Received: 10 February 2020 | Revised: 7 February 2021 | Accepted: 3 March 2021

IET Communications

DOI: 10.1049/cmu2.12162

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH PAPER

Artificial intelligence enhances the performance of chaotic baseband wireless communication

Hai-Peng Ren[1] | Hui-Ping Yin[1] | Hong-Er Zhao[1] | Chao Bai[2] | Celso Grebogi[1,3]

[1] Shaanxi Key Laboratory of Complex System Control and Intelligent Information Processing, Xi'an University of Technology, Xi'an, People's Republic of China

[2] Xi'an Technological University, Xi'an, People's Republic of China

[3] Institute for Complex Systems and Mathematical Biology, King's College, University of Aberdeen, Aberdeen, UK

Correspondence:
Hai-Peng Ren, Shaanxi Key Laboratory of Complex System Control and Intelligent Information Processing, Xi'an University of Technology, Xi'an 710048, People's Republic of China.
Email: renhaipeng@xaut.edu.cn

Abstract

It was reported recently that chaos properties could be used to relieve inter-symbol interference caused by multipath propagation in chaos-based wireless communication system. Although there exists the optimal decoding threshold to theoretically eliminate the inter-symbol interference, its practical implementation is still a challenge due to the strong requirement to know the future symbols to be transmitted. To tackle this almost 'impossible' task, convolutional neural network with deep learning structure is proposed to predict future symbols based on the received signal, to further reduce inter-symbol interference and to obtain a better bit error rate performance. Due to the short time predictability of chaotic signal, the proposed method is able to predict short-term future symbols and get a better threshold suitable for the time-variant channel. The analytical bit error rate of the proposed method is derived. The contributions of the paper are as follows: firstly, a convolutional neural network with deep learning structure is proposed for the first time to predict the future symbols in the chaos baseband wireless communication system, which does not require much training in this important application; secondly, the future bits predicted by the trained convolutional neural network are used together with the past decoded bits to calculate more accurate decoding threshold compared with the existing methods, yielding a better bit error rate performance. Numerical simulations and experimental results validate the effectiveness of our theory and the superiority of the proposed method.

1 | INTRODUCTION

Artificial intelligence (AI) is a technology for simulating and emulating human cognition. It was proposed in 1956, but it has experienced a tortuous development [1, 2]. Recently, AI has been demonstrating its superior performance in a variety of practical applications [3]. Convolutional neural network (CNN) is one of the most popular deep learning network structures used in AI and it has a wide range of applications in various fields. In the field of communication, the increase in hardware computing speed, especially in the field-programmable gate array (FPGA) with high performance to price ratio, makes it possible to use AI in communication systems [4–7].

Chaos has been applied in communication since 1993 [8, 9], but most early research concentrated on the theoretical analysis of bit error rate (BER) and security in the ideal channel, possibly with white noise. Since a performance improvement in fibre-optical communication using chaos was reported [9], attention has been drawn into practical communication channel applications. The wireless channel, the topic of this work, is a practical channel widely used worldwide. Due to the serious distortion of the wireless communication channel caused by limited bandwidth, multipath propagation, and complicated noise, the wireless communication requires more sophisticated techniques to achieve satisfactory performance compared to the wired communication counterpart. Although chaotic signals have been reported to possess some desirable properties for communication, such as broad band and orthogonality, whether the information in the chaotic signal suffers loss has been a pending problem. However, this issue was resolved [10] by the proof of the invariance of the Lyapunov spectrum of the chaotic signal while being transmitted through a wireless channel. The Lyapunov spectrum invariance property meant that the information content in the signal is not lost during transmission. Further research efforts have boosted the applicability of chaos-based wireless communication systems after some new features

of chaos were reported [11–13]. They include the application of chaos in the conventional wireless communication systems [13] and the invariant Lyapunov exponent spectrum, which can be used to relieve the inter-symbol interference (ISI) caused by multipath propagation [11].

Recently, ISI resistance performance of chaotic baseband wireless communication systems (CBWCS) was theoretically and experimentally investigated [11–13]. In particular, some properties from chaotic dynamics could be used to relieve ISI, resulting in superior performance, compared to the conventional methods such as channel equalisation. The existing ISI relief technique only uses the available information [11–13], that is, the past received bits (symbols), because the future symbols are not available at the current time, even though it affects the ISI relief performance. How to improve the ISI relief performance by predicting or estimating the future information bit? This has become a critical issue, which we address here. There existed some AI methods applied to communication systems for dealing with multipath propagation. An ISI reduction scheme based on multilayer neuron network and back-propagation (BP) algorithm in orthogonal frequency division multiplexing system was proposed [14], which achieved a better performance. Neural network decoder based on multilayer neural network [15] and radial basis function [16] were proposed for decoding the information that did not need complicated theoretical decoding threshold. By learning the function of conventional equalisation, Gaussian kernel deep neural network and recurrent neural network were trained as channel equaliser [17, 18]. Potential applications of machine learning in wireless communication for Internet of things were reviewed in [19]. Compared to nonchaotic baseband signal communication, the chaotic baseband signal has special properties such as short-term predictability. Therefore, the neural networks were proposed for short-term time series prediction of the chaotic system [20–22]. Recently, predicting medium to relatively long-term chaotic time series using the echo state network (ESN) has been reported for better performance [23]. To explore the ESN property, the short-term baseband waveform prediction was performed in [24]. Then, according to the predicted waveform, one future bit was decoded and used in the calculation of a more accurate decoding threshold in order to achieve a better BER performance [12]. Because the waveform or time series prediction can be done for only one sampling point iteratively [24], it is hard to predict additional future bits with relatively high precision.

Different from the idea in [24], where the ESN is used for dynamical time sequence prediction, by exploring the mapping function between the image constructed from sampling points of the filtered baseband waveform and the corresponding information bits, this work uses the CNN. Its use enhances local image feature extraction ability derived by the convolution operations (layers) and it results in low training cost compared with conventional (or deep) neural network rendered by pooling operation and shared weights. The CNN is trained to infer the future symbol sequence directly, avoiding the intermediate iterative steps in the prediction as done in [24] and thus simplifying the operation. Besides this advantage, in our wireless communication system configuration, the data are transmitted frame by frame with a dedicated probe data, which is used for clock synchronisation as done in the conventional wireless communication system. The probe data is also used for the CNN training to avoid the time-varying channel parameters effect on the ISI relief performance.

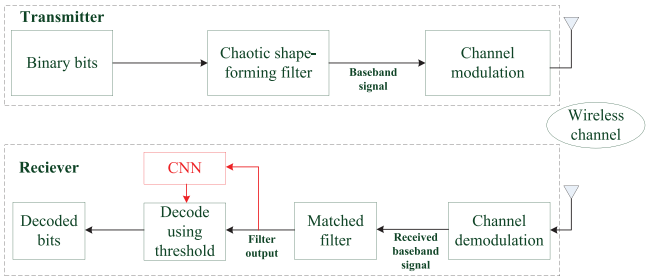


FIGURE 1 Block diagram of chaotic baseband wireless communication system

The remainder of the paper is organised as follows. In Section 2, the configuration of our CBWCS and the predictability of the future symbol are given. In Section 3, the ISI resistance performance of CBWCS is analysed. In Section 4, the future bit is predicted directly using CNN, where the CNN training is explained in detail. Simulation and experimental results are given in Sections 5 and 6, respectively. The conclusions are given in Section 7.

2 | THE CONFIGURATION OF CHAOTIC BASEBAND WIRELESS COMMUNICATION SYSTEM

The block diagram of our CBWCS is given in Figure 1.

In CBWCS, the chaotic baseband signal is generated by chaotic shape forming filter given by the following equation [12]:

$$u(t) = \sum_{m=-\infty}^{\infty} s_m \cdot p(t-m), \tag{1}$$

where $p(t)$ is the basis function:

$$p(t) = \begin{cases} \left(1 - e^{-\beta/f}\right) e^{\beta t} \left(\cos \omega t - \dfrac{\beta}{\omega} \sin \omega t\right), & (t < 0) \\ 1 - e^{-\beta(t-1/f)} \left(\cos \omega t - \dfrac{\beta}{\omega} \sin \omega t\right), & (0 \le t < 1/f), \\ 0, & (t \ge 1/f) \end{cases} \tag{2}$$

where $\omega = 2\pi f$, f is the base frequency, $\beta = f \times \ln 2$, $s_m \in [-1, 1]$ is the information symbol to be transmitted. The base function waveform for $f = 1$ is given in Figure 2.

In Figure 1, the binary information bits to be transmitted are fed into the chaotic shape forming filter given by Equation (1) to form the baseband signal. Then the baseband signal is passed

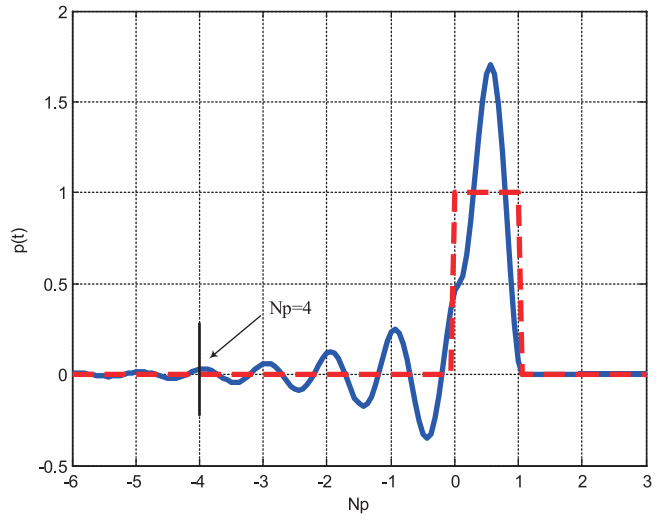


FIGURE 2 The base function waveform of chaotic shape forming filter for $f = 1$

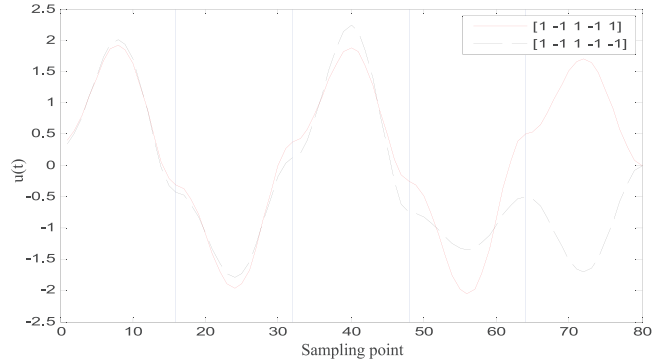


FIGURE 3 The difference in the past baseband signal caused by the predicted future 1 bit

through the channel modulation to yield the signal to be transmitted in the physical wireless channel. After the wireless channel transmission, at the receiver, the received signal undergoes the channel demodulation to remove the channel modulation frequency and to obtain the received baseband signal, which passes through the matched filter and is sampled at the same oversampling rate as that of the shape forming filter. Here, the oversampling rate is $n_{\text{samp}} = 16$.

From Figure 1, the matched filter output is fed into the CNN, the CNN output is the information symbols including one future and two currently received symbols, which are described in detail in Section 3. In the following, we explain the basic principle of the future symbol predictability.

The baseband signal of the current information symbol is affected by both the past information symbols and the future information symbols. This fact makes it possible for us to use the (past) received signal to predict the future symbols. In Figure 3, the past four symbols are same, that is, $[1, -1, 1, -1]$. However, the difference in the baseband signal corresponding to past symbols caused by future 1 bit (1 or -1) difference can be observed from the waveform difference corresponding to the past four symbols.

From Figure 3, we draw two conclusions: first, importantly, the future symbol (bit) difference causes the current waveform difference, which is perceivable by the feature collection property of the CNN; second, the further the future symbol is, the less influence it has on the past waveform, which could also be obtained from Equation (2) and Figure 2. These two points are very important for our work. First, it is possible for us to use the current waveform to predict the future symbols because different future symbols cause the difference on the current waveform. Second, it is easy for us to use not very long past waveform to predict near future symbols because the further in the future the symbol is, the less effect it has on the past waveform. This point reduces the data used for prediction, which in turn reduces the computation requirement.

3 | INTER-SYSMBOL INTERFERENCE RESISTANCE PERFORMANCE ANALYSIS

As described in [11, 12], the ISI in CBWCS is given by

$$\theta_n = \sum_{l=0}^{L-1} \sum_{\substack{i \neq 0 \\ i=-\infty}}^{i=\infty} s_{n+i} C_{l,i} = I_{\text{past}} + I_{\text{future}} = I, \tag{3}$$

where L is the number of paths, $s_{n+i}(i < 0)$ are the past transmitted symbols and $s_{n+i}(i > 0)$ are the future symbols to be transmitted. θ_n is the optimal threshold to decode the current received symbol, which is the sum of I_{past} and I_{future}, where

$$I_{\text{past}} = \sum_{l=0}^{L-1} \sum_{i=-\infty}^{i=-1} s_{n+i} C_{l,i}$$

is the ISI caused by past symbols and

$$I_{\text{future}} = \sum_{l=0}^{L-1} \sum_{i=1}^{i=\infty} s_{n+i} C_{l,i}$$

is the ISI caused by future symbols. $C_{l,i}$ can be calculated as follows:

$$C_{l,i} = \begin{cases} c_1, & \text{for}\left(\left|\tau_l + i/f\right| \geq 1/f\right) \\ c_2, & \text{for}\left(0 \leq \left|\tau_l + i/f\right| < 1/f\right) \end{cases}, \tag{4}$$

where

$$c_1 = \alpha_l e^{-\beta|\tau_l + \frac{i}{f}|} \frac{(\omega^2 - 3\beta^2)f}{4\beta(\omega^2 + \beta^2)} \times \left(2 - e^{-\frac{\beta}{f}} - e^{\frac{\beta}{f}}\right) \left(\frac{(\omega^2 - 3\beta^2)f}{4\beta(\omega^2 + \beta^2)} \cos(\omega\tau_l) + \frac{(3\omega^2 - \beta^2)f}{4\omega(\omega^2 + \beta^2)} \sin(\omega\tau_l)\right),$$

and

$$c_2 = \alpha_l \left\{ \frac{(\omega^2 - 3\beta^2)f}{4\beta(\omega^2+\beta^2)} \left(e^{-\beta|\tau_l + \frac{i}{f}|}(2 - e^{-\frac{\beta}{f}}) - e^{\beta|\tau_l+\frac{i}{f}|}e^{-\frac{\beta}{f}} \right) \times \cos(\omega\tau_l) + \frac{(3\omega^2-\beta^2)f}{4\omega(\omega^2+\beta^2)} \times \left(e^{-\beta|\tau_l+\frac{i}{f}|}(2 - e^{-\frac{\beta}{f}}) + e^{\beta|\tau_l+\frac{i}{f}|}e^{-\frac{\beta}{f}} \right) \times \sin(\omega\tau_l) + 1 - |\tau_l f + i| \right\}, \alpha_l$$

and τ_l are the channel parameters obtained by the least squares (LS) channel parameter estimation method [25]. From Equations (3) and (4), the optimal threshold is determined by three factors, namely, the past received information symbols (bits), the future symbols, which have not been received yet at the current time, and the channel parameters needed in Equation (4). The past received symbols, $s_i (i < 0)$, have been decoded, which are available at the current time. However, the future information symbols, $s_i (i > 0)$, which have not been received yet and are undetermined at the current time. Therefore, the existing technique, using the threshold determined by the past information symbols, that is, I_{past}, decodes the information bit to achieve sub-optimal performance compared to that using the optimal threshold, that is, the sum of I_{future} and I_{past}. In spite of using I_{past}, it still achieves significantly better performance compared to the conventional method, which uses 0 as the decoding threshold together with channel equalization [11, 12]. However, it does leave space for further improvement.

This work further improves the performance using CNN to predict the future symbols as the block with red border in Figure 1. This point is one of the main contributions of this work. Thus, a more accurate decoding threshold is obtained and used to improve the BER performance.

From [11], we learn that the BER using the optimal threshold $\theta_n = I$ for decoding s_n is

$$p(\text{error}|\theta_n = I) = \frac{1}{2} erfc\left(\frac{P}{\sqrt{2\sigma_W^2}} \right), \tag{5}$$

where $erfc(\cdot)$ is the complementary error function, which is defined as $erfc(\cdot) = 1 - erf(\cdot)$, where $erf(\cdot)$ is the error function which expresses the error probability. $P = \sum_{l=0}^{L-1} C_{l,0}$ is the sum of the multipath power for s_n, σ_W^2 is the variance of W, where W is Gaussian noise with zero mean. $P^2/(2\sigma_W^2)$ is the signal-to-noise ratio (SNR) of the filtered signal, The optimal threshold I given in Equation (3) contains both past and future symbols. Equation (5) is the optimal threshold provided that I_{future} is correct. But in practice, the future symbols are unknown and I cannot be obtained. In our method, the future symbols are predicted by CNN. So the BER for decoding s_n should be calculated using Equation (6):

$$p\{\text{error}|\theta_n = I_{\text{past}} + I_{\text{future}}\} = \frac{e^{\beta/f} - 1}{4|K|} \int_{-\frac{|K|}{e^{\beta/f}-1}}^{\frac{|K|}{e^{\beta/f}-1}} erfc\left(\frac{P + I_{\text{future}}}{\sqrt{2\sigma_W^2}} \right) d(I_{\text{future}}), \tag{6}$$

where $K = \sum_{l=0}^{L-1} \alpha_l (2 - e^{-\beta/f} - e^{\beta/f}) e^{-\beta\tau_l} (A\cos(\omega\tau_l) + B\sin(\omega\tau_l))$ depends on the channel parameters α_l and τ_l, d represents integral variable, and I_{future} represents the ISI from future symbols, which can be calculated by

$$I_{\text{future}} = K \sum_{i=1}^{\infty} s_m e^{-\beta m}. \tag{7}$$

By dividing the future symbols into two parts: three symbols predicted by CNN (s_1, s_2, and s_3) and others (s_4 to s_∞); I_{future} can be calculated by adding these two parts. Using p to represent the probability function, and assuming the symbols '−1' and '+1' have equal possibility, we conclude that $p(s_m = 1) = p(s_m = -1) = \frac{1}{2}$. Because I_{future} is an uniform distribution in the range given by $[-|\frac{Ke^{-\beta 3}}{e^{\beta}-1}|, |\frac{Ke^{-\beta 3}}{e^{\beta}-1}|]$, the effect of $I_{4\text{ to }\infty}$ can be assumed to be close to 0, so $I_{\text{future}} \approx I_{1\text{ to }3}$, where $I_{1\text{ to }3}$ represents the ISI from the first future three symbols, and $I_{4\text{ to }\infty}$ represents the ISI from the future symbols starting from the fourth future symbol. We use p_n to represent the error rate of s_n ($n = 1, 2, 3$) that is predicted by CNN because each of future symbols, s_1, s_2, s_3, has two possibilities: being predicted correctly and being predicted wrongly. And if s_1, s_2, s_3 are all predicted wrongly, the ISI is the most serious, it can be calculated by

$$I_b = 2K\left(p_1 e^{-\beta 1} + p_2 e^{-\beta 2} + p_3 e^{-\beta 3}\right). \tag{8}$$

If s_1, s_2, s_3 are predicted correctly, it can be calculated by

$$I_a = K\left[\left(1 - p_1\right)e^{-\beta 1} + \left(1 - p_2\right)e^{-\beta 2} + \left(1 - p_3\right)e^{-\beta 3}\right]. \tag{9}$$

So I_a and I_b are the lower and upper limits in Equation (6), respectively. The BER for decoding s_n should be calculated by

$$\begin{aligned} p\{\text{error}|\theta_n = I_{\text{past}} + I_{1\text{ to }3}\} &= \frac{e^{\beta/f} - 1}{4|K|} \int_{I_a}^{I_b} erfc\left(\frac{P + I_{\text{future}}}{\sqrt{2\sigma_W^2}} \right) d(I_{\text{future}}) \\ &= \frac{e^{\beta/f} - 1}{4|K|} \cdot \sqrt{2\sigma_W^2} \cdot \left[t_1 \cdot erfc(t_1) - e^{-t_1^2}/\sqrt{\pi} \right. \\ &\quad \left. - t_2 \cdot erfc(t_2) + e^{-t_2^2}/\sqrt{\pi} \right], \end{aligned} \tag{10}$$

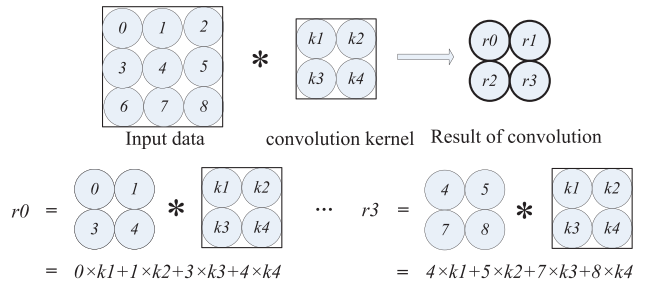


FIGURE 4 Convolution example of a 3 × 3 input data with a 2 × 2 convolution kernel, where ∗ represents convolution

where $t_1 = \frac{P+I_a}{\sqrt{2\sigma_W^2}}$, $t_2 = \frac{P+I_b}{\sqrt{2\sigma_W^2}}$. Equation (10) gives the analytical BER using the predicted future symbols. From Equations (8)–(10), we learn that the error probability of the CNN prediction only contributes to the final BER in a partial way. Therefore, the BER using the threshold calculated by $\theta_n = I_{\text{past}} + I_{1 \text{ to } 3}$ is much lower than the BER using CNN direct prediction, which is also shown in Figure 9 in Section 4. This is the reason why we do not use the CNN to directly decode the symbols.

4 | SYMBOL PREDICTION USING CONVOLUTIONAL NEURAL NETWORK

4.1 | Basics of convolutional neural network and training principle

CNN was first conceived in 1962 by Hubel and Wiesel in connection with the cat visual cortex, which effectively reduced the learning complexity [26]. Based on the result in [27], Yann LeCun et al. designed an error gradient-based algorithm to train CNN, which showed superior performance in pattern recognition compared to other methods [28]. This result led to growing interest in the research on CNN [29–32]. CNN consists of three basic layers: convolutional layer, pooling layer and fully connected (output) layer.

The convolutional layer is used to extract the features of the input data. Using a 3 × 3 input data as an example, if the convolution kernel size is 2 × 2, the weights of the convolution kernel are $k1$, $k2$, $k3$, $k4$. Then the convolution process, using the given convolution kernel and input data, is pictorially shown in Figure 4. As seen from Figure 4, for this 3 × 3 input data, we use a 2 × 2 convolution kernel to convolve. By setting the slide stride size as 1, that is, by sliding the 2 × 2 window to the right or down by one element at a time, the convolution result is obtained. Different convolution kernels extract different features of the input data.

The input data to the pooling layer are the result of the previous convolutional layer. Figure 5 is an example of a 2 × 2 window size average pooling. In Figure 5, $[r0, r1; r2, r3]$ is the result of former convolution layer, after the average pooling, we get the result of pooling layer as given by $p0 = (r0 + r1 + r2 + r3)/4$. The number of nodes (or neurons) in the network has changed from 2 × 2 to 1, which compresses the data while extracting features by calculating the average value

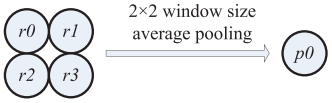


FIGURE 5 Average pooling result for 2 × 2 window

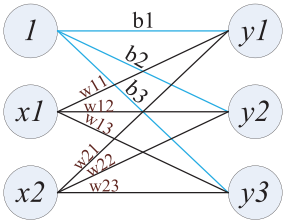


FIGURE 6 Structure of the fully connected layer

of several nodes. The pooling operation helps to improve the robustness against data perturbation. The pooling layer acts as a further feature extraction, it reduces the risk of over-fitting, and also reduces the dimension of the feature map, improving the robustness of feature extraction. Generally, the high-level features can be extracted by stacking the convolutional layers and the pooling layers several times.

The fully connected layer has all-to-all connections with the nodes from the previous pooling layer. Figure 6 shows an example of the fully connected layer. All nodes in the former layer, x_n, are connected to all nodes in the next layer, y_m, where b_m is the bias of y_m and w_{nm} is the weight from x_n to y_m. The output calculation formula is given in Equation (11). So this part acts as a classifier, which is referred as the fully connected layer with respect to the previous layers [33].

$$\begin{cases} y_1 = b_1 \times 1 + w_{11} \times x_1 + w_{21} \times x_2 \\ \cdots \\ y_3 = b_3 \times 1 + w_{13} \times x_1 + w_{23} \times x_2 \end{cases}. \tag{11}$$

The training process of the CNN consists of two stages. The first stage is a feed-forward calculation process, where data are transmitted from input to output layers via intermediate layers. The second stage is the error BP from the output layer to the input layer when the current output is not equal to the expectation [34]. When the CNN output does not match the expectation, the error between the current output and the expected value is propagated layer by layer back in order to calculate the error contribution to the parameters at each layer, and then the weights of each layer and the bias are updated accordingly. After updating, the input and the new weights are used to calculate the new output. If the new output matches the expectation, the training is completed. But, if not, the error BP and weight updating are repeated again until the output is matched with the preset tolerance. The structure of the CNN is shown in Figure A1.

4.2 | Training data collection

In our CBWCS, the data are transmitted frame by frame. The channel parameters are assumed to be unchanged within one frame but are varied from one frame to the next, so the chan-

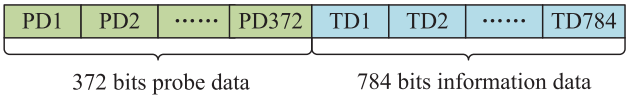


FIGURE 7 Data frame structure

nel can be treated as time-invariant during one frame. One data frame consists of the $(2^6 - 2) \times 6 = 372$ bits probe data and the 784 bits information data, as shown in Figure 7.

The probe data are generated by the shape forming filter using the training data, which is used for clock synchronisation, frequency bias compensation and channel parameter identification, as done in the conventional wireless communication system. In our proposed method, the probe data is also used for CNN training. From Figure 3, we gather that the further the future bits are, the less effect they have on the current waveform. According to the base function waveform given in Figure 2, we also learn that the one future bit difference mainly causes a significant difference in the waveform corresponding to past 4 bits, and further the future bits are, they have less influence on the first future bit. Based on these facts, we only include the future second bit for training purpose. Thus, we have 6 bits with all possible combination (from $[-1\ -1\ -1\ -1\ -1\ -1]$ to [1 1 1 1 1 1]) for training the data. By removing two training data sets, $[-1\ -1\ -1\ -1\ -1\ -1]$ and [1 1 1 1 1 1], (which is not allowed in our configuration), we obtain $(2^6 - 2) \times 6 = 372$ data to train the CNN. The length of the information data is 784 bits in our work, as shown in Figure 7, which could be even larger according to the time-varying properties of the channel. The 372 training data waveform is transmitted and received at the receiver, which has known symbol information and can be used for CNN training.

4.3 | The proposed convolutional neural network method

4.3.1 | The input of convolutional neural network

From Figure 3, we learn that different future bits cause the difference on the waveform mainly in the past 4 symbols; the further the future symbol is, the less influence it has on the past waveform, which makes it possible for us to use a limited past waveform, say past 4 bits, to predict the near future symbols. Thus, the matched filter output waveform corresponding to the 4 past symbols is sampled at the oversampling rate equal to 16, and the sampling points are used as the input of the CNN. Therefore, $16 \times 4 = 64$ sampling points are derived. Since the input of the CNN should be a matrix, we reshape the 64 sampling points as an 8×8 data matrix as the input to the CNN.

4.3.2 | The structure and parameters of convolutional neural network

For the CNN layers selection, the trade-off is between the accuracy and the training cost. Generally speaking, the more layers, the higher is the accuracy and the computation cost. In our case, the input image is an 8×8 data matrix, it has a small input size compared to the commonly larger image processing applications. Therefore, it is not assigned too many layers. In terms of the prediction accuracy, five layers are sufficient in our work. The use of less (three) layers does not reach the expected prediction accuracy. By increasing the number of layers causes the computation cost to increase, but, at the same time, the accuracy does not increase significantly. Therefore, in our application, we select five layers in the CNN, as shown in Figure A1.

TABLE 1 The structure and parameters of the CNN

Layer	Type	Units	x	y	Receptive field x	Receptive field y
1	Convolutional	6	6	6	3	3
2	Sub-sampling	6	3	3	2	2
3	Convolutional	12	2	2	2	2
4	Sub-sampling	12	1	1	2	2
5	Fully connected	3	1	1	1	1

As depicted in Figure A1, the CNN contains five layers, the first four are convolutional and the remaining one is fully connected. The first layer is the first convolutional layer with six feature maps of size 6×6, each unit of each feature map is connected to a 3×3 neighbourhood of the input. The second layer is the first sub-sampling layer with six feature maps of size 3×3, each unit in each feature map is connected to a 2×2 neighbourhood in the corresponding feature map in the previous layer. The third layer is the second convolutional layer with 12 feature maps of size 2×2, and the fourth layer is the second sub-sampling layer with 12 feature maps of size 1. Then 13 features are organised as a vector, including a bias unit. The vector is fully connected to the fifth layer (output layer), which is composed of three units. The structure of CNN used in our work is modified based on the basic architecture of LeNet-5 CNN model to obtain a trade-off between time-cost and the BER performance. More details about the CNN structure and parameters used here are shown in Table 1.

As for the other hyper-parameters of the CNN, it is also a trade-off problem, which can be decided according to the general guidance and the application requirements. For example, the batch size of training data is generally set from 64 to 512. In our application, the training data size is 62, then, the batch size is set as 62. The learning rate could be a fixed value or a gradually decreasing value in the learning process. We have tested different values to choose the best choice in the sense of fast convergence. Finally, it is set to be 1.0 and it is kept constant in the training process for our application. The training epoch is decided by the tolerance setting, which is determined by the BER obtained using the tolerance.

4.3.3 | The output of convolutional neural network

Three output data can be obtained through convolution layer(s), pooling layer(s), and a fully connected layer, which is given in

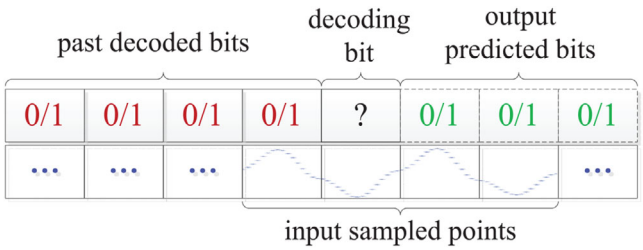


FIGURE 8 The input and output of the convolutional neural network (CNN) and the position of decoding bit

Figure A1. Three outputs are the predicted bits, as shown by last three 0/1 in the first row of Figure 8, where 0/1 means that the bit could be 0 or 1 according to the transmission data. The input data of the CNN are the 64 sampling points from the matched filter (in Figure 1) output waveform corresponding to 4 bits, as indicated by the second row of Figure 8, where each bit is sampled by 16 points. The 64 sampled points in Figure 8 are a row vector with 64 entries, which are reformulated as a matrix with eight rows and eight columns, and used as the input to the CNN, as shown in Figure A1.

In the training stage of the CNN, the sampling points corresponding to 372 bits waveform are used to train the CNN, repeatedly, as indicated by the flowchart in Figure 10, where the tolerance is 0.01 in our work. After training, the CNN is used to predict decoded symbols, using the remaining data in the frame. The predicted 3 bits are used in a manner as shown by Figure 8. The decoding threshold of 1 bit, before these predicted 3 bits, as indicated in Figure 8 by '?', is calculated using

$$\theta_n = I_{\text{past}} + I_{\text{future}} \approx \sum_{l=0}^{L-1}\sum_{i=-1}^{i=-4} s_{n+i}C_{l,i} + \sum_{l=0}^{L-1}\sum_{i=1}^{i=3} s_{n+i}C_{l,i}. \quad (12)$$

Comparing Equation (12) to Equation (3), we know that, in Equation (12) we only use the four past symbols and three future symbols to remove the ISI effect. This operation is reasonable because even past symbols have little effect on the current symbol, as shown in Figure 2. Similarly, the future symbols also have similar function as in Equation (12).

Discussion 1: Using Equation (12) to decode the symbol is more accurate than the methods in [11, 12], because only past four symbol effects are used in [11, 12], while, the future three symbols are included in the threshold calculation in the proposed method. The threshold in Equation (12) is also more accurate than the method in [24], where only one future symbol was used rather than the three used here.

Discussion 2: One question might be raised here, namely, why not directly use the predicted symbol as the decoding result? The answer in fact has been given in the analysis in Section 3. The decoding error rate of using Equation (12) is theoretically considerably lower than that of using the direct predicted result. To test the prediction effectiveness using the aforementioned data and training method, the CNN is trained. In the same channel parameters, 1,000,000 bits are transmitted, which

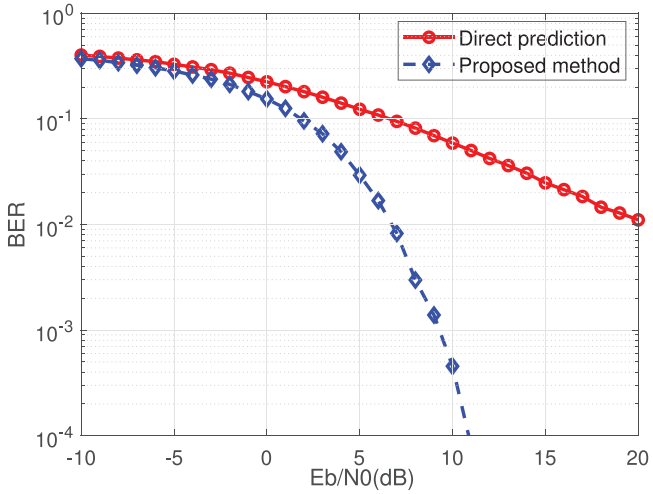


FIGURE 9 The bit error rate (BER) comparison result between the direct decoding method using convolutional neural network (CNN) prediction and the method proposed here under three paths with $\tau_0 = 0, \tau_1 = 1, \tau_2 = 2$, $\gamma = 0.6$ *Note*: The blue dashed line with diamond markers is the BER result using our proposed method, the red solid line with circle markers is the BER result using CNN direct prediction method

is a uniform distribution. Then the matched filter output is sampled and used to test the effectiveness of the method under different signal-to-noise ratio (SNR) cases. Here, we test two methods, including the proposed method described above and the method using the CNN to predict the symbol directly. The results in Figure 9 show that the error rate using the CNN to predict the symbol is about 5% in the case of Eb/N0 equals to 10 dB. However, if we use the predicted result as parameters in Equation (12), then, the calculated threshold is used to decode the information bit, its error rate is about 0.05% in the same Eb/N0. In each Eb/N0, we generate 1,000,000 bits of the signal, then we count the number of error decoded, and divide the number of error decoded by the total number of generated symbols, we get the BER at each Eb/N0. Every bit has much less contribution to the right decoding threshold than that when decoding directly, as described in the Section 3.

4.4 | The training of the convolutional neural network

The CNN training is given in Figure 10. For the first frame of the transmitted data, the initial weight of CNN is set randomly. From the second frame, the initial weight is the weight trained for the last frame, it just needs to slightly update the weight to adapt to the slight changed channel situation (because of the short time duration of one frame transmission), to reduce the training time.

The training data are divided into 62 matrixes input for the CNN in our work. We trained CNN using stochastic gradient descent with a batch size of 62 data matrixes, and the learning rate is initialised at 1.0. For the first frame of data, we initialised the weights in each layer from a zero-mean Gaussian distribution with standard deviation 0.01, and the neuron bias

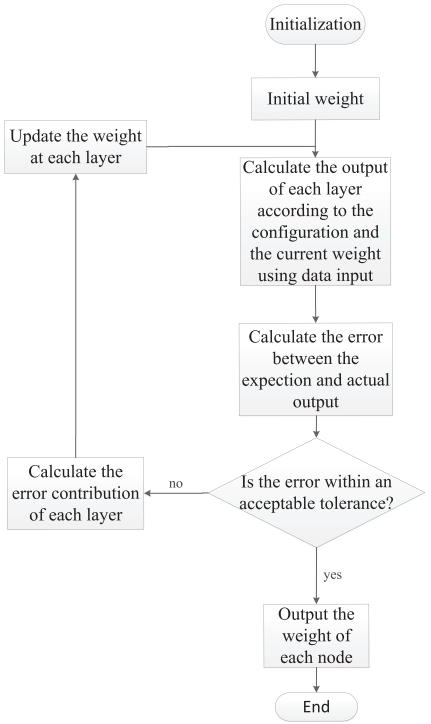


FIGURE 10 Flow chart of the convolutional neural network (CNN) training process

in the fifth fully connected layer is initialised as 1. We train the network until the training data error reaches the tolerance. The acceptable tolerance error is set to be 0.01, which effectively guarantees the BER performance improvement in our work. The weights derived for the last data frame are used as initial values of the CNN weights for the current data frame training.

The training process includes two phases, the forward propagation phase and backward propagation stage.

(1) Forward propagation, the input data matrix is firstly sent to the CNN, and the actual output is calculated by the network, which is also the operation for the prediction process.
(2) Backward propagation, the prediction error e between the actual output and the corresponding expected output is calculated, then the gradient for each weight of each layer is calculated, and the weights are updated using the gradient descent method.

The training times are quite different for the first frame and for the remaining ones. Generally speaking, the training time for the first frame is much larger than that of the remaining frames. The first frame training means that the CNN is from the random initialisation, that is, the weights are randomly initialised. From this state to the suitable weights after training, it takes long time to train, like 5000 rounds. One point to be emphasised is that, for every frame, the training is conducted if the tolerance is not satisfied, whether it is the first frame. This is a very conservative setting to test the performance. In fact, for the slow time-varying channel, this can be extended to many more frames or larger frame data size, where the slow time-varying means that the channel parameters do not change dramatically in the subsequent data frames. In natural environment, if there is no relative motion, the time invariance can be guaranteed for electromagnetic radio wave transmission. After the first frame training, the network weights contain the basic structural information about the problem. From the second frame, it takes on average about 10 rounds for the training to adjust the weights to meet the setting tolerance. This is calculated by averaging the training epochs (batch operations) of 1000 data frames in the natural environment without relative motion after the first frame training. For the sparse communication, if the transmitter and receiver do not move relatively to each other, it does not affect the training rounds much, as we also observed from the practical experiments. We have tested the case under which, after the CNN is trained for the first scenario, the receiver is moved to other places far from the original one, the training of the first frame for the new location is on average 500 rounds, less than the random initialisation that we started with. According to the above analysis, this work is applicable for the mobile wireless communication systems.

4.5 | Computation requirement

To give a general idea of the computational complexity and time requirement of the proposed method, we define l as the index of a convolutional layer, and D as the depth (number of convolutional layers). N_l is the number of convolution kernels (also known as 'width') in the lth layer. K_l is the spatial size (length) of the convolution kernel. M_l is the spatial size of the output feature map. The number of floating-point operations (FLOPs) is used to evaluate the computational requirement. The total time required of all convolutional layers [35] is the order of

$$O\left(\sum_{l=1}^{D} N_{l-1} \cdot K_l^2 \cdot N_l \cdot M_l^2\right). \tag{13}$$

In our method, the inner area of the 8×8 input matrix is scanned for a 3×3 local receptive field. A total of 6×6 local receptive fields are scanned. The 6×6 results of the sigmoid function are then 2×2 sub-sampled with an average-pooling function, and become a 3×3 feature map. Therefore, there are $6 \times 6 \times 6$ neurons in the first convolutional layer (layer 1), each with 3×3 inputs (synapses), that is, $3 \times 3 \times 6 \times 6 \times 6 = 1944$ multiplications are required in computing the weighted inputs of neurons in layer 1. The inner areas of six feature maps, which are the output of the previous layers (layer 2), are scanned for 2×2 local receptive fields. The local receptive fields produced from the same position of the feature maps become the common inputs to 12 neurons, each with six kernels. Therefore, in layer 3, there are $2 \times 2 \times 12$ neurons each with $2 \times 2 \times 6$ synapses, which requires at least $2 \times 2 \times 6 \times 2 \times 2 \times 12 = 1152$ multiplications. The 2×2 sigmoid outputs are sub-sampled with an average-pooling function resulting in a 1×1 feature map in

TABLE 2 Complexity of the proposed method

Cost	FLOPs
Training for one frame	$583{,}110N_e$
One prediction	3135

layer 4. The produced feature maps are merged into a feature vector with 13 elements that includes a bias term, and the elements of the feature vector are fully connected to three output neurons. The neuron with the higher output value than a given threshold is picked for '1', otherwise '−1'. Because this layer has 3 neurons, each with 13 synapses, $13 \times 3 = 39$ multiplications are required. Totally, 3135 multiplications are required for one data matrix input.

This time requirement applies to both training and prediction time. The training time per data matrix is roughly three times of the prediction time per data matrix (one for forward propagation and two for backward propagation). In the training process, the batch size is 62, thus, $62 \times 3 \times 3135 = 583,110$ multiplications are required for one epoch. In the prediction, just 3135 multiplications are required for the three future symbols prediction used for one symbol decoding.

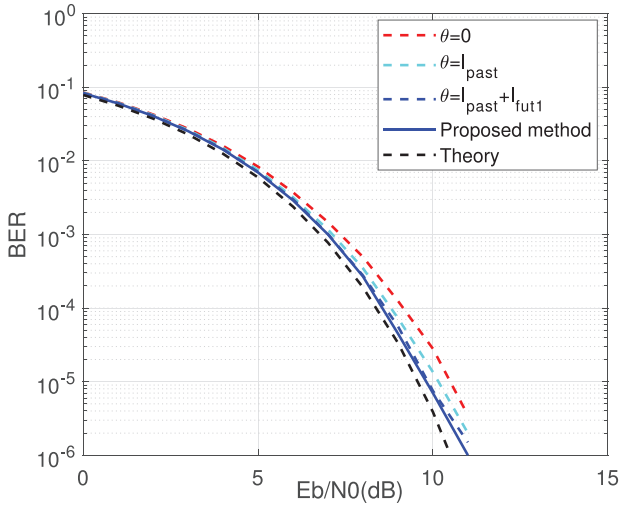
We assume the number of epochs to be N_e , the complexity is shown in Table 2.

This paper aims to present and test the novel idea, the real-time application is not the main concern of this work. This calculation requirement seems large, but with the fast development of high-performance microprocessors and hardware, such as FPGA, its real-time application should be feasible in the near future. For the CNN computation in FPGA, there is some reported example in [36], where VC707 FPGA chip is used to achieve the peak performance of 61.62 GFLOPS under 100 MHz working frequency using much more complex configuration than that used here. As the complexity analysis and the data in Table 2, in the practical application, one can assume that the cost for first frame can be neglected and a 50 GFLOPS performance of the FPGA. Then, 10 rounds for training are less than 100 μs, 1 prediction is less than 0.06 μs, which allows quite considerable data frame transmission rate.

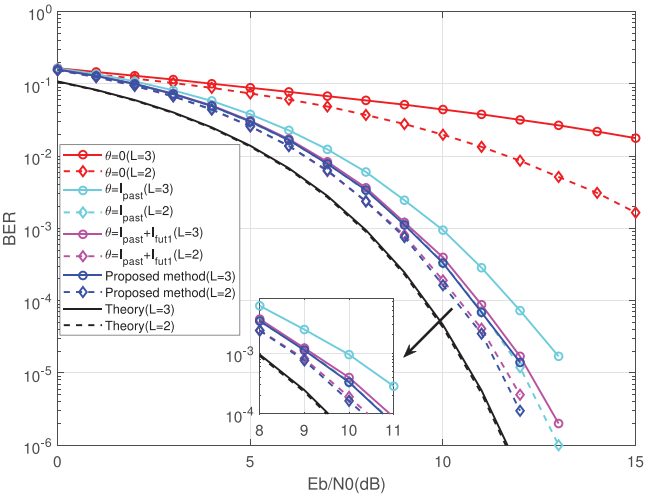
5 | SIMULATION RESULTS

In our simulations, the MATLAB R2018b is used, the wireless channel is modelled as $h(t) = \sum_{l=0}^{L-1} \alpha_l \delta(t - \tau_l)$, where $\alpha_l = e^{-\gamma\tau_l}$ is the attenuation of path l, τ_l is the time delay of path l. L is the multipath number. $\delta(\cdot)$ is the Dirac delta function [37]. Here the base frequency is $f = 1Hz$ and the parameter $\beta = \ln 2$. The BER comparison results under single path and different multipaths are plotted in Figure 11.

Figure 11(a) shows the BER using different decoding thresholds versus bit energy-to-noise density (Eb/N0) under single path with the channel parameters $L = 1$, $\tau_0 = 0$. $\theta = 0$ indicates the decoding threshold that does not consider ISI, $\theta = I_{\text{past}}$ indicates the decoding threshold that only considers



(a)



(b)

FIGURE 11 Bit error rate (BER) comparison results using different decoding thresholds (a) BER results under single path, red dashed line is the BER curve using $\theta = 0$, cyan dashed line is the BER curve using $\theta = I_{\text{past}}$ in [11, 12], blue dashed line is the BER curve using $\theta = I_{\text{past}} + I_{\text{fut1}}$ in [24], blue solid line is the BER curve using the proposed method here, black dashed line is the theoretical BER; (b) BER results under different multipaths, red lines are the results using $\theta = 0$, cyan lines are the results using $\theta = I_{\text{past}}$ in [11, 12], magenta lines are the results using $\theta = I_{\text{past}} + I_{\text{fut1}}$ in [24], blue lines are the results using the proposed method here, the dashed lines with diamond markers are the BER curves corresponding to $L = 2$, the solid lines with circle markers are the BER curves corresponding to $L = 3$, the black dashed line is the theoretical BER corresponding to $L = 2$, and the black solid line is the theoretical BER corresponding to $L = 3$

ISI caused by the past four symbols, as done in [11, 12]. The result from the method in [24] is also given in the same figure, where $\theta = I_{\text{past}} + I_{\text{fut1}}$. Our proposed method considers the ISI caused by both past four symbols and future three symbols predicted by the CNN. Their legend indicates the theoretical result by removing all ISI. It can be seen from Figure 11(a) that the

proposed method has the lowest BER, and it is close to that of the theoretically optimum by removing most ISI.

Figure 11(b) shows the BER using different decoding thresholds versus Eb/N0 under different multipath channels, whose parameters are $\tau_0 = 0$, $\tau_1 = 1$, $\gamma = 0.6$ for $L = 2$ and $\tau_0 = 0$, $\tau_1 = 1$, $\tau_2 = 2$, $\gamma = 0.6$ for $L = 3$. Different methods are discriminated by the legend given in the figure. It can be seen from Figure 11(b) that the proposed method has the lowest BER. We can also see that the decoding threshold without considering ISI becomes even worse compared to the single path case, since the ISI in multipath case becomes stronger compared to the single path case.

6 | EXPERIMENTAL RESULTS

In the experiment, two Wireless Open-Access Research Platform (WARP) with Virtex-6 LX240T FPGA are used [38]. The Xilinx Virtex-6 LX240T FPGA is provided for digital signal processing, two MAX2829 RF chips are used to support dual channel, and a 2.4/5 GHz dual-band transceiver. The maximum transmission power is 20 dBm using the dual-band power amplifier, the 12-bit low power analog/digital converter AD9963 is used to provide two ADC channels with sample rates of 100 MSPS and two DAC channels with sample rates of 170 MSPS. Two 10/100/1000 M Ethernet interfaces (Marvell 88E1121R) are used to carry out the high-speed digital signal exchange with the personal computer. Each WARP node has two radio antennas, which are referred as radio frequency antenna A and radio frequency antenna B; in our experiments only the RFA is used. Here $n_{\text{samp}} = 16$, the sampling frequency is 40 MHz, and the carrier frequency is 2.4 GHz.

Our experimental data are generated in MATLAB. The data are sent to WARP using TCP/IP port and then sent out from the transmitter and received by the receiver (as seen in Figures 12(a) and 13(a)). All the data received by TX are sent to MATLAB through TCP/IP port for unified data processing. In order to verify the effectiveness of the method, we send 128 frames of data to each transmitted power, which corresponds to $128 \times 784 = 100,352$ bits test.

First, the transmitter and the receiver are placed in the laboratory to test the single path case, the photo of the test scenario is given in Figure 12(a), where TX represents the transmitter and RX represents the receiver. Two WARPs are very close to simulate single path transmission. In our experiment, the bandwidth is about 4 MHz, thus, the delay resolution of the system is 0.25 μs. The channel parameters, that is, delay and fading, are estimated by the LS algorithm, and the estimated channel parameters in this test scenario in Figure 12(a) are shown in Figure 12(b). There is one main path with channel fading 1, the other paths are relatively weak (i.e. the channel fading is less than 0.1). Figure 14(a) shows the BER for different methods versus the transmission power. In WARP, the transmission power can be adjusted by the parameters TX_RF_Gain and TX_BB_Gain to simulate the SNR variation. It can be seen from Figure 14(a) that the proposed method has the lowest BER, showing that the proposed method reduces the ISI in the case of single path channel (with noise) by predicting future symbols.

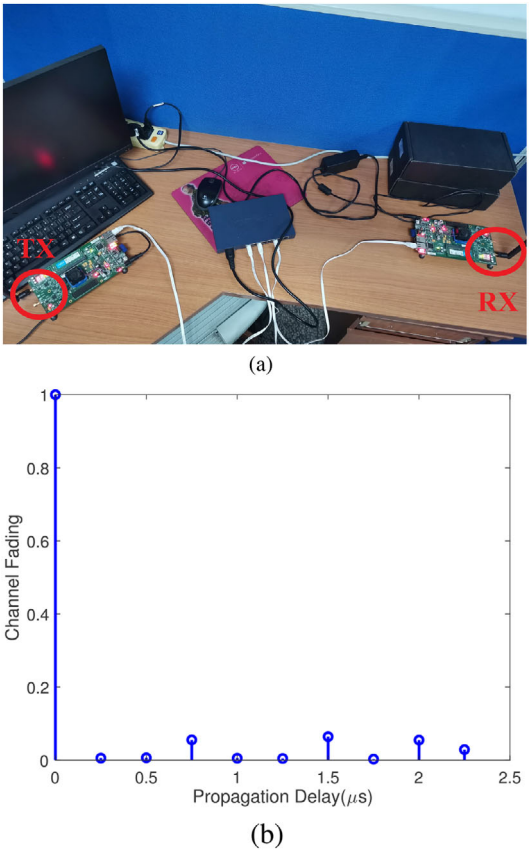


FIGURE 12 (a) The photo of laboratory test set-up for single path; (b) the normalised estimated channel parameters

We also tested experimentally the BER performance under the real multipath channel, where multipath is present as shown by the test scenario given in Figure 13(a). The transmitter is located near the tree, and the receiver is located at some distance. The estimated channel parameters are shown in Figure 13(b). There are two main paths identified by channel estimation with channel fading 1, 0.2193, respectively, and the delay of the two paths are 0 and 0.25 μs, respectively. We have verified that the channel does not vary during the transmission of one frame in our experiment [12]. Figure 14(b) shows BER for the different methods versus the different transmission power. It can be seen that the proposed method has the lowest BER in the case of the multipath channel. Compared with the single path case, the performance improvement is more significant since in the multipath case ISI is stronger.

From Figure 14, when the transmission power is very low, the BER performance of the proposed method is closer or lower compared to that without ISI, that is, $\theta = 0$. The reason for this phenomenon lies in that the channel information identification results are not accurate for a lower SNR case, which makes the accurate calculation of the decoding threshold more difficult. The low SNR also affects the future bit prediction accuracy. The decoding threshold $\theta = 0$ does not require any channel parameter information. In Figure 14(a), the improvement of the BER performance is less than that in Figure 14(b), which is reasonable, since the multipath causes serious ISI.

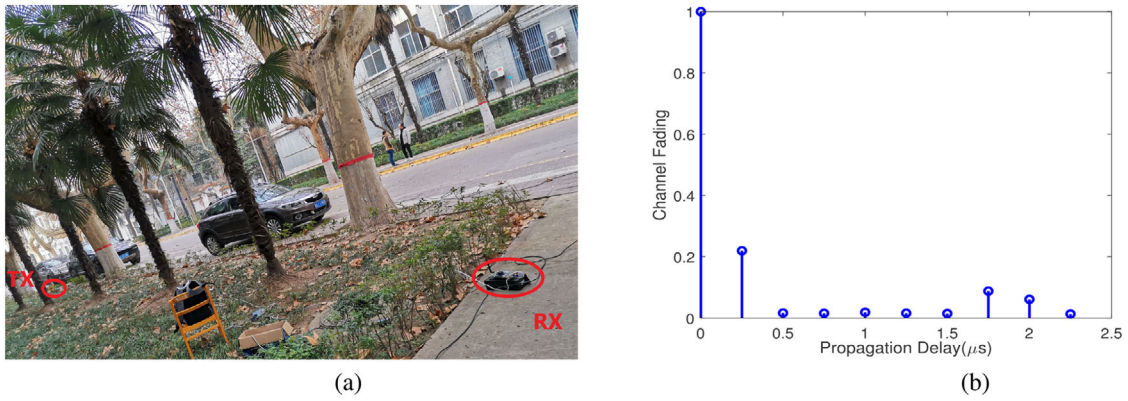


FIGURE 13 (a) The photo of test scenario; (b) the normalised estimated channel parameters

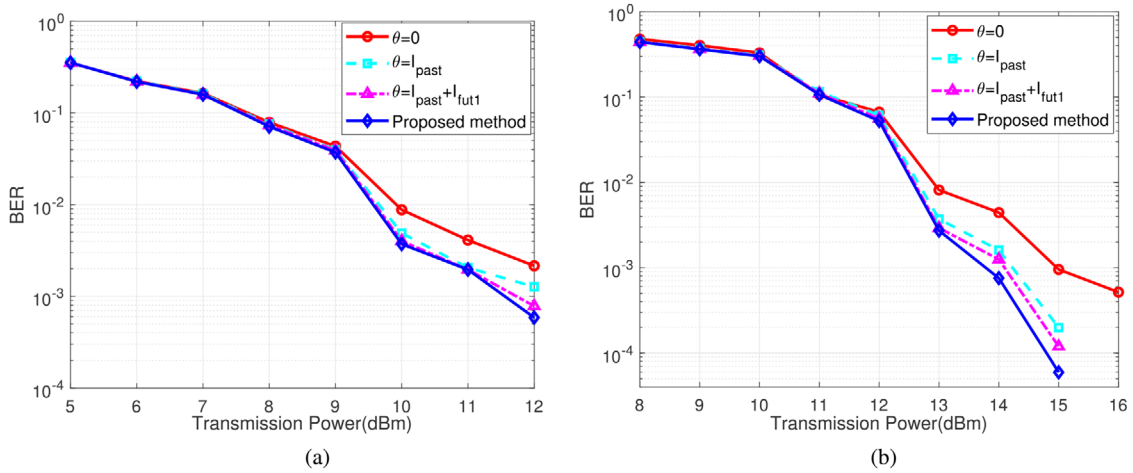


FIGURE 14 (a) Bit error rate (BER) comparison results for the experiment in single path *Note*: Blue solid line with diamond markers is the BER curve using our proposed method, magenta dash-dotted line with lower triangular markers is the BER curve using $\theta = I_{past} + I_{fut1}$ in [24], cyan dashed line with square markers is the BER curve using $\theta = I_{past}$ in [11, 12], and red solid line with circle markers is the BER curve using $\theta = 0$; (b) BER comparison results for the experiment in multipath channel. The different lines description is the same as those of subplot (a)

7 | CONCLUSION

This paper proposes a method based on a CNN to reduce the ISI in CBWCSs and to decrease the BER. The previous method can only eliminate the ISI caused by the past symbols because the future information symbols are unavailable. In order to further decrease the ISI caused by the future information symbols and enhance the performance of chaotic baseband wireless communication, ESN has been used in [24] to predict the waveform iteratively. The predicted waveform is used to decode the future 1 bit, and then used in calculating the decoding threshold including the future 1 bit effect. Different from the idea in [24], this work proposes a new AI-based method by using CNN and its powerful ability to learn of a mapping between the waveform sampling points and the future bits. By training, the CNN has the ability to recognise the features of the input data, and then predict the future symbols to be received based on the waveform corresponding to past symbols. Another difference between the method here and that in [24] lies in that the future three bits are used to calculate the more accurate decoding threshold in the proposed method. Extensive tests also show that, under different delay and attenuation conditions, a lower BER can be achieved using the proposed method. The experimental tests pave the way for practical applications of chaotic baseband wireless communication.

ACKNOWLEDGEMENTS

This work was supported in part by Shaanxi Provincial Special Support Program for Science and Technology Innovation Leader. Dr Bai was supported in part by China Postdoctoral Science Foundation Funded Project (2020M673349), and Open Research Fund from Shaanxi Key Laboratory of Complex System Control and Intelligent Information Processing (2020CP02).

ORCID

Hai-Peng Ren https://orcid.org/0000-0003-3834-5103

REFERENCES

1. Kline, R.: Cybernetics, automata studies, and the Dartmouth Ccnference on artificial intelligence. Ann. Hist. Comput. 33(4), 5–16 (2011)
2. Rao, P.V.S.: Fifth generation computers and artificial intelligence. IETE J. Res. 34(3), 159–165 (1988)
3. Zhao, D., et al.: Special issue on deep reinforcement learning and adaptive dynamic programming. IEEE Trans. Neural Netw. Learn. Syst. 29(6), 2038–2041 (2018)
4. Chow, P., et al.: The design of a SRAM-based field-programmable gate array – part II: circuit design and layout. IEEE Trans. Very Large Scale Integr. VLSI Syst. 7(3), 321–330 (1999)
5. Shiratori, N., et al.: Using artificial intelligence in communication system design. IEEE Softw. 9(1), 38–46 (1992)
6. Di Caro, G., Dorigo, M.: AntNet: Distributed stigmergetic control for communications networks. J. Artif. Intell. Res. 9, 317–365 (1999)
7. Han, Z., et al.: Artificial intelligence based handoff management for dense WLANs: a deep reinforcement learning approach. IEEE Access 7, 31688–31701 (2019)
8. Hayes, S., Grebogi, C., Ott, E.: Communicating with chaos. Phys. Rev. Lett. 70(20), 3031–3034 (1993)
9. Argyris, A., et al.: Chaos-based communications at high bit rates using commercial fibre-optic links. Nature 438, 343–346 (2005)
10. Ren, H.P., et al.: Wireless communication with chaos. Phys. Rev. Lett. 110(18), 184101 (2013)
11. Yao, J.L., et al.: Chaos-based wireless communication resisting multipath effects. Phys. Rev. E 96(3), 032226 (2017)
12. Yao, J.L., et al.: Experimental wireless communication using chaotic baseband waveform. IEEE Trans. Veh. Technol. 68(1), 578–591 (2018)
13. Bai, C., et al.: Experimental phase separation differential chaos shift keying wireless communication based on matched filter. IEEE Access 7, 25274–25287 (2019)
14. Makka, J., et al.: Reduction of inter-symbol interference using artificial neural network system in multicarrier OFDM system. Electric Electron Tech Open Acc J. 2(1), 20–23 (2018)
15. Waseem, A., Hossain, A.H.M.: MIMO channel equalization and symbol detection using multilayer neural network. Master Thesis, Blekinge Institute of Technology (2012)
16. Shaerbaf, S., Seyedin, S.A.: Nonlinear multiuser receiver for optimized chaos-based DSCDMA systems. Iran. J. Electr. Electron. Eng. 7(3), 149–160 (2011)
17. Chi, N., et al.: Gaussian kernel-aided deep neural network equalizer utilized in underwater PAM8 visible light communication system. Opt. Express 26(20), 26700–26712 (2018)
18. Feng, J.C., et al.: Reconstruction of chaotic signals with application to channel equalization in chaos-based communication systems. Int. J. Commun. Syst. 17, 217–232 (2004)
19. Jagannath, J., et al.: Machine learning for wireless communications in the Internet of things: a comprehensive survey. Ad Hoc Networks 93. https://doi.org/10.1016/j.adhoc.2019.101913 (2019)
20. Principe, J.C., et al.: Prediction of chaotic time series with neural networks and the issue of dynamic modeling. Int. J. Bifurcation Chaos 2(04), 989–996 (1992)
21. Maguire, L.P., et al.: Predicting a chaotic time series using a fuzzy neural network. Inf. Sci. 112(1–4), 125–136 (1998)
22. Mukherjee, S., et al.: Nonlinear prediction of chaotic time series using support vector machines. Proceedings of the 1997 IEEE Signal Processing Society Workshop, Amelia Island, FL, pp. 511–520 (1997)
23. Pathak, J., et al.: Model-free prediction of large spatiotemporally chaotic systems from data: a reservoir computing approach. Phys. Rev. Lett. 120(2), 024102 (2018)
24. Ren, H.P., et al.: Performance improvement of chaotic baseband wireless communication using echo state network. IEEE Trans. Commun. 68(10), 6525–6536
25. Barhumi, I., et al.: Optimal training design for MIMO OFDM systems in mobile wireless channels. IEEE Trans. Signal Process. 51(6), 1615–1624 (2003)
26. Hubel, D.H., Wiesel, T.N.: Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat. J. Neurophysiol. 28(2), 229–289 (1965)
27. Fukushima, K.: Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol. Cybern. 36(4), 193–202 (1980)
28. Lecun, Y., et al.: Gradient-based learning applied to document recognition. Proc. IEEE 86(11), 2278–2324 (1998)
29. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. European Conference on Computer Vision, Springer, Cham, pp. 818–833 (2014)
30. Szegedy, C., et al.: Going deeper with convolutions. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, pp. 1–9 (2015)
31. žbontar, J., LeCun, Y.: Stereo matching by training a convolutional neural network to compare image patches. Learn. Res. 17(1), 2287–2318 (2016)
32. Chen, Y.H., et al.: Eyeriss: an energy-efficient reconfigurable accelerator for deep convolutional neural networks. IEEE J. Solid-State Circuit 52(1), 127–138 (2016)
33. Guo, T., et al.: Simple convolutional neural network on image classification. 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA), Beijing, pp. 721–724 (2017)
34. Kim, J., Nguyen, A.D., Lee, S.: Deep CNN-based blind image quality predictor. IEEE Trans. Neural Netw. Learn. Syst. 30(1), 11–24 (2018)
35. He, K., Sun, J.: Convolutional neural networks at constrained time cost. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, pp. 5353–5360 (2015)
36. Zhang, C., et al.: Optimizing FPGA-based Accelerator Design for Deep Convolutional Neural Networks. Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, pp. 161–170 (2015)
37. Döttling, M., et al.: Radio technologies and concepts for IMT-advanced. John Wiley & Sons, Hoboken, NJ (2009)
38. Wireless Open Access Research Platform. http://warp.rice.edu. Accessed Feburary 2020

How to cite this article: Ren Hai-Peng, Yin Hui-Ping, Zhao Hong-Er, Bai C, Grebogi C. Artificial intelligence enhances the performance of chaotic baseband wireless communication. IET Commun. 2021;15:1467–1479. https://doi.org/10.1049/cmu2.12162

APPENDIX A

The proposed CNN structure to predict the symbols is shown in Figure A.1. In Figure A.1, the CNN input is an 8×8 matrix, which is reformulated sampling data from the four past symbols. The outputs of CNN are the predicted three future symbols with respect to the decoding one.

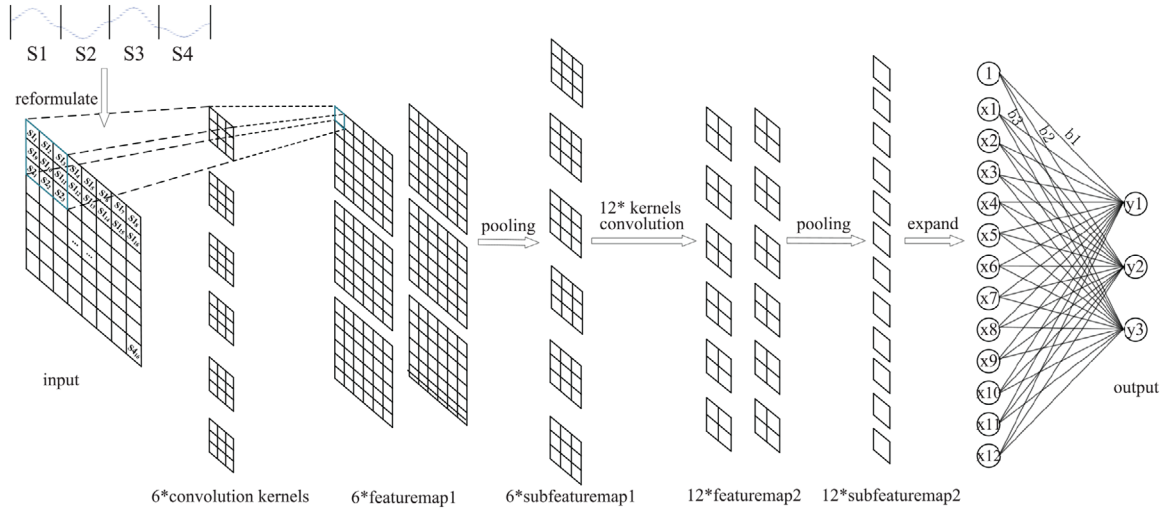


FIGURE A.1 The proposed CNN structure *Note*: Sixty-four sampling data corresponding to four past symbols are reformulated into 8×8 matrix, then used as CNN input. Three outputs of the CNN are the predicted future symbols (1 or −1)