Deep Learning-Based SC-FDMA Channel Equalization

Mohamed A. Mohamed[1,2,*], Hassan A. Hassan[1,2], Mohamed H. Essai[1], Hamada Esmaiel[2], Ahmed S. Mubarak[2], and Osama A. Omer[2]
[1] Department of Electrical Engineering, Faculty of Engineering, Al-Azhar University, Qena 83513, Egypt
[2] Department of Electrical Engineering, Faculty of Engineering, Aswan University, Abulrish 81542, Egypt
Email: mohammed.anbar@azhar.edu.eg (M.A.M.), hassanali2720.el@azhar.edu.eg (H.A.H.), mhessai@azhar.edu.eg (M.H.E.), h.esmaiel@aswu.edu.eg (H.E.), ahmed.soliman@aswu.edu.eg (A.S.M.), omer.osama@aswu.edu.eg (O.A.O.)

***Abstract*—It is very challenging to design an effective wireless communication system. That's because of numerous factors affecting the performance of a typical wireless communication system, such as nonlinear channel distortions and impairments. single carrier frequency division multiple access (SC-FDMA) is a multiple access scheme that is an important part of the long-term evolution (LTE) standard for uplink transmission. An advanced mobile radio system's multiple access schemes should indeed meet stringent requirements, such as a low bit error rate (BER). In this article, we investigate the equalization problem for nonlinear channel distortions and impairments using deep neural networks (NN). We introduce a novel combined deep neural network channel equalization and symbol detection scheme based on a deep learning (DL) recurrent feedback (RF) long short-term memory (LSTM) neural network to achieve blind equalization and decoding for SC-FDMA systems without knowing the channel state information (CSI). To train the model efficiently, the training data is gathered by simulation, with channel effects and noise treated as a complete black box. CSI and constellation demapping are learned by a deep neural network (DNN) model. Then, the frequency-domain sequences that have been corrupted are implicitly equalized to get the broadcasted signal back. Our specified SC-FDMA system, which uses a quadrature phase-shift keying (QPSK) modulation method and the suggested Deep Learning-based model channel equalizer, performs better than the existing equalizers by an average of 1 to 4 dB at moderate signal-to-noise (SNR) ratios, according to simulation data. A complexity comparison between the proposed and the conventional equalizers was conducted in terms of training time, execution time, and number of operations. On combined channel equalization and symbol detection, the suggested system delivers state-of-the-art performance.**

***Index Terms*—Deep learning, channel equalization, Long Short-Term Memory (LSTM), Quadrature Phase-Shift Keying (QPSK), recurrent neural network, single-carrier frequency-division multiple access**

I. Introduction

In recent years, the communication standards and innovations are driven by the eager desire of customers to elevate their access to broadband wireless communication service. Accordingly, mobile networks witness a huge demand in terms of larger data rates and massive connected devices. It is anticipated for global mobile data traffic to margin 230 exabytes (EB) per month, and the connected devices to pass 90 million by 2026 [1].

Consequently, new wireless transmission techniques with large data rates and resistance to radio frequency (RF) impairments have gotten a lot of attention as a response to this demand. Multicarrier orthogonal frequency division multiple access (OFDMA) strategies have become the dominant essence for wireless broadband applications in recent decades because of their superior spectral efficiency, which is achieved using a special set of overlapped orthogonal subcarriers, and their resistance to channel selectivity [2].

Despite of their multiple advantages, OFDMA has some downsides, such as a high peak-to-average power ratio (PAPR), which makes it hard for mobile devices to save energy [3]. To address this issue, a modified version of OFDMA, discrete Fourier transform (DFT) pre-coded OFDM, known as single-carrier FDMA, was investigated (SC-FDMA). It also has the same effectiveness and complexities as OFDMA but with a lower PAPR condition. As time passed, SC-FDMA has proven to be a great design and has been used for uplink transmission in the long-term evolution (LTE) standard.

Because of the large amount of inter-symbol interference (ISI) among the transmitted symbols, the multipath environment of wireless communication channels makes it difficult to recover rapidly transmitted data at the receiver. As a result, overcoming the problem of inter-symbol interference is a critical task for wireless communications systems. Powerful equalization procedures are unavoidable in order to offset the negative consequences of ISI. There have been a lot of new equalization algorithms in the last few years that can help cut down the amount of interference (ISI) in fading channels. The ISI could indeed span hundreds of symbols when the data rate is extremely high, and the cost of designing and making these filters might be too high [4, 5].

Equalizer design problem get more attentions both academically and in the industrial field since it is crucial

Manuscript received July 2, 2023; revised August 22, 2023; accepted September 8, 2023; published February 2, 2024.
*Corresponding author

to deal with dynamic and rapidly varying channels. Linear equalizers often use transversal or lattice linear filters and adaptation techniques like recursive least square (RLS), fast RLS, least mean square (LMS), square-root RLS, gradient RLS, and so on. Linear equalizers, on the other hand, perform poorly on channels with profound spectral nulls.

Artificial deep learning neural networks (ADLNN) algorithms in wireless communications systems have evolved considerably in recent years, particularly in the physical layer, owing to their tremendous capability to learn, recognize, and forecast [6]. In the next subsection, a brief overview for the literature of neural networks (NN) in equalizer design problem will be introduced.

A. Related Work

Modulation classification problem has been tackled in multiple research works. In [7–9], an effective modulation classifier has been developed using convolutional neural networks (CNN). The authors of [10–12] demonstrated higher performance and faster convergence for NN-based decoders for BCH and polar code channel decoding. Additionally, in [13], signal detection in multiple-input-multiple-output (MIMO) orthogonal frequency-division multiplexing (OFDM) systems was presented to attain highly developed accuracy with much less complexity while giving robustness under intricate channel interference. Additionally, [14–16] provide examples of their opportunities for enhancing wireless communication systems.

Artificial DNNs have recently attracted attention in the domain of channel equalization due to their abilities to accomplish the mapping between input and output domains in a way that is not linear and because equalization and symbol detection might be considered as, classification tasks [17–19].

In the equalization process that uses an adaptive technique based on a NN to recover the target signal, the NN minimizes the difference exist between the equalizer outcome and the delayed signal to recover the channel's nonlinear properties from the data that was received. Patra *et al*. demonstrated in [20–22] that for signals with either pulse amplitude modulation (PAM) or quadrature amplitude modulation (QAM), NN-based nonlinear equalizers outperformed conventional linear equalizers in terms of bit error rate (BER).

For a long time, NNs have been employed for channel equalization [18, 23–25], therefore it appears reasonable to use DNN as an end-to-end approach for optimizing channel equalization and decoding simultaneously.

Numerous machine learning techniques were used to tackle the nonlinear equalization problem in the absence of accurate channel state information (CSI). Among these techniques are DNN [26, 27], Gaussian processes for classification (GPC) [28], convolutional neural network (CNN) [19], and support vector machine (SVM) [29, 30], which allow the receiver to achieve adaptive equalization. The DNN equalizer described in [26] requires a significant number of parameters and is only useful for short codes. In [28, 31, 32] these equalizers determine the channel filter coefficients of ISI using multiple training sequences. Additionally, these methods require prior knowledge of channel filter coefficient distributions and the variance of AWGN. In actuality, such assumptions may be false. As a result, without a priori information, it is difficult to generate correct CSI analysis results. As shown in [31], after the decoder, the SVM equalizer performs poorly.

Given the aforementioned positive developments, DL (deep learning) has a lot of power and is very promising for dealing with more difficult situations and meeting the strict needs of 5G and beyond communication systems.

B. Motivations and Contributions

Deep learning has lately gained wide popularity in the areas of natural language processing, text translation, and computer vision [33, 34]. The application of deep learning in wireless communications has also experienced significant growth [35].

The basic feedforward NN only creates the weighted connections between the cascading layers, but in the recurrent neural networks (RNNs), the adjacent neurons in the same layer are also connected to each other. RNNs could be trained to learn sequential or time-varying patterns since they have memory or feedback connections. In the last few years, RNNs have been prevented from being a mainstream network model because of the challenges in training and computing complexity. Due to the advancement of deep learning theory, RNNs are growing quickly right now.

The main feature of RNNs is that they have a hidden layer with memory capabilities, which provides them with a structural advantage when processing time-series data. Therefore, using an RNN as a channel equalizer can help the CS equalizer learn faster and work better. Deep learning techniques are much better for channel equalization than other methods because they can automatically find problem features without needing a lot of information in advance.

Standard RNNs have the drawback of being unable to use information about future input [36]. In tasks like text translation, the content that comes after the current text often has a big effect on how it is understood, and it is evident that the standard RNN is unable to account for this. The bidirectional RNN (B-RNN) was suggested in [36, 37] as a remedy for this shortcoming. In order to accomplish this, the B-RNN uses two distinct hidden layers that analyses the data in both directions before feeding the results into a single output layer [33]. Standard RNNs also have the issue that they do not have much of a chance of picking up informational features over long distances. This is because as the time step increases during the training phase, they might run into gradient elimination and gradient explosion. A long short-term memory (LSTM) architecture was suggested in [38] as a solution to this issue. One could consider the LSTM network to be an improved form of the RNN's simple hidden unit [33, 39]. Long-distance dependencies can be handled well by the LSTM [40]. A traditional LSTM has three gates (forget, input, and output), a memory cell, a block input, and an output activation

function (AF). Each of the gates and the block's input are linked to the block's output [39].

In this work, we model the channel equalization problem in SC-FDMA systems as a DL task, and propose a new idea for combined channel equalization and signal detection (CE-SD) based on DL recurrent feedback LSTM-NN. This idea takes features from the SC-FDMA system's received messages and labels them based on the constellation map used at the transmitter. Compared to conventional non-neural network approaches, the neural network approach is more flexible since it can be used for various channel circumstances and does not need to worry about channel details. In the proposed scheme, channel equalization and signal detection are treated as a complete black box, and the box functions are continuously approached by a DNN model. The DNN model has the ability to perform equalization and symbol decoding simultaneously, even in the absence of channel state information (CSI). In terms of bit error rate (BER), simulation results showed that our suggested scheme outperforms other commonly used signal equalization approaches. This successful example highlights the utility of DL in SC-FDMA systems.

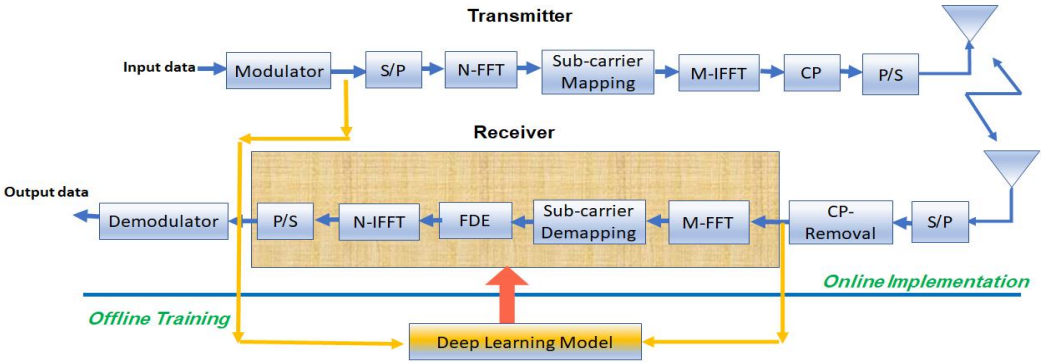

Fig. 1. The proposed SC-FDMA scheme.

The most significant contributions are listed below.

1) We embed the DL strategy into the SC-FDMA system to equalize the channel effects and detect symbols to exploit the capabilities of the DNN at recognizing and representing things, the training process can make the CS equalization at data subcarriers work better.
2) We evaluate the performance of the suggested framework for channel equalization under various conditions. The accuracy of the channel equalization is explicitly assessed using a simulation of the bit error rate (BER). Additionally, simulations and comparisons have demonstrated that the suggested framework is effective and trustworthy across a variety of channel circumstances.
3) We provide a dataset that will help the research community to evaluate against linear equalizers (LEs), such as the minimum main square error (MMSE) and zero forcing (ZF) and optimization algorithms as well.

The following sections will organize the remainder of the paper: The second section, which follows this one, is devoted to describing the system. The third and fourth sections introduce the Deep Learning model and the offline training of the suggested model, respectively. The simulation results are then shown. Finally, to conclude the study.

II. System Model

The SC-FDMA system is demonstrated in Fig. 1, as in [41]. The overall system subcarriers are M. From among those N_u users, each N subcarriers are designated to only one user, where $M = N_u \times N$. All of this is accomplished following the N points fast Fourier transform (FFT). A length L_{cp} of cyclic prefix (CP), longer than or equal to the length of the channel's transfer function L_{ch}, will be added following the M points inverse fast Fourier transform (IFFT). The time domain (TD) transmitted signal that corresponds to the kth user without the L_{cp} in vector form is $\mathbf{G}_k = \mathbf{F}_M^H \mathbf{T}_k \mathbf{F}_N \mathbf{s}_k$, where $\mathbf{s}_k$ is the kth user's ($N\times1$) symbol vector, $\mathbf{T}_k$ is an $M\times N$ sub-carrier mapping matrix, and $\mathbf{F}_N$ and $\mathbf{F}_M^H$ are the FFT and IFFT matrices, respectively, with dimensions $N\times N$ and $M\times M$. Assume that $\mathbf{H}_k$ is the transfer function of the channel between the kth user and the base station, with maximum delay spread L_{ch} less than the L_{cp}, to entirely remove the ISI.

The procedure will be reversed at the receiving end. The CP is removed first, then the FFT transforms the SC-FDMA symbols into FD by M points FFT accompanied by sub-carrier demapping to extract the FD signal received for the kth user.

The TD signal received relating to all N_u users for the tth SC-FDMA symbol presented by

$$\mathbf{r}^t = \sum_{l=0}^{N_u-1} \mathbf{H}_l^t \, \mathbf{G}_l^t + \mathbf{n}^t, \tag{1}$$

where $\mathbf{H}_l^{\text{th}}$ is the ($M\times M$) channel circular convolution matrix where the first column contains the impulse response of the channel between the lth user and the BS, and $\mathbf{n}^t$ is an ($M\times1$) AWGN vector with variance σ_n^2. After M FFT, the transform of the received signal is expressed as

$$\mathbf{R}^t = \sum_{l=0}^{N_u-1} \mathbf{F}_M \mathbf{H}_l^t \, \mathbf{G}_l^t + \mathbf{F}_M \, \mathbf{n}^t \, . \tag{2}$$

It will be easier to follow if we remove the symbol index from Eq. (2), as shown below:

$$\mathbf{R} = \sum_{l=0}^{N_u-1} \widehat{\mathbf{H}}_l \, \mathbf{T}_k \mathbf{F}_N \mathbf{s}_k + \mathbf{F}_M \, \mathbf{n} \tag{3}$$

where $\mathbf{F}_M \mathbf{H}_l \mathbf{F}_M^H = \text{diag}\big(\mathbf{FFT}(h_l)\big) = \widehat{\mathbf{H}_l}$.

The FD signal received for the k^{th} user is calculated

after demapping as:

$$\mathbf{T}_k^H \mathbf{R} = \mathbf{T}_k^H \sum_{l=0}^{N_u-1} \widehat{\mathbf{H}}_l \, \mathbf{T}_k \mathbf{F}_N \mathbf{s}_k + \mathbf{T}_k^H \mathbf{F}_M \, \mathbf{n} \,. \quad (4)$$

Because of the mapping matrix's orthogonality property,

$$\mathbf{T}_k^H \mathbf{T}_l = \begin{cases} \mathbf{I}_N, & k = l \\ \mathbf{0}_N, & k \neq l \end{cases}$$

The FD signal received for the *k*th user could well resemble

$$\mathbf{R}_k = \mathbf{T}_k^H \widehat{\mathbf{H}}_k \mathbf{T}_k \mathbf{F}_N \mathbf{s}_k + \mathbf{T}_k^H \mathbf{F}_M \, \mathbf{n} = \bar{\widehat{\mathbf{H}}}_k \mathbf{F}_N \mathbf{s}_k + \mathbf{T}_k^H \mathbf{F}_M \, \mathbf{n} \quad (5)$$

where $\bar{\widehat{\mathbf{H}}}_k = \mathbf{T}_k^H \widehat{\mathbf{H}}_k \mathbf{T}_k$ is a $N \times N$ diagonal sub-matrix of the $M \times M$ $\widehat{\mathbf{H}}_k$ matrix.

The signal received can be expressed using the *b*th frequency bin as follows:

$$R_{k,b} = \bar{\widehat{H}}_{k,b} S_{k,b} + N_b, \;\; b = 0,1, \ldots N-1. \quad (6)$$

Then perform frequency domain equalization by any traditional linear techniques (ZF or MMSE) like in [41] to counteract the ISI's effects. Where the frequency domain channel equalization is carried out on a sub-carrier basis. The FD-LE is conducted through multiplying $\mathbf{W}$, to the signal received. The signal equalized in FD domain is given by

$$\widehat{\mathbf{R}}_k = \mathbf{W}_k \, \boldsymbol{R}_k = \widetilde{\mathbf{H}}_{\mathrm{k}}{}^{-1} \mathbf{R}_k \quad (7)$$

where $\mathbf{W}_k = \widetilde{\mathbf{H}}_{\mathrm{k}}{}^{-1}$ and $\widetilde{\mathbf{H}}_{\mathrm{k}}$ represent the diagonal matrix used for channel equalization. The *b*th entry along the diagonal is given by $1/H_{k,b}$ for the ZF criterion and $H^*{}_{k,b}/\left(\left|H_{k,b}\right|^2 + 1/SNR\right)$ for the MMSE criterion [42].

Then perform FD equalization by any traditional technique like in [41] to counteract the ISI's effects. Then demodulate and detect the *k*th user original transmitted symbols after *N* points IFFT TD transformation. However, in the suggested method, a DNN is used instead of conventional channel equalization techniques, which results in an end-to-end scheme that can get back the original information from the information that was sent.

III. Deep Learning Model

Due to deep learning's powerful capabilities, it has been successfully applied in a wide variety of applications, including natural language processing [43], computer vision [44], speech recognition [33], and others. Here, we will discuss the most fundamental theories and concepts underlying deep learning and how they apply to our model. For a detailed explanation of deep learning and machine learning, please see [45].

LSTM NNs are discussed in this section for combined channel equalization and symbol detection. Offline training with simulated data is used to train the proposed DL-LSTM-based channel equalizer.

The LSTM network is a form of RNN that is smart enough to learn long-term correlations between time step sequences [38]. Numerous LSTM-based systems have been developed to address issues such as speech recognition [46], handwriting recognition [47], and online translation with tools like Facebook translation systems [48], and Google neural machine translation [49].

Input, output, and forget gates, as well as a memory cell, comprise the LSTM-NN structure. The LSTM-NN properly stores the long-term memory via the forget and input gates. The LSTM cell's primary structure is depicted in Fig. 2 in [38]. The forget gate allows the LSTM-NN to eliminate the unwanted data from the last process by using the present used input x_t and the cell output h_t. On the basis of the preceding cell output h_{t-1} and the present cell's input x_t, the input gate determines the data that will be utilized in conjunction with the preceding LSTM cell state c_{t-1} to generate a new state of the cell c_t. LSTM may decide which data is discarded and which is maintained by using the forget and input gates.

The output gate determines the present cell output h_t by utilizing the preceding cell's output h_{t-1} at the present state of the cell c_t and input x_t.

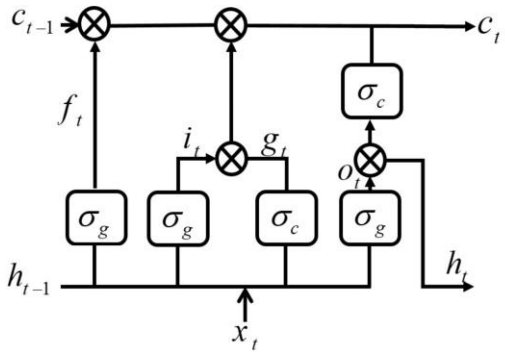

Fig. 2. LSTM neural network architecture.

The mathematical formulation for the LSTM-NN configuration is given by (8) to (13) as in [50].

$$\mathbf{i}_t = \boldsymbol{\sigma}_g(\mathbf{w}_i \mathbf{x}_t + \mathbf{R}_i \mathbf{h}_{t-1} + \mathbf{b}_i) \quad (8)$$

$$\mathbf{o}_t = \boldsymbol{\sigma}_g(\mathbf{w}_o \mathbf{x}_t + \mathbf{R}_o \mathbf{h}_{t-1} + \mathbf{b}_o) \quad (9)$$

$$\mathbf{g}_t = \boldsymbol{\sigma}_c\left(\mathbf{w}_g \mathbf{x}_t + \mathbf{R}_g \mathbf{h}_{t-1} + \mathbf{b}_g\right) \quad (10)$$

$$\mathbf{f}_t = \boldsymbol{\sigma}_g\left(\mathbf{w}_f \mathbf{x}_t + \mathbf{R}_f \mathbf{h}_{t-1} + \mathbf{b}_f\right) \quad (11)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \quad (12)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \boldsymbol{\sigma}_c(\mathbf{c}_t) \quad (13)$$

where $\mathbf{x}_t \in \mathbb{R}^{n_i}$ is the input vector at time t with n_i entries. $\mathbf{h}_t, \mathbf{h}_{t-1} \in (1,-1)^{n_h}$ are the hidden layer states vectors of the LSTM at time t and t–1, respectively. $\mathbf{w}_i$, $\mathbf{w}_f$, $\mathbf{w}_g, \mathbf{w}_o \in \mathbb{R}^{n_h \times n_i}$ are the forward or the input trainable weight matrices. $\mathbf{R}_i, \mathbf{R}_f, \mathbf{R}_g, \mathbf{R}_o \in \mathbb{R}^{n_h \times n_i}$ are the recurrent trainable weight matrices. $\mathbf{b}_i, \mathbf{b}_f, \mathbf{b}_g, \mathbf{b}_o \in \mathbb{R}^{n_h}$ are the biases vectors. *i*, *o*, and *f* represent the input, output, and forget gates, respectively. $\mathbf{g}_t, \boldsymbol{\sigma}_g,$ and $\boldsymbol{\sigma}_c$ represent the cell candidate, the gate activation function (sigmoid function), and the state activation function (tanh function), respectively. ⊙ denote the Hadamard Product (Elementwise Multiplication). LSTM-NN evaluates only the prior sequence's impact on the present sequence, neglecting the subsequent information.

The DL LSTM-NN was constructed using an array of the following five layers as shown in Fig. 3, to perform

combined channel equalization and symbol detection: A layer for the sequence input (with a size equal to the amount of features in the input data, which is 128), an LSTM layer (with 128 hidden units), and eventually, 256 classes are accomplished by the use of a 256 fully connected layer, accompanied by a SoftMax layer and a classification layer.

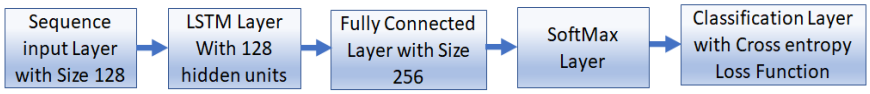

Fig. 3. DL LSTM-NN framework for the proposed joint channel equalizer and symbol detector.

In the current work each user is assigned four subcarriers from the SC-FDMA frame, and each subcarrier can be one of four QPSK constellation points. The number of labels in the training equals M_s^N, where M_s is the constellation (modulation) order and N is the subcarriers that are assigned to only one user. Therefore, the number of labels in the training equals 4^4=256, so the number of classes is 256. Consequently, the size of the fully connected layer in the LSTM-NN should be 256 to match the number of classes. If we use higher-order modulations or give each user more subcarriers, the number of labels will also go up. This will make the system harder to use, take longer to train, and lead to an impractical system. As a result, we recommend using QPSK.

As the proposed DL LSTM-based channel equalizer and symbol detector is created, the weights and biases of the recommended equalizer must be adjusted (tuned) before deployment, using the appropriate optimization algorithm. The optimal parameters (weights and biases) are learned on a training set with predefined outputs. To figure out the best parameters, a loss function is used to figure out how far the network output is from the desired output, and then three different DLNN optimization methods are used to change the parameters. A number of different optimization methods are used to get the best possible channel equalization and symbol detection for the SC-FDMA wireless communication system. Some of them are stochastic gradient descent with momentum (SGDm), root mean square propagation (RMSProp), and adaptive moment estimation (Adam).

A. Optimization Algorithms

The method of back-propagation (BP) is frequently utilized to train NNs. The training method repeats two phases: propagation and weight update. The error propagates from the output layer backward to the rest of the nodes. Using these errors, we can determine the gradient of the loss function relative to the network weights. In order to minimize the loss function, the optimization method takes the gradient and uses it to make adjustments to the weights [51].

Learning processes are greatly aided by optimization algorithms. By adjusting the model's weights and biases to minimize the loss function, the learning process seeks to find a model that will yield improved results. Learning deep neural networks is analogous to solving an optimization problem, with the goal of achieving global optimization via a stable training trajectory and rapid convergence with gradient descent algorithms [52].

It is generally usual to use a gradient descent optimization algorithm to seek the minimum loss function. The gradient descent method changes the weights and biases incrementally in small steps in the direction of the negative gradient of the loss function.

$$w^l(t+1) = w^l(t) - \alpha \nabla L\left(w^l(t)\right) \tag{14}$$

$$b^l(t+1) = b^l(t) - \alpha \nabla L\left(b^l(t)\right) \tag{15}$$

where t is the iteration number, l and α represents the layer number and learning rate, respectively. The learning rate, $\alpha \in [0,1]$, determines how much the weights is updated each time. When this parameter's value is too large, the output does not converge to the solution but instead floats around it. To the contrary, if it is set too low, the calculation will take too long to arrive at a solution.

In the typical gradient descent approach, the gradient of the loss function is evaluated using the complete training set at once and updates the parameters (w, and b) after scanning the whole training set, this method is referred to as "batch gradient descent".

In the case of a convex problem, batch gradient descent is ensured to converge to the global minimum, while in the case of a non-convex problem, it is confirmed to converge to a local minimum. However, in deep learning-related tasks, the training set typically contains thousands or even trillions of samples, making it impractical to calculate the gradient via a quick scan. This makes even a single update to the parameters too time-consuming. It is also challenging to feed all the data into the model at once due to the computation memory's capacity limitations. As a result, the batch gradient descent method for solving the optimization problem is rarely used in deep learning models.

Stochastic gradient descent (SGD) is a technique that can be used to calculate the gradient and update the parameters for each training sample rather than using the entire set of training data. However, due to the large variation between the training samples, updating parameters often results in significant fluctuations in the objective function. However, while a low learning rate allows SGD to converge to a great point, it also slows down training. The frequent data transfers between GPU memory and local memory also reduce efficiency when we are employing GPUs to carry out the computation.

Iterations utilize a different subset of the data, referred to as a "mini batch". The benefits of both batch gradient descent and stochastic gradient descent are combined in mini-batch gradient descent, which also updates the parameters after obtaining the gradient of a small group of samples. Unfortunately, good convergence is not guaranteed by mini-batch gradient descent, and tuning the learning rate also requires some expertise. So, to further enhance the convergence, some researchers have added

additional helpful tricks and techniques. All mini batches of the entire training dataset are processed by the training algorithm during one epoch.

Algorithms using stochastic gradient descent permit oscillations along the path taken by the steepest fall to reach the ideal. One mechanism for stabilizing this oscillation is the use of momentum [53]. The weights and biases of a neural network can be adjusted using stochastic gradient descent with momentum (SGDm) in the following manner:

$$w^l(t+1) = w^l(t) - \alpha\,\nabla L\big(w^l(t)\big) + \epsilon\big(w^l(t) - w^l(t-1)\big) \quad (16)$$

$$b^l(t+1) = b^l(t) - \alpha\,\nabla L\left(b^l(t)\right)\epsilon\left(b^l(t) - b^l(t-1)\right) \quad (17)$$

where the contribution of the prior gradient step to the current iteration is given by ϵ.

All of SGDm's parameters are learned at the same rate. By implementing learning rates that vary according to parameters and can automatically adapt to the loss function being utilized, network training can be made better. Root mean square propagation (RMSProp) is one such approach. To do this, it calculates the squares of the gradients of the parameters at each element using the formula:

$$q_\theta(t) = \mu_2 q_\theta(t-1) + (1-\mu_2)\left(\nabla L\big(w^l(t)\big)\right)^2 \quad (18)$$

$$q_b(t) = \mu_2 q_b(t-1) + (1-\mu_2)\left(\nabla L\big(w^l(t)\big)\right)^2 \quad (19)$$

where μ_2 is the rate of decay of the moving average. Overall, the rate of decay seems to be 0.9, 0.99, or 0.999. The associated squared gradient averaging lengths are equivalent to $1/(1-\mu_2)$, particularly, 10, 100, or 1000 parameter updates, respectively.

With the help of a moving average, the RMSProp method normalizes the updates of the weights and bias parameters, as shown below:

$$w^l(t+1) = w^l(t) - \frac{\alpha\,\nabla L\big(w^l(t)\big)}{\sqrt{q_\theta(t)}+\varphi} \quad (20)$$

$$b^l(t+1) = b^l(t) - \frac{\alpha\,\nabla L\big(b^l(t)\big)}{\sqrt{q_b(t)}+\varphi} \quad (21)$$

This allows RMSProp to increase learning rates for parameters with small gradients while decreasing learning rates for parameters with large gradients. A minor constant φ is added to prevent a division by zero. where an element-by-element division is used.

Momentum terms have been added to the parameter updates in Adam, similar to RMSProp. When it comes to deep learning, Adam is one of the most popular optimization algorithms used. The term "Adam" comes from the term "adaptive moment estimation", which describes how Adam calculates first and second moment estimates of gradients to determine individual adaptive learning rates for various parameters [54]. An element-wise moving average of both the parameter gradients and their squared values is retained by the algorithm. The moving average of the parameter gradients can indeed be described as follows:

$$c_\theta(t) = \mu_1 c_\theta(t-1) + (1-\mu_1)\left(\nabla L\big(w^l(t)\big)\right) \quad (22)$$

$$c_b(t) = \mu_1 c_b(t-1) + (1-\mu_1)\left(\nabla L\big(w^l(t)\big)\right) \quad (23)$$

where μ_1 is rate of decay. Adam updates the network parameters utilizing moving averages as follows:

$$w^l(t+1) = w^l(t) - \frac{\alpha\, c_\theta(t)}{\sqrt{q_\theta(t)}+\varphi} \quad (24)$$

$$b^l(t+1) = b^l(t) - \frac{\alpha\, c_\theta(t)}{\sqrt{q_b(t)}+\varphi} \quad (25)$$

If the gradients over multiple iterations are consistent, a moving average of the gradient can be used to gain momentum with parameter updates in a particular direction. If the gradients are noisy, the moving average of the gradient will be smaller, leading to smaller parameter updates.

IV. Offline Training of the Suggested DL Model

While DLNNs are the state-of-the-art approach for wireless communication systems, they have a huge amount of computational complexity and a lengthy training period. GPUs have become the most effective training equipment for DLNNs.

Due to the lengthy training period required for the proposed DL model and the large number of parameters that must be tuned during training, like weights and biases, training must be conducted offline. The trained model is utilized to extract the transmitted data during online implementation.

Because machine learning-based models aren't built by experts, they learn from data instead of being made by them. For the bulk of machine learning tasks, obtaining a huge amount of labeled data for training is a difficult challenge. Alternatively, training data for channel equalization issues can be easily gotten by simply conducting a simulation. Obtaining the training data is straightforward once the channel parameters and model are known.

Offline training of the neural networks is carried out using simulated data. When you run a simulation, you start with a random message s and send the SC-FDMA frames to the receiving end through a simulated channel model. Each frame has one SC-FDMA symbol in it. To retrieve the received SC-FDMA signal, SC-FDMA frames with varying channel defects are used. After undergoing the distortion of the channel and removing the CP, the incoming signals y are gathered as training samples. As shown in Fig. 1, the network's input data are the signals that are received y, and the actual information messages s. These signals act as the supervision labels.

By minimizing the loss function and updating the weights and biases, the optimization algorithms train the model. The loss function, in its simplest form, is the difference between the network's output and the original messages, which can be expressed in a variety of ways. The neural network toolbox in MATLAB gives the user the option of selecting a loss function from a list of possible options (i.e., MAE, crossentropyex, and MSE).

The loss function that we used in our experiments is the crossentropyex, and it can be expressed as:

$$Loss = -\sum_{i=1}^{N}\sum_{j=1}^{c} s_{ij}(k)\log\log\left(\hat{s}_{ij}(k)\right) \quad (26)$$

where c is the class number, N is the sample number, s_{ij} is the ith transmitted data sample for the jth class and $\hat{s}_{ij}$ is the DLLSTM model response for sample i class j.

Because the activation function (AF) is a sigmoid function, each output element has been compressed into the range [0, 1], which can be understood as the likelihood that the output bit is 1.

All classical equalizers are heavily reliant on tractable channel models that are believed to be stationary, linear, and Gaussian in nature. Practical wireless communication systems, on the other hand, contain additional defects and unknown environmental influences that precise channel models are incapable of adequately addressing. As a result, researchers have devised numerous channel models capable of accurately describing practical channel statistics. In this way, it is possible to get reliable and relevant training datasets by modelling with these channel models.

Pilot symbols can be used to figure out the channel models in the wireless communication systems. Then, the data broadcasted can be found by using the channel that was figured out. The vehicular A channel model is being used in this work to figure out how a real wireless channel works and how this can hurt the performance of the suggested model and the whole SC-FDMA wireless communication system.

Finally, the model can recover data automatically, without the need for explicit channel estimation and symbol detection processes. These processes are accomplished together. Fig. 4 shows how to train offline to get a learned model based on LSTM-NN.

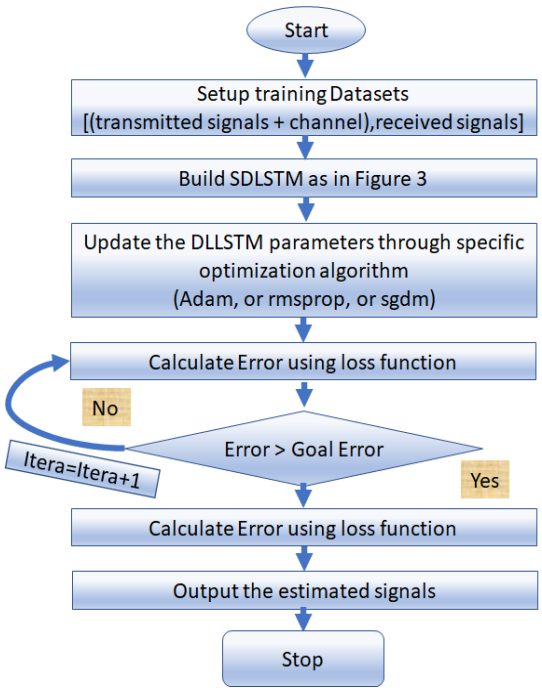

Fig. 4. Offline training of the DLLSTM-NN.

V. Simulation Results

Several experiments were carried out to demonstrate the efficiency of the proposed DLNN-based channel equalizer and symbol detector technique for the SC-FDMA wireless communication system. The proposed DLNN-based equalizer was trained and compared to the conventional Zero-Forcing (ZF) and Minimum Mean Square Error (MMSE) equalizers in terms of bit error rates (BERs) at different signal-to-noise ratios (SNRs) using the collected data sets. The training dataset is gathered for four subcarriers. The transmitter sends the SC-FDMA packets to the receiver, each containing one SC-FDMA data symbol. The SC-FDMA system and channel specifications are listed in Table I. The employed DL LSTM-NN architecture parameters and training settings are summarized in Table II.

Table I: SC-FDMA System Specifications

Parameter	Value
No. of Subcarrier = M-IFFT	64
Subcarriers allocated to each user = N-IFFT	4
Subcarrier spacing	15KHz
Cyclic prefix length	20
Modulation Format	QPSK
Channel model	Vehicular A
Channel estimation	Perfect
Equalization	ZF, MMSE, and proposed DL Model

Table II: DL Model Architecture

Parameter	Value
Sequence input t size	128
LSTM layer size	128
Fully connected layer size (No. of Classes)	256
Loss function	Crossentropyex
Mini-batch size	1000
Numbers of Epochs	3
Optimization approaches	Adam, RMSProp, and SGdm
Gate Activation Function (GAF)	Sigmoid
State Activation Function (SAF)	Tanh
Training Options	
Initial learning rate	0.05
Learning rate drop factor	0.45

Also, in the present simulations, several learning optimizers namely: The SGdm, RMSProp, and Adam will be used to train the proposed equalizer to investigate how well it performs under these optimization techniques [55]. Also, the learning rate has been chosen after an enormous number of trials to ensure convergence for all learning algorithms, and it is fixed for all channel conditions.

In the case of deep fading channels, it is well known that the linear equalization may amplify the noise at the spectral null, which has a negative impact on the performance of the SC-FDMA system. So, it is clear from Fig. 5, that the proposed equalizer using the Adam learning algorithm and crossentropyex loss functions outperforms both the ZF and the MMSE equalizers at SNRs ranging from 7 to 20 dB. The proposed equalizer can achieve BER=10^{-4} at 13 dB compared to 16 dB by MMSE and 20 dB by ZF, respectively. Additionally, for higher SNR (i.e., >14 dB) the DL model was successfully able to detect the signal perfectly such that the BER is zero for these SNR values, which reflects the capabilities of our DL model.

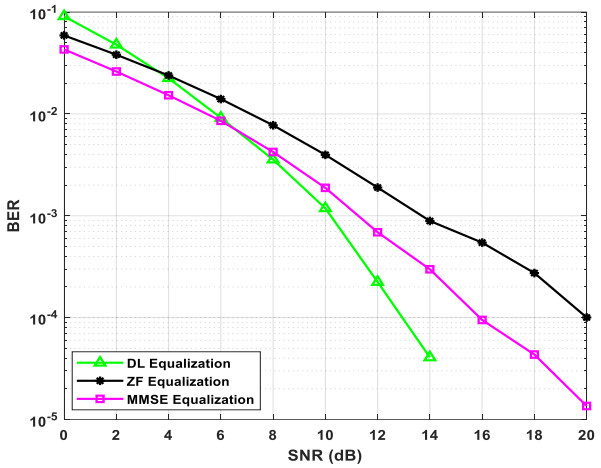

Fig. 5. BER curves of the proposed DL LSTM-based equalizer and the traditional linear equalizers using the Adam learning algorithm and the crossentropyex loss function.

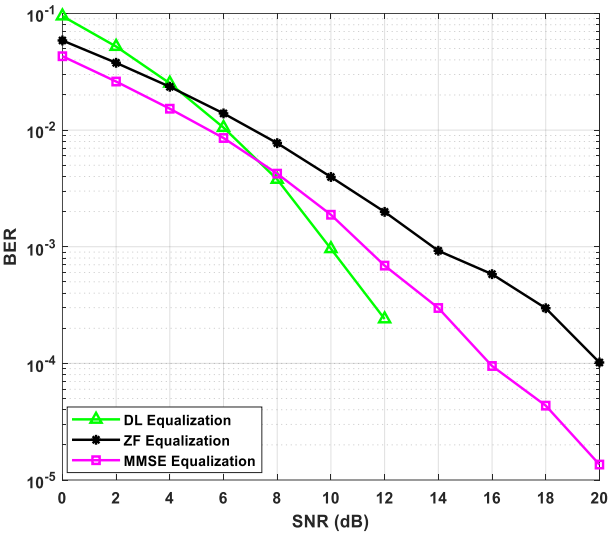

Fig. 6. BER curves of the proposed DL LSTM-based equalizer and the traditional linear equalizers using the RMSProp learning algorithm and the crossentropyex loss function.

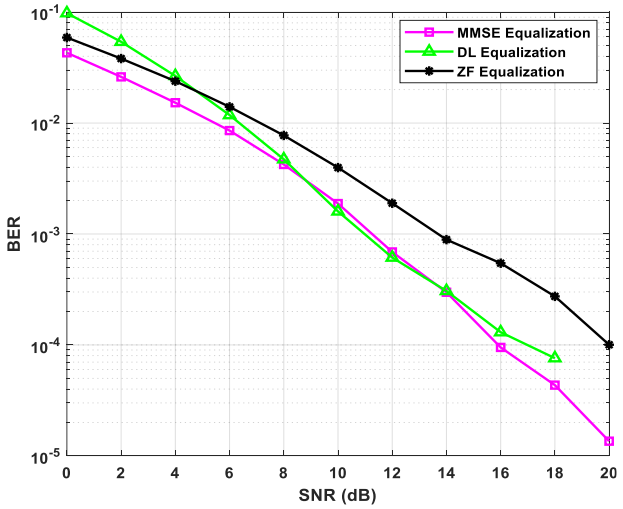

Fig. 7. BER curves of the proposed DL LSTM-based equalizer and the traditional linear equalizers using the SGdm learning algorithm and crossentropyex loss functions.

On the other hand, the MMSE outperforms the proposed model when the SNRs drop by less than 7 dB, and both linear equalizers outperform the proposed model when the SNRs drop by less than 3 dB.

Moreover, it is clear from Fig. 6 that the proposed equalizer using the RMSProp learning algorithm and the crossentropyex loss functions outperforms both the ZF and the MMSE equalizers at SNRs ranging from 7.5 to 20 dB. The proposed equalizer can achieve BER=10^{-3} at 10 dB compared to 11 dB by MMSE and 14 dB by ZF, respectively. Additionally, for higher SNR (i.e., >12 dB) the DL model was successfully able to detect the signal perfectly such that the BER is zero for these SNR values, which reflects the capabilities of our DL model. However, when the SNRs decrease by less than 7.5 dB, the MMSE outperform the suggested model, and when the SNRs drop by less than 4 dB, both linear equalizers beat the suggested model.

Furthermore, it is obvious from Fig. 7 that the proposed equalizer using the SGdm learning algorithm and the crossentropyex loss function have approximately comparable performance to the MMSE equalizer at SNRs ranging from 8.5 dB to 14.25 dB. After SNR = 18 dB, the proposed equalizer produces zero BER, while the MMSE and ZF produce a specific value of error, which reflects the capabilities of our DL model to detect the signal perfectly. On the contrary, the MMSE outperforms the suggested model when the SNRs decrease by less than 8.5 dB, and both linear equalizers outperform the suggested model when the SNRs drop by less than 4.25 dB.

Also, it is obvious from Fig. 5, Fig. 6, and Fig. 7 that the MMSE equalizer outperforms the ZF equalizer at all SNR examination ranges because it uses the channel second-order statistics in the equalization process.

It is obvious from Fig. 8 that all models have approximately comparable performance at SNR from 0 to 6 dB. After SNR = 6 dB, the SGDm model begins to have the worst performance, while both the Adam and RMSProp models were almost the same until SNR=12 dB. After SNR=12 dB, the RMSProp model outperforms both the Adam and SGDm models. Finally, we can say that the SGDm models have the worst performance in all SNR ranges.

Optimization techniques are critical for the improvement of deep learning systems. DNN training can be viewed as an optimization issue, with the objective of achieving a global optimum via a trustworthy training trajectory and rapid convergence via gradient descent techniques [55]. The goal of the DL method is to develop a model that produces more accurate and faster outcomes by modifying the biases and weights to minimize the loss function. Selecting the best optimizer for a certain scientific issue is a difficult task. By selecting an inadequate optimizer, the network may remain in the local minima (stay in the same place) during training, resulting in little progress in the learning process. As a result, the inquiry is required to look at how different optimizers perform based on the model and dataset used to make the best DL model.

This section compares the performance of three optimization algorithms: Adam, RMSProp, and SGDm, using an experimental approach.

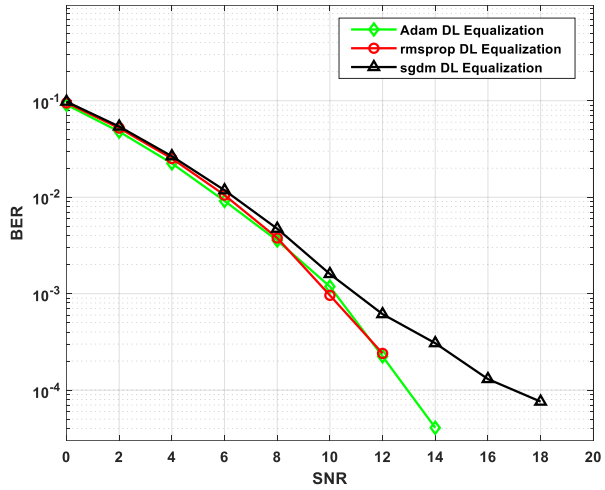

Fig. 8. Performance comparison of the proposed DL Equalizers using different optimization algorithms.

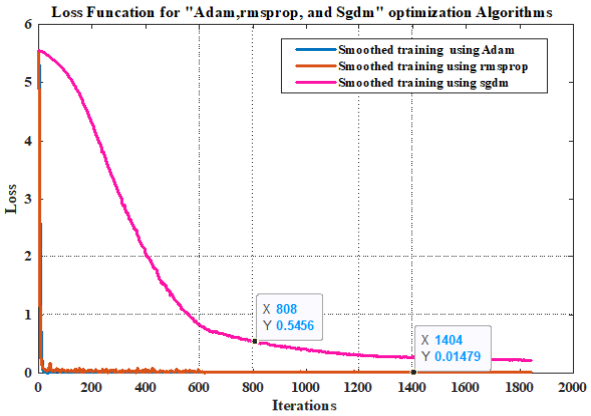

Fig. 9. Loss function comparison of the DL equalizers using different optimization algorithms.

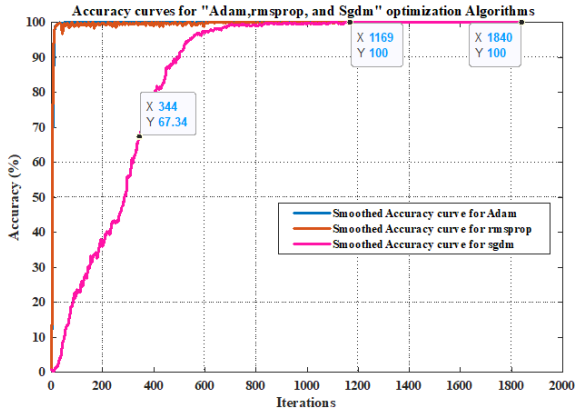

Fig. 10. Accuracy curves comparison of the DL equalizers using different optimization algorithms.

It is beneficial to monitor the training processes of the DL equalizers during the training process. By plotting loss and accuracy measures during the training process, we can monitor how the training is proceeding. Fig. 9, and Fig. 10 show that the sgdm optimization approach achieves the highest loss (worst performance) when compared to the Adam and rmsprop optimization strategies and takes a long time to converge to 100% accuracy as compared to both Adam and rmsprop, which is supported by Fig. 8, which shows that the trained DL equalizer using the sgdm algorithm has the greatest BER values. Furthermore, the loss and accuracy curves of both the Adam and rmsprop optimization techniques emphasize the obtained results in Fig. 8.

A. Computational Complexity Comparison

The computational complexity of the proposed LSTM-based channel equalization and symbol detection deep learning models in the SC-FDMA is provided empirically in terms of the training time which is performed offline.

Then these models' computational complexity is compared to the conventional models during the online deployment in terms of the execution time and number of operations (complex multiplications) of each model.

1) Training time

Training time can be defined as the amount of time expended to get the best NN parameters (e.g., weights and biases) that will minimize the error using a training dataset. Because it involves continually evaluating the loss function with multiple parameter values, the training procedure is computationally complex.

Table III lists the consumed training time for LSTM-based channel equalization and symbol detection deep learning models. The used computer is equipped with Windows 10 operating system and an Intel(R) Core(TM) i5-2450M CPU @ 2.50GHz, and 8 GB of RAM.

TABLE III: COMPARISON OF OFFLINE TRAINING TIME

LSTM-based CE-SD Adam (Min: Sec)	LSTM-based CE-SD RMSprop (Min: Sec)	LSTM-based CE-SD SGDm (Min: Sec)
14:06	14:29	15:25

From Table III, the LSTM-based CE-SD trained with Adam optimizer consumes the lowest training time, followed by LSTM-based CE-SD trained with RMSprop optimizer, while the highest training time is consumed by LSTM-based CE-SD trained with SGDm optimizer, at the same training options. The LSTM-based CE-SD trained with SGDm optimizer training time indicates its high computational complexity in comparison to its peers.

The conventional models have no computational complexity in the offline phase since they do not need training.

2) Execution time

Here we will compare the execution time that is necessary for our proposed model and that is necessary for the conventional ways that use ZF or MMSE equalizers, based on MATLAB MathWorks execution time measurement functions.

At this point, the proposed model and the traditional methods will be compared based on how long it takes for each to run on the same computer settings as in offline training.

From Table IV, the proposed LSTM-based CE-SD consumes the highest execution time, while the

conventional ways that use conventional equalizers such as ZF or MMSE consume lower execution time. The execution time of the ZF equalizer is the lowest one as well. All of this is because of the number of operations $\mathcal{O}(N)$ (complex multiplications and complex additions) in the proposed LSTM-based CE-SD function in the input size, hidden units' size, and the fully connected layer size. While the number of operations $\mathcal{O}(N)$ in the ways that use conventional equalizers (ZF or MMSE) is linearly dependent on the input size only. Therefore, the computational complexity of the proposed LSTM-based CE-SD will be the highest. Also, the number of operations $\mathcal{O}(N)$ in the conventional way that use the ZF equalizer is lower than that used with the MMSE equalizer.

TABLE IV: COMPARISON OF EXECUTION TIMES

LSTM-based CE-SD	With MMSE Equalizer	With ZF Equalizer
0.172989 s	0.011331 s	0.009307 s

3) Number of operations:

The computational complexity of our proposed model can be computed in terms of the number of operations (multiplications).

The goal of this high-level metric is to solely take into account multipliers, neglecting additions, because the latter can be implemented in hardware or software for a lower cost, whereas multipliers are typically the slowest component in the system and take up the most chip space [56, 57].

According to the mathematical formulations and the structure of our proposed deep learning model, mainly depending on the LSTM-NN in Fig. 3, we can drive the computational complexity. As in [58], the total number of operations in a standard LSTM network can be calculated as follows:

$$\text{Number of Operations}_{\text{LSTM}} = n_h(4n_i + 4n_h + 3) \quad (27)$$

where n_h is the number of hidden (memory cells) units in the LSTM cell. n_i, is the dimensional or the number of features in the input vector at time *t*.

Then we have to compute the number of operations of the fully connected layer to compute the overall computational complexity of the proposed model.

Also as in [58], the size of the fully connected layer is equal to the number of the classes we have to classify, let us symbolize it with n_k. The input to the fully connected layer equal to the number of hidden states n_h from the LSTM layer. Hence, the number of operations of the fully connected layer equal:

$$\text{Number of Operations}_{\text{FC}} = n_k\, n_h \quad (28)$$

By ignoring the SoftMax layer computational complexity for simplicity, the overall complexity of the proposed model will be

$$\begin{aligned}\text{Number of Operations} &= \text{Number of Operations}_{\text{LSTM}} + \\ &\quad \text{Number of Operations}_{\text{FC}} \\ &= n_h\left(4n_i + 4n_h + 4\right) + n_k n_h\end{aligned} \quad (29)$$

As we can see, the computational complexity depend on the input size, the hidden state size, and the number of classes we have to classify.

Now we will find the computational complexity of both ZF and MMSE equalizers. Since the inverse of $\widetilde{\mathbf{H}}_{\mathrm{k}}$ $N\times N$ diagonal matrices in (7) require, a complexity of $\mathcal{O}(N)$ [59]. And the FFT and IFFT require, a complexity of $\mathcal{O}(M/2\log_2 M)$ and $\mathcal{O}(N/2\log_2 N)$, respectively, the overall complexity will be [60]:

$$\begin{aligned}\text{Overall complexity}_{\text{LE}} = \mathcal{O}(N) + \mathcal{O}(M/2\log_2 M) + \\ \mathcal{O}(N/2\log_2 N)\end{aligned} \quad (30)$$

Therefore, the overall computational complexity of the proposed and conventional models as a function of the number of operations (mainly the complex multiplications) can be summarized in Table V.

The number of addition operations in the denominator of MMSE is more than that in ZF, so the execution time for MMSE will be more than ZF, which confirms our findings in Table IV.

TABLE V: COMPLEXITY COMPARISON OF THE PROPOSED AND CONVERSATIONAL MODELS

LSTM-based CE-SD	$n_h(4n_i + 4n_h + 4) + n_k\, n_h$
With MMSE equalizer	$\mathcal{O}(N) + \mathcal{O}\big((M/2)\log_2 M\big) + \mathcal{O}\big((N/2)\log_2 N\big)$
With ZF equalizer	$\mathcal{O}(N) + \mathcal{O}\big((M/2)\log_2 M\big) + \mathcal{O}\big((N/2)\log_2 N\big)$

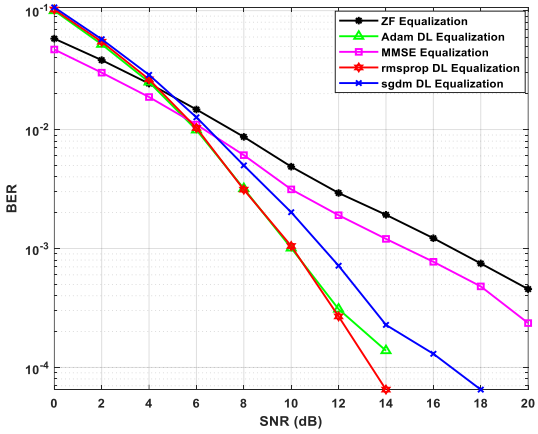

Fig. 11. BER curves of the proposed DL LSTM-based equalizer and the traditional linear equalizers under ITU Indoor channel model.

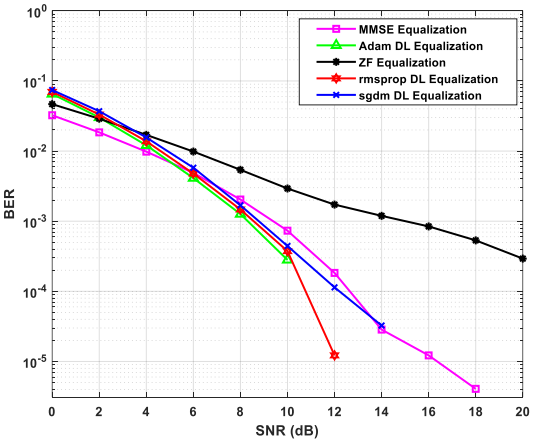

Fig. 12. BER curves of the proposed DL LSTM-based equalizer and the traditional linear equalizers under ITU Pedestrian channel model.

B. Generalization Ability and the Robustness

In these experiments, several practical channel models have been adopted. These channel models have been established based on lots of measurements (such as the indoor and pedestrian models) released by ITU [61, 62].

Fig. 5, Fig. 6, Fig. 7, Fig. 11, and Fig. 12 illustrate the BER performance versus the SNR of the proposed scheme and linear MMSE and ZF equalizers under three different ITU channel models. It can be noted that the proposed equalizer provides stable performance where it outperforms the ZF equalizer in all studied channel models. Also, it strongly competes with the MMSE equalizer and beats it in most of the studied channel models. The obtained results emphasize the generalization ability and robustness of the proposed equalizer as it has been tested using datasets (corrupted by the 3 different ITU channel models) that it has not used in the training process before.

VI. Conclusion

In conclusion, an online DL-LSTM-CE-SD-based SC-FDMA system is proposed. The suggested equalizer is first trained offline, then used in the communication system to keep track of the channel statistics. Finally, the channel is equalized, and the transmitted symbol is recovered from the transmitted data stream. The suggested equalizer's performance is studied and compared to other standard equalizer approaches, such as ZF and MMSE. The proposed equalizer beats both the ZF and MMSE equalizers in terms of BER, exhibits significant enhancements, and is adaptable for various channel conditions. In addition, a comparison of three different optimization methods for DL was done to study how the proposed equalizer performs at each. A complexity comparison between the proposed and conventional equalizers was investigated. Since recovering the transmitted signal and delivering the information to the receiving end is crucial in most cases, the focus of such applications is on good performance rather than mathematical complexity. Even though the computational complexity of the proposed DL model is higher than that of traditional models, the fast growth of technology in designing and manufacturing high-speed GPUs gives priority to the proposed model. Due to the exceptional learning and generalization characteristics of the proposed DL LSTM-CE-SD model, the suggested equalizer seems promising for channel equalization in SC-FDMA communication systems, especially in worse channel circumstances.

Here are some suggestions for future research:

- Evaluating the performance of the proposed equalizer utilizing multiple optimization techniques, including Adagrad, AdaMax, and Nadam.
- Evaluating the efficacy of the proposed equalizer with different states and gate activation functions.
- Developing and employing loss functions that are more robust than the cross-entropy function to enhance the efficacy of the proposed model.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

All of the authors of this research paper took part in planning, carrying out, and analyzing the study. They have all read and approved the final version.

References

[1] A. Abdelmoaty, D. Naboulsi, G. Dahman *et al.*, "Resilient topology design for wireless backhaul: A deep reinforcement learning approach," *IEEE Wireless Communications Letters*, vol. 11, no. 12, pp. 2532–2536, Dec. 2022.

[2] C. An and H.-G. Ryu, "Spectrum efficient multidimensional OFDM-CDIM communication system," in *Proc. 2020 23rd Int. Symposium on Wireless Personal Multimedia Communications*, 2020. doi: 10.1109/WPMC50192.2020.9309497

[3] T. Kebede, Y. Wondie, J. Steinbrunn, H. B. Kassa and K. T. Kornegay, "Multi-carrier waveforms and multiple access strategies in wireless networks: Performance, applications, and challenges," *IEEE Access*, vol. 10, pp. 21120–21140, 2022.

[4] J. G. Proakis and M. Salehi, *Digital Communications*, McGraw-hill New York, 2001.

[5] N. Benvenuto, G. Cherubini, and S. Tomasin, *Algorithms for Communications Systems and Their Applications*, John Wiley & Sons, 2021. DOI: 10.1002/0470855509.fmatter

[6] T. O'shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[7] R. Liu, Y. Guo, and S. Zhu, "Modulation recognition method of complex modulation signal based on convolution neural network," in *Proc. 2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference*, 2020, pp. 1179–1184.

[8] J. Jiao, X. Sun, Y. Zhang, L. Liu, J. Shao, J. Lyu, L. Fang , "Modulation recognition of radio signals based on edge computing and convolutional neural network," *Journal of Communications and Information Networks*, vol. 6, no. 3, pp. 280–300, Sept. 2021.

[9] K. Ma, Y. Zhou, and J. Chen, "CNN-based automatic modulation recognition of wireless signal," in *Proc. 2020 IEEE 3rd International Conference on Information Systems and Computer Aided Education*, 2020, pp. 654–659.

[10] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be'ery, "Deep learning methods for improved decoding of linear codes," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 119–131, Feb. 2018.

[11] W. Xu, Z. Wu, Y.-L. Ueng, X. You, and C. Zhang, "Improved polar decoder based on deep learning," in *Proc. 2017 IEEE International Workshop on Signal Processing Systems*, 2017. doi: 10.1109/SiPS.2017.8109997

[12] C.-F. Teng, C.-H. D. Wu, A. K.-S. Ho, and A.-Y. A. Wu, "Low-complexity recurrent neural network-based polar decoder with weight quantization mechanism," in *Proc. 2019 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2019, pp. 1413–1417.

[13] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection," in *Proc. 2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications*, 2017. doi: 10.1109/SPAWC.2017.8227772

[14] X. Yi and C. Zhong, "Deep learning for joint channel estimation and signal detection in OFDM systems," *IEEE Communications Letters*, vol. 24, no. 12, pp. 2780–2784, Dec. 2020.

[15] M. H. E. Ali, "Deep learning-based pilot-assisted channel state estimator for OFDM systems," *IET Communications*, 2021. https://doi.org/10.1049/cmu2.12051

[16] H. A. Hassan, M. A. Mohamed, M. H. Essai, H. Esmaiel, A. S. Mubarak, and O. A. Omer, "Effective deep learning-based channel state estimation and signal detection for OFDM wireless systems," *Journal of Electrical Engineering*, vol. 74, no. 3, pp. 167–176, 2023.

[17] K. Burse, R. N. Yadav, and S. Shrivastava, "Channel equalization using neural networks: A review," *IEEE Trans. on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 3, pp. 352–357, May 2010.

[18] D. F. Carrera, C. Vargas-Rosales, N. M. Yungaicela-Naula and L. Azpilicueta, "Comparative study of artificial neural network based channel equalization methods for mmWave communications," *IEEE Access*, vol. 9, pp. 41678–41687, 2021.

[19] R. Mei, Z. Wang, and W. Hu, "Robust blind equalization algorithm using convolutional neural network," *IEEE Signal Processing Letters*, vol. 29, pp. 1569–1573, 2022. doi: 10.1109/LSP.2022.3189319

[20] J. C. Patra, W. B. Poh, N. S. Chaudhari, and A. Das, "Nonlinear channel equalization with QAM signal using Chebyshev artificial neural network," in *Proc. 2005 IEEE International Joint Conference on Neural Networks*, 2005, vol. 5, pp. 3214–3219.

[21] J. C. Patra, W. C. Chin, P. K. Meher, and G. Chakraborty, "Legendre-FLANN-based nonlinear channel equalization in wireless communication system," in *Proc. 2008 IEEE International Conference on Systems, Man and Cybernetics*, 2008, pp. 1826–1831.

[22] C.-Y. Lo and W.-D. Weng, "Application of neural network techniques on nonlinear channel equalization for 16-QAM modulation systems," in *Prof. 2008 Eighth International Conference on Intelligent Systems Design and Applications, Kaohsuing, Taiwan*, 2008, pp. 356–361.

[23] S. Chen, G. Gibson, C. Cowan, P. Grant, "Adaptive equalization of finite non-linear channels using multilayer perceptrons," *Signal Processing*, vol. 20, no. 2, pp. 107–119, 1990.

[24] J. C. Patra and R. N. Pal, "A functional link artificial neural network for adaptive channel equalization," *Signal Processing*, vol. 43, no. 2, pp. 181–195, 1995.

[25] H. Shi and T. Yan, "Adaptive equalization for QAM signals using gated recycle unit neural network," in *Proc. 2021 3rd International Conference on Advances in Computer Technology, Information Science and Communication*, 2021, pp. 210–214.

[26] H. Ye and G. Y. Li, "Initial results on deep learning for joint channel equalization and decoding," in *Proc. 2017 IEEE 86th Vehicular Technology Conference*, 2017, pp. 1–5. doi: 10.1109/VTCFall.2017.8288419

[27] Y. Hu, L. Zhao, and Y. Hu, "Joint channel equalization and decoding with one recurrent neural network," in *Proc. 2019 IEEE Int. Symposium on Broadband Multimedia Systems and Broadcasting*, 2019. doi: 10.1109/BMSB47279.2019.8971938

[28] X. Li, Y. Zhang, D. Li, P. P. Shum, and T. Huang, "Nonlinear channel equalization using gaussian processes regression in IMDD fiber link," *IEEE Photonics Journal*, vol. 14, no. 6, pp. 1–6, Dec. 2022.

[29] S.-Y. Kwon, J.-H. Kim, and H.-N. Kim, "SVR-based blind equalization on HF channels with a Doppler spread," in *Proc. 2022 International Conference on Artificial Intelligence in Information and Communication*, 2022, pp. 357–359.

[30] E. Giacoumidis, A. Tsokanos, M. Ghanbarisabagh, S. Mhatli and L. P. Barry, "Unsupervised support vector machines for nonlinear blind equalization in CO-OFDM," *IEEE Photonics Technology Letters*, vol. 30, no. 12, pp. 1091–1094, Jun. 2018.

[31] P. M. Olmos, J. J. Murillo-Fuentes and F. Perez-Cruz, "Joint Nonlinear Channel Equalization and Soft LDPC Decoding with Gaussian Processes," *IEEE Trans. on Signal Processing*, vol. 58, no. 3, pp. 1183–1192, March 2010.

[32] L. Salamanca, J. J. Murillo-Fuentes, and F. Pérez-Cruz, "Channel decoding with a Bayesian equalizer," in *Proc. 2010 IEEE International Symposium on Information Theory*, 2010, pp. 1998–2002.

[33] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada*, 2013, pp. 6645–6649.

[34] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, *MIT* Press, 2016.

[35] H. Huang, S. Guo, G, Gui, Z. Yang, J. Zhang, H. Sari, F. Adachi, "Deep learning for physical-layer 5G wireless techniques: Opportunities, challenges and solutions," *IEEE Wireless Communications*, vol. 27, no. 1, pp. 214–222, Feb. 2020.

[36] C. Nathwani, "Online signature verification using bidirectional recurrent neural network," in *Proc. 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India*, 2020, pp. 1076–1078.

[37] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[38] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 15 Nov. 1997.

[39] K. Greff, R. K. Srivastava, J. Koutn ık, B. R. Steunebrink and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[40] X. Wang, Z. Yu and S. Mao, "Deep ML: Deep LSTM for indoor localization with smartphone magnetic and light Sensors," in *Proc.* 2018 *IEEE International Conference on Communications (ICC), Kansas City*, MO, USA, 2018, pp. 1–6. doi: 10.1109/ICC.2018.8422562

[41] M. Anbar, N. Iqbal, A. Zerguine, and M. S. Alouini, "Iterative SC-FDMA frequency domain equalization and phase noise mitigation," in *Proc. 2018 Int. Symposium on Intelligent Signal Processing and Communication Systems*, 2018, pp. 91–95.

[42] H. Sari, G. Karam and I. Jeanclaude, "Transmission techniques for digital terrestrial TV broadcasting," *IEEE Communications Magazine*, vol. 33, no. 2, pp. 100–109, Feb. 1995.

[43] K. Cho, B. Merri ënboer, C. Gulcehre *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc. the 2014 Conf. on Empirical Methods in Natural Language Processing*, pp. 1724–1734.

[44] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[45] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015. https://doi.org/10.1016/j.neunet.2014.09.003

[46] T. Zia and U. Zahid, "Long short-term memory recurrent neural network architectures for Urdu acoustic modeling," *Int. J. Speech Technol.*, vol. 22, pp. 21–30, 2019. https://doi.org/10.1007/s10772-018-09573-7

[47] A. Graves, M. Liwicki, S. Fern ández *et al.*, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, May 2009.

[48] T. Ong. (2017). Facebook's translations are now powered completely by AI. [Online]. Available: https://www.theverge.com/2017/8/4/16093872/facebook-ai-translationsartificial-intelligence

[49] Y. Wu, S. Mike, C. Zhifeng *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint*, arXiv:1609.08144, 2016.

[50] M. H. Essai, A. B. Abdel-Raman, and E. A. Badry, "Developing novel activation functions based deep learning LSTM for classification," *IEEE Access*, vol. 10, pp. 97259–97275, 2022.

[51] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[52] R. Atienza, *Advanced Deep Learning with TensorFlow 2 and* Keras*: Apply DL, GANs, VAEs, Deep RL, Unsupervised Learning, Object Detection and Segmentation, and More*, Packt Publishing Ltd, 2020.

[53] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*, MIT Press, 2012.

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

[55] E. Dogo, O. Afolabi, N. Nwulu *et al.*, "A comparative analysis of gradient descent-based optimization algorithms on convolutional neural networks," *in Proc. 2018 Int. Conf. on Computational Techniques, Electronics and Mechanical Systems*, 2018, pp. 92–99.

[56] E. Jacobsen and P. Kootsookos, "Fast, accurate frequency estimators [DSP Tips & Tricks]," *IEEE Signal Processing Magazine*, vol. 24, no. 3, pp. 123–125, May 2007.

[57] S. Mirzaei, A. Hosangadi, and R. Kastner, "FPGA implementation of high-speed FIR filters using add and shift method," in *Proc. 2006 International Conference on Computer Design*, 2006, pp. 308–313.

[58] P. J. Freire, S. Srivallapanondh, A. Napoli *et al.*, "Computational complexity evaluation of neural network applications in signal processing," 2022. https://doi.org/10.48550/arXiv.2206.12191

[59] C. R. Vogel, *Computational Methods for Inverse Problems*, Philadelphia: Society for Industrial and Applied Mathematics 3600 University City Science Center Philadelphia, PA, United States, 2002.
[60] G. Cuypers, M. Moonen, and Networking, "Frequency-domain equalizers with zero restoration for zero-padded block transmission with high SNR," *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, no. 1, pp. 1–14, 2016.
[61] ITU-R, Guidelines for evaluation of radio transmission technologies for IMT-2000, 1997.
[62] X. Cheng, D. Liu, S. Yan, W. Shi and Y. Zhao, "Channel estimation and equalization based on deep BLSTM for FBMC-OQAM systems," in *Proc. 2019 IEEE International Conference on Communications, Shanghai, China*, 2019, pp. 1–6, doi: 10.1109/ICC.2019.8761647.

Mohamed A. Mohamed Omar was born in Sohag, Egypt in 1987. He received his B.Sc. with an honour degree in electronics and communication engineering from Al-Azhar University, Egypt, in 2010. He received his M.Sc. degree in telecommunications engineering from King Fahd University of Petroleum and Minerals (KFUPM), Dhahran, KSA, in 2018. From 2012 to 2015 and from 2018 to 2020, he was a teaching assistant at Al-Azhar University. Since 2020, he has been an assistant lecturer at Al-Azhar University. Currently, he is a Ph.D. student at Aswan University. His research interests include Wireless Communications, Communication & Signal Processing, and machine learning.

Hassan A. Hassan was born in Luxor, Egypt in 1984. He received a B.S. degree in electrical engineering from Al-Azhar University, Egypt, in 2007, and an M.S. degree in electrical engineering from Al-Azhar University, Egypt, in 2017. From 2012 to 2017 he was a teaching assistant with Al-Azhar University. Since 2017, he has been an assistant lecturer with Al-Azhar University. Currently, he is a Ph.D. student at Aswan University. His research interests include wireless communication, channel estimation of signals, and machine learning.

Mohamed H. Essai Ali was born in El Balyana town, Sohag, Egypt in 1978. He received the B.S. degree in electrical engineering from the Al-Azhar University, Egypt, in 2001, M.S. degree in electrical engineering from Assuit University, Egypt, in 2007, Egypt and the Ph.D. degree in mechanical engineering from Novosibirsk State Technical University, Novosibirsk, Russia, in 2012. From 2001 to 2008, he was a demonstrator and lecturer assistant with Al-Azhar University. From 2009 to 2012 he was a PhD student with Novosibirsk State Technical University. From 2012 to 2018 he was Assistant Professor with Al-Azhar University. From 1st April 2014 to 25 Dec. 2014, he was a Guest Researcher, with Novosibirsk State Technical University. Since 2018, he has been an Associate Professor with the Electrical Engineering Department, Al-Azhar University, Faculty of Engineering. He is the author of five textbooks, more than 50 articles. His research interests include theory and applications of robust statistics, wireless communication, channel estimation of signals in terms of a priori uncertainty for the problems of telecommunications, optical wireless communication, artificial intelligence-based signal processing applications, and FPGA based applications.

Hamada Esmaiel received the B.Sc. degree in electrical engineering from South Valley University, Egypt, in 2005, and the MSc in wireless communications from South Valley University, Egypt in 2010, and the Ph.D. degree in communication engineering from University of Tasmania, Australia in 2015. In 2011 he was a researcher assistant in wireless communication Lab. Wonkwang University, Iksan, South Korea. Since 2015, he has been an assistant professor at Aswan University, Egypt. He is the general co-chair of IEEE IEEE ISWC' 18. He is a technical committee member in many international conferences and a reviewer in many international conferences, journals and transactions. His current research interests are 5G networks, Li-Fi technology, millimeter wave transmissions, underwater communication and MIMO systems. He is an IEEE member.

Osama A. Omer (Member, IEEE) received the B.Sc. and M.Sc. degrees from South Valley University, in 2000 and 2004, respectively, and the Ph.D. degree from the Tokyo University of Agriculture and Technology, in 2009. He has spent six months as a Postdoctoral Researcher with the Medical Engineering Department, Luebeck University, Germany. He has also spent three months as a Postdoctoral Researcher at Kyushu University, Japan. The last but not least, he has spent six months as a Research and Development Scientist Engineer at the NOKIA Research and Development Center, Tokyo/Japan, in 2008. He is currently a Full Professor at the Faculty of Engineering, Aswan University. His research interests include wireless communications, deep learning, and image/signal processing.

Ahmed S. Mubarak received his B.Sc. degree in electrical engineering from South Valley University, Egypt, Egypt, in 2001, and his M.Sc. degree in electrical engineering from Assiut University, Egypt, in 2012, and his Ph.D. degree in electrical engineering from Aswan University, Egypt, in 2019. Currently, he is an assistant professor in Aswan Faculty of Engineering, Aswan, Egypt. His research interests are in the broad areas of wireless communications including 5G & 6G networks, millimeter wave/THz transmissions and MIMO systems, communication theory, signal processing, and deep learning.