Intelligent Receiver for Frequency Hopping Signals Using Deep Learning

Mahmoud M. Qasaymeh[1,*], Ali Alqatawneh[1], and Ahmad F. Aljaafreh[1,2]
[1] Computer Engineering and Communication Department, Faculty of Engineering, Tafila Technical University, Tafila, Jordan
[2] Department of Computer Science and Software Engineering, College of Engineering, University of Detroit Mercy, Detroit, USA
Email: qasaymeh@ttu.edu.jo (M.M.Q.), ali.qatawneh@ttu.edu.jo (A.A.), aljaafah@udmercy.edu (A.F.A.)

***Abstract*—This paper presents a promising Deep-Learning (DL) approach for accurate symbol detection in a slow Frequency Hopping (SFH) wireless communication System under a Narrow Band (NB) multipath channel fading. A feedforward neural network with three layers of input, hidden, and output was employed for deep learning. The neural network is designed to take 80 features as input, representing the received signal samples at the receiver. The neural network is trained to anticipate the transmitted symbol based on the provided training dataset, utilizing different modulation techniques, including Binary Phase Shift Keying (BPSK), Quadrature Phase Shift Keying (QPSK), 8-PSK, and 16-PSK. Additionally, computer simulations are conducted to verify the effectiveness of the proposed method across various modulation schemes. The generated training loss and validation loss curves confirmed the ability of the receiver to learn.**

***Index Terms*—categorical cross-entropy loss, channel gains, confusion matrix machine learning, frequency hopping, loss function, Narrow Band (NB) multipath channel, neural network, time delay estimation**

I. Introduction

Wireless networks have become an essential part of modern communication infrastructure. Wireless networks provide vital services and play an important role in improving the quality of life. They are currently crucial drivers of Internet-of-Things (IoT) applications in a wide range of sectors and industries, whether for military purposes or civilian purposes, which may include health care, manufacturing, smart cities, transportation, and intelligent anticipating and responding to natural disasters [1]. In IoT network, information is collected from remote sensors and devices and sent to a central unit for processing over a wireless network. Despite the widespread prevalence of wireless networks, they face critical security threats, with radio-jamming attacks being the most significant threat. Low-cost devices such as Software-Defined Radio (SDR) can be utilized to perform jamming by emitting high radiation energy on the same channel used by the legitimate signal [2]. Jamming attacks make it difficult for wireless devices on the receiving end to retrieve the transmitted information from the legitimate signal that has been jammed.

Different anti-jamming techniques have been proposed to mitigate the impact of physic-layer jamming on wireless networks [3–6]. These techniques can be classified into four categories: coding using orthogonal pseudo-random codes to improve Signal-to-Interference Ratio (SINR), power control by optimizing the transmitted, space diversity using multi-antenna signal processing, and spreading the signal in the frequency domain [3].

Spread spectrum techniques such as Frequency Hopping (FH) and Direct Sequence (DS) were employed in commercial wireless communications to reduce the impact of jamming [4]. However, some jamming techniques can recognize the spreading sequence from the cyclostationary behavior of the transmitted signal, leading to the complete jamming of the legitimate signal. To overcome this issue, the anti-jamming scheme in [4] utilized a secret shared code to create a varying spreading sequence. Compared to the DS spread spectrum, the FH spreading spectrum scheme is more immune to narrow-band interference but has less implementation complexity than the DS spread spectrum [5]. FH-based wireless communications provide robustness against frequency-selective fading in millimeter-wave cellular communications [6].

Furthermore, FH spread spectrum communication is appropriate for IoT applications due to its flexibility in handling dense wireless networks, as in the case of the IoT paradigm. A Long-Range FH (LR-FH) scheme was proposed in [7] to handle distant communications, such as satellite-linked IoT devices. An overview and performance analysis for LR-FH were presented in [8]. The result of the study demonstrated that the LR-FH-based wireless network has significant flexibility for expansion and growth. However, it comes at the expense of capacity for each device connected to the network. Recently, the FH spread spectrum has been deployed to enable a single communication system to provide radar and communication tasks simultaneously [9]. Identifying the Channel State Information (CSI) and hopping parameters such as hopping sequences, hopping frequencies, and hopping rates is crucial to accurately recover the data symbol from the received signal in FH-based wireless networks. Therefore, various signal detection and

Manuscript received February 20, 2024; revised May 26, 2024; accepted April 2, 2024.
*Corresponding author

doi: 10.18178/ijeetc.13.5.366-373

frequency hopping parameter estimation schemes have been proposed [10–17].

The article in [10] proposed a joint signal parameter estimation technique for FH-based communication with M-ary Frequency Shift Keying (MFSK). Also, the proposed scheme employs a Maximum Likelihood Estimation (MLE) along with a Smooth Pseudo-Wigner–Ville Distribution (SPWVD) to estimate hopping parameters. An adaptive smoothed Wigner Ville does not require any former experience of the signal parameters since the kernel parameters can be identified from the signal features [11]. Despite the Wigner–Ville distribution offering a significant degree of resolution in both time and frequency, its computational complexity is high.

To enhance the estimate of signal parameters under low Signal-to-Noise (SNR) scenarios, authors in [12] developed a frequency hopping spectrum estimation scheme based on a sparse Bayesian approach. The approach in [12] involves partitioning the received signal into overlap measurements; the Sparse Bayesian Learning (SBL) approach was used to exploit the frequency hopping sequence through statistical methods. Based on Space-time Frequency Analysis (STFA) and Matrix Joint Diagonalization (JDM), a blind FH signal parameter estimation algorithm was presented in [13]. Results show that under low SNR as -4 dB, the proposed scheme was able to estimate hop period, hop start time, hop end time, and frequency hopping, where the frequency predicted with an accuracy reaches 73.26%; the estimation accuracy can reach 97.374% at an SNR of 5dB. The computational complexity of STFA is acceptable. However, the STFA can only satisfy the high accuracy requirement for time and frequency domains at a different time [14].

Energy detection-based FHSS signal detection was proposed in [15–17]. The proposed signal detection method in [17] uses the cross-correlation between the frequency-domain noisy-received signals to extract the decision statistic for energy detection; however, the adopted method in [17] cannot address the signal detection for slow-frequency hopping systems. In Unmanned Aerial Vehicles (UAVs), frequency hopping is used for data transmission and control [18]. The paper in [18] presented an adaptive noise-threshold calculation for frequency hopping signal detection for UAVs. For drones, a compressive sampling method that comprises frequency hopping signal spectrum extraction and soft detection was proposed in [19] to address the detection of FH-based Radio Control (RC) signals.

Machine Learning (ML) has demonstrated great potential in wireless networks [20]. ML can be utilized in two different ways. Firstly, ML can be employed directly to improve the performance of specific task modules in a wireless network, such as modulation, error detection and correction, channel estimation, and signal detection. In the second way, the ML can be employed as an End-to-End (E2E) system. In the E2E system, the transmission and reception modules are replaced with ML. Employing ML in a communication system can significantly improve the system's overall efficiency [20].

Adopting a Short-Time Fourier Transform (STFT) to acquire the signal spectrogram for FH signal detection was presented in [21] under SNR and interference. A Deep Learning (DL) image processing algorithm was combined with signal estimation and detection to reduce the estimation error. Deep learning using neural networks was adopted in [22–28] for signal detection and channel parameter estimation. In [22], a hybrid Convolution Neural Network (CNN) and Recurrent Neural Network (RNN) were used for FH signal detection under unknown hopping rates to mitigate problems caused by the lack of time-frequency resolution and spectral leakage. Using CNN, the existence of the frequency hopping signal was presented in [23]. However, the proposed scheme in [23] could not identify the hopping frequencies [24].

An intelligent anti-jamming receiver for FH-based wireless networks was proposed in [25]. The proposed method combines time–frequency signal processing and deep learning to achieve accurate frequency-hopping sequence estimations. The combined model comprises a convolution CNN and a Gated Recurrent Unit (GRU). Results in [25] showed that the proposed method can flexibly estimate the frequency hopping sequence regardless of sequence length. Reference [26] also combined deep learning with time-frequency analysis to identify abnormal signals successfully. Finally, radio Frequency Jamming detection and classification using machine learning were introduced in [27] and [28].

This paper introduces an intelligent receiver designed for a frequency-hopping wireless communication setup, which operates within a multipath fading channel. The proposed receiver utilizes a deep learning algorithm to decode the received signal. Specifically, a Feedforward Neural Network (FFNN) consisting of input, hidden, and output layers is utilized for deep learning training purposes. The collected dataset underwent a prepossessing procedure to guarantee consistency and simplify model training.

The rest of the paper is organized as follows: Section II outlines the problem formulation. Section III details the development of the proposed model. Section IV presents simulation results and discussions. Finally, Section V concludes the paper.

II. Problem Formulation

In this paper, we consider the baseband model of multipath FH system as shown in Fig. 1, modeled as a time-varying linear filter [29–31]. The equivalent low-pass channel impulse response is given by:

$$h(\tau,t)=\sum_{i=1}^{P(t)}\beta_i(t)e^{-j\emptyset_i(t)}\,\delta(t-\tau_i(t)) \tag{1}$$

where $\beta_i(t), \emptyset_i(t)$ and $\tau_i(t)$ are respectively the channel magnitude, channel phase, and the associated time delay of the ith multipath, which are assumed to be independent. The received signal is the convolution of the input signal $x(t)$ with the equivalent low-pass channel impulse response $h(\tau,t)$

$$y(t)=x(t)*h(\tau,t) \tag{2}$$

where * is the convolution operator. The baseband model of the FH received signal in a multipath environment of the sample version form can be expressed as

$$y^{(n)}(kT) = \sum_{i=1}^{P(t)} \beta_i(t) e^{-j\phi_i(t)} e^{-j2\pi f_n \tau_i} s(kT - \tau_i; b_n) + w^{(n)}(kT) \quad (3)$$

where $y^{(n)}(kT)$ is the received signal in the nth hop, T is the sampling period, f_n is the frequency in the nth hop, b_n is the sequence of the transmitted bits. $s(kT; b_n)$ is the transmitted baseband signal and $w^{(n)}(kT)$ is the white Gaussian noise parameter. Parameter P denotes the total number of multi paths considered in the model. Channel parameters (channel magnitude, channel phase, and time delay), and the transmitted bit the sequence are unknown. The hop frequency is known for both transmitter and receiver [30]. The discrete time version of Eq. (3) is given by:

$$y^{(n)}(t) = \sum_{i=1}^{P} \beta_i e^{-j2\pi f_n \tau_i} s_i(t) + w^{(n)}(t), t = 1,2, \ldots, M \quad (4)$$

where $s_i(t)$ is the delayed version of the transmitted signal though the ith multipath, β_i is considered complex standard Gaussian random, τ_i is considered uniform random distribution and the AWGN is considered complex standard Gaussian random.

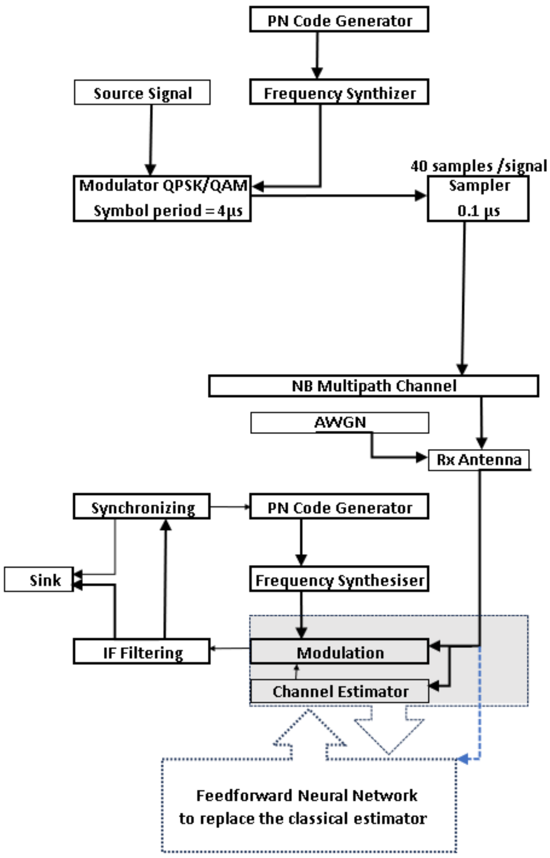

Fig. 1. Baseband model of multipath FH system.

The FH communication system depicted in Fig. 1, the parameters listed in Table I and the full details were previously discussed in [30]. In the transmitter, the data information is modulated by the modulator. The study involved a progressive exploration of modulation schemes, commencing with the baseline Binary Phase Shift Keying (BPSK) and subsequently extending to more complex schemes, including Quadrature Phase Shift Keying (QPSK), 8-PSK, and 16-PSK.

TABLE I: PARAMETERS OF BASEBAND MODEL OF MULTIPATH FH SYSTEM

Parameters	Description
P	Random Number of Multipath P=1,2,3,4,…
i	Multipath index i=1:1:P
n	Frequency Hop index
t	Time index $t = 1,2, \ldots M$
β_i	The ith path channel gain, Complex Gaussian random distribution
τ_i	The ith path time delay, uniform random distribution
f_n	The frequency in the nth hop Known for both T_x and R_x, Randomly selected from 75 frequencies 1899 to 1929 MHz.
s_k	The kth symbol taken from QAM
$s_k(t)$	The kth continuous time signal representing the kth Symbol
$s_{k,i}(t)$	The delayed version of the the kth transmitted signal though the ith multipath.
$h(\tau, t)$	The Impulse response of NBFH multipath model
$y^{(n)}(k)$	The kth received signal in the n-th consists of M samples
$w^{(n)}(k)$	AWGN to the k-th received signal in the nth
y	The received M samples of the the kth signal in the nth hop
yy	NN input

As an example, let's take quadrature phase shift keying (QPSK), where the binary signal yields four unique input combinations: 00, 01, 10, and 11, corresponding to four symbols: s_0, s_1, s_2, and s_3. Consequently, in QPSK, the binary input data are grouped into pairs of two bits to create a symbol. Within the modulator, each symbol produces one of the four potential output phases: +45°, +135°, –45°, and –135°.

The problem addressed in this article is the estimation of the transmitted bits/symbols, in other words, referring to Eq. (4), estimating the transmitted baseband signal $s_i(t)$ based on the received signal $y^{(n)}(t)$ under additive white Gaussian noise and multi path scenario via machine learning. The received M samples of the nth hop is collected in vector **y** given by:

$$y = [y_0 \quad y_1 \quad y_2 \; \ldots \ldots y_{M-1}] \quad (5)$$

The real part of y is concatenated with the imaginary part to form a vector of length 2M which will be the input to ML-based receiver.

$$\text{yy} = [\text{Real } (y), \text{Imaginary } (y)] \quad (6)$$

III. DEVELOPMENT OF ML METHOD

In recent years, there has been increasing interest in using machine learning models to replace the conventional channel estimator. Machine learning models can be trained on a large dataset of channel measurements, and they can learn to estimate the channel response more accurately than traditional statistical models.

There are several potential advantages to using a machine learning model to replace the conventional channel estimator. First, machine learning models can be more accurate than traditional statistical models, especially in complex or rapidly changing channels. Second, machine learning models can be more adaptive than traditional statistical models, and they can learn to adapt to changes in the channel over time. Third, machine learning models can estimate a broader range of channel characteristics than traditional statistical models, such as the channel's directionality or polarization. However, there are also some challenges associated with using machine learning models for channel estimation. First, machine learning models can be computationally expensive to train and run. Second, machine learning models can be sensitive to the quality of the training data, and they may only perform well if the training data is representative of the actual channel conditions. Third, machine learning models can be challenging to interpret, and it can be difficult to understand how they are making their predictions. The most straightforward deep architecture is the Multi-Layer Perceptron (MLP), which consists of a succession of fully connected layers separated by activation functions. Despite their simplicity, MLPs remain an essential tool when the dimension of the signal to be processed is manageable.

A. Datasets Generation

The first step of the research is to create a synthetic dataset that mimics the complexity of actual signal delays. A random number of multipath models with a wide range of SNR (SNR=0, 5, 10, … dB) is considered. The transmission was confined to the range 1899 MHz to 1929 MHz, the uplink frequency range for the PCS system. A total of 75 frequencies were considered, with a 400 kHz frequency separation between carriers. The symbol period was set to 4 μs. Twenty thousand signals were generated, with each signal/symbol having a length of 40 complex samples; real and imaginary parts are concatenated on 80-length input feature vectors. The time delay is taken randomly from a uniform distribution between zero and one, and the channel gain is taken as a random complex normal distribution.

B. Dataset Preprocessing

The dataset went through preprocessing procedures to guarantee consistency and make model training easier. Categorical labels are used for training. After that, the dataset was split into training (80%) and testing (20%) groups [32]. To accommodate this variability, the output layer utilizes a linear activation function, allowing for the flexibility required to capture such experiment-specific nuances.

C. Neural Network Architecture

A feedforward neural network is an artificial neural network where the information moves in only one direction forward from the input layer, through the hidden layers, and finally to the output layer. Each layer consists of nodes (neurons), and connections between nodes have associated weights. During training, the weights are adjusted to minimize the difference between predicted and actual output. The architecture of the FFNN employed in this study shown in Fig. 2, comprises three hidden layers, each playing a pivotal role in capturing the intricate relationships within the synthesized signals.

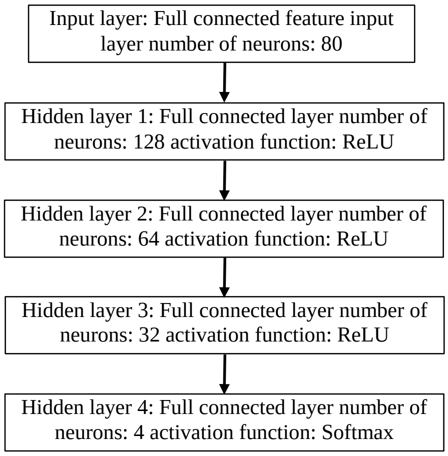

Fig. 2. Feedforward Neural Network (FNN) architecture, the diagram automatically generated by ChatUML.

1) Input layer

The input layer is designed to handle 80 features, representing the real and imaginary parts of the 40 complex samples. The values associated with these neurons would be the actual values of the features in your input data. The primary role of the input layer is to pass this information forward to the subsequent layers in the neural network. The subsequent layers, often hidden and an output layer, perform computations and learn patterns from the input data by adjusting weights associated with the connections between neurons.

2) Hidden layers

The hidden layers consist of three layers:

A) Hidden layer1

This is the first hidden Fully Connected Layer with 128 neurons. Each of the 128 neurons in this layer is connected to all 80 neurons in the input layer. So, the fully connected layer serves as the first hidden layer, and it directly receives input from the features in the input data. The weights associated with these connections are learned during training to allow the network to capture complex patterns and relationships in the data. The rectified linear unit (ReLU) activation function is applied after the first hidden layer; the ReLU is a common activation function used in neural networks. It introduces non-linearity by outputting the input for positive values and zero for negative values. This non-linearity is important for the network to learn complex patterns. The ReLU activation function is defined as follows:

$$f(x)=max(x,0) \tag{7}$$

B) Hidden layer 2

This is the second hidden fully connected layer, with 64 neurons directly connected to the first hidden layer. This implies that the 64 neurons in the second hidden layer receive input from the 128 neurons in the first hidden layer.

The weights associated with these connections are learned during training, allowing the neural network to capture and model complex patterns in the data. The ReLU activation is applied after this layer.

C) Hidden layer 3

This is the third fully connected layer with 32 neurons: The values from the ReLU activation function in the second layer are connected to each of the 32 neurons in this third hidden layer. During training, the network learns the weights associated with these connections to capture patterns and relationships in the data. The additional non-linear transformations to the input data aid the network's ability to learn complex representations. The ReLU activation function is used after this layer.

3) Output layer

This is a fully connected output layer with a number of neurons equal to the number of classes. In the QPSK case, the number of classes is set to 4, representing the possible transmitted symbols. Softmax activation is applied to the output of the fully connected layer. To convert the raw scores into class probabilities. For deep learning classification purposes, the Softmax of the output of the classifier network is the probability distribution, which is then converted to the binary matrix in which each class is represented by a unique binary number for classification purposes. The SoftMax activation function is commonly employed in the output layer for multi-class classification problems. It converts the raw output scores of the network into probabilities, ensuring that the sum of the probabilities for each class is 1. This makes it suitable for classification tasks. It specifies the loss function and performance metrics for multi-class classification tasks. The SoftMax activation function can be described mathematically as:

$$\sigma_i = \frac{e^{x_i}}{\sum_{j-1}^{C} e^{x_j}} \tag{8}$$

where the vector $x = [x_1, x_2, \dots, x_C]$ represents the raw scores for every single class, $\sigma = [\sigma_1, \sigma_2, \dots, \sigma_C]$ is the Softmax function output vector with x_i being the raw score for the class i, and C is the number of classes. The Softmax activation function normalizes the raw scores x_i so that the output values lie between 0 and 1 such that the sum of the probabilities of the raw scores is one, i.e. $\sum_i^C \sigma_i = 1$.

The categorical cross-entropy loss is a popular and effective performance metric. It is a widely utilized metric for addressing multi-class classification challenges. It quantifies the disparity between the predicted probability distribution, determined through the SoftMax activation function, and the actual probability distribution represented by one-hot encoded labels. The objective function is often categorical cross-entropy for a multi-class classification task such as modulation recognition. Categorical cross-loss is a measure of the difference between two probability distributions. The SoftMax activation function and Categorical cross-entropy work together in multi-class classification tasks. The Categorical Cross-Entropy Loss can be represented in mathematical form as [32]:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(p_{ij}) \tag{9}$$

where N is the total number of samples in the validation set, C is the number of classes, y_{ij} is the indicator function that is 1 if the true label for sample i is class j and 0 otherwise, and p_{ij} is the predicted probability assigned by the model to sample i being of class j.

D. Neural Network Training Process

The adaptive moment estimation (Adam) optimization algorithm is a popular and effective optimization algorithm used to minimize the loss of function during the training of neural networks. Adam combines ideas from root mean square propagation (RMSprop) and momentum. Adam maintains two moving averages for each parameter: the first moment (mean) and the second moment (uncentered variance). These moving averages are computed using exponential decay and are used to adjust the learning rates for each parameter during training adaptively. The algorithm helps overcome some limitations of other optimization techniques, such as being sensitive to the choice of learning rates. The key features of Adam optimizer include adaptive learning rates, which can be beneficial when dealing with sparse data and noisy gradients. It also incorporates momentum-like behaviours to help accelerate the optimization process. Initial learning rate is set to 3e-4. Maximum number of epochs is set to 15. Mini-batch size is set to 64. Validation is performed every 5 epochs. The train Network function in MATLAB is used to train the neural network using the specified architecture, training data, and options. By plotting various metrics during training, we can learn how the training is progressing. For example, we can determine if and how quickly the network accuracy is improving, and whether the network is starting to overfit the training data.

IV. Simulation Results

We launch an extensive computer simulation to assess the performance of the proposed estimator. Both training and validation loss metrics are crucial for evaluating the model's effectiveness. The training loss guides parameter updates, while validation loss provides an independent assessment of the model's generalization to unseen data. Monitoring both ensures the model is well-performing and generalizable. Fig. 3 illustrates the training loss and validation loss for various modulation schemes and dataset sizes specified in Table II.

TABLE II: THE TRAINING PROGRESS TABLE

Results	2-PSK	4-PSK	8-PSK	16-PSK
Number of train signals	500	1000	10000	20000
Validation accuracy	100%	99.29	96.8%	94.6%
Elapsed time	2.0 s	4.0 s	11.0 s	34.0 s
Epoch	15	15	15	15
Iterations	30/30	15/15	540	1220
Iteration per epoch	2	17	125	250
Maximum iterations	30	569	1875	3750
Validation frequency	5	5	5	5
Constant learning rate =0.0003				

A decreasing training loss indicates effective learning from the provided data. The stability or reduction of the validation loss indicates the model's expected performance on real-world scenarios beyond the training set. The experimentation involved assessing the performance of various Phase Shift Keying (PSK) modulation schemes, namely 2-PSK, 4-PSK, 8-PSK, and 16-PSK, using a machine learning model. Each modulation scheme was trained with a different number of signals, ranging from 500 for 2-PSK to 20,000 for 16-PSK. The elapsed time for training increased as the complexity of the modulation scheme grew, with 2-PSK completing in 2.0 s and 16-PSK taking 34.0 s. The training process was conducted over 15 epochs for all schemes, with validation occurring every 5 epochs. However, the number of iterations varied significantly between the schemes, with 2-PSK and 4-PSK completing 30 iterations each, while 8-PSK required 540 iterations and 16-PSK necessitated 1220 iterations to reach convergence. This discrepancy in iteration counts translated to varying iteration per epoch values, with 2-PSK requiring two iterations per epoch, 4-PSK 17 iterations per epoch, 8-PSK125 iterations per epoch, and 16-PSK 250 iterations per epoch. Additionally, a constant learning rate of 0.0003 was maintained throughout the training process for all modulation schemes. These results offer insights into the performance and computational requirements of different PSK modulation schemes, crucial for optimizing communication system designs and machine learning model training strategies. The accuracy of a neural network shown in Fig. 4 is commonly computed as the ratio between the number of correct predictions and the total number of predictions. The validation accuracy varied across the schemes, with 2-PSK achieving a perfect accuracy of 100%, followed by 99.29% for 4-PSK, 96.8% for 8-PSK, and 94.6% for 16-PSK. The experiment may be repeated considering FSK, DPSK, QAM and a new set of figures will be generated. The trend is the same with some minor findings that we may show later.

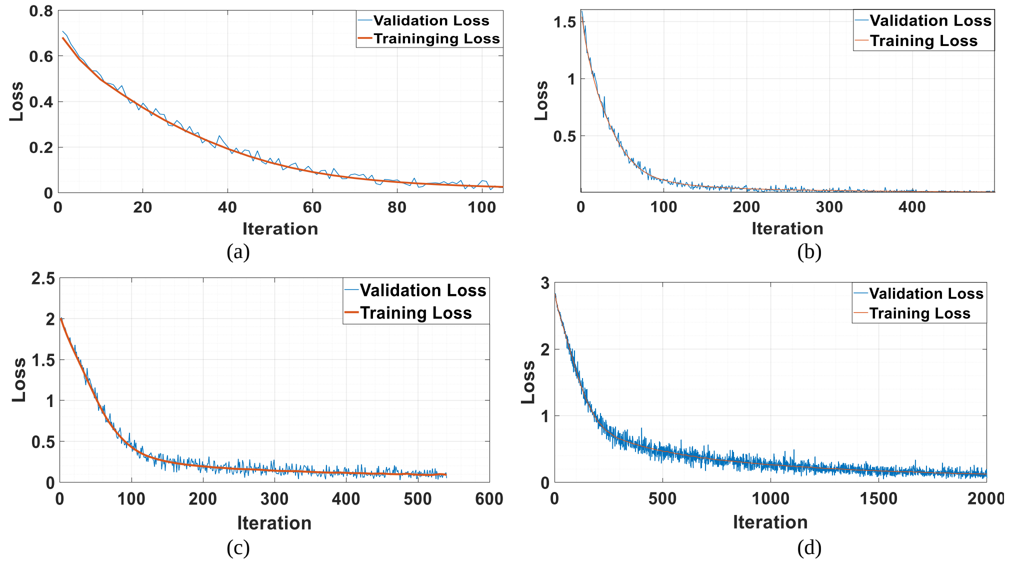

Fig. 3. Validation and training loss for (a) 2-PSK, (b) 4-PSK, (c) 8-PSK, and (d) 16-PSK.

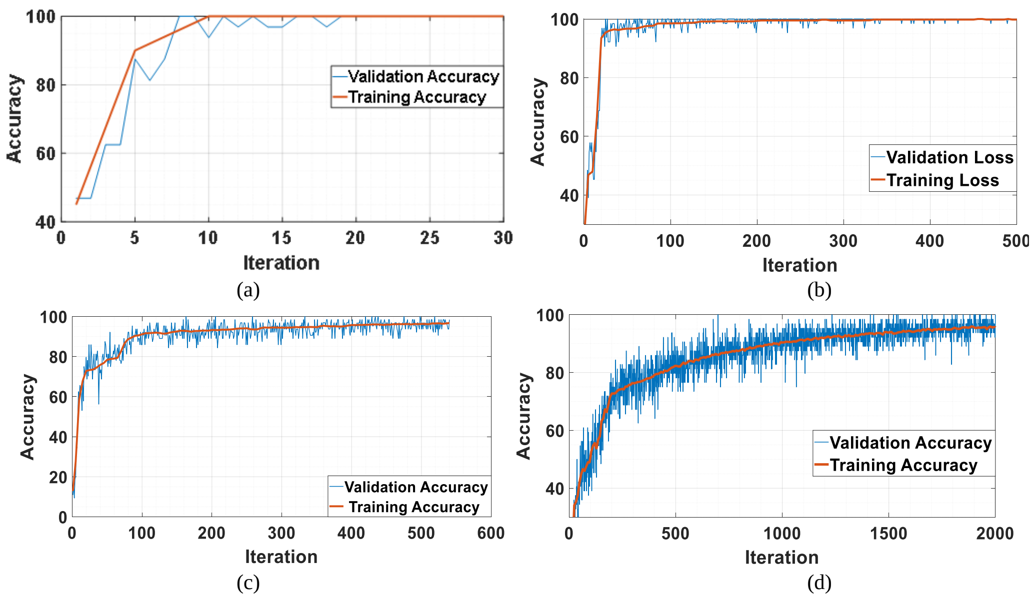

Fig. 4. Validation and training accuracy for (a) 2-PSK, (b) 4 PSK, (c) 8-PSK, and (d) 16-PSK.

Fig. 5 is a confusion matrix that is often used to evaluate the performance of a classification algorithm on a set of data for which the true values are known. The probability matrix is closely related to the confusion matrix, typically refers to the matrix of predicted probabilities for each class given by a classification model. Each row in the probability matrix corresponds to an actual class, and each column corresponds to the predicted probability for a specific class. This confusion matrix helps evaluate the performance of a classification model across four different classes, providing insights into the model's strengths and weaknesses in predicting each class. Accuracy is inversely related to the Symbol Error Rate (SER), as higher accuracy implies a lower likelihood of incorrectly detected symbols. The model will be revisited, as the primary goal of this article was to evaluate the potential of using machine learning algorithms. A new DNN will be built to recognize the hop frequency. Security threats and anti-Jamming techniques will be investigated. Also, a detailed comparative study to investigate the performance and the complexity of the traditional methods and the DNN/ ML methods will be introduced soon.

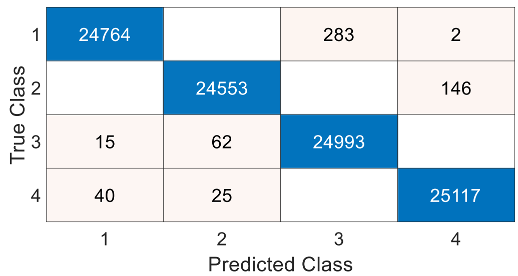

Fig. 5. Confusion matrix for a three multi path and zero dB SNR.

V. Conclusion

This work develops an intelligent receiver for SFH system using deep learning, The neural network will replace the conventional receiver, it is designed to take 80 features (received signal samples) as input and predict one of 2, 4, 8, or 16 classes, representing the transmitted symbol depending on the modulation technique. The training process of the feedforward neural network aims to optimize the network's weights to minimize the classification error on the provided training and validation datasets. The generated training loss and validation loss curves confirmed the ability of the receiver to learn. In our next research article, the performance, and the complexity of the DNN estimator will be compared with the conventional ones.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Mahmoud Qasaymeh is the primary author of this paper, and he has written the entire paper, compiled all necessary information, and presented it in an understandable manner. Ahmad Aljaafreh has developed the machine learning code. Ali Alqatawneh has provided the introduction, literature, and paper revision.

References

[1] H. Pirayesh and H. Zeng, "Jamming attacks and anti-jamming strategies in wireless networks: A comprehensive survey," *IEEE Commun. Surv. Tutor*., vol. 24, no. 2, pp. 767–809, 2022.

[2] S. Sciancalepore and R. D. Pietro, "Bittransfer: Mitigating reactive jamming in electronic warfare scenarios," *IEEE Access*, vol. 7, pp. 156175–156190, 2019.

[3] X. Wang, J Wang, Y Xu *et al.*, "Dynamic spectrum anti-jamming communications: Challenges and opportunities," *IEEE Commun. Mag*., vol. 58, no. 2, pp. 79–85, 2022

[4] M. A. Munir and A. R. M. Maud, "Direct-sequence spread spectrum with variable spreading sequence for jamming immunity," in *Proc. 2019 16th Int. Bhurban Conf. on Applied Sciences and Technology*, Islamabad, Pakistan, 2019, pp. 933–937.

[5] M. Letafati, A. Kuhestani, H. Behroozi, and D. W. K. Ng, "Jamming-resilient frequency hopping-aided secure communication for internet-of-things in the presence of an untrusted relay," *IEEE Trans. Wirel. Commun*., vol. 19, no. 10, pp. 6771–6785, 2020.

[6] D. Torrieri, S. Talarico, and M. C. Valenti, "Analysis of a frequency-hopping millimeter-wave cellular uplink," *IEEE Trans. Wirel. Commun*., vol. 15, no. 10, pp. 7089–7098, 2016.

[7] A. Yegin, T. Kramp, P. Dufour *et al.*, "LoRaWAN protocol: Specifications, security, and capabilities," *LPWAN Technologies for IoT and M2M Applications*, Elsevier, 2020, pp. 37–63.

[8] G. Boquet, P. Tuset-Peiro, F. Adelantado, T. Watteyne, and X. Vilajosana, "LR-FHSS: Overview and performance analysis," *IEEE Commun. Mag*., vol. 59, no. 3, pp. 30–36, 2021

[9] K. Wu, J. A. Zhang, X. Huang, and Y. J. Guo, "Frequency-hopping MIMO radar-based communications: An overview," *IEEE Aerosp. Electron. Syst. Mag*., vol. 37, no. 4, pp. 42–54, 2022.

[10] T.-C. Chen, "Joint signal parameter estimation of frequency-hopping communications," *IET Commun*., vol. 6, no. 4, p. 381, 2012.

[11] A. Kanaa and A. Z. Sha'ameri, "A robust parameter estimation of FHSS signals using time–frequency analysis in a non-cooperative environment," *Phys. Commun*., vol. 26, pp. 9-20, 2018. doi: 10.1016/J.PHYCOM.2017.10.013

[12] L. Zhao, L. Wang, G. Bi, L. Zhang, and H. Zhang, "Robust frequency-hopping spectrum estimation based on sparse Bayesian method," *IEEE Trans. Wirel. Commun*., vol. 14, no. 2, pp. 781–793, 2015.

[13] J. Wan, D. Zhang, W. Xu, and Q. Guo, "Parameter estimation of multi frequency hopping signals based on space-time-frequency distribution," *Symmetry (Basel)*, vol. 11, no. 5, 648, 2019.

[14] Y. Li, F. Wang, G. Fan, Y. Liu, and Y. Zhang, "A fast estimation algorithm for parameters of multiple frequency-hopping signals based on compressed spectrum sensing and maximum likelihood," *Electronics (Basel)*, vol. 12, no. 8, 1808, 2023.

[15] F. Liu, M. W. Marcellin, N. A. Goodman, and A. Bilgin, "Compressive sampling for detection of frequency-hopping spread spectrum signals," *IEEE Trans. Signal Process*., vol. 64, no. 21, pp. 5513–5524, 2016.

[16] T. Anjali, T. S. Aparna, M. Meera, A. Parvathy, and G. Narayanan, "Implementation of energy detection technique for spread spectrum systems," *Intelligent Manufacturing and Energy Sustainability*, Singapore: Springer Singapore, 2021, pp. 443–454.

[17] K.-G. Lee and S.-J. Oh, "Detection of fast frequency-hopping signals using dirty template in the frequency domain," *IEEE Wirel. Commun. Lett*., vol. 8, no. 1, pp. 281–284, 2019.

[18] J. Ye, J. Zou, J. Gao *et al.*, "A new frequency hopping signal detection of civil UAV based on improved K-means clustering algorithm," *IEEE Access*, vol. 9, pp. 53190–53204, 2021.

[19] D. Mototolea, R. Youssef, E. Radoi, and I. Nicolaescu, "Non-cooperative low-complexity detection approach for FHSS-GFSK drone control signals," *IEEE Open J. Commun. Soc*., vol. 1, pp. 401–412, Mar. 2020.

[20] S. Liu, T. Wang, and S. Wang, "Toward intelligent wireless communications: Deep learning—based physical layer technologies," *Digit. Commun. Netw*., vol. 7, no. 4, pp. 589–597, 2021.

[21] Z. Chen, Y. Shi, Y. Wang, X. Li, X. Yu, and Y. Shi, "Unlocking signal processing with image detection: A frequency hopping detection scheme for complex EMI environments using STFT and CenterNet," *IEEE Access*, vol. 11, pp. 46004–46014, 2023.

[22] K.-G. Lee and S.-J. Oh, "Detection of frequency-hopping signals with deep learning," *IEEE Commun. Lett*., vol. 24, no. 5, pp. 1042–1046, 2020

[23] C. Li, Z. Zhao and Y. Chen, "Detection algorithm of frequency hopping signals based on S Transform and Deep Learning," in *Proc. 2022 16th IEEE International Conference on Signal Processing (ICSP)*, Beijing, China, 2022, pp. 310–313.

[24] Z. Yuan, Z. Zhao, Y. Zhang, S. Zheng, and S. Dai, "Intelligent reception of frequency hopping signals based on CVDP," *Appl. Sci. (Basel)*, vol. 13, no. 13, 7604, 2023.

[25] J. Zhu, A. Wang, W. Wu *et al.*, "Deep-learning-based recovery of frequency-hopping sequences for anti-jamming applications," *Electronics*, vol. 12, no. 3, 496, 2023. doi: 10.3390/electronics12030496

[26] T. Kuang, H. Chen, L. Han *et al.*, "Abnormal signal recognition with time-frequency spectrogram: A deep learning approach," *China Communications*, 2022. doi: 10.48550/arXiv.2205.15001

[27] Z. Feng and C. Hua, "Machine learning-based RF jamming detection in wireless networks," in *Proc. 2018 Third Int. Conf. on Security of Smart Cities, Industrial Control System and Communications*, 2018. doi: 10.1109/SSIC.2018.8556709

[28] G. S. Kasturi, A. Jain, and J. Singh, "Detection and Classification of Radio Frequency Jamming Attacks using Machine learning," *J. Wirel. Mob. Netw. Ubiquitous Comput. Dependable Appl*., vol. 11, no. 4, pp. 49–62, 2020.

[29] M. Qasaymeh and M. Khodeir, "Blind channel estimation for frequency hopping system using subspace-based method," *International Journal of Electronics and Communication Engineering*, vol. 9, no. 4, pp. 959–962, 2015.

[30] M. M. Qasaymeh, G. Hiren, T. Nizar, R. Pendse, and M. E. Sawan, "Time delay estimator for frequency hopping system using rank-revealing triangular factorization," in *Proc. IEEE 69th Vehicular Technology Conf.*, Barcelona, Spain, 2009. doi: 10.1109/VETECS.2009.5073886

[31] M. M. Qasaymeh, A. A. Alqatawneh, and A. F. Aljaafreh, "Channel estimation methods for frequency hopping system based on machine learning," *Journal of Communications*, vol. 19, no. 3, pp. 143–151, 2024.

[32] M. M. Qasaymeh and A. F. Aljaafreh, "Joint time delay and frequency estimation based on deep learning," *Journal of Communications*, vol. 19, no. 1, pp. 1–6, 2024.

Mahmoud Qasaymeh is an associate professor in Computer and Communication Engineering Department, Tafila Technical University Jordan, Jordan. He received his Ph.D. degree in electrical engineering, wireless communication from Wichita State University, KS, USA and the M.S. degree communications and electronics from Jordan University of Science and Technology, Jordan. His research interests are in the wireless communication system, array signal processing, radio communication systems, blind estimation techniques, frequency hopping systems, OFDM, MIMO and coding techniques.

Ali Alqatawneh is an assistant professor at Computer Engineering and Communication Department at Tafila Technical University, Jordan. He received a B.Sc. degree in communication engineering from Mu'tah University, Jordan, an M.Sc. degree in communication engineering from Mu'tah University, and a Ph.D. degree in electrical engineering with a specialization in wireless communication from the University of Arkansas, USA. His research experience covers various topics in wireless communications, such as multicarrier multiple access techniques, channel estimation techniques, high mobility wireless communications, cognitive radio networks, and orthogonal frequency division multiplexing with index modulation.

Ahmad Aljaafreh is a full professor at Computer Engineering and Communication Department at Tafila Technical University, Jordan. He received his PhD in computer engineering in 2010 from Western Michigan University, MI, USA. His research experience covers various topics in Artificial Intelligence applications. Currently, he is in sabbatical leave at College of Engineering and Science, University of Detroit Mercy.