a superior balance between error-correction performance and hardware complexity. Since the data in the VLC system is transmitted serially, we apply a recurrent neural network (RNN), which has shown great success in a wide variety of sequential tasks, to the proposed decoder.

• Validation of the performance-complexity trade-off, demonstrating that the proposed neural decoder effectively recovers original data streams with only marginal degradation in error-correction capability while achieving a substantial reduction in computational overhead relative to the state-of-the-art.

• Extensive benchmarking of execution latency across diverse hardware environments, confirming the practical feasibility of deploying RNN-based decoders with integrated flicker suppression in real-time VLC applications.

2. System Model

The architectural framework of the proposed VLC system is illustrated in Figure 1, highlighting the integration of a concatenated FEC-RLL encoding stage at the transmitter. At the receiver, the information recovery process is handled by a deep learning-driven decoder, which replaces conventional sequential decoding methods.

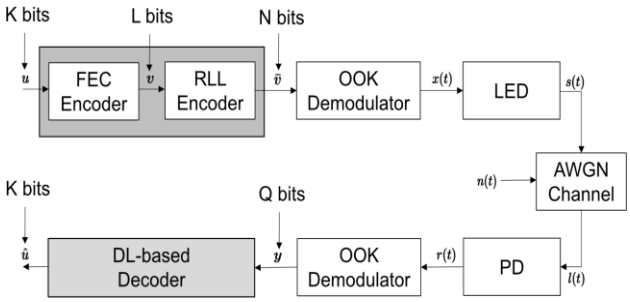

Figure 1. *Block diagram of the proposed VLC system*

On the transmitter side, an information sequence of *K* bits, denoted as $\mathbf{u} \triangleq [u_0, u_1, ..., u_{K-1}]$, initially undergoes FEC encoding to produce an *L*-bit codeword $\mathbf{v} \triangleq [v_0, v_1, ..., v_{L-1}]$. These encoded bits are subsequently mapped into an *N*-bit RLL codeword using a mapping scheme designed to maintain DC balance. The resulting digital sequence is modulated into an On-Off Keying (OOK) signal, which drives the LED front-end to generate the optical output $\mathbf{s}(t)$. This signal effectively facilitates concurrent data transmission and general illumination.

After propagation through the VLC channel, the signal is captured by a photodetector (PD). The PD converts the incident light into an electrical signal $\mathbf{r}(t)$ characterized by the equation $\mathbf{r}(t) = \gamma.\mathbf{s}(t) * \mathbf{l}(t) + \mathbf{n}(t)$, where γ is PD responsivity (A/W); $\mathbf{l}(t)$ is the impulse response of the channel; $*$ denotes convolution; and $\mathbf{n}(t)$ is additive white gaussian noise (AWGN) which contains shot noise, thermal noise, and intersymbol interference (ISI) [15]. For systems employing On-Off Keying, the receive signal-to-noise ratio SNR_{rx} can be expressed as:

$$SNR_{rx} = \frac{(\gamma P_r)^2}{\sigma_{shot}^2 + \sigma_{thermal}^2 + (\gamma.P_{ISI})} \tag{1}$$

where, P_r is the average power of the electrical signal $\mathbf{x}(t)$, P_{ISI} is the average power of intersymbol interference, σ_{shot} and $\sigma_{thermal}$ are the variances of the shot noise and thermal noise, respectively.

For a given electrical input electrical signal, the OOK demodulator produces the output sequence $\mathbf{y}$. Using a hard-decision technique, the *i-th* value of $\mathbf{y}$ is derived as shown below:

$$y_i = \begin{cases} 1, & if\ r_i \geq V_{th} \\ 0, & otherwise \end{cases} \tag{2}$$

where, V_{th} is the hard-decision threshold voltage.

In contrast, the OOK demodulator utilizes a *Gaussian approximation* to calculate soft information (*log-likelihood ratios*), defined by:

$$LLR = \ln \frac{p(y_i | \bar{c}_i = 0)}{p(y_i | \bar{c}_i = 1)} \tag{3}$$

where the conditional probabilities are generally calculated as:

$$p(y_i | \bar{c}_i = \Delta) = \frac{1}{\sqrt{2\pi\sigma_\Delta^2}} e^{-\frac{(y_i - \mu_\Delta)^2}{2\sigma_\Delta^2}} \tag{4}$$

where, μ_Δ and σ_Δ are the statistical mean and standard deviation for $\Delta = 0, 1$.

In VLC systems, unidirectional communication relies on non-negative signals that are directly proportional to the LED's optical intensity. Upon reaching the photodetector, these light signals are converted into a voltage level reflecting the incident intensity. Given the practical constraints, employing *Gaussian approximation* for LLR calculation becomes impractical. To address this, we introduce a low-bit OOK demodulator designed to quantize the received voltage into binary sequences, which are subsequently processed by a deep learning-based decoder. The architecture of this demodulator is elaborated in the following section.

3. Low-bit OOK demodulator

The practical realization of the proposed VLC architecture utilizes a low-resolution (1–3 bits) OOK demodulator, as depicted in Figure 2. For an *S*-bit precision configuration, 2^{S-1} decision thresholds are implemented: the central threshold facilitates the hard-decision bit, whereas the auxiliary thresholds determine the reliability or confidence level of that bit. As illustrated in the block diagram, the photodetector first converts optical incidence into electrical signals, which are subsequently scaled by a trans-impedance amplifier (TIA). This amplified signal is fed into an *S*-bit decision module comprising a bank of comparators $\left(D_{-(2^{S-1}-1)}, ..., D_{-1}, D_0, D_1, ..., D_{2^{S-1}-1}\right)$. While the hard-decision bit is extracted via the central threshold

V_{th0}, the confidence-related bits are derived from peripheral levels including $D_{-(2^{S-1}-1)}, D_{-(2^{S-1}-2)}, ..., D_{-2}, D_{-1}$ a $D_1, D_2, ..., D_{(2^{S-1}-2)}, D_{(2^{S-1}-1)}$. The resulting signals from the comparators are concatenated into a binary vector and passed to the RNN-based decoder to perform data recovery.

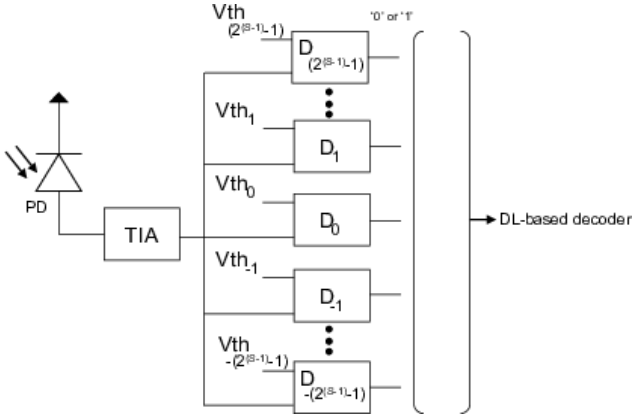

Figure 2. *Proposed low-bit OOK demodulator configuration*

In the final stage, the RNN-based decoder leverages the input bit vectors to reconstruct the original information sequences $\hat{\mathbf{u}} \triangleq [\hat{u}_0, \hat{u}_1, ..., \hat{u}_{K-1}]$. It is observed that $Q = N$ when the OOK demodulator generates hard-decision bits or soft LLR values; $Q = 2^S - 1$ if the *S-bit* OOK demodulation technique is employed.

4. Proposed RNN-based Decoder

In this section, we introduce the fundamental concepts of RNNs and describe how the RNN-based decoder is structured to support OOK demodulation in VLC systems.

4.1. Overview of Recurrent Neural Networks (RNNs)

Specifically designed for sequence modeling, Recurrent Neural Networks (RNNs) are artificial neural architectures that maintain an internal state to process successive inputs in a time-dependent manner. Unlike memoryless architectures, where inputs and outputs are assumed to be independent, RNNs utilize feedback loops (recurrent edges) to ensure that the output at the current time step is conditioned on both the current input and the information from previous states. Training is typically performed using the *Backpropagation Through Time* (BPTT) algorithm. However, standard RNNs often struggle to capture long-term dependencies due to the vanishing or exploding gradient problems. To mitigate these issues, Long Short-Term Memory (LSTM) networks - a specialized RNN architecture capable of learning long-term dependencies - are widely adopted. In this study, we utilize LSTMs to implement the proposed decoder, and the terms RNN and LSTM are used interchangeably hereafter.

A typical LSTM network consists of multiple memory blocks, commonly referred to as LSTM cells. Within each cell, the current input, the previous cell state $\mathbf{c^{t-1}}$, and the hidden state $\mathbf{h^{t-1}}$ are processed together to compute the updated current cell state $\mathbf{c^t}$ and hidden state $\mathbf{h^t}$ that are being transferred to the subsequent cell. Both the hidden and cell states share a common vector dimensionality, determined by the number of nodes within the cell ($\mathbf{c^t}, \mathbf{h^t} \in \mathbb{R}^{M\times 1}$). As illustrated in Figure 3, the memory block regulates the flow of information - specifically adding or removing data from the cell state - via specialized structures termed 'gates.' Each gate consists of a sigmoid neural network layer paired with a pointwise multiplication operator, driven by the concatenation of the previous hidden state $\mathbf{h^{t-1}}$ and the current input $\mathbf{y^t} \in \mathbb{R}^{Z\times 1}$. The LSTM architecture fundamentally incorporates three such gating mechanisms: A forget gate determines which information from the previous cell state should be discarded. An input gate controls the extent to which new information from the current input is added to the cell state. An output gate decides which part of the current cell state will be exported to the hidden state. The mathematical operations for these three gates at time step *t* are defined as follows:

$$\mathbf{i}^t = \sigma(\mathbf{W}^{yi}\mathbf{y}^t + \mathbf{W}^{hi}\mathbf{h}^{t-1} + \mathbf{b}^i) \quad (5)$$

$$\mathbf{f}^t = \sigma(\mathbf{W}^{yf}\mathbf{y}^t + \mathbf{W}^{hf}\mathbf{h}^{t-1} + \mathbf{b}^f) \quad (6)$$

$$\mathbf{o}^t = \sigma(\mathbf{W}^{yo}\mathbf{y}^t + \mathbf{W}^{ho}\mathbf{h}^{t-1} + \mathbf{b}^o) \quad (7)$$

$$\mathbf{e}^t = \tanh(\mathbf{W}^{ye}\mathbf{y}^t + \mathbf{W}^{he}\mathbf{h}^{t-1} + \mathbf{b}^e) \quad (8)$$

where $\sigma(\bullet)$ and $\tanh(\bullet)$ denote the element-wise logistic sigmoid and hyperbolic tangent activation functions, respectively. $\mathbf{i}$, $\mathbf{f}$, and $\mathbf{o} \in \mathbb{R}^{M\times 1}$ are the input gate, forget gate, and output gate, respectively. In addition, $\mathbf{W}^{yi}, \mathbf{W}^{yf}, \mathbf{W}^{yo}, \mathbf{W}^{yc} \in \mathbb{R}^{M\times Z}$; $\mathbf{W}^{hi}, \mathbf{W}^{hf}, \mathbf{W}^{ho}, \mathbf{W}^{hc} \in \mathbb{R}^{M\times M}$; $\mathbf{b}^i, \mathbf{b}^f, \mathbf{b}^o, \mathbf{b}^e \in \mathbb{R}^{M\times 1}$ are the variable weight matrices and bias vectors, which are optimized iteratively throughout the training phase. Consequently, the temporal evolution of the cell state $\mathbf{c^t}$ and hidden state $\mathbf{h^t}$ are formulated as follows, according to expressions (9) and (10):

$$\mathbf{c}^t = \mathbf{f}^t \odot \mathbf{c}^{t-1} + \mathbf{i}^t \odot \mathbf{e}^t \quad (9)$$

$$\mathbf{h}^t = \mathbf{o}^t \odot \tanh(\mathbf{c}^t) \quad (10)$$

In these formulations, the symbol $\odot$ represents the Hadamard product, facilitating the element-wise multiplication of the constituent vectors.

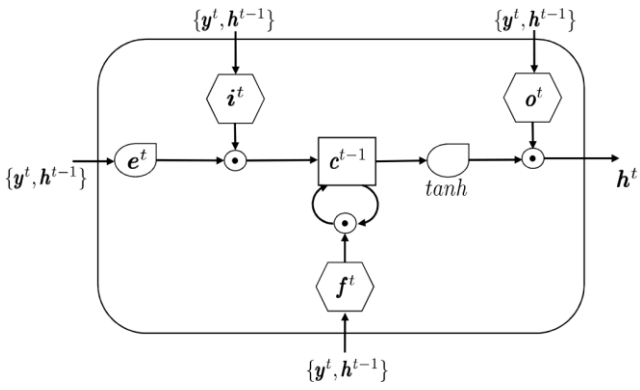

Figure 3. *A basic LSTM cell*

4.2. The architecture of the proposed RNN-based Decoder

To facilitate simultaneous RLL and FEC decoding, we employ an LSTM-based framework that accepts Q-dimensional data from OOK demodulator data $\mathbf{y} \in \mathbb{R}^{1\times Q}$ and yields K-dimensional estimated $\hat{\mathbf{u}} \in \mathbb{R}^{1\times K}$ as output. The decoder architecture, shown in Figure 4, features an LSTM layer coupled with a fully-connected layer. The LSTM layer comprises Q repeated cells with an identical internal structure. At each discrete time step, only one cell operates. The current input of LSTM is $y^t \in \mathbf{y} \triangleq \left[y_0, y_1, ..., y_{Q-1} \right]$, while the outputs $\mathbf{h}^t$ and $\mathbf{c}^t$ are $M \times 1$ vectors.

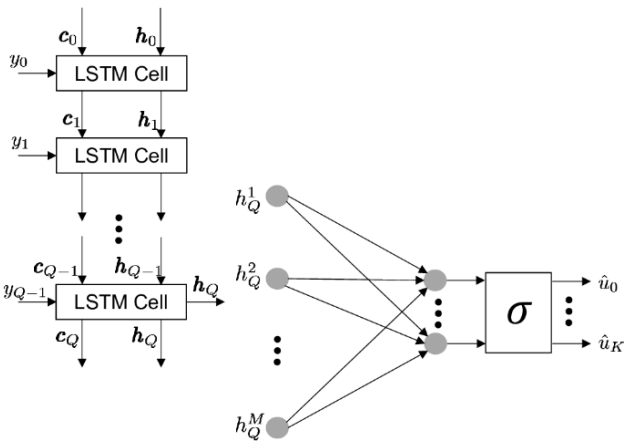

Figure 4. *Proposed RNN-based decoder architecture*

The final fully-connected layer, which consists of M inputs $\mathbf{h}^Q$ and K outputs, $\hat{\mathbf{u}}$ performs the mapping $\mathbf{f}: \mathbb{R}^{M\times 1} \to \mathbb{R}^{K\times 1}$ from the hidden representation to the output space. Since the output of DL-based decoder represents the information bits, a sigmoid activation function is employed to squash the output values into the [0, 1] range. These values can then be interpreted as probabilities, from which the estimated bits are inferred as "1" or "0".

4.3. Training Strategy

The objective of the training process is to determine the optimal network parameters, specifically the weights and biases, that minimize the discrepancy between the decoder's predicted output $\hat{\mathbf{u}}$ and the ground-truth transmission bits $\mathbf{u}$. This optimization relies heavily on the selection of an appropriate loss function. In this study, the Mean Squared Error (MSE) is employed as the objective function, defined as:

$$L_{MSE} = \frac{\left\| \mathbf{u} - \hat{\mathbf{u}} \right\|^2}{K} \tag{11}$$

To facilitate network training, a dataset was constructed comprising transmitted information sequences $\mathbf{u}$ and their corresponding received vectors $\mathbf{y}$. In the proposed VLC system, training samples are generated by randomly selecting information sequences $\mathbf{u}$, which are then processed through a signal chain including a channel encoder, a Run-Length Limited (RLL) encoder, an OOK modulator, and an AWGN channel, as illustrated in Figure 1. For the numerical evaluations, 1-bit hard-decision sequences derived from Eq. 2 are initially employed as the received vectors $\mathbf{y}$ for training and testing the DL-based decoder. Furthermore, additional datasets were generated using 2-bit and 3-bit OOK demodulators, as described in Figure 2, to assess the impact of multi-bit precision.

In the proposed demodulation framework, both the 2-bit and 3-bit configurations utilize a fixed-threshold scheme for the decision circuits. For the 2-bit demodulator, $2^2 - 1 = 3$ comparators are implemented: the primary hard-decision threshold is set at $V_{th0} = V_P / 2$, while the auxiliary confidence levels are established at $V_{th+1} = (V_P - V_{th0}) / 2$, and $V_{th-1} = V_{th0} / 2$. In contrast, the 3-bit architecture employs an array of $2^3 - 1 = 7$ to achieve finer quantization. The decision space is partitioned by a set of upper threshold voltages, denoted as $V_{th+3} = 3/4(V_P - V_{th})$, $V_{th+2} = 1/2(V_P - V_{th0})$, $V_{th+1} = 1/4(V_P - V_{th0})$, alongside a corresponding set of lower thresholds, $V_{th-1} = 3/4V_{th0}$, $V_{th-2} = 1/2V_{th0}$, $V_{th-3} = 1/4V_{th0}$, respectively.

4. Numerical Results and Analysis

In this section, we conduct a performance evaluation of the DL-based decoder within the proposed VLC framework. All experiments utilize a combination of rate-1/2 Polar codes and Manchester RLL encoding. The output dimension for the LSTM units is configured at 256. For the optimization phase, we adopt the Adam optimizer with a batch size of 128 and a learning rate $\eta = 0.001$. Network parameters, including weight matrices and bias vectors, are initialized via the Xavier method. Additionally, a dropout layer with a 0.2 probability is integrated into the architecture as a regularization technique to prevent overfitting.

Table 1. *Basic experiment setting*

Channel	AWGN
Modulation	OOK
LSTM cell size	256
Mini-batch size	128
SNR range	{-2, 0, 2, 4,6, 8,10, 12} dB
Training samples per SNR	10^6
Testing samples per SNR	10^4
Dropout probability	0.2
Initialization method	Xavier initialization
Optimization method	Mini-batch SGD with Adam

The channel environment and simulation parameters are adopted from [16] to ensure the validity and consistency of the data generation process. To develop the proposed deep learning (DL)-based decoder, distinct datasets were generated for the training and evaluation phases. Specifically, the training set was utilized for model optimization, while a separate test set was employed to assess the Symbol Error Rate (SER). It is important to note that the additive noise for the training and testing procedures was synthesized using independent random

seeds to ensure statistical independence. The simulation environment was developed in Python 3.6 using the TensorFlow 1.8.0 library. A comprehensive summary of the experimental parameters is provided in Table 1. To benchmark performance, our proposed DL-based decoders are compared against a baseline configuration consisting of SISO Manchester combined with 1/2 Polar decoders (Ref. Man-Polar [5]).

4.4. Error-Correction Performance

The first experiment evaluates the performance of various DL-based decoder configurations across different information sequence lengths $K=4, N=16$. As illustrated in the SER curves of Figure 5, the 1-bit demodulation DL-based decoder (1bit-RNN) consistently exhibits the least favorable performance across all tested scenarios. Conversely, the 3-bit variant (3bit-RNN) achieves significantly higher decoding accuracy, providing a coding gain of over 1.5 dB compared to the 1bit-RNN at an SER of 10^{-5}. Furthermore, the 2-bit demodulation decoder (2bit-RNN), performs competitively, showing only a marginal 0.4 dB degradation relative to the 3-bit configuration.

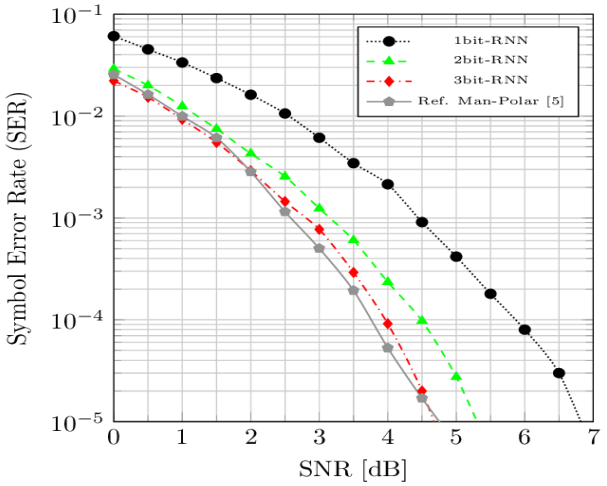

Figure 5. *SER performance of K = 4, N = 16*

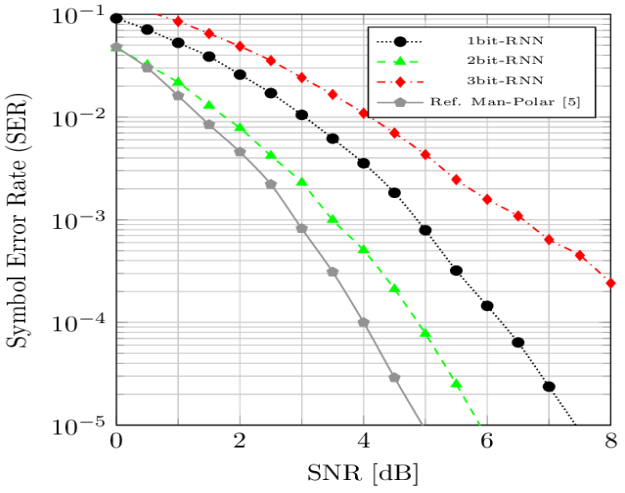

Figure 6. *SER performance of K = 8, N = 32*

Figure 6 presents a comparative analysis of the SER performance for the proposed decoders under the condition of $K=8, N=32$. It is observed that the system exhibits a noticeable performance degradation relative to the results obtained for $K=4, N=16$. A remarkable performance gap is observed, where the 3-bit RNN outperforms the 1-bit RNN decoder by roughly 0.8 dB at an SER level of 10^{-3}. This improvement highlights the advantage of higher quantization precision in the proposed VLC system. Furthermore, the SER performance of the 3bit-RNN decoder exhibits saturation at 5.10^{-3} in a high SNR regime. within the high-SNR regime. This occurs because the high dimensionality of the input vector, specifically $(2^3-1)\times 32=224$ introduces an excessively complex mapping between the input and output sequences that the RNN fails to capture effectively. This phenomenon indicates a state of model underfitting, where the neural network's architecture lacks the necessary capacity or training depth to generalize the underlying patterns in such a high-dimensional feature space.

4.5. Computation Time of Proposed Decoders

To assess the computational efficiency and practical feasibility of the 1-bit, 2-bit, and 3-bit RNN configurations, we benchmarked their performance across three distinct hardware platforms: a high-performance PC (Intel Core i7-3970X CPU @ 3.90GHz and NVIDIA GeForce GTX 1080 GPU), an NVIDIA Jetson TX2, and a Raspberry Pi 3 Model B. Figure 7 illustrates the average training time per mini-batch for the scenario where $K=4, N=16$. The results demonstrate that the proposed decoders are compatible with all evaluated platforms. Notably, the training throughput of the 1bit-RNN is approximately two and four times greater than that of the 2bit-RNN and 3bit-RNN, respectively. Furthermore, a significant performance gap is observed between the platforms; the Raspberry Pi 3, equipped with a 1.2GHz ARMv8 Cortex-A CPU, exhibits substantially higher latency compared to the GPU-accelerated Jetson TX2 and PC. Specifically, the training time per mini-batch on the Raspberry Pi is 6.3s, whereas the Jetson TX2 and PC require only 0.157s and 0.041s, respectively.

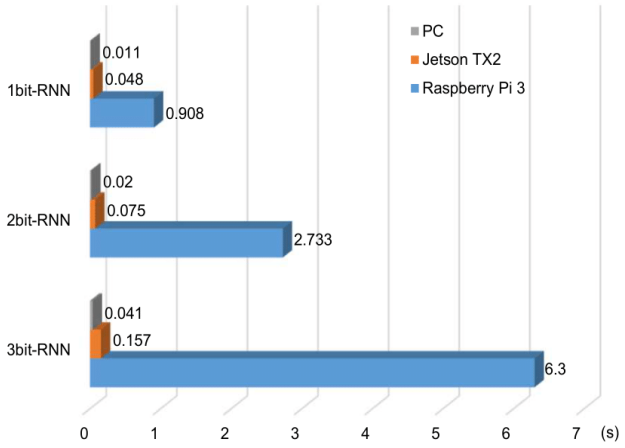

Figure 7. *The computation time of 3 configurations in the training phase*

Figure 8 presents a comparative study of the average computational latency across the selected hardware

environments for 10,000 samples with $K = 4, N = 16$. To maintain parameter consistency, the training weights were exported from a workstation and deployed on the edge devices (Jetson TX2 and Raspberry Pi 3). The 1-bit RNN variant demonstrated the highest efficiency, with a 0.045s processing time (900 kbps bandwidth), while the 2 and 3-bit configurations suffered from increased computational complexity (0.136s and 0.316s). While the Jetson TX2 demonstrated a tenfold increase in processing time relative to the PC, the Raspberry Pi 3 experienced a more drastic performance penalty, with execution times escalating to 20.2s ($B \approx 1{,}98$ *kbps*), 61.4s ($B \approx 650$ *bps*), and 142.9s ($B \approx 280$ *bps*). Table 2 summarizes the calculated optical clock rates necessary to maintain the proposed systems' coding efficiency.

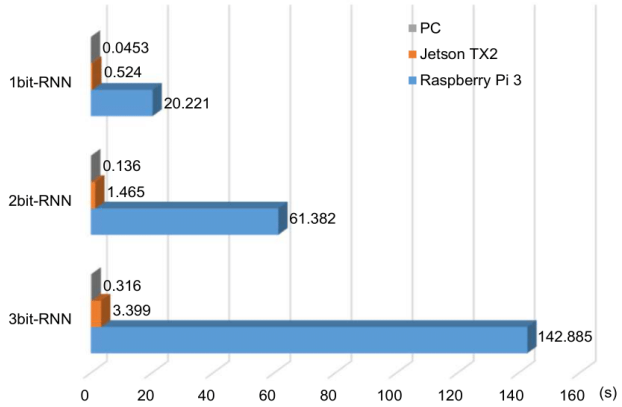

Figure 8. *The computation time of the 3 configurations in the testing phase*

Table 2. *Estimated Optical Clock Rates of our VLC systems on different platforms*

	1bit-RNN	***2bit-RNN***	***3bit-RNN***
PC	3.5 Mhz	1.2 Mhz	500 kHz
Jetson TX2	305 kHz	110 kHz	47 kHz
Raspberry Pi 3	8kHz	2.6 Khz	1.1 kHz

The results illustrated in Table 2 verify that all optical clock frequencies surpass 200 Hz, maintaining a maximum flickering time period (MFTP) of less than 5ms. Since these values align with established human eye safety thresholds, this confirms that the proposed RNN-based decoders are viable for real-world OOK-VLC implementations. This remains true even when utilizing budget-friendly embedded hardware where computation resources may be limited.

While deep learning (DL)-based decoders demonstrate a slight performance trade-off relative to the Ref. Man-Polar [5] benchmark, they offer significant advantages in terms of computational latency. As illustrated in Figure 8, which evaluates the average decoding time for 10,000 test samples (K=4) on a PC, DL-based architectures exhibit superior efficiency. Specifically, when operating solely on a CPU, the 3-bit, 2-bit, and 1-bit RNN decoders achieve execution time reductions of 38%, 73%, and 91%, respectively, compared to the reference decoder. Furthermore, the integration of GPU acceleration significantly enhances the throughput of these DL-based models.

5. Conclusion

In this work, we have developed and analyzed an RNN-based RLL-FEC decoding strategy for OOK-based VLC systems, focusing on 1-bit, 2-bit, and 3-bit configurations. Our multi-platform evaluation reveals that these proposed structures effectively learn decoding functions that rival the performance of the traditional SISO RLL and Polar concatenation method. Despite achieving similar SER for short-packet transmissions, the RNN-based approach significantly lowers processing requirements. These outcomes underscore the feasibility of utilizing deep learning for real-time decoding in VLC systems. Future research will explore several optimization pathways as follows.

- In the current study, the RNN-based decoders were implemented using baseline LSTM cell configurations, standard optimization algorithms, and default loss functions. To achieve a more robust and high-performance framework, future research should focus on a systematic investigation of alternative LSTM architectural variants and the rigorous tuning of hyperparameters. Specifically, evaluating the impact of different learning rate schedules, dropout rates, and customized objective functions will be essential in identifying an optimal neural structure tailored for high-reliability decoders.

- While the current RNN-based decoders exhibit exceptional error-correction capabilities for short message lengths ($K = 4$ or $K = 8$), the data in Section V indicates a progressive performance degradation as the information block length increases. To mitigate this scalability challenge, future research must move beyond standard RNN configurations. Exploring hybrid architectures, such as integrating attention mechanisms or Transformer-based blocks, could potentially enhance the model's ability to maintain long-range dependencies and robust decoding accuracy across larger data frames.

- While the current investigation assumes a stationary channel environment, practical VLC deployments are often characterized by temporal fluctuations in channel parameters. In real-world scenarios, these characteristics may vary significantly even within a single symbol interval. To enhance the practical utility of our proposed decoders, subsequent work will prioritize the analysis of unstable channel conditions of VLC systems.

- Since the current results are collected from a simulation, another essential future work is the hardware implementation of the proposed architecture. The VLC's transceiver must be designed and implemented in the field-programmable gate arrays (FPGAs) to experimentally investigate different parameters such as power consumption and hardware resources. Moreover, in a low-bit OOK demodulator design, the comparator power, noise, and latency must be taken into consideration.

Acknowledgement: This work is supported by The university of Danang - VN-UK Institute for Research and Executive Education (grant number T2023-VNUK-02).

REFERENCES

[1] A. Jovicic, J. Li, and T. Richardson, "Visible light communication: opportunities, challenges and the path to market," *IEEE Communications Magazine*, vol. 51, no. 12, pp. 26–32, Dec. 2013, doi: https://doi.org/10.1109/mcom.2013.6685754.

[2] "802.15.7-2011 - IEEE Standard for Local and Metropolitan Area Networks--Part 15.7: Short-Range Wireless Optical Communication Using Visible Light," *IEEE Standard*, pp. 1-309, Sep. 2011. DOI: 10.1109/IEEESTD.2011.6016195

[3] H. Wang and S. Kim, "Soft-Input Soft-Output Run-Length Limited Decoding for Visible Light Communication," *IEEE Photonics Technology Letters*, vol. 28, no. 3, pp. 225–228, Oct. 2015, doi: https://doi.org/10.1109/lpt.2015.2492607.

[4] H. Wang and S. Kim, "Bit-Level Soft Run-Length Limited Decoding Algorithm for Visible Light Communication," *IEEE Photonics Technology Letters*, vol. 28, no. 3, pp. 237–240, Feb. 2016, doi: https://doi.org/10.1109/lpt.2015.2493203.

[5] H. Wang and S. Kim, "Dimming Control Systems With Polar Codes in Visible Light Communication," *IEEE Photonics Technology Letters*, vol. 29, no. 19, pp. 1651–1654, Aug. 2017, doi: https://doi.org/10.1109/lpt.2017.2737026.

[6] D. D Le, D. P. Nguyen, T. H. Tran, and Y. Nakashima, "Log-Likelihood Ratio Calculation Using 3-Bit Soft-Decision for Error Correction in Visible Light Communication Systems," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 101, no. 12, 2210 - 2212, 2018.

[7] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: https://doi.org/10.1038/nature14539.

[8] T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, Dec. 2017, doi: https://doi.org/10.1109/tccn.2017.2758370.

[9] T. Gruber, S. Cammerer, J. Hoydis, and S. ten Brink, "On deep learning-based channel decoding," 2017 51st Annual Conference on Information Sciences and Systems (CISS), Mar. 2017, doi: https://doi.org/10.1109/ciss.2017.7926071.

[10] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be'ery, "Deep Learning Methods for Improved Decoding of Linear Codes," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 119–131, Feb. 2018, doi: https://doi.org/10.1109/jstsp.2017.2788405.

[11] W. Lyu, Z. Zhang, C. Jiao, K. Qin, and H. Zhang, "Performance Evaluation of Channel Decoding with Deep Neural Networks," *2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, May 2018, doi: https://doi.org/10.1109/icc.2018.8422289.

[12] F. Liang, C. Shen, and F. Wu, "An Iterative BP-CNN Architecture for Channel Decoding," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 144–159, Feb. 2018, doi: https://doi.org/10.1109/jstsp.2018.2794062.

[13] S. Dorner, S. Cammerer, J. Hoydis, and S. ten Brink, "Deep Learning Based Communication Over the Air," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 132–143, Feb. 2018, doi: https://doi.org/10.1109/jstsp.2017.2784180.

[14] H. Lee, I. Lee, T. Q. S. Quek, and S. H. Lee, "Binary signaling design for visible light communication: A deep learning framework," *Opt. Express*, vol. 26, no. 14, pp. 18131–18142, Jul. 2018.

[15] Chowdhury, M. Z., Joha, M. I., Rahman, M. M., Kabir, M. S., & Jang, Y. M., "Machine learning and deep learning in VLC systems: A comprehensive survey," IEEE Open Journal of the Communications Society, Aug. 2025.

[16] T. Komine and M. Nakagawa, "Fundamental analysis for visible-light communication system using LED lights," *IEEE Transactions on Consumer Electronics*, vol. 50, no. 1, pp. 100–107, Feb. 2004, doi: https://doi.org/10.1109/tce.2004.1277847