11-3-2022

DeepDeMod: BPSK Demodulation Using Deep Learning Over Software-Defined Radio

Arhum Ahmad
Indian Institute of Technology

Satyam Agarwal
Indian Institute of Technology

Sam Darshi
Indian Institute of Technology

Sumit Chakravarty
Kennesaw State University, schakra2@kennesaw.edu

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 12 October 2022, accepted 29 October 2022, date of publication 3 November 2022, date of current version 8 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3219090

RESEARCH ARTICLE

DeepDeMod: BPSK Demodulation Using Deep Learning Over Software-Defined Radio

ARHUM AHMAD[1], (Graduate Student Member, IEEE), SATYAM AGARWAL[1], (Senior Member, IEEE), SAM DARSHI[1], (Senior Member, IEEE), AND SUMIT CHAKRAVARTY[2], (Member, IEEE)

[1]Department of Electrical Engineering, Indian Institute of Technology Ropar, Rupnagar 140001, India
[2]Department of Electrical and Computer Engineering, Kennesaw State University, Kennesaw, GA 30144, USA

Corresponding author: Arhum Ahmad (arhum.19eez0005@iitrpr.ac.in)

This work was supported in part by the Department of Science and Technology under Grant DST/INSPIRE/04/2016/001127 and Grant CRG/2020/005749.

ABSTRACT In wireless communication, signal demodulation under non-ideal conditions is one of the important research topic. In this paper, a novel non-coherent binary phase shift keying demodulator based on deep neural network, namely DeepDeMod, is proposed. The proposed scheme makes use of neural network to decode the symbols from the received sampled signal. The proposed scheme is developed to demodulate signal under fading channel with additive white Gaussian noise along with hardware imperfections, such as phase and frequency offset. The time varying nature of hardware imperfections and channel poses a additional challenge in signal demodulation. In order to address this issue, additionally we propose transfer learning based DeepDeMod scheme. Pilot symbols along with data is transmitted in a packet which is used to learn the time varying parameters from the pilot reception followed by data demodulation. Results show that compared with the conventional demodulators and other machine learning based demodulators, our proposed DeepDeMod provides significantly better performance in term of bit error rate. We also implement the proposed DeepDeMod on software defined radio and present the experimental results.

INDEX TERMS Binary phase shift keying (BPSK), deep neural network (DNN), demodulation, software-defined radio (SDR), transfer learning, universal software radio peripheral (USRP).

I. INTRODUCTION

Phase shift keying (PSK) is one of the digital modulation techniques where the carrier phase changes according to the message signal. Binary phase shift keying (BPSK) is the basic PSK scheme that uses two-phase states to transmit digital information. As PSK schemes are more resilient to noise and provide better bit error rate (BER) performance among other modulation schemes, it has found immense applications in satellite/deep-space communication, navigation, mobile communications, underwater communications [1], biomedical implant transceivers [2], [3], and so forth. Since BPSK is the primary modulation scheme, employing it for the proposed work lays the framework for expanding it to other digital modulation schemes such as ASK, FSK, QPSK, QAM, and MSK, among others.

The conventional BPSK receiver uses the coherent detection technique, where the knowledge of the carrier frequency and phase must be known to the receiver. Hence, the demodulation performance of the BPSK receiver depends on various modules, such as filters, phase-locked loop (PLL), product modulator, analog-to-digital converter (ADC), oscillators, phase/frequency detectors, etc., which introduces significant delay in detection, and its implementation is complex [3], [4]. With factors like vibrations, acceleration, temperature fluctuations, aging, instability in discrete components, etc., the performance of these modules varies, which causes degradation in the receiver's overall performance [5], [6].

In this paper, we propose a deep neural network (DNN) based BPSK demodulator, termed as *DeepDeMod*, which

The associate editor coordinating the review of this manuscript and approving it for publication was Abdel-Hamid Soliman.

learns a mapping between bit transmitted and received BPSK modulated signals and detects bit transmitted. A DNN is a neural network with multiple (more than two) layers between the input and output layers. It uses mathematical modeling such as statistical (linear/non-linear regression, discriminant analysis, etc.) and predictive modeling (classification model, clustering model, random forest, etc.) to process data and develop a relation between input and output.

The proposed work selects the type of neural network from many options, such as DNN, convolutional neural network (CNN), recurrent neural network (RNN), k-nearest neighbors (KNN), and deep belief network (DBN), among others. We select the DNN based machine learning framework due to its ability to satisfactorily classify and predict the data using supervised learning. On the other hand, CNN is apt for image-based problems, RNN is apt for the time-series-based problem, and so on. Also, the complexity of DNN over other neural networks is low [7], [8]. Further, based on prior works [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], the performance of other machine learning (ML) based schemes do not guarantee significant accuracy improvement. While employing DNN, we start with a basic 1-hidden layer model, then keep increasing to get the desired performance. We observed that for our problem, five-layer DNN provides the best accuracy versus complexity in terms of training and testing accuracy.

Compared to the conventional BPSK demodulator, the DeepDeMod is more resilient to parameter fluctuations, and its demodulation function is achieved through learning a big dataset. Hence, it gives more flexibility and self-adaptability. Once DNN is trained, it is able to produce a fast and reliable output even when the received signal is noisy i.e., it has better anti-noise capability. This is because the DNN learns the features in the received signal under various imperfections during the training process.

Since the real-time execution environment constantly changes, the results begin to deteriorate if we do not update the trained neural network according to the environment. To address this problem, we propose an adaptive demodulation algorithm (DeepDeMod-TL) based on transfer learning. DeepDeMod-TL tracks the variations in the received signal and updates the weights in the DNN model to maintain the optimal performance of DeepDeMod.

A. RELATED WORKS

In recent years, many researchers have focused on applying neural networks (NN) in wireless communications, such as signal modulation recognition, optimization, demodulation, etc. [21], [22]. For instance, [23], [24], [25] proposed a deep learning-based model for modulation recognition. The authors in [26], [27], and [28] proposed a machine learning method to optimize performance of wireless communication system. Nakayama and Imai [9] proposed an amplitude shift keying demodulator based on a neural network to combine the wideband noise rejection, pulse waveform shaping, and decoding into a single neural network. Multi-layer perceptron (MLP) based demodulator was proposed in [10] and [11]. Moreover, He et al. [10] used multiple MLPs to construct a demodulator named MaxMLP classifier, which automatically detected different modulated signals (BPSK, QPSK, and GMSK) without using complex signal processing algorithms. Önder et al. [11] proposed a NN-based receiver to decode multiple phase-shift keying (MPSK) signal using a three-layer MLP structure. Fan and Wu [12] proposed a deep belief network based demodulator for BPSK and QPSK. They evaluated the demodulator's behavior under a band-limited Rayleigh fading channel. Furthermore, Wang et al. [13] proposed a flexible end-to-end wireless communications prototype using a deep belief network - support vector machine demodulator and adaptive boosting based demodulator, which could detect eight different modulation schemes, namely, BPSK, 4-QAM, 8-QAM, 16-QAM, 32-QAM, 64-QAM, 128-QAM, and 256-QAM.

Convolutional neural network (CNN) based demodulator was proposed in [14], [15], and [16]. Mohammad et.al. [14] demodulated binary frequency shift keying via implementing a deep convolutional neural network. Zhang et al. [15] proposed a 1-D CNN-based non-coherent BPSK demodulator, which detected the signal's phase shift and extracted bit information from the phase message. Their simulation results showed accurate detection of phase shifts in a BPSK modulated signal. As an extension, Liu et al. [16] showed the hardware implementation of 1D-CNN demodulator for BPSK using FPGA. Leonard et al. [17] demonstrated an approach to replace a radio receiver's physical layer functions with NN. They detected BPSK, QPSK, and 8-PSK modulated signals under various channel conditions along with frequency and timing correction. Chen et al. [18] proposed a CNN-based architecture for demodulating the modulated signals by integrating CNN into the communication system. Zheng et.al. [19] proposed a new receiver model called DeepReceiver. They used a 1-D convolution network architecture with multiple binary classifiers to obtain data during detection. Zhang et.al. [20] proposed an intelligent deep neural network (DNN)-based demodulator with an LSTM unit to detect the received signals.

This paper presents a unified signal demodulation framework using machine learning (ML). Specifically, we propose a novel preprocessing followed by a DNN model to detect bits in the BPSK signal (termed DeepDeMod). We also propose a novel way to update DNN weights on the fly (termed DeepDeMod-TL). We implement the DeepDeMod and DeepDeMod-TL in the radio system using software-defined radio (SDR). This work is a novel contribution to this field, and it differs from the existing works in the following aspects:

- A band pass filter (BPF) is used in the current proposal to select the cosine component of the message signal in order to generate the rich features needed for the proposed DNN to learn/detect from the received signal. On the other hand, the performance of existing works

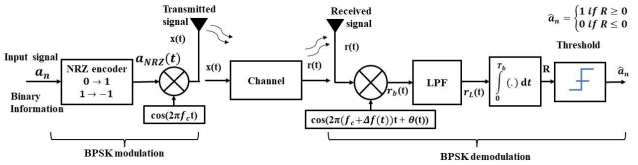


FIGURE 1. Conventional BPSK modulation and demodulation.

is limited because they use the matching filter's output which is based on a DC message bits.

- Training data in our model considers all possible signal variations, and its performance is resilient to the training data set. In contrast, the other works proposed in the literature have their performance depending on the trained data set. For example, [15] uses training with a signal strength of -2 dB for optimal results.
- The proposed DeepDeMod method for signal demodulation adapts itself to the changes in the signal parameters and therefore provides high detection accuracy over time. The existing works have not discussed such time-dependent variations in frequency, phase, and channel parameters over time.
- The proposed method can be easily implemented on the existing conventional system, and the performance of DeepDeMod and DeepDeMod-TL show significant improvement in BER.

From the results, we show that our proposed DeepDeMod and DeepDeMod-TL achieve BER of the order of 10^{-4} at a signal strength of -14 dB and -18 dB, respectively. On the other hand, conventional coherent demodulation achieves this BER at 8 dB of signal strength.

The remaining part of this paper is organized as follows. Conventional BPSK detection and its problems are described in Section II. Section III and IV present the proposed DeepDeMod and DeepDeMod-TL, respectively. Simulation results are provided in Section V. Hardware implementation and results are shown in Section VI. The paper is concluded in Section VII.

II. CONVENTIONAL BPSK DETECTION

In BPSK, binary bits are transmitted by changing the phase of the carrier signal. Binary bit '0' is represented with a raw carrier signal, while bit '1' is represented by carrier phase shifted by π radians. Therefore, within a bit duration T_b seconds, the BPSK modulated signal with two different phase states of the carrier signal are represented as,

$$x(t) = \begin{cases} \cos(2\pi f_c t) & 0 \le t \le T_b \quad \text{for} \quad a_n = \text{'0'} \\ \cos(2\pi f_c t + \pi) & 0 \le t \le T_b \quad \text{for} \quad a_n = \text{'1'} \end{cases} \tag{1}$$

where f_c is the carrier frequency (Hz), t is the instantaneous time in seconds, T_b is the bit period in seconds, a_n is binary message signal such that $a_n \in \{0, 1\}$ for the n^{th} bit in the message signal.

Fig. 1 represents a conventional BPSK modulator and demodulator. As seen from Fig. 1, the BPSK modulator operates on binary message signal a_n and produces a signal $x(t)$ as given in (1). The above equation (1) can also be represented in terms of non-return-to-zero (NRZ) signal ($a_{NRZ}(t)$) as $a_{NRZ}(t)cos(2\pi f_c t)$, where $a_{NRZ}(t) \in \{1, -1\}$ with respect to a_n. This signal is transmitted through a wireless fading channel with additive white Gaussian noise (AWGN).

At the receiver, the received signal is multiplied by a reference carrier signal and passed through a low pass filter (LPF). This output is integrated over a bit period (T_b) using an integrator. A threshold detector decides on the transmitted bit by comparing the integrated signal against a threshold value. This threshold value is 0 in case of bits being equiprobable.

The perfect knowledge of frequency and phase at the receiver is required for coherent demodulation. However, in a practical setting, it is impossible to achieve perfect synchronization of carrier frequency and phase [29], [30]. An analysis of the effect of phase and frequency offset is shown below. Let the transmitted signal $x(t)$ is given as

$$x(t) = a_{NRZ}(t)\cos(2\pi f_c t) \tag{2}$$

This signal is received as $r(t)$ and is given as

$$r(t) = h(t)a_{NRZ}(t)\cos(2\pi f_c t) + n(t) \tag{3}$$

where $h(t)$ is the channel fading coefficient and $n(t)$ is AWGN. For demodulation, $r(t)$ is first passed through a product modulator, where carrier has time varying frequency ($\Delta f(t)$) and phase ($\theta(t)$) offset. The output of the product modulator is given as

$$\begin{aligned} r_b(t) = {} & h(t)a_{NRZ}(t)\cos(2\pi f_c t) \\ & \times \cos(2\pi(f_c + \Delta f(t))t + \theta(t)) + n'(t) \end{aligned} \tag{4}$$

where $n'(t)$ is processed AWGN by product modulator. Next the signal passes through a LPF. The output of LPF is given as

$$r_L(t) = \frac{a_{NRZ}(t)}{2}h'(t)cos(2\pi\,\Delta f(t)t + \theta(t)) + n''(t) \tag{5}$$

where $h'(t)$ and $n''(t)$ are processed fading coefficient and AWGN by LPF, respectively.

From (5), the LPF output contains time-varying frequency and phase offset component instead of only the message signal. These offsets cause bit misdetection in case of non-coherent detection of BPSK signal. Generally, a phase lock loop (PLL) is used for carrier synchronization, which is not very robust to the variation in carrier frequency, and phase [31]. As PLL uses a voltage-controlled oscillator (VCO), variations in temperature, supply voltage, and manufacturing processes affects the stability and operating performance. In the following, we propose a DNN-based demodulation scheme that does not require carrier frequency and phase synchronization at the receiver yet performs better than the coherent detector.

III. PROPOSED DeepDeMod

Conventional BPSK coherent demodulators require carrier synchronization, leading to demodulation error when phase and frequency offsets exist at the receiver. In this section,

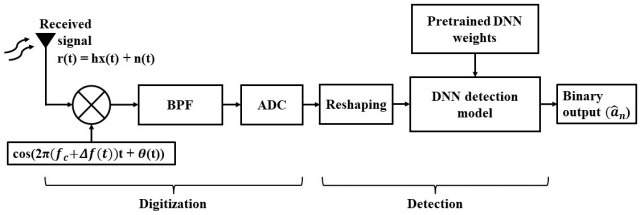


FIGURE 2. Receiver structure of the proposed DeepDeMod.

we present a novel DNN based demodulator (DeepDeMod) which directly loads the received modulated sampled signal into the neural network without carrier synchronization. The neural network accomplishes bit detection by matching the features in the input signal.

It is convenient to subdivide the receiver into two parts, signal digitization and detection. The function of the signal digitizer is to convert the received analog waveform $r(t)$ into a discrete signal vector. The function of the detector is to decide which of the possible signal waveform was transmitted based on the observed signal vector. A block diagram of the proposed DeepDeMod demodulator is shown in Fig. 2. In Fig. 2, $\Delta f(t)$ and $\theta(t)$ are frequency and phase offset, respectively in the local oscillator as observed in a realistic receiver.

A. SIGNAL DIGITIZATION

The signal digitization part takes received signal $(r(t))$ as input and passes it through the product modulator and then to the band pass filter (BPF). Instead of using a LPF as in the conventional demodulation, here we use BPF with center frequency $2f_c$ and bandwidth (BW) equal to the BW of BPSK signal i.e., $2/T_b$. The output of BPF is in the form of '$a_{NRZ}(t) \times cos(4\pi f_c t)$'. On the other hand, the output of a LPF in the conventional demodulator is constant i.e., $a_{NRZ}(t)$. Output of both the filters also include channel and noise effects. Since, the output of LPF is a constant message signal with noise and other effects, the DNN is unable to learn and map message signal at low signal strength (i.e., SNR), as noise is dominant over constant signal. However, the output of BPF has both message and carrier signal component, which helps the DNN to distinguish the message signal in the form of carrier phase from noise at low signal strength. It also helps to learn the distribution of other effects while training. Hence, BPF component provides features to the proposed DNN to learn and predict with high accuracy. The sample rate must be taken into consideration while sampling signals. Here a sampler or ADC is used with sampling rate K/T_b to convert continuous time signal into a discrete form for detection by the DNN. This sampling rate is selected to get exactly K samples over one bit period.

B. SIGNAL DETECTION

DNN is used as a detector, which maps K samples per bit of received data into one bit. The reshaping block is used to shape the sampled signal to suitably map to the input of the DNN. Prior to detection, the DNN is trained with the known signals to get the optimal weights for the DNN. This pre-trained DNN is used to detect the bits transmitted in the received signal. Further, as discussed later in section IV, this pre-trained DNN is fine tuned using transfer learning to improve the detection performance.

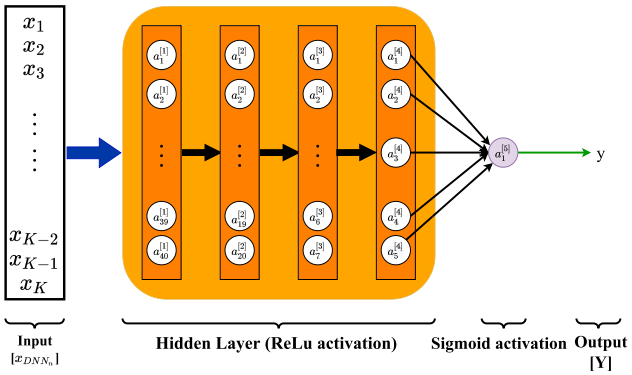


FIGURE 3. Architecture of the proposed DNN detector.

1) ARCHITECTURE OF THE PROPOSED DNN DETECTOR

The proposed DNN detector is composed of an input layer, four hidden layers and an output layer. The DNN detector takes sampled signal input for a bit duration and detects the bit transmitted. The total number of samples in one bit period is K. This implies that the proposed DNN has K inputs. Major part of learning in DNN is carried out by the hidden layers which is sandwiched between the input and the output layer. Four hidden layers are used in this proposed DNN in such a way that the first hidden layer contains 40 neurons, second contains 20 neurons, third contains 7 neurons and fourth contains 5 neurons. The function of the hidden layer is to assign weights to the inputs and direct them via an activation function as the output. In summary, the hidden layers conduct non-linear transformations on the network's inputs. Hidden layers are layers of mathematical functions, each of which is designed to produce an output particular to the desired outcome. Hidden layers allow a neural network's function to be broken down into particular data transformations. Each hidden layer function is tuned to generate a specific output. The output layer contains a single neuron. The proposed DNN detector is shown in Fig. 3. The process of moving data through the neural network is known as forward propagation, and this is described as follow.

1) Multiply the input value x_i by the corresponding weights w_i for each input, then add up all the multiplied values. Weights represent the strength of the connections between neurons and determine how much input will affect a neuron's output. Hence, the summation is equal to the dot product of the input vector and corresponding weights for the first neuron of the first hidden layer.

$$u = \sum_{i=1}^{K} x_i . w_i \tag{6}$$

2) Bias b is added to the sum of multiplied values. Bias, also known as offset, moves the entire activation function to the left or right to create the required output values.

$$z = \sum_{i=1}^{K} x_i.w_i + b \tag{7}$$

3) Pass the value of z to a non-linear activation function. The input to the next layer neurons is given as

$$a = f(\sum_{i=1}^{K} x_i.w_i + b) \tag{8}$$

where, $f(.)$ is the activation function.

In general, the DNN operation can be expressed as

$$a_j^l = f(\sum_k a_k^{l-1} w_{jk}^l + b_j^l) \tag{9}$$

where, w_{jk}^l denotes the weight for the connection from the k^{th} neuron in the $(l-1)^{th}$ layer to the j^{th} neuron in the l^{th} layer.

There are many possible activation functions for use in a DNN. From multiple simulations, our observation is that the Rectified Linear Unit (ReLU) activation function at all hidden layers give the best result. ReLu activation function is simple to implement and it is less susceptible to vanishing gradients that prevent deep models from being trained. Backpropagation, also referred to as backward error propagation, is the method for computing the gradient of the loss function with respect to the weights. As the output is binary, the binary crossentropy function is used as loss function in the DNN detector as shown in (10).

$$Loss = -\frac{1}{\mathrm{N_b}} \sum_{i=1}^{\mathrm{N_b}} y_i \cdot \log \hat{y}_i + (1 - y_i) \cdot \log (1 - \hat{y}_i) \tag{10}$$

where $\hat{y}_i$ is the i^{th} scalar value in the model output, y_i is the corresponding label value, and N_b is the number of scalar values in the model output (i.e., total number of bits considered).

The binary crossentropy is used for binary classification task. Based on the binary output, the output layer uses the sigmoid activation function. Adam optimizer is used to minimize the loss function and update the weights. The DNN is trained for 100 epochs, after which the training loss saturates. We implement the proposed DNN model over Tensorflow platform [32]. It is an open-source software to implement machine learning algorithm utilizing various tools, libraries, and community resources.

2) TRAINING DATA

To have an efficient DNN detector, we first need to train it with an appropriate training data set. Supervised learning is employed where the input and output data sets are generated via computer simulations and fed to the DNN. Random message bits are generated, which is modulated and passed through the wireless fading channel. At the receiver, the signal is multiplied by the reference carrier signal, passed through a BPF, and then sampled by the ADC with a sampling rate K/T_b. These samples with known bit boundaries are used for training the proposed DNN, where we have assumed that boundaries of bits are known.

The samples can be expressed as $r_s(n)$ $(n = 1, 2, 3, \cdots, N)$, where N is the total number of samples, which is an integer multiple of K, such as $N = KN_b$, where N_b is the total number of bits considered. The bits transmitted are arranged in an array ($Label_{DNN}$) for providing labels to the training data ($Train_{DNN}$) as specified below. The input sampled signal to the DNN for the m^{th} bit is given as:

$$\mathbf{r_{DNN_m}} = \begin{bmatrix} r_s((m-1)K+1) \\ r_s((m-1)K+2) \\ r_s((m-1)K+3) \\ \vdots \\ r_s(mK) \end{bmatrix}_{K\times 1} \tag{11}$$

where subscript DNN_m represents m^{th} bit samples.

For the training data-set of DNN, the received signal experiences multiple effects like noise, fading, hardware imperfections (phase and frequency offset at the local oscillator), etc. We consider a large number of bits (2.5×10^6 bits) experiencing those effects or combination of those effects under various channel condition and noise (i.e., SNR). Based on our model as shown in Fig. 2, output of ADC samples are in the form of a row vector. From known bit boundaries each set of oversampled received data is adjusted as shown in (11) and combined together to form a 2-D array as input of the DNN named as $Train_{DNN}$ shown in (12). Label set for training is named as $Label_{DNN}$ shown in (13). It contains bits 1's and 0's with respect to $Train_{DNN}$. This $Train_{DNN}$ and $Label_{DNN}$ is used as *primary* data set while applying transfer learning to the DNN. The 2-D train array and its corresponding label are given as:

$$Train_{DNN} = \begin{bmatrix} r_s(1) & r_s(K+1) & \cdots & r_s(N-K+1) \\ r_s(2) & r_s(K+2) & \cdots & r_s(N-K+2) \\ \vdots & \vdots & \ddots & \vdots \\ r_s(K) & r_s(2K) & \cdots & r_s(N) \end{bmatrix}_{K\times N_b} \tag{12}$$

$$Label_{DNN} = \begin{bmatrix} a_1 & a_2 & \cdots & a_{N_b} \end{bmatrix}_{1\times N_b} \tag{13}$$

To generate the dataset for simulation, first we generate the samples as shown in (11) without any noise. Then, we add noise corresponding to various SNRs to the signal and generate the samples. Finally we add all the impairments under various SNR, i.e., phase, and frequency offset and channel fading. Thus, we concatenate all the data generated and create a large data set to train the proposed DNN.

Generation of the training dataset for the hardware implementation is a challenge. For this, first, we collect data via simulation for a wide range of SNRs and then we collect the realistic data set using SDR under different environments. Finally, we merge all the data and create a composite data set to train the proposed DNN. Based on such data-set the DNN

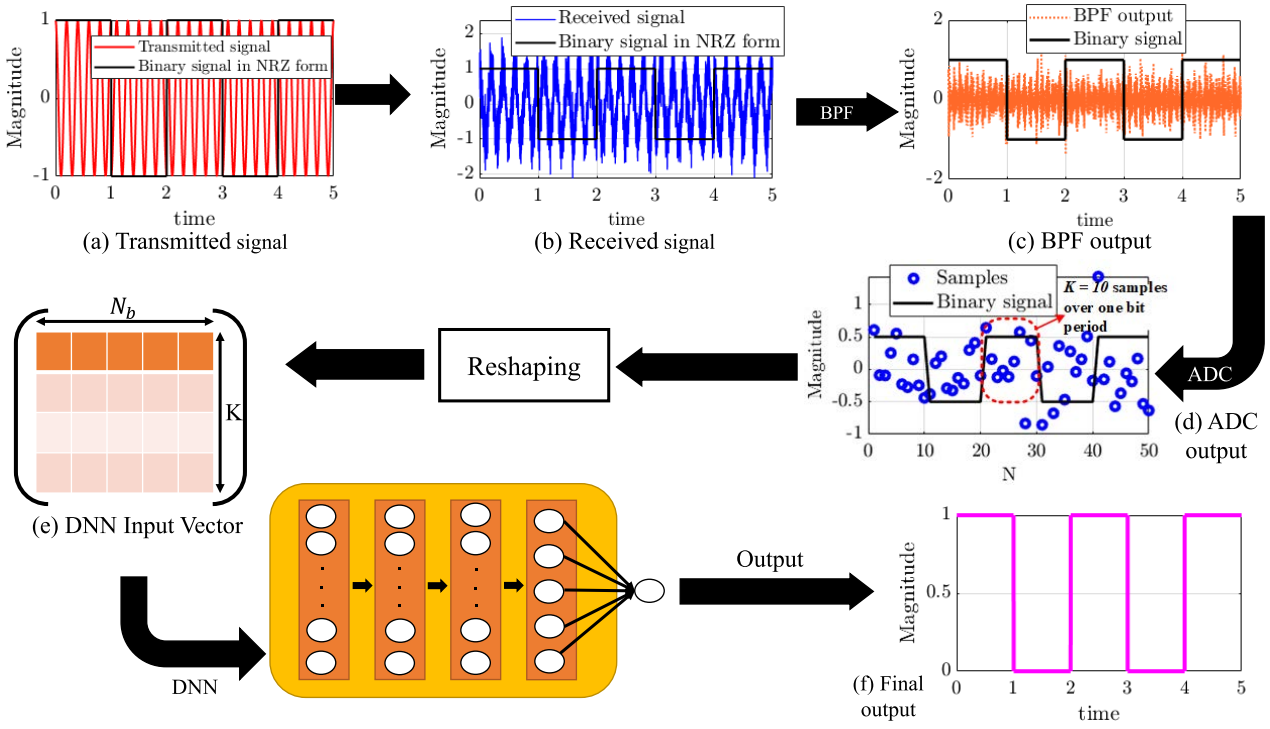


FIGURE 4. Output of different stages in DeepDeMod.

TABLE 1. DNN model and training parameters.

Parameter name	Parameter Value
Input size	K (= 10)
Number of hidden layer	4
Number of neurons in each hidden layer	40, 20, 7, 5
Activation function in hidden layer	Rectified Linear Unit (ReLu)
Learning rate (α)	0.01
Loss function	Binary cross-entropy
Optimizer	Adam
Output size	1
Activation function in output layer	Sigmoid
Batch size	250
Epochs	100

is able to learn the correct mapping between the input and the desired output.

The DNN model and it's training parameters are listed in Table 1. The DNN model is trained using the data set as specified above. Steps followed for model training is describe in Algorithm. 1. Once the training is complete, the trained weights and bias are obtained such that they are able to predict binary bits under various circumstances.

3) MODEL WORKING

To understand the complete working of the proposed DeepDeMod detector shown in Fig. 2, we present a pictorial representation of the output of every block in Fig. 4.

Algorithm 1 Training Process of Proposed DNN

Require: $\mathbf{r}_{\mathbf{DNN_m}}$ for m^{th} bit input samples. [$\mathbf{r}_{\mathbf{DNN_1}}$, $\mathbf{r}_{\mathbf{DNN_2}}$, $\mathbf{r}_{\mathbf{DNN_3}}$, . . .,$\mathbf{r}_{\mathbf{DNN_{N_b}}}$] represent the set of received signal samples for training as shown in (12). The respective labels for backpropagation are shown in (13).

Ensure: DNN trained network

```
Initialize the proposed DNN with random weights and bias.
Select the hyper-parameters as shown in Table. 1.
Load the Training and Label data set.
while Number of epochs are completed do
    for each training vector r_DNN_m do
        Forward propagation, using Equation (9).
        Compute loss, using Equation (10).
        Backward propagation.
        Update parameters using optimizer (Adam).
    end for
end while
```

Let a BPSK signal $x(t)$ (as shown in Fig. 4 (a)) is transmitted through a noisy fading channel. The signal received at the receiver is shown in Fig. 4 (b). The received signal is passed through a product multiplier and a BPF as shown in Fig. 4 (c). This signal is still analog in nature. For processing by the DNN model, we sample this signal by passing it through an ADC with sampling rate K/T_b as shown in Fig. 4 (d). This sampled data is reshaped into the DNN input vector as shown in Fig. 4 (e), which is sent to the DNN for detection. The DNN detector outputs the binary bit as shown in Fig. 4 (f).

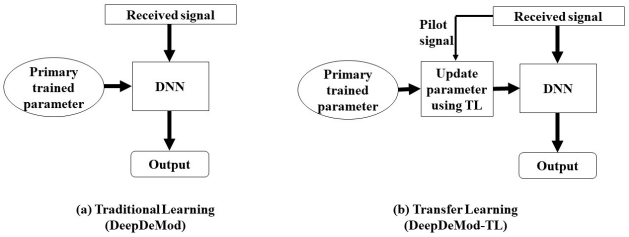


FIGURE 5. DeepDeMod with and without transfer learning.

In this work, we transmit signal in the form of packets where basic packet information is known to the receiver. Also we have assume that boundaries of bits in the packet is known to the receiver.

IV. ONLINE DeepDeMod USING TRANSFER LEARNING

In practical scenario, the transmitter, receiver and channel parameters keep changing. This gives rise to a need to re-train our DNN model with updated parameters for better performance. If the DNN model is not updated, the results will degrade. In order to overcome such a situation, we propose transfer learning (TL) based DeepDeMod (DeepDeMod-TL) in this section.

A. TRANSFER LEARNING

As an essential branch of machine learning, transfer learning aims to use the similarity between data, tasks, or models to transfer the knowledge learned from the original data set (model) to the new data set (model). Simply, it defines as an approach in deep learning where knowledge is transferred from one model to another. Fig. 5 presents our proposed DeepDeMod with and without transfer learning method. In normal DeepDeMod mode, the learning happens only once. These trained parameters are used to detect the received signals without considering any changes in the received signal over time. On the other hand, transfer learning enabled DeepDeMod (DeepDeMod-TL) considers the changes in the received signal and updates the DNN weights accordingly so that the detection performance is better.

Transfer learning is classified into four types: sample-based learning, feature-based learning, model-based learning, and relationship-based learning [33], [34]. Also the transfer learning approach is mainly divided into two classes based on the similarity of domains, regardless of the availability of labelled and unlabelled data: Homogeneous transfer learning and heterogeneous transfer learning [33]. Homogeneous transfer learning [35] approaches are developed and proposed for handling the situations where the domains are of the same feature space. Heterogeneous transfer learning [35] refers to the knowledge transfer process in the situations where the domains have different feature spaces. The DNN with feature-based transfer learning is used in this work in which the weight matrix is shared from the pre-trained model and is re-trained to achieve fine-tuning. Feature-based approach is flexible in adopting different classification strategies according to the cases, which motivate us to focus on deriving a feature-based transfer learning approach for updating the signal detector DNN model in real time. It help us to train the DNN with small amount of available data (i.e., pilot) and saves the training time of DNN. Also, feature-based approach is applicable to both homogeneous and heterogeneous problems, selecting feature-based approach help us in upgrading the DeepDeMod model, such as using different modulation scheme (such as QPSK, QAM, ASK, FSK etc) or different task (such as modulation recognition, encoding/decoding, system parameter optimization etc). A similar method can be applied to other modulation schemes as well.

B. WORKING

In a conventional packet radio, pilot symbols are transmitted for the receiver to estimate the channel parameters. In transfer learning we use the same set of pilot symbols to re-train our DeepDeMod detector for better accuracy. A packet consists of two parts, namely pilot bits and actual data. At the receiver, the transmitted pilot bits corresponding to the received pilot signal is known. This is used to fine tune the weights in our DNN model. DNN is pre-trained with the primary data set. When a new packet arrives, the pilot signal is first used to retrain our DNN for achieving fine tuning. Thereafter, the detection of actual data bits take place.

DNN re-training may take some time which introduces delay in detection. To overcome this, we do not re-train the DNN model with every input of packet. From the pilot signal, we can estimate the BER using the pre-trained DNN. If the BER is within allowable threshold (λ), we use the same model for data bit detection. If the BER $\geqslant \lambda$, we re-train the model using the pilot signal to improve its performance. A flow graph is shown in Fig. 6, which explains the transfer learning based updation of DNN. The steps in the flow graph is described below.

- Step 1: A BPSK modulated packet is received at the receiver.
- Step 2: The received signal is passed through the product modulator and the BPF.
- Step 3: The signal is analog at this point. It is sampled and quantized (using ADC) to get K number of samples per bit period.
- Step 4: In this step, received signal corresponding to the pilot symbols are extracted. Pilot, length of pilot (number of pilot bits), length of data is known at the receiver.
- Step 5: In this step, sampled signal is reshaped for input to DNN.
- Step 6: The signal is provided as input to the DNN for detection. Initially, DNN is loaded with primary weights $[W_p]$. DNN uses this set of weights $[W_p]$ to check BER of this packet using pilot.
  - If BER $<$ allowable threshold (λ), DNN continues the detection of rest of the data with same weights $[W_p]$.

TABLE 2. List of parameters and values for simulation.

Parameter Name	Parameter Value
Modulation Mode	BPSK
Carrier frequency	915 MHz
Time period	0.75 μs
Number of samples per bit period	10
Sampling frequency	13.33 MHz
SNR range	10 dB to -40 dB

- If BER $> \lambda$, the DNN uses primary weight $[W_p]$ as starting point to re-train itself using pilot data. Once the DNN is re-trained it has a new set of weights $[W_{TL}]$, which is learned based on the current signal condition. Thereafter, this is used for signal detection. Detection continues with model weights $[W_{TL}]$ until the BER becomes greater than λ (in which case the model is retrained using the primary weights $[W_p]$ as described earlier). This is to avoid biasing, over-fitting, correlation and corruption during successive trainings.

V. SIMULATION RESULTS

In this section, a series of simulations have been carried out to verify the BER performance of the proposed DeepDeMod and DeepDeMod-TL. Matlab is used to generate data for training and testing purpose where carrier frequency (f_c) of the BPSK modulated signal is set to 915 MHz, number of samples per bit period (K) ranges from 10-50, and SNR (γ) ranges between 10 dB to -40 dB. Table 2 list the the parameters and their value used for the simulation.

The training set for the proposed DeepDeMod is generated via simulation, where various channel conditions, noise, and transmitter/receiver imperfection such as phase and frequency offset are considered. The number of binary symbols for each case is considered to be 10,000. Data under different cases are combined together randomly to form a *primary data set*. Primary data set has overall 2.5×10^6 binary symbols to train the DeepDeMod. Further, the network trained by the primary data set is used as the starting point for DeepDeMod-TL to fine tune the model. For testing purpose, 10,000 binary symbols are considered. Packet transmissions are considered wherein a packet has 1200 binary symbols which includes 200 pilot symbols and rest is data payload.

Below we present the BER performance of DeepDeMod and DeepDeMod-TL under various scenarios, which are given as follows:

- *Scenario 1 (AWGN channel)*: Channel considered is AWGN. Perfect synchronization between the transmitter and the receiver oscillators is considered.
- *Scenario 2 (Fading channel)*: The modulated signal is passed through a multipath fading channel with fading modelled as Rayleigh distribution. Perfect synchronization between the transmitter and the receiver oscillators is considered.
- *Scenario 3 (Hardware Imperfection)*: The modulated signal is passed through a fading channel. There exists phase and frequency offset between the transmitter and the receiver depicting the case of hardware imperfections.

For the above scenarios 1-3 the BER versus SNR curve of DeepDeMod and DeepDeMod-TL is presented for K equal to 10 and 50. These results are compared with the performance of the conventional coherent demodulator such as optimum detector and other machine learning based demodulators. ML based demodulators include one-dimensional convolutional neural network (1-D CNN) [15], artificial neural network (ANN) based demodulator named as neural network demodulator (NND) [11], and multiple layper perceptron (MLP) based classifier named as MaxMLP [10]. All these ML based demodulator take oversampled received signal as input to process and detect the respective binary bits. 1-D CNN [15] demodulates the BPSK modulated received signal by detecting the phase shift in the signal. It takes M (= 8) number of samples over one bit period and processes the received sampled data through 1D-CNN to detect the bits. NND [11] demodulates the BPSK modulated received signal using ANN. It take N_f (= 16) number of received signal samples over one bit period to demodulate. A feed-forward ANN is trained for multiple possible inputs of M-PSK modulated symbols. Thereafter, by using this ANN, the transmitted bits is detected by finding the maxima of the ANN outputs. MaxMLP [10] is comprised of multiple identical MLPs and each MLP acts as a binary classifier. The MLP is a feed forward ANN that maps a set of input data onto a set of appropriate output. It consists of multiple layers of nodes in a directed graph, with each layer fully connected to the next one. It takes multiple samples (= 20) over a bit period to demodulate the signal.

Apart from BER performance of the proposed method, we also show the performance of DeepDeMod for different values of K in Section V-D. A discussion on model complexity is presented in Section V-E.

A. DEMODULATION PERFORMANCE UNDER AWGN CHANNEL

Fig. 7 shows the BER performance of the proposed DeepDeMod, DeepDeMod-TL, conventional coherent demodulator and ML based demodulators such as, 1-D CNN [15], NND [11], and MaxMLP [10] under AWGN channel. It is observed from Fig. 7 that the proposed DeepDeMod and DeepDeMod-TL achieves better performance compared to the other methods. This is because, the proposed DeepDeMod performs demodulation by learning and mapping the over-sampled signal. During the training process, the DNN learns the signal pattern embedded in noise and develops anti-noise capability towards detecting message bits under low SNR. In simple words, the DNN learns the effect of noise on message signal and successfully maps the relation of such distorted signal to the desired output. Therefore, the

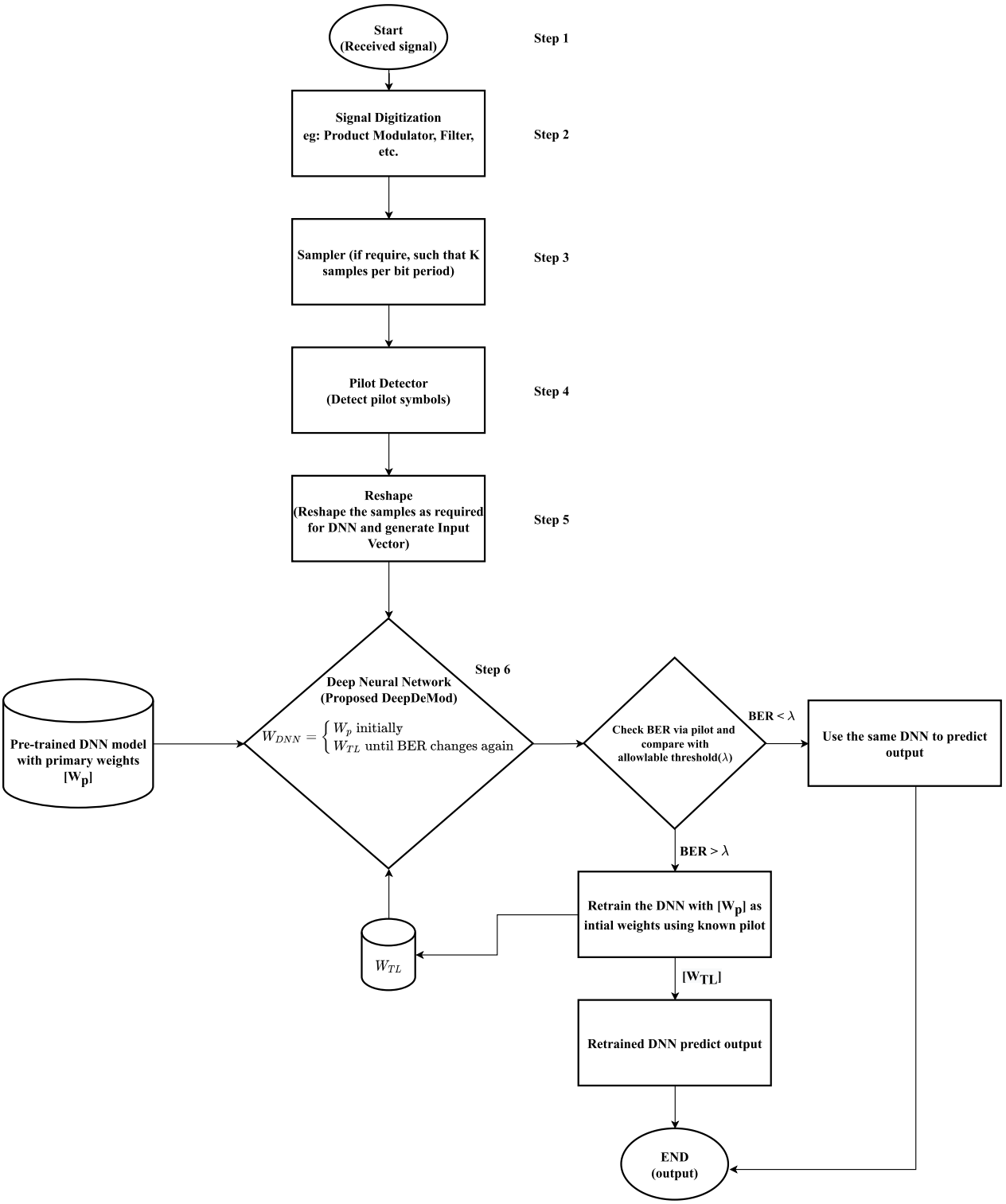


FIGURE 6. Flow graph of DNN updation using transfer learning.

performance of the proposed method outperforms the other methods. We also observe from Fig. 7 that the BER of 1-D CNN is lower than the conventional coherent demodulator when the SNR is greater than 4 dB. This is because as 1-D CNN detects the phase shift of the signal for demodulation, at higher SNR the phase shift can be easily recognised but at low SNR sudden peaks due to noise could be considered as phase change which explains the behaviour of BER curve of the 1D-CNN. The performance of NND and MaxMLP is nearly same as the performance of the conventional coherent demodulator. Here, the training process and data used by NND and MaxMLP are based on the output of matched

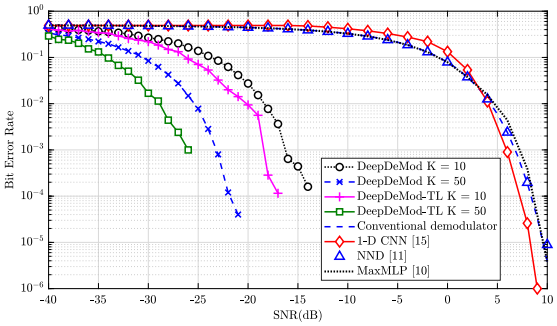


FIGURE 7. BER performance under AWGN Channel.

TABLE 3. BER comparison of neural network demodulators and conventional coherent demodulator under AWGN.

SNR	BER					
(dB)	DeepDeMod (K = 10)	DeepDeMod-TL (K = 10)	Conventional method	1-D CNN	NND	MaxMLP
-17	0.0036	0.00011	0.4208	0.4907	0.4208	0.4208
-18	0.0077	0.00019	0.4293	0.4907	0.4293	0.4293
-20	0.0273	0.00939	0.4437	0.4907	0.4437	0.4437
-22	0.0624	0.02000	0.4552	0.4907	0.4552	0.4552
-25	0.1384	0.07059	0.4208	0.4907	0.4683	0.4683

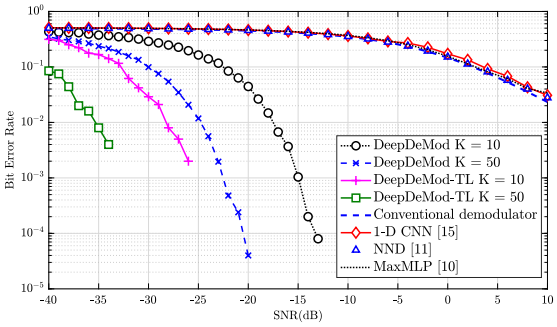


FIGURE 8. BER performance under Rayleigh fading channel.

filter receiver which makes its performance biased toward such a receiver. Also, the BER of DeepDeMod-TL is lower than the DeepDeMod. This is because the DeepDeMod is trained with the *primary dataset* which has a variety of sample points drawn from different parameters. Thus its learning is more general. On the other hand, DeepDeMod-TL refines the previously trained DeepDeMod model with the current pilot transmissions. Thus its performance is better.

The comparison of the proposed DeepDeMod, DeepDeMod-TL, conventional coherent demodulator, and other neural network demodulator [10], [11], [15] is shown in Table 3. The BER performance of the proposed demodulator provides better BER performance even in low SNR conditions.

B. DEMODULATION PERFORMANCE UNDER FADING CHANNEL

In this section, the BER performance of the proposed DeepDeMod, DeepDeMod-TL along with conventional coherent demodulator, 1-D CNN [15], NND [11], and MaxMLP [10] under Rayleigh fading channel is studied and results are shown in Fig. 8. It can be clearly observed from Fig. 8 that the DeepDeMod and DeepDeMod-TL perform better than the other demodulators under Rayleigh fading channel as well. In the conventional coherent demodulator, the channel estimation is a crucial part in the demodulation and it is estimated using pilot symbols via algorithms like minimum mean square error (MMSE), least squares (LS), maximum likelihood estimation (MLE), etc. In practical systems, channel estimation is not perfect which affects the performance of the conventional coherent demodulator. On the other hand, the DeepDeMod and DeepDeMod-TL network is trained using a wide range of dataset with varied fading and noise patterns, which helps the network comprehend the nature of received signals and improve performance. In other words, the network learns the impact of variation in signal amplitude and phase on the received signal caused by a noisy fading channel and updates its parameter accordingly. Hence, the performance of the proposed DeepDeMod is better than the existing methods. In a similar way other ML-based demodulators demodulate the signal without any channel estimation and there performance is close to the conventional coherent demodulator. This is because, the ML-based demodulators seems to learn the distribution of faded signal but their prediction is biased towards matched filter receiver. DeepDeMod-TL, in turn, updates itself with the most recent channel state, which is why its BER performance is superior to the DeepDeMod and other methods.

C. DEMODULATION PERFORMANCE UNDER HARDWARE IMPERFECTIONS

Synchronization is critical in digital signal demodulation, and its failure may have catastrophic effects on the system's performance. During demodulation, it is generally assumed that a perfect phase reference is available at the receiver. Two pairs of ideal identical oscillators at the transmitter and receiver sides could ensure such synchronization. However, the signal emitted by a pair of oscillators with the same nominal frequency start drifting soon from each other due to various factors, such as, *temperature variation, device non-linearity, aging, power supply ripples, etc* [36]. Such effect can cause phase offset ($\theta(t)$) and frequency offset ($\Delta f(t)$) at the local oscillator at the receiver. In this section, different parameters are considered together to analyse the performance of the proposed method. We consider a Rayleigh fading channel under AWGN noise with Tikhonov distributed phase offset [31], and Uniformly distributed frequency offset ($[-20, 20]$ ppm).

BER performance under hardware imperfections is computed and the results are shown in Fig. 9. The BER curve of a DeepDeMod, DeepDeMod-TL, conventional non-coherent demodulator, 1-D CNN [15], NND [11], and MaxMLP [10] is shown. There are time varying shifts in the phase of the received signal due to hardware imperfections. In conventional non-coherent demodulator, these time-varying shifts increase the probability of misdetection. However, in our proposed DeepDeMod and DeepDeMod-TL, the DNN has already learnt these time-varying shifts through the primary

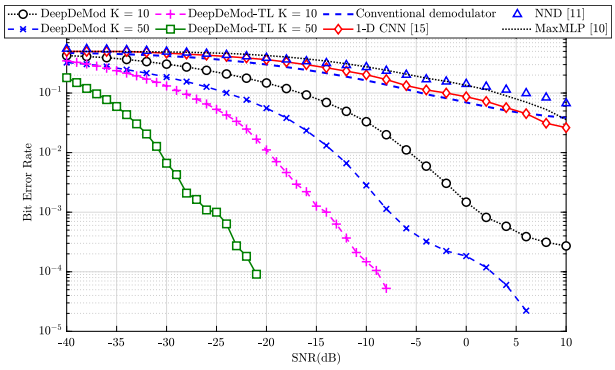


FIGURE 9. BER performance under frequency and phase offset and Rayleigh fading channel.

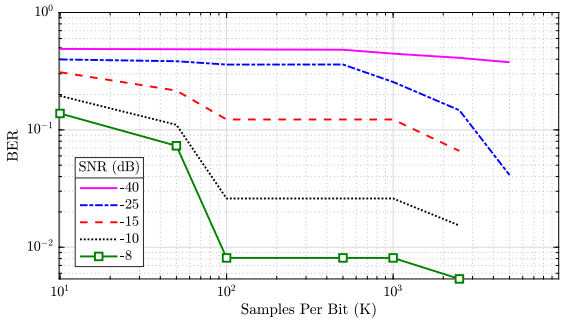


FIGURE 10. BER versus number of samples per bit (K).

dataset and therefore it can easily demodulate the received signal. In 1-D CNN, the BER is lower than the conventional non-coherent demodulator when SNR of the signal is higher than 7 dB, afterwards the 1-D CNN's performance degrades due to its inability to detect the correct phase shift. The BER performance of NND and MaxMLP decreases due to the ambiguity between two closely set of received input samples. Further, DeepDeMod-TL fine tunes the model parameters to effectively detect the signal based on the current frequency and phase offsets.

D. EFFECT OF NUMBER OF OVER-SAMPLES (K)

Number of samples per bit period (K) is an important factor for the DNN demodulator. For each bit, input to the DNN has 'K' number of samples which are used to detect the bit and hence its detection accuracy. Fig. 10 shows, the graph of BER versus K for different SNR values for our proposed DeepDeMod demodulation.

The result shows that the demodulation BER initially decreases with an increase in the number of samples then, it saturates. Beyond this BER again decreases. In general, the BER decreases with an increase in K. The reason for such a behaviour is that more number of samples provide DNN more opportunity to understand the underlying mapping of input to output, and, in turn, a better performing model. The performance saturates due to the fact that the increased number of samples do not provide any additional feature in the data. Increasing the number of samples results in improved performance albeit with increase in complexity and memory requirements. Therefore, the selection of 'K' is a critical trade-off between performance and complexity.

E. MODEL COMPLEXITY

The proposed algorithm in this work is based on the deep neural network. The complexity of a deep neural network can be divided into *time complexity* and *space complexity*. The time complexity includes the number of layers in the network, training time, testing time, etc. The overall time complexity is the sum of the time complexity of all layers. The space complexity includes the total number of parameters and the characteristic (such as activation function, loss function, etc.) of each layer.

Table 4 shows the proposed DeepDeMod DNN layout and its complexity. Our proposed DNN comprises of four hidden layers and one output layer. The number of neurons for each layer and their activation function are listed in Table 4. The time complexity determines the training time and the prediction time of the model. If the complexity is too high, the training and prediction of the model will cost a lot of time, and fast prediction cannot be achieved. In terms of Big-O notation, the time complexity of back-propagation is $O(n \cdot m \cdot h^p \cdot o \cdot i)$ [37], where n is the number of training samples, m is the number of input samples ($m = K$), p is the number of hidden layers each containing h number of neurons - for simplicity, o is the number of output neurons, and i is the number of iterations.

The space complexity determines the number of parameters in the network model. The more parameters in the network, the more data required for training, which will easily lead to model overfitting. The spatial complexity of a fully connected model is closely related to the size of the input data. The larger the size of the input data, the larger the size of the model and hence its complexity. In our proposed DeepDeMod, the total number of trainable parameters (or weights) for $K = 10$ are 1,453 and for $K = 50$ are 3,053, which shows that with a increase in the number of input samples (K), complexity also increases.

The computational complexity in terms of operations is also shown in Table 4, which includes the number of multiplication, addition, and floating point operations (FLOPs). For BPSK, the computational complexity of DeepDeMod is K^2, while for 1-D CNN [15], NND [11], MaxMLP [10] is $(K+1)^2$, $2(K+1)^2$ and $2N(K+1)^2$, respectively. FLOPs means the total number of floating point operations required for a single forward pass. It can be used to measure the complexity of the model. The higher the FLOPs, slower the model. FLOPs depends on input, output size of layers, filter size, kernel size, etc. based on neural network. In the proposed DeepDeMod the total number of FLOPs are 2833 FLOPs per iteration for K equal to 10, while 1-D CNN [15] has approx. 31.5172×10^6 FLOPs, NND [11] has 39 FLOPs, since it use single layer single neuron structure, also MaxMLP [10] has 3500 FLOPs. In the proposed DeepDeMod model, training time is 2963.01015 sec and prediction time for each bit is 3.521×10^{-5} secs for K equal to 10. Also for DeepDeMod-TL

TABLE 4. Layout of the DeepDeMod for signal detection for $K = 10$.

Layer	Output dimension	# parameters	# multiplication operation	# addition operation	FLOPs
Input	$K = 10$	NA	NA	NA	NA
1^{st} Hidden layer - Dense + ReLu	40	440	400	41	840
2^{nd} Hidden layer - Dense + ReLu	20	820	800	21	1620
3^{rd} Hidden layer - Dense + ReLu	7	147	140	8	287
4^{th} Hidden layer - Dense + ReLu	5	40	35	6	75
Output - Dense + sigmoid	1	6	5	2	11
Trainable parameters - 1,453			Total FLOPs - 2833 FLOPs		
Training time DeepDeMod : 2963.01015 sec			Prediction time DeepDeMod: 35.21 μsec		
Training time DeepDeMod-TL : 0.9 sec			Prediction time DeepDeMod-TL: 35.21 μsec		

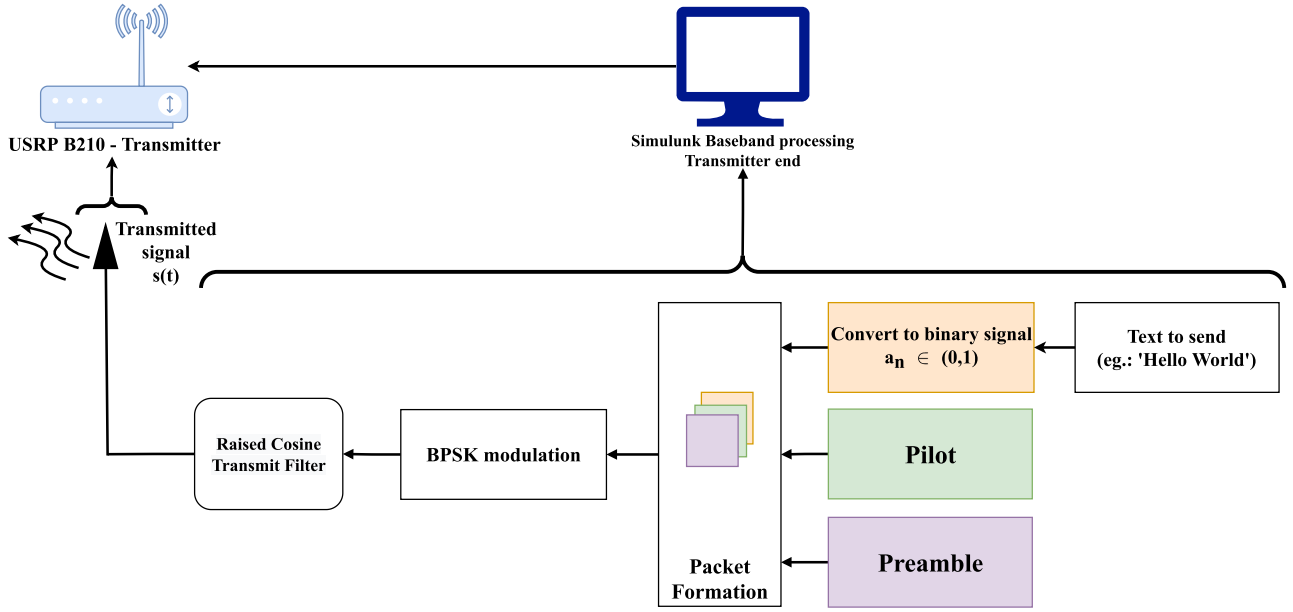


FIGURE 11. Block diagram of transmitter as implemented in the SDR.

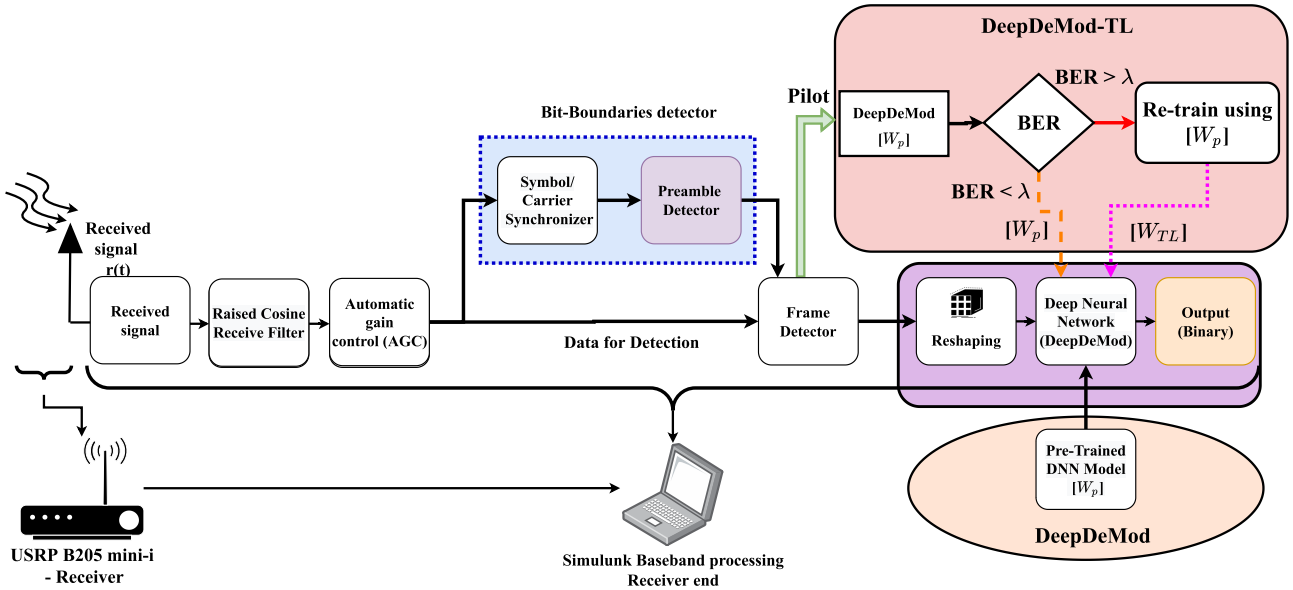


FIGURE 12. Block diagram of receiver as implemented in the SDR. The receiver implements the proposed DeepDeMod and DeepDeMod-TL methods.

model, training time is 0.9 sec per pilot and prediction time for each bit is 3.521×10^{-5} secs for K equal to 10. Based on overall comparison of complexity and performance, DeepDeMod is seen to be superior to other ML based models. Also DeepDeMod presents a better trade off between complexity and performance.

A complexity comparison of Conventional and DeepDeMod methods is shown in the Table 5. The complexity

TABLE 5. Complexity comparison of conventional and DeepDeMod method.

Modules based complexity	Conventional method	DeepDeMod Method
Filter	LPF (matched/ correlation filter)	BPF
Integrator	Yes (based on filter use)	Not required
Sampler	Yes, 1-sample	Yes, K- samples
Threshold device/detector	Yes	No
Synchronization device (PLL)	Yes	No
Channel estimator	Yes	No
Reshaping block	No	Yes

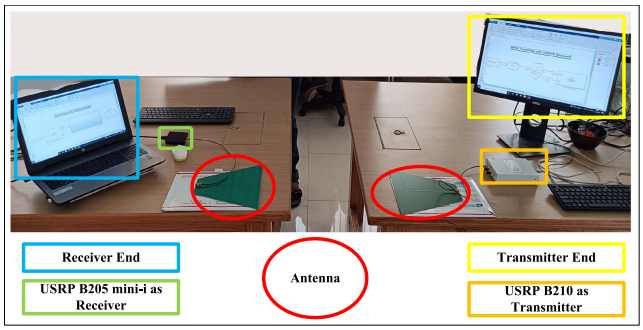
trade-off of the conventional coherent demodulator and proposed DeepDeMod is based on sampling the received signal to process by the decision module. The conventional method uses one sample per bit period, whereas DeepDeMod uses K samples per bit period. As shown in Section V-D the BER decreases with increase in number of samples per bit period, thus increasing the complexity of the proposed DNN. Hence, proper selection of K is a trade-off between performance and complexity. The conventional coherent BPSK receiver is based on the phase-lock loop and VCO to recover the carrier signal [38]. Such a module increases the cost, power consumption, and processing time of the receiver and is complex to implement [3], [38]. In order to maximize the performance, the conventional receiver utilizes a channel estimator to track the channel state to correctly demodulate the binary data [39], [40]. Such modules add up the computational complexity of the conventional receiver. However, in the DeepDeMod, the data bit-stream has been extracted without needing a PLL, and channel estimator. Thus, the proposed DeepDeMod provides better performance and simplified receiver design however at the expense of higher memory requirements.

VI. HARDWARE IMPLEMENTATION AND EXPERIMENTAL RESULTS

For the hardware implementation of the proposed method, we use software defined radio (SDR), which is a generic radio communication system with most of the physical layer (PHY) functionality written in software [41]. In our prototype, a packet-based radio communication system is implemented. It consists of two USRPs namely B210 (as a transmitter) and B205 mini-i (as a receiver). Each of these are connected to their respective host processing device running a baseband processing algorithm using Matlab/Simulink software. This system is designed and verified using Simulink's Communication System Toolbox and Signal Processing Toolbox.

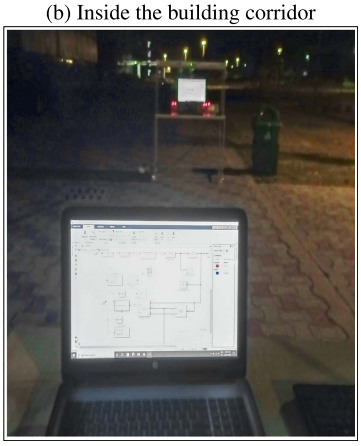
A. DeepDeMod HARDWARE IMPLEMENTATION

In this implementation, packet transmission is carried out wherein a packet contains three parts namely, the preamble, the pilot and the payload/data. We assume the preamble, the pilot, length of the preamble, length of the pilot and length of the data is known to the receiver. Selection of the preamble for the packet is subtle. Simulink/SDR is unable to process the randomly generated preamble. If we use the randomly generated preamble, the receiver is unable to detect the packet.



(a) Inside the Lab

(b) Inside the building corridor



(c) Outside (open area)

FIGURE 13. DeepDeMod experimental set-up at different locations.

Further, the bit-boundaries are unknown. We use barker code to generate the preamble for the packet. Whereas, pilot is randomly selected in this implementation. Fig. 11 presents the block diagram of the transmitter part implemented in SDR while Fig. 12 shows the receiver block diagram as implemented in software.

Payload message is converted into binary signal (data) using 7-bit ASCII code and is transmitted. The packet is formed by concatenating preamble, pilot and data. This packet is modulated using BPSK scheme and passed through raised cosine transmit filter to up-sample the signal. The signal is transmitted over the air using USRP B210.

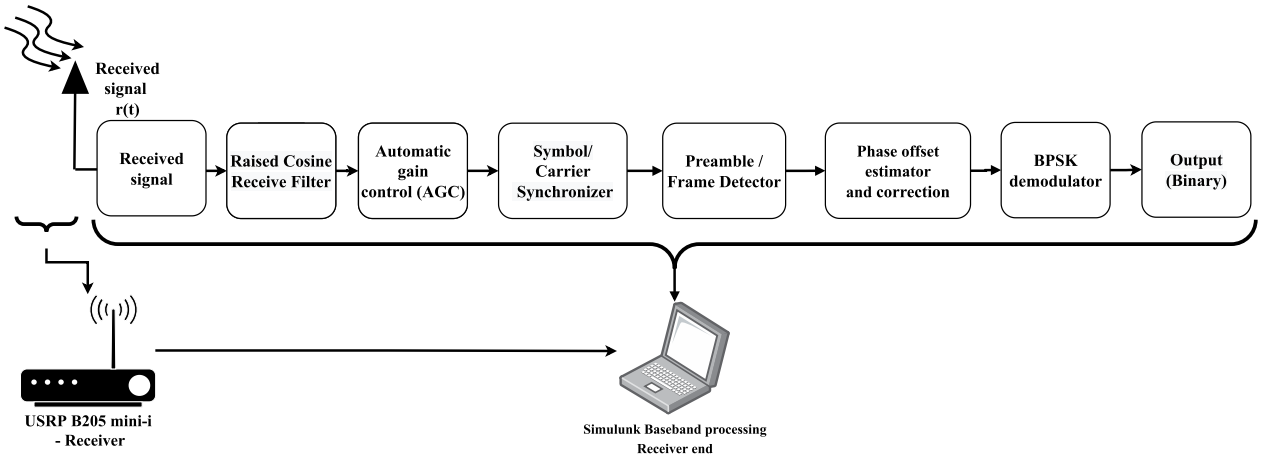


FIGURE 14. Block diagram of the conventional coherent demodulator implementation on SDR.

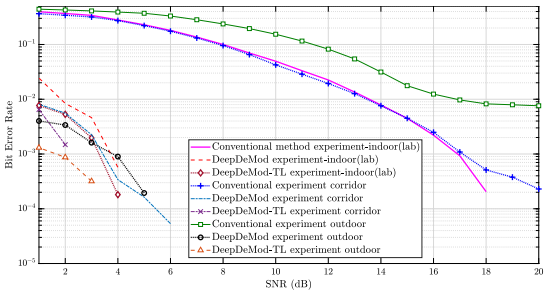


FIGURE 15. Experimental BER for DeepDeMod, DeepDeMod-TL ($K = 5$) and conventional coherent demodulator.

The signal is received by USRP B205 mini-i. We oversample the received signal by sampling it K-times in a bit period. Once the sampled signal is received, it is passed through a raised cosine filter and an automatic gain controller (AGC). In this work, we assume that the bit-boundaries of the signal are known at the receiver. To have this knowledge at the receiver, we use symbol/carrier synchroniser block along with preamble detector. The original data and bit-boundaries are inputted to the frame detector in order to align the data along the correct bit-boundaries in the signal stream. Next, the reshaping block reshapes the signal stream into the desired shape and is processed by the pretrained DNN model. In DeepDeMod, DNN uses the pre-trained weights $[W_p]$ to predict binary output. On the other-hand, DeepDeMod-TL first calculates the BER using pre-trained weights $[W_p]$ and known pilot. If the BER is less than the desired threshold (λ), the DNN continues to predict with the same weights $[W_p]$. However, if the BER is greater than the desired threshold re-training of the DNN model with W_p as initial weights and pilot as training data is carried out using transfer learning. The updated weights W_{TL} are then used for bit prediction. Finally, the predicted binary data stream can be converted back to the desired message.

Setting correct parameters on the SDR is critical and sometimes challenging. Even though the DNN is able to adapt, its performance does depend on choosing the right SDR parameters. Table 6 details the design parameters of the

TABLE 6. Prototype description for implementing DeepDeMod.

Experiment setup	Type and parameter
Transmitter USRP	B210
Receiver USRP	B205 mini - i
Carrier frequency	915 MHz
Antenna type	Log periodic
Antenna Gain	5 dBi
Amplifier	CN0522

DeepDeMod prototype. Log periodic directional antenna is used to transmit the high frequency signals. At the transmitter, an amplifier is connected for long range communication. Fig. 13 shows the set-up of our experiment at different locations viz., *inside the lab, in corridor inside the building, and outside (open area)*. We operate at 915 MHz carrier frequency. At receiver the signal is demodulated using the conventional coherent method and the proposed DeepDeMod, and DeepDeMod-TL method.

B. CONVENTIONAL COHERENT DEMODULATION IMPLEMENTATION ON HARDWARE

In this sub-section, a brief description on the implementation of the conventional coherent demodulator on USRP is discussed. A block diagram of its implementation on software is shown in Fig. 14. Here, the received signal is passed through a raised cosine filter and an AGC. Then the signal is passed through the symbol synchroniser which corrects the symbol timing clock between a single-carrier transmitter and receiver. Then the time corrected signal is passed through the carrier synchronizer block which compensates for the carrier frequency and phase offset. Then the signal passes through the preamble detector to detect the start/end position of the preamble in the packet. The data payload is separated from the packet and forwarded to the phase offset estimator block to correct the phase offset in the received signal. Finally, the BPSK demodulator block is used to map the received signal to its binary form using hard decision.

C. EXPERIMENTAL RESULTS

The demodulation performance of our proposed DeepDeMod prototype at different locations as shown in Fig. 13 is presented in Fig. 15. Fig. 15 shows the BER performance of the proposed DeepDeMod and DeepDeMod-TL, along with the performance of the conventional coherent demodulator in the same environment.

It is observed from Fig. 15 that the proposed DeepDeMod and DeepDeMod-TL achieves better performance compared to the conventional coherent method. The trained DeepDeMod matches the features in the input signal which may include channel effect, hardware imperfections (phase/frequency offsets), etc., and compensates for these effects. Hence, the performance of the proposed method outperforms the conventional coherent demodulator. The DeepDeMod-TL performance shows a significant reduction in BER with respect to the DeepDeMod upto SNR = 3 dB. The low SNR (i.e., 2 & 1 dB) performance of DeepDeMod-TL is close to DeepDeMod performance. The reason for this behaviour is that a lesser number of features (i.e., $K = 5$) are used for training and fine-tuning, because of which the model is unable to learn the signal's anomaly distribution/information in sufficient amount. However, the performance by proposed method is enhanced significantly as compared to the conventional coherent demodulator.

VII. CONCLUSION

This paper proposes a DNN-based BPSK detector composed of preprocessing the received signal (using bandpass filter), a reshaping block, and a 5-layer DNN. The DNN detects the transmitted bits by learning the received signal. Results indicate that the proposed DeepDeMod and DeepDeMod-TL can accurately detect BPSK signals, even if carrier frequency and phase offsets exist. Employing the modulated signals with additive white Gaussian noise and Rayleigh fading, the DNN can detect the signal accurately. We also observed the performance enhancement with increasing number of samples per bit, albeit at a higher computational complexity. The proposed method can be applied to various applications such as industrial IoT, military communication, etc., where stringent BER is required under adverse channel and environmental condition.

In this work, we have assumed that the bit boundaries and preamble is detected prior to passing the signal through the DeepDeMod. Our future task would be to develop a complete deep learning based receiver which is able to perform all receiver functionalities.

REFERENCES

[1] C.-U. Baek, J.-W. Jung, and D.-W. Do, "Study on the structure of an efficient receiver for covert underwater communication using direct sequence spread spectrum," *Appl. Sci.*, vol. 8, no. 1, p. 58, Jan. 2018.

[2] Z. Luo and S. Sonkusale, "A novel BPSK demodulator for biological implants," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 6, pp. 1478–1484, Jul. 2008.

[3] Y. Hu and M. Sawan, "A fully integrated low-power BPSK demodulator for implantable medical devices," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 12, pp. 2552–2562, Dec. 2005.

[4] G. A. Leonov, N. V. Kuznetsov, M. V. Yuldashev, and R. V. Yuldashev, "Nonlinear dynamical model of Costas loop and an approach to the analysis of its stability in the large," *Signal Process.*, vol. 108, pp. 124–135, Mar. 2015.

[5] M. Zhao, W. Zhou, W. Zhang, and M. Dou, "Aging drift compensation," in *Proc. IEEE Int. Freq. Control Symp.*, Baltimore, MD, USA, May 2012, pp. 1–4.

[6] H. Zhou, C. Nicholls, T. Kunz, and H. Schwartz, "Frequency accuracy & stability dependencies of crystal oscillators," Syst. Comput. Eng., Carleton Univ., Ottawa, ON, Canada, Tech. Rep., SCE-08-12, pp. 1–15, Nov. 2008.

[7] Z. Xu, C. Sun, T. Ji, J. H. Manton, and W. Shieh, "Computational complexity comparison of feedforward/radial basis function/recurrent neural network-based equalizer for a 50-Gb/s PAM4 direct-detection optical link," *Opt. Exp.*, vol. 27, no. 25, pp. 36953–36964, 2019.

[8] P. J. Freire, S. Srivallapanondh, A. Napoli, J. E. Prilepsky, and S. K. Turitsyn, "Computational complexity evaluation of neural network applications in signal processing," 2022, *arXiv:2206.12191*.

[9] K. Nakayama and K. Imai, "A neural demodulator for amplitude shift keying signals," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 6. Orlando, FL, USA, Jun. 1994, pp. 3909–3914.

[10] F. He, X. Xu, L. Zhou, and H. Man, "A learning based cognitive radio receiver," in *Proc. Mil. Commun. Conf.*, Baltimore, MD, USA, Nov. 2011, pp. 7–12.

[11] M. Önder, A. Akan, and H. Doğan, "Advanced~neural network receiver design~to combat multiple channel~impairments," *TURKISH J. Electr. Eng. Comput. Sci.*, vol. 24, no. 4, pp. 3066–3077, Jan. 2016.

[12] M. Fan and L. Wu, "Demodulator based on deep belief networks in communication system," in *Proc. Int. Conf. Commun., Control, Comput. Electron. Eng.*, Khartoum, Sudan, Jan. 2017, pp. 1–5.

[13] H. Wang, Z. Wu, S. Ma, S. Lu, H. Zhang, G. Ding, and S. Li, "Deep learning for signal demodulation in physical layer wireless communications: Prototype platform, open dataset, and analytics," *IEEE Access*, vol. 7, pp. 30792–30801, 2019.

[14] A. S. Mohammad, N. Reddy, F. James, and C. Beard, "Demodulation of faded wireless signals using deep convolutional neural networks," in *Proc. IEEE 8th Annu. Comput. Commun. Workshop Conf.*, Las Vegas, NV, USA, Jan. 2018, pp. 969–975.

[15] M. Zhang, Z. Liu, L. Li, and H. Wang, "Enhanced efficiency BPSK demodulator based on one-dimensional convolutional neural network," *IEEE Access*, vol. 6, pp. 26939–26948, 2018.

[16] Y. Liu, Y. Shen, L. Li, and H. Wang, "FPGA implementation of a BPSK 1D-CNN demodulator," *Appl. Sci.*, vol. 8, no. 3, pp. 26939–26948, Mar. 2018.

[17] W. Leonard, A. Saunders, M. Calabro, and K. Olsen, "A multi-waveform radio receiver, an example of machine learning enabled radio architecture and design," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Norfolk, VA, USA, Nov. 2019, pp. 634–639.

[18] W.-J. Chen, J. Wang, and J.-Q. Li, "End-to-end PSK signals demodulation using convolutional neural network," *IEEE Access*, vol. 10, pp. 58302–58310, 2022.

[19] S. Zheng, S. Chen, and X. Yang, "DeepReceiver: A deep learning-based intelligent receiver for wireless communications in the physical layer," *IEEE Trans. Cognit. Commun. Netw.*, vol. 7, no. 1, pp. 5–20, Mar. 2021.

[20] L. Zhang, H. Zhang, Y. Jiang, and Z. Wu, "Intelligent and reliable deep learning LSTM neural networks-based OFDM-DCSK demodulation design," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 16163–16167, Dec. 2020.

[21] T. Erpek, T. J. O'Shea, Y. E. Sagduyu, Y. Shi, and T. C. Clancy, "Deep learning for wireless communications," in *Development and Analysis of Deep Learning Architectures*. New York, NY, USA: Springer, 2020, pp. 223–266.

[22] S. Ali et al., "6G white paper on machine learning in wireless communication networks," 2020, *arXiv:2004.13875*.

[23] K. Jiang, J. Zhang, H. Wu, A. Wang, and Y. Iwahori, "A novel digital modulation recognition algorithm based on deep convolutional neural network," *Appl. Sci.*, vol. 10, no. 3, p. 1166, Feb. 2020.

[24] W. Zhang and M. Krunz, "Machine learning based protocol classification in unlicensed 5 GHz bands," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, May 2022, pp. 752–757.

[25] P. Ghasemzadeh, M. Hempel, and H. Sharif, "A robust graph convolutional neural network-based classifier for automatic modulation recognition," in *Proc. Int. Wireless Commun. Mobile Comput. (IWCMC)*, Dubrovnik, Croatia, May 2022, pp. 907–912.

[26] K. Oshima, D. Yamamoto, A. Yumoto, S.-J. Kim, Y. Ito, and M. Hasegawa, "Online machine learning algorithms to optimize performances of complex wireless communication systems," *Math. Biosci. Eng.*, vol. 19, no. 2, pp. 2056–2094, 2021.
[27] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2457–2471, Apr. 2021.
[28] W. Xia, G. Zheng, Y. Zhu, J. Zhang, J. Wang, and A. P. Petropulu, "A deep learning framework for optimization of MISO downlink beamforming," *IEEE Trans. Commun.*, vol. 68, no. 3, pp. 1866–1880, Mar. 2020.
[29] J. Sykora, A. Czylwik, L. Clavier, and A. Burr, "Coding and processing for advanced wireless networks," in *Inclusive Radio Communications for 5G and Beyond*, C. Oestges and F. Quitin, Eds. New York, NY, USA: Academic, 2021, ch. 5, pp. 121–140.
[30] A. Zaidi, F. Athley, J. Medbo, U. Gustavsson, G. Durisi, and X. Chen, "NR waveform," in *5G Physical Layer*. New York, NY, USA: Academic, 2018, ch. 6, pp. 159–198.
[31] A. Chandra, A. Patra, and C. Bose, "Performance analysis of BPSK over different fading channels with imperfect carrier phase recovery," in *Proc. IEEE Symp. Ind. Electron. Appl. (ISIEA)*, Penang, Malaysia, Oct. 2010, pp. 106–111.
[32] M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," 2015, *arXiv:1603.04467*.
[33] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *J. Big Data*, vol. 3, no. 1, pp. 1–40, Dec. 2016.
[34] M. Long, J. Wang, J. Sun, and P. S. Yu, "Domain invariant transfer kernel learning," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 6, pp. 1519–1532, Jun. 2015.
[35] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, Jul. 2020.
[36] A. Chandra, A. Patra, and C. Bose, "Performance analysis of PSK systems with phase error in fading channels: A survey," *Phys. Commun.*, vol. 4, no. 1, pp. 63–82, Mar. 2011.
[37] *Neural Network Models (Supervised)*. Accessed: Jun. 4, 2022. [Online]. Available: https://scikit-learn.org/stable/modules/neural_networks_supervised.html?highlight=complexity
[38] A. Moeinfar, H. Shamsi, M. Taradeh, S. Gholami, and S. Afrancheh, "Novel high-data-rate low-complexity BPSK demodulator for telemetry systems," in *Proc. Int. Conf. Comput. Tool*, Lisbon, Portugal, Apr. 2011, pp. 4–5.
[39] D. Tyagi and N. P. Shrivastava, "Receiver designing and channel estimation," *Int. J. Comput. Sci. Inf. Technol.*, vol. 3, no. 3, pp. 4059–4063, May 2012.
[40] P. Hammarberg, F. Rusek, and O. Edfors, "Iterative receivers with channel estimation for multi-user MIMO-OFDM: Complexity and performance," *EURASIP J. Wireless Commun. Netw.*, vol. 2012, pp. 1–17, Mar. 2012.
[41] R. W. Stewart, K. W. Barlee, D. S. Atkinson, and L. H. Crockett, *Software Defined Radio Using MATLAB & Simulink and the RTL-SDR*, 1st ed. New York, NY, USA: Academic, 2015.

ARHUM AHMAD (Graduate Student Member, IEEE) received the B.E. degree (Hons.) in electronics and telecommunication engineering from Chhattisgarh Swami Vivekanand Technical University (CSVTU), Durg, Chhattisgarh, India, in 2018. He is currently pursuing the Ph.D. degree with the Department of Electrical Engineering, Indian Institute of Technology Ropar, India. His research interests include wireless communication networks, including next-generation networks, 5G networks and architecture, UAV communications, machine learning, artificial Intelligence, and the IoT.

SATYAM AGARWAL (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from IIT Delhi, in 2016. He is currently an Assistant Professor with the Department of Electrical Engineering, IIT Ropar, India. Prior to this, he was an Assistant Professor with IIT Guwahati. In 2017, he was a Postdoctoral Researcher with Politecnico di Torino, Turin, Italy. His research interests include wireless communication networks, including next-generation networks, 5G networks and architecture, and air-borne networks.

SAM DARSHI (Senior Member, IEEE) received the B.Tech. degree (Hons.) in electronics and communication from M.J.P. Rohilkhand University, Bareilly, Uttar Pradesh, and the Ph.D. degree from the Department of Electronics and Electrical Engineering (EEE), IIT Guwahati, India. He is currently working as an Assistant Professor at IIT Ropar, Punjab. Prior to joining IIT Ropar, he served as a Faculty Member at IIIT Guwahati. His research interests include infrastructure-less multihop and relay networks, cooperative communication, next generation wireless networks, and intelligent transportation systems.

SUMIT CHAKRAVARTY (Member, IEEE) received the bachelor's, master's, and Ph.D. degrees in electrical and computer engineering. He has an extensive background in electrical and computer engineering. He served as a Postdoctoral Researcher at UPENN. He has used his expertise in signal processing machine learning and communications in his role as a Principal Scientist in industry as well as a Faculty Member in academia. He is currently an Associate Professor of ECE. He has published multiple journals and peer reviewed conference papers in venues like *Pattern Recognition*, IEEE Sensors, and *Journal of Medical Imaging*. His academic experiences include working for NASA-Goddard, University of Maryland, and NYIT. His industry experience in engineering and research include working in various roles, such as an Instrumentation Engineer and a Research Intern with Siemens CAD and Apex Eclipse Communications, and a Principal Scientist for SGT Inc., Honeywell Research (Automatic Control Solutions-Advanced Technology Labs). He has served as a Guest Editor for *Electronics* journal Special Issue on "Signal Processing in Wireless Communications" and Special Issue on "Towards Reliable and Scalable Smart Cities: Internet of Things Meets Big Data and AI."