Reduced-Complexity Estimation of FM Instantaneous Parameters via Deep-Learning

Huda Saleem Razzaq
Computer Science Department
Computer Science and Mathematics
Najaf, Iraq
hudas.alsarayefi@uokufa.edu.iq
https://orcid.org/0009-0003-5135-4837

Zahir M. Hussain
Computer Science Department
Computer Science and Mathematics
Najaf, Iraq
zahir.hussain@uokufa.edu.iq
https://orcid.org/0000-0002-1707-5485

DOI: http://dx.doi.org/10.31642/JoKMC/2018/100107

Received Oct. 20, 2022. Accepted for publication Jan. 28, 2023

***Abstract*—** *Parameter estimation is a fundamental problem in signal processing. Deep learning is a fundamental method to solve this problem. This paper used five Deep Learning (DL) methods and three datasets including different singles Single Tone (ST), Linear- Frequency-Modulated (LFM), and Quadratic-Frequency-Modulated (QFM). This signal is affected by a mixture of Additive White Gaussian (AWG) noise and Additive Symmetric alpha Stable (SαS) noise. The recurrent neural network includes Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), Bi-Direction Long Short-Term Memory (BiLSTM), and Convolution Neural Network includes 1D-CNN and 2D-CNN. DL can significantly reduce estimation complexity, memory cost, and power consumption as compared to the classical methods based on time-frequency analysis, which are important requirements for many systems, such as some Internet of Things (IoT) sensor applications. All DL methods are designed with fewer layers to reduce complexity, especially in hardware. Geometric SNR (GSNR) is used to determine the impulsiveness of mixture noise in a Gaussian and SαS noise. When compared to a deep learning classifier with few layers to get on high accuracy and complexity reduces for Instantaneous Frequency (IF) estimation, Linear Chirp Rate (LCR) estimation, and Quadratic Chirp Rate (QCR) estimation. IF, LCR, and QFM estimation for ST, LFM, and QFM signals. The results show that 2D-CNN is better than other deep methods for parameter estimation in LFM signals and QFM signals, and the GRU is better for parameter estimation in ST signals, where the accuracy of the ST dataset in GRU is 58.09. The accuracy of the LFM and CFM datasets in 2D-CNN is 98.26 and 98.2 respectively.*

Keywords*— *Instantaneous frequency estimation, ST, LFM, QFM signal, sensors, Gaussian noise, SαS noise, GRU, LSTM, BiLSTM,1D- CNN, 2D-CNN, deep learning, ROC, and GSNR.

I. INTRODUCTION

The non-stationary frequency modulated (FM) signals can be used to represent many realistic signals utilized in radar, sonar, medicinal applications, and wireless communications [1, 2]. The nonparametric Instantaneous Frequency (IF) estimation of a single tone (mono-component), linear (multi-component), and non-linear (multi-component) frequency modulated (FM) nonstationary signals are present in this paper. Huang B., & et al. (2020) FreqEnet (Frequency estimation network) is a suggested framework for estimating frequency using deep learning. The signal frequency estimation is a regression problem that can be predicted using the LTSM module. The architecture is extremely compact, with only three LSTM and one fully linked layer. For training our model, two periodic signals are generated. as well as to evaluate the resilience and generalization of the original signal, uniform and Gauss white noise are introduced. Bin Huang et al. (2020) introduce a network for estimating frequency based on RNN deep learning. The signal frequency estimation is predicted with the LTSM module. the layers of the RNN model are three LSTM layers and one fully connected layer. Two periodic signals generated are the first single sine wave signal with adding random noise, second mixed periodic signal is the fusion of three sinusoidal signals with different phases and noise. the Gaussian noise with $SNR \in (2, 4, 6, 8, 10)$. The Adam optimization is applied and different epochs and learning rates. The dataset with 18000 samples for training, 2000 samples for validation, and 2000 samples for testing. The sampling frequency of the sample is 60Hz, and the length of each sample is 60 points [3]. Vu et al. (2016) For relation categorization, examine CNN and simple RNN (no gating methods). They show that CNN outperforms RNN and present proof that CNN and RNN provide complementary information: while RNN computes a weighted combination of all words in the sentence, CNN isolates the most informative NGRAMS for the relation and only analyzes their resulting activations [4]. Yin W. (2017) Deep neural

networks (DNNs) have transformed natural language processing (NLP). The two basic types of DNN architectures, Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), are intensively investigated to handle diverse NLP problems. CNN is thought to be good at extracting position-invariant characteristics, but RNN is good at modeling units in sequence [5]. Wu H. & et al. (2020) use one-dimensional Convolution Neural Networks (1D-CNNs) to extract detailed temporal structure information at each signal node and a bidirectional Long Short Term Memory (BiLSTM) network to extract the spatial relationship among the different signal nodes, and then propose a novel identification method by treating the spatial- and temporal-information in a different way [6]. Huda Saleem, and Zahir M. (2021) It is researched how to estimate the instantaneous frequency (IF) under Additive White Gaussian Noise (AWGN) and Additive Symmetric -alpha Stable Noise (SαSN). Based on the highest probability for the Short Time Fourier Transform, two TFD MATLAB algorithms (pspectrum and spectrogram) were investigated (STFT) [7]. ElMoaqet H., & et al. (2020) For automatic feature extraction and detection of apnea occurrences from single respiratory channel inputs, a deep recurrent neural network (RNN) architecture is created. The suggested deep RNN model investigates long short-term memory (LSTM) and bidirectional long short-term memory (BiLSTM). The suggested architecture is tested using three different respiratory signals: oronasal thermal airflow (FlowTh), nasal pressure (NPRE), and abdomen respiratory inductance plethysmography (ABD) [8]. Zahraa C., and Zahir M. (2017) the performance of instantaneous frequency estimators of mono-component FM signals (a single-tone sinusoid) under AWGN with various types of multiplicative noise was investigated. Two basic estimators are considered for single-tone signals: maximum likelihood (ML) estimator employing Discrete Fourier Transform (DFT) with interpolated peak estimate, and autocorrelation approach [9]. Iman Sajedian and Junsuk Rho (2019) a noisy sinusoidal wave frequency was determined using a deep learning network. The wave effect by Gaussian noise with SNRdB equal 25. A three-layer neural network was used to extract the frequency of sinusoidal waves mixed with white noise at a signal-to-noise ratio of 25 dB. The two neurons for the first hidden and second layer, and three neurons for the third hidden layer. ANN used a very small number of neurons to prevent the model from overfitting. Many methods can be used to prevent overfitting; examples include using dropout, reducing the complexity of the model, or increasing the amount of data. The deep-learning algorithm can find the frequency of a sinusoidal wave that is polluted by Gaussian noise. Neural Networks (NNs), belong to the family of deep-learning methods. Datasets with 100,000 noisy waves from 1 kHz to 10 kHz were generated for training and testing the model. Used 72% of the waves as the training dataset, 18% as the validation dataset, and 10% as the test dataset. The Nesterov–Adam optimizer was used with a learning rate of 0.001 [10].

The rest of this paper is structured as follows: Section II shows the problem statement. Section III shows its objectives. Section IV is about contributions. Section V shows challenges. Section VI shows Instantaneous Frequency (IF) and Frequency Modulation (FM). Section VII is Gated Recurrent Unit (GRU). Section VIII introduces Long Short-Term Memory (LSTM). Section IX explains Bidirectional LSTM (BiLSTM). Section X discusses Convolution Neural Network (CNN). Section XI discusses parameter estimation by deep learning methodologies. Section XII is the dataset generated. Section XIII introduces the discussion of the results. Section XIV is a conclusion.

II. PROBLEM STATEMENT

This work will discuss the Instantaneous parameters estimation problem for FM signals under noise environments. Impulsive noise is the real problem. An essential kind of impulse noise is the symmetric α-stable noise. The FM signals affected mixture of α-stable noise is a kind of non-Gaussian noise and Gaussian noise. Impulse noise is typically associated with Gaussian noise, making the estimation problem more difficult.

III. OBJECTIVES

The aim is the estimation of the instantaneous parameters including IF, LCR, and QCR for noisy FM signals, where the objectives are: First, the FM sinusoidal waves generate such as LFM and QFM signals. Second, these signals are affected by a mixture of noise AWGN and SαSN. Third, RNN includes GRU, LSTM, and BiLSTM models that are applied to noisy signals for instantaneous parameter estimation. Finally, CNN includes ID-CNN and 2D-CNN models applied for the noisy input signals to instantaneous parameter estimation. Applications of this work are RADAR and medical SONAR. The effects are better localization for RADAR and better diagnosis in medical SONAR. The improvement of this work is an accurate and fast estimation of the parameters.
The proposed approach limitation is required powerful hardware, high processing for the processor, and storage memory.

IV. CONTRIBUTIONS

This work is the first attempt to employ ANN including CNN to estimate instantaneous parameters for noisy FM waves has been developed. The efficient structure development for CNN and RNN models, where reducing the complexity of the system by reducing the number of network layers. The previous works focus on classifying the types of signals or it is to estimate the frequency using deep neural networks. As for our work, we employ convolutional neural networks and recurrent neural networks to estimate the parameters depending on the prediction result.

V. CHALLENGES

The challenges of this work include first: Noisy FM signals were processed in the time domain without converting to the frequency domain to avoid the complexity of the conversion methods. Second, the noisy one-dimensional signals are converted into two-dimensional signals without using TFD. Third, obtain high accuracy in the presence of

impulsive noise without the use of denoise methods, under low GSNR.

VI. INSTANTANEOUS FREQUENCY AND FREQUENCY MODULATION

The instantaneous frequency, which describes the frequency content's variations with time, is an essential characteristic of FM signals. The IF of a signal is a derivative of its instantaneous phase (θ(t)) concerning time [11-12]:

$$f_i(t) = \frac{1}{2\pi}\frac{d\theta(t)}{dt} \quad (1)$$

$$\theta = 2\pi(f_o t + \delta\frac{t^2}{2} + \rho\frac{t^3}{3}) \quad (2)$$

The signal model having Linear Frequency Modulation (LFM) law is:

$$s(t) = A\, e^{j2\pi\left(f_o t+\frac{\delta}{2}t^2\right)} \quad (3)$$

where δ is the linear modulation index, f_o is the initial frequency (in Hertz), and A is the amplitude. Using Eq. (1), the LFM signal IF will be:

$$f_i(t) = f_o + \delta t \quad (4)$$

Quadratic Frequency Modulation (QFM) signal has also been considered in this work with quadratic IF law as follows:

$$s(t) = A\, e^{j2\pi\left(f_o t+\frac{\delta}{2}t^2+\frac{\rho}{3}t^3\right)} \quad (5)$$

where ρ is the quadratic modulation index of the QFM signal, with the quadratic IF law:

$$f_i(t) = f_o + \delta t + \rho t^2 \quad (6)$$

VII. GATED RECURRENT UNIT

GRU is from RNN type an abbreviation for "Gated Recurrent Unit." GRUs were first used in 2014. They are comparable to LSTMs but less complex. GRU generates the current value of the hidden state h_t by performing linear interpolation between an intermediate candidate hidden state $\tilde{h}_t$ and the prior value of the hidden state h_{t-1}. A GRU has two gates that are updated gate z_t that controls how much of the previous state is overwritten, and a reset gate r_t that controls how much of the previous state is forgotten when computing the candidate's hidden state. Fig. (1) shows GRU architecture [13]. The feed-forward procedure of GRU is as follows:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \quad (7)$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \quad (8)$$

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \times h_{t-1}, x_t] + b_h) \quad (9)$$

$$h_t = (1 - z_t) \times h_{t-1} + z_t \times \tilde{h}_t \quad (10)$$

Where σ & tanh are activation functions, h_{t-1} is the hidden state at the previous time t-1, $\tilde{h}_t$ is the hidden state at the current time t, x_t is the input vector, W_z, W_r, W_h are weights parameters, and b_z, b_r, b_h are bias vectors, r_t is reset gate function, z_t is update gate function, and h_t is the hidden state.

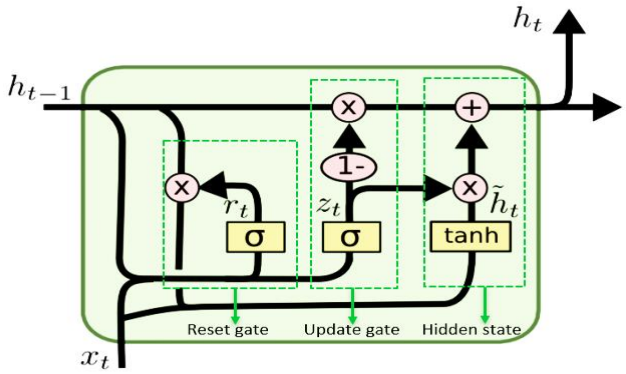

Fig. 1. Gated Recurrent Unit RNN (LSTM) Architecture [14].

VIII. LONG SHORT-TERM MEMORY

Hochreiter and Schmidhuber [15] introduced LSTM, a type of RNN, in 1997. LSTM architecture replaces the traditionally hidden layers with LSTM cells. The cells are made up of numerous gates that can control the flow of input. An LSTM cell is made up of an input gate, a cell state, a forget gate, and an output gate. It also includes a sigmoid layer, a tanh layer, and point-wise multiplication. The input gate is made up of the input. Cell State: Runs throughout the network and can add or remove information using gates. Forget gate layer: Determines the fraction of information that will be allowed. Output gate: This is made up of the LSTM's output. The sigmoid layer yields numbers ranging from 0 to 1, indicating how much of each component should be allowed through. The tanh layer creates a new vector that is added to the state. The cell state is updated based on the gate outputs [16]. Fig. (2) shows LSTM architecture. The feed-forward procedure of LSTM is as follows:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}.x_t] + b_f\right) \quad (11)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}.x_t] + b_i) \quad (12)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}.x_t] + b_o) \quad (13)$$

$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}.x_t] + b_c) \quad (14)$$

$$C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t \quad (15)$$

$$h_t = o_t \times \tanh(C_t) \quad (16)$$

Where σ & tanh are activation functions, h_{t-1} is the hidden state at the previous time t-1 (short-term memory), h_t is the hidden state at the current time t, c_{t-1} is hidden cell state at previous time t-1 (long-term memory), $\tilde{C}_t$ is hidden cell state at current time t, x_t is the input vector, $W_f.W_i.W_c.W_o$ are weights parameters, $b_f.b_i.b_c.b_o$ are bias vectors, f_t is forget gate function, i_t is input gate function, and o_t is output gate function.

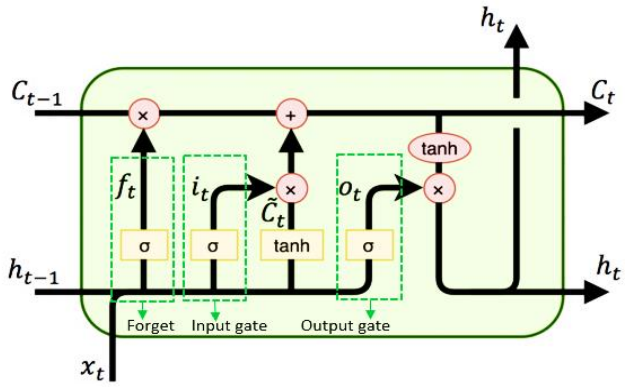

Fig. 2. Long Short-Term Memory RNN (LSTM) Architecture [17].

IX. BIDIRECTIONAL LSTM

To increase classification performance, improve the LSTM unit and provide a BiLSTM model. In unidirectional LSTMs, information travels from backward to forward, whereas bidirectional LSTMs use hidden states to forward information from backward to forward and forward to backward. This aids in the learning of LSTM networks. apply bidirectional training to a Long Short-Term Memory (LSTM) network for the first time [18]. It consists of two LSTMs, one LSTM is trained by taking the sequential input data from the forward, and the other LSTM trains by taking data from the backward direction. Doing this increases the amount of information available for classifying the data, improving the performance compared to a traditional LSTM. Fig. (3) shows bidirectional recurrent neural network (BiLSTM) architecture.

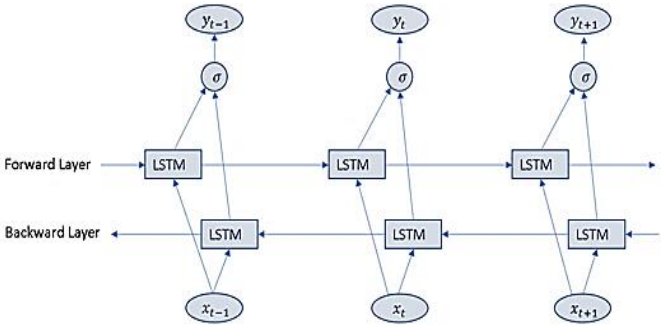

Fig. 3. Bidirectional Recurrent Neural Network (BiLSTM) Architecture [19].

X. CONVOLUTION NEURAL NETWORK

Convolutional neural networks, or CNNs, are a type of neural network that is used to process data using a known, grid-like architecture. This includes time-series data, which is a 1D grid of pixels, and image data, which is a 2D grid of pixels. Because the network incorporates a mathematical procedure known as convolution [20-22]:

First, the convolutional layer is the foundation of a convolutional neural network and is designed to overcome the constraints of fully linked layers.

$$\boldsymbol{x}_f^{(\ell)} = \sum_{u,v} \boldsymbol{x}_{uv}^{(\ell-1)} * \boldsymbol{w}_f^{(\ell)} + \boldsymbol{b}_f^{(\ell)} \tag{17}$$

where $\boldsymbol{x}_f^{(\ell)}$ represents the current layer's output for a given filter f, $\boldsymbol{x}_{uv}^{(\ell-1)}$ the output of the previous layer and the spatial extent of the filter in the horizontal and vertical direction is given by u and v.

Second, the pooling layer divides the input volume into a collection of non-overlapping rectangles and outputs the maximum activation for each subregion, hence the name max-pooling.

$$x^{(\ell)} = \max_{u,v}\left(x^{(\ell-1)}\right)_{uv} \tag{18}$$

where u and v denote the spatial extent of the non-overlapping regions in width and height.

The third Full Connect (FC) layer is comparable to the hidden layer. The fully connected layer of a convolutional neural network is responsible for high-level reasoning and is hence typically put after the convolutional layers.

$$x^{(\ell)} = \left(w^{(\ell)}\right)^T x^{(\ell-1)} + b^{(\ell)} \tag{19}$$

where $x^{(\ell-1)}$ denotes the activations of the previous layer and $x^{(\ell)}$, $w^{(\ell)}$, and $b^{(\ell)}$ denote the activations, weights, and biases of the current layer, respectively.

The fourth Linear Unit Rectified (ReLU) activation function is linear and has a simple threshold at zero, it may be written as:

$$ReLU(x) = \max(0, x) \tag{20}$$

Fifth the loss of cross entropy adapted to many classes via the softmax function and the negative log-likelihood. The cross-entropy loss has the following mathematical formula:

$$\text{LF} = -\frac{1}{N}\sum_{i}^{N}\sum_{j}^{M} y_{ij}\log\left(p_{ij}\right) \tag{21}$$

Where N is the number of rows, and M is the number of classes. Batch normalization places after a convolutional or fully connected layer. It is added to compute the mini-batch mean and variance during the forward pass.

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{22}$$

$$y_i = \gamma\hat{x}_i + \beta \tag{23}$$

where offset γ =1, Epsilon ε =0, scale β =0, variance σ^2, and mean μ.

XI. THE PARAMETERS ESTIMATION BY DEEP LEARNING METHODOLOGIES

In this section, we demonstrate the implementation of deep learning algorithms for the parameter estimation of noisy FM signals. The dataset is generated with noise as explained in [7], where three noisy datasets are generated including ST, LFM, and QFM signals. Then five deep learning methods are applied including GRU, LSTM, BiLSTM, 1D-CNN, and 2D-CNN. CNN and RNN are implemented with the parameters including learning rate equal 1e-4, it is used in the backward stage of learning improvement. Max epochs = 10, mini-batch size = 8, and Adam used initially learn rate= 0.001, Epsilon= 0.00000001, squared gradient decay factor = 0.999, and gradient decay factor = 0.9000. Hyperparameters in convolution are the number of filters, size of the filter, and

padding. The batch normalization is included ε, β, and γ. The max pool is included in size and stride. The dropout layer is included probability. The procedures of RNN and CNN illustration as follows:

1. GRU Model

- The input dataset ($Data\_X$) includes ST, LFM, & QFM and labels ($Data\_L$), where $Data\_X$ include 12120 noisy FM signals with length 1×201 and $Data\_L$ include ten labels from zero to nine.
- Divided the dataset into 85% training (XTr, LTr), 5% validation (XVa, LVa), and 10% testing (XTe, LTe).
- The sequence input layer with length 1×201.
- GRU layer is applied with the number of hidden units is 180 units.
- Fully connected is used with an output equal to ten.
- Softmax is applied on the output network with a range between zero and one.
- Calculate the error of the output layer by cross-entropy.

$$ETr_j = -\sum_{i=1}^{n} LTr_i \cdot \log\left(\widehat{LTr}_i\right) \quad (24)$$

Where n is the number of classes, $\widehat{LTr}_i$ is predicate labels, LTr_i is actual labels in one-hot encoding, $j \in [1. \dots N]$, N is the number of samples training.

- Run backward stage includes calculating Adam relies on the gradient for softmax and loss function, and calculating the weight (w_t) updates as follows:

$$w_t = w_{t-1} - \eta \frac{\widehat{m}_t}{\sqrt{\widehat{v}_t + \epsilon}} \quad (25)$$

- Evaluate the training data by using validation data for each iteration:

$$EVa_j = -\sum_{i=1}^{n} LVa_i \cdot \log\left(\widehat{LVa}_i\right) \quad (26)$$

- Repeat implement the steps for all training data, that access the last iteration:
- The classification layer is used cross entropy to compute the loss of the network. Training loss is calculated by taking the sum of errors for each sample in the training set:

$$Loss_{Tr} = \frac{1}{N}\sum_{i=1}^{N} ETr_i \quad (27)$$

Also, find the loss of validation as follows:

$$Loss_{Va} = \frac{1}{Nv}\sum_{i=1}^{Nv} EVa_i \quad (28)$$

- Repeat and implement the steps until reach into last epoch (finish training).
- The GRU model trained is returned.
- The testing set (XTe, LTe) used the GRU model trained with optimal weights to predict the labels, where each label represents one IF, LCR, and QCR according to the use of the dataset.
- P_f, P_S, P_q represent the predicted IF, LCR, & QFM respectively according to the predicted label. Then, Estimate IF as follows:

$$IF = P_f + P_s\, t + P_q\, t^2$$

- Evaluation of the GRU model trained using metrics is accuracy, precision, recall, F-score, ROC, and confusion matrix.

Other models of recurrent neural networks (LSTM & BiLSTM) implement the same procedure described for the GRU model, but the difference is that instead of the GRU layer, we put the LSTM layer with some hidden units of 180 units. As well as for the BiLSTM layer with the number of hidden units is 180 units.

2. 1D-CNN Model

- The input dataset ($Data\_X$) includes ST, LFM, & QFM and labels ($Data\_L$), where $Data\_X$ include 12120 noisy FM signals with length $201 \times 1 \times 1$and $Data\_L$ include ten labels from zero to nine.
- Divided the dataset into 85% training (XTr, LTr), 5% validation (XVa, LVa), and 10% testing (XTe, LTe).
- The sequence input layer with size $201 \times 1 \times 1$.
- The batch normalization is applied to the input.
- The convolutional layer is applied, with the number of filters 30 and size 1×9.
- The batch normalization is applied to the above step.
- ReLU is used as an activation function.
- Convolution, batch normalization, and ReLU are applied to find feature maps with the number of filters 60, 90, and 128, respectively, and size 1×9.
- Fully connected with 500 output and dropout with probability equal to 50%.
- Fully connected is used with an output equal to ten.
- Softmax is applied to the output network.
- Calculate the error of the output layer by cross-entropy.
- Run backward stage includes calculating Adam relies on the gradient for the softmax loss function, and calculating the weight (w_t) updates.
- Evaluate the training data by using validation data for each iteration.
- Repeat the implement steps for all training data, that access the last iteration.
- The classification layer is used cross entropy to compute the loss of the network.
- Repeat the implement steps, until reaches into last epoch (finish training).
- The 1D-CNN model trained is returned.
- The testing set (XTe, LTe) used the 1D-CNN model trained with optimal weights to predict the labels, where each label represents one IF, LCR, and QCR according to the use of the dataset.

- P_f, P_S, P_q represent the predicted IF, LCR, & QFM respectively according to the predicted label. Then, Estimate IF as follows:

$$IF = P_f + P_s\, t + P_q\, t^2$$

- Evaluation of the 1D-CNN model trained using metrics is accuracy, precision, recall, F-score, ROC, and confusion matrix.

3. 2D-CNN Model

- The input dataset ($Data\_X$) includes ST, LFM, & QFM and labels ($Data\_L$), where $Data\_X$ include 12120 noisy FM signals with length $28 \times 28 \times 1$and $Data\_L$ include ten labels from zero to nine.
- Divided the dataset into 85% training (XTr, LTr), 5% validation (XVa, LVa), and 10% testing (XTe, LTe).
- The image input layer with a size of $28 \times 28 \times 1$.
- The batch normalization is applied to the input.
- The convolutional layer is applied, with the number of filters being 30 and size 3×3.
- Batch normalization is applied to the result of step 2.
- ReLU is used as an activation function.
- Max pooling is applied to reduce the features map.
- Convolution, batch normalization, ReLU, and max pooling are applied to find feature maps with the number of filters 60, 90, and 128, respectively, and size 3×3.
- Fully connected with 100 output and dropout with probability equal to 50%.
- Fully connected is used with an output equal to ten.
- Softmax is applied to the output network.
- Calculate the error of the output layer by cross-entropy.
- Run backward stage includes calculating Adam relies on the gradient for the softmax loss function, and calculating the weight (w_t) updates.
- Evaluate the training data by using validation data for each iteration.
- Repeat the implement steps for all training data, that access the last iteration (one epoch completed).
- The classification layer is used cross entropy to compute the loss of the network.
- Repeat the implement steps, until reaches into last epoch (finish training).
- The 2D-CNN model trained is returned.
- The testing set (XTe, LTe) used the 2D-CNN model trained with optimal weights to predict the labels, where each label represents one IF, LCR, and QCR according to the use of the dataset.
- P_f, P_S, P_q represent the predicted IF, LCR, & QFM respectively according to the predicted label. Then, Estimate IF as follows:

$$IF = P_f + P_s\, t + P_q\, t^2$$

- Evaluation of the 2D-CNN model trained using metrics is accuracy, precision, recall, F-score, ROC, and confusion matrix.

XII. DATASETS GENERATED

Noisy ST, LFM, and QFM signals are generated with a Geometric SNR range $[-50\ \ 50]$. In this work, signals generate with additive white Gaussian noise and symmetric stable noise. Please note that the selection of initial frequency $f1$ and final frequency $f2$ (or slope $\delta\ or\ e1$, instead) can be different for different radar/sonar systems, and for bio applications, they are generally low [23], as we considered in the current paper. In classical radar systems, Authors use the normalized frequency f/fs (fs being the sampling frequency), hence, specific values of f will not be effective, as confirmed in Ref [24]. Therefore, FM estimation approaches should be generic, not limited to specific $f1$ or slope/LCR $\delta/e1$. In [25], values of slopes over [0,1] have been considered, although higher values are also possible. In deep learning, we need to generate huge data, so 12120 noisy FM signals were generated, and distributed over ten classes, and each class contains 1212 noisy FM signals. The recurrent neural networks and one-dimensional convolutional neural networks deal with the data serial time or sequential data, so noisy FM signals were generated with one dimension and length 201, and then trained by training models that include GRU, LSTM, BiLSTM, and 1D-CNN. The convolutional neural networks deal with two-dimensional images, so one-dimensional noisy FM signals with a length of 784 were generated, and then converted into a two-dimensional image of size 28×28, then trained by training models that include 2D-CNN. The datasets for ST, LFM, and QFM signals are generated with the same parameters for noise. Each dataset training by five deep-learning methods. The dataset was randomly divided into a training ratio of 85%, a validation ratio of 5%, and a test ratio of 10%. The IF range includes the initial frequency is $f1 = 10,$ and the final frequency is $f2 = 19$. The number of frequencies is $f = 10$, different frequency is $df = \left\lfloor \frac{f2-f1}{nf} \right\rfloor$. The range IF is $fr = [\, f1 : increasing\ by\ df : f2\,]$. The LCR range includes the initial LCR is $e1 = 0.1,$ the final LCR is $e2 = 0.9$. The number of LCR is $ne = 10,$ different LCR is $de = \frac{e2-e1}{ne}$. The range LCR is $er = [e1 : increasing\ by\ de : e2\,]$. The QCR range includes the initial QCR is $q1 = -0.9,$ and the final QCR is $q2 = -0.1$. The number of QCR is $nq = 10$, different QCR is $dq = \frac{q2-q1}{nq}$. The range QCR is $qr = [q1 : increasing\ by\ dq : q2\,]$.

XIII. DISCUSSION OF THE RESULTS

The results show high accuracy for instantaneous parameter estimation. For measuring the efficiency of the supervised training model or the performance efficiencies of the prediction algorithms, there are many metrics including accuracy, precision, recall, F-score, confusion matrix, and ROC [26], as shown in some Figures and Table I. The accuracy of the supervised model is simply the number of correct predictions on the total number of predictions. Precision measures how

good the model is at correctly identifying the positive class. Recall shows how good the model is at correctly predicting all the positive signals in the dataset. The F1-score is the harmonic mean of precision and recall. An accuracy, precision, F-score, and recall scores will give a value between 0 and 100, a value of 100 would indicate a perfect model. The confusion matrix is a matrix that compares the number of predictions for each class that are correct and those that are incorrect. ROC curves plot the accuracy of the model and therefore are best suited to diagnose the performance of models where the data is balanced. In SαS is an impulsive model, where alpha is more harmful even if it is of small value, where it affects the parameters guess. Deep CNN outperformed artificial neural networks in estimating the instantaneous frequency of non-stationary data. Fig. (4) shows the ROC of the ST dataset by using GRU, where ROC curves are an important tool for evaluating the performance of a machine-learning model. The ROC figure shows the relationship between the True Positive Rate (TPR) for the model and the False Positive Rate (FPR). The best classifications will show the receiver operating line hugging the left and top sides of the plot axis. Fig. (5) shows the FE of the ST dataset by using GRU. Fig. (6) shows the ROC of the LFM dataset by using 2D-CNN. Fig. (7) shows the FE and LCR of the LFM dataset by using 2D-CNN. Fig. (8) shows the ROC of the QFM dataset by using 2D-CNN. Fig. (9) shows the FE and LCR of the QFM dataset by using 2D-CNN. Fig. (10) shows the confusion matrix of the ST signals by using GRU, where confusion matrix rows of a confusion matrix correspond to the predicted class and the columns correspond to the target class. The diagonal cells correspond to observations that are correctly classified. The off-diagonal cells correspond to incorrectly classified observations. The column on the far right of the plot shows the percentages of all the samples predicted to belong to each class that is correctly and incorrectly classified. The row at the bottom of the plot shows the percentages of all the samples belonging to each class that is correctly and incorrectly classified. The cell in the bottom right of the plot shows the overall accuracy. Fig. (11) shows the confusion matrix of the LFM signals by using 2D-CNN. Fig. (12) shows the confusion matrix of the QFM signals by using 2D-CNN. Fig. (13) shows the accuracy and loss rate for noisy ST signals by GRU. Fig. (14) shows the accuracy and loss rate for noisy LFM signals by 2D-CNN. Fig. (15) shows the accuracy and loss rate for noisy QFM signals by 2D-CNN. Fig. (16) shows the accuracy and loss rate for noisy LFM signals by LSTM. Fig. (17) shows the accuracy and loss rate for noisy QFM signals by BiLSTM. Tab. 1 shows the measurement results for ST, LFM, & QFM datasets and GRU, LSTM, BILSTM, 1D-CNN, & 2D-CNN deep learning methods.

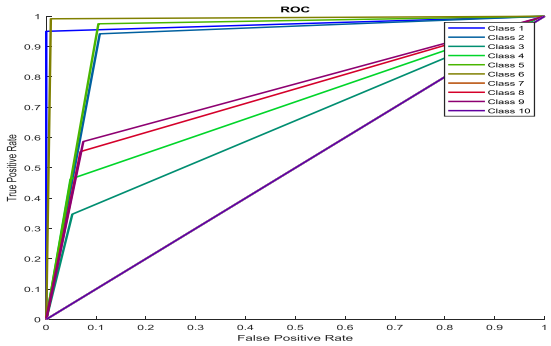

Fig. 4. ROC of the ST Dataset by Using GRU.

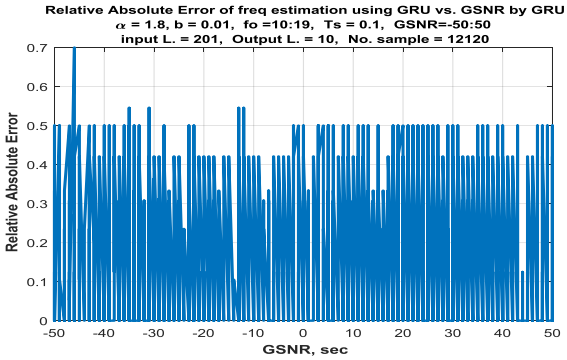

Fig. 5. FE of the ST Dataset by Using GRU.

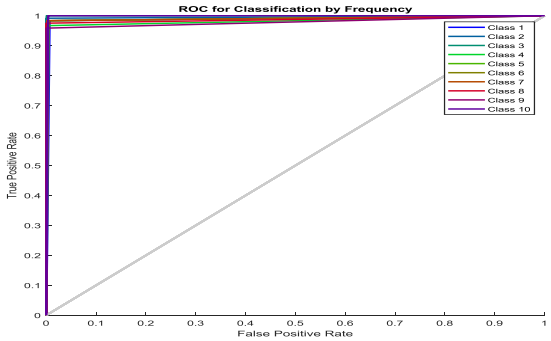

Fig. 6. ROC of the LFM Dataset by Using 2D-CNN.

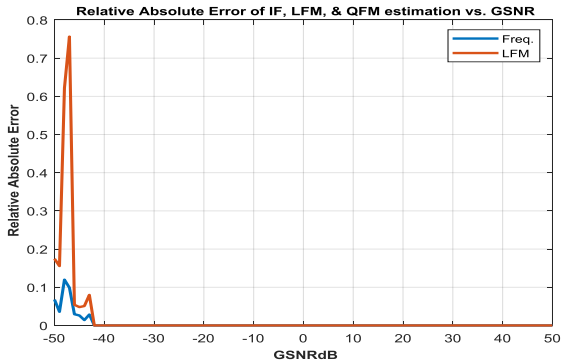

Fig. 7. FE and LCR of the LFM Dataset by Using 2D-CNN.

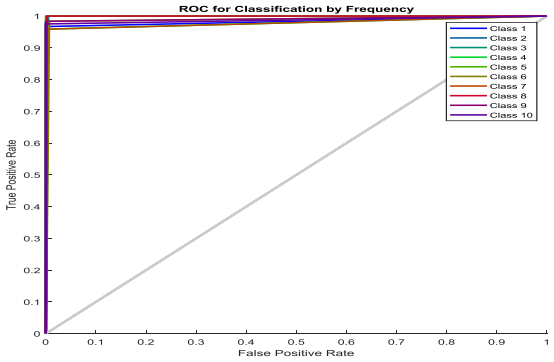

Fig. 8. ROC of the QFM Dataset by Using 2D-CNN.

TABLE I. THE MEASURES RESULTS FOR ST, LFM, & QFM DATASET AND GRU, LSTM, BILSTM, 1D-CNN, & 2D-CNN DEEP LEARNING METHODS.

Dataset	Methods	Accuracy	Precision	Recall	F1-score	Epoch	Time (Sec)
ST dataset	GRU	**58.0992**	**48.0214**	**58.0992**	**52.5818**	**200**	**316**
	LSTM	46.6116	31.7648	46.6116	37.7820	200	429
	BiLSTM	45.9504	34.2707	45.9504	39.2603	200	1514
	1D-CNN	51.4876	42.0942	51.4876	46.3195	10	4118
	2D-CNN	54.1322	50.3689	54.1322	52.1828	10	4935
LFM dataset	GRU	82.8926	77.9568	82.8926	80.3489	200	297
	LSTM	66.2810	50.7212	66.2810	57.4665	200	422
	BiLSTM	69.9174	64.0148	69.9174	66.8360	200	914
	1D-CNN	74.7934	72.8796	74.7934	73.8241	10	35826
	2D-CNN	**98.2645**	**98.3087**	**98.2645**	**98.2866**	**10**	**5420**
QFM dataset	GRU	78.7603	67.6727	78.7603	72.7968	200	293
	LSTM	67.8512	51.3631	67.8512	58.4670	200	425
	BiLSTM	69.9174	52.4809	69.9174	59.9572	200	1097
	1D-CNN	75.8678	73.2224	75.8678	74.5216	10	3872
	2D-CNN	**98.2645**	**98.3062**	**98.2645**	**98.2853**	**10**	**5629**

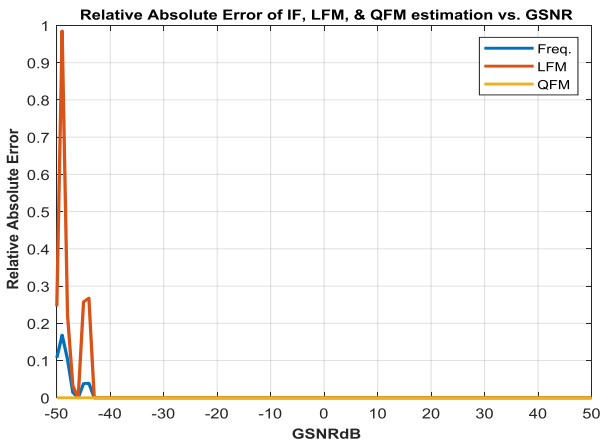

Fig. 9. FE and LCR of the QFM Dataset by Using 2D-CNN.

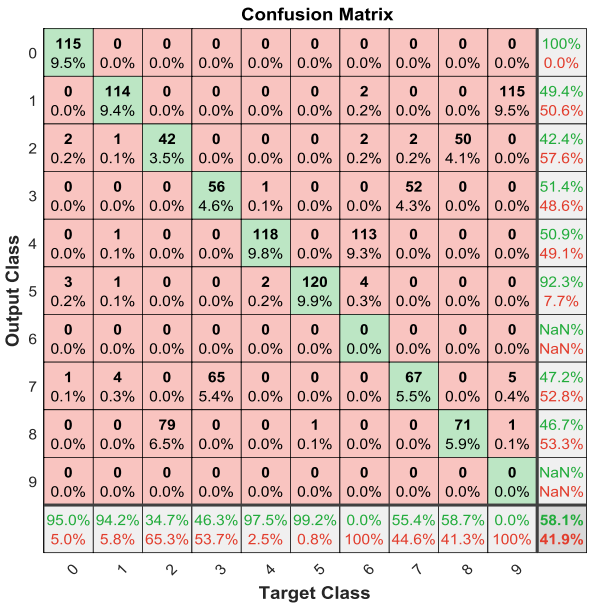

Fig. 10. A Confusion Matrix of the ST Signals by Using GRU.

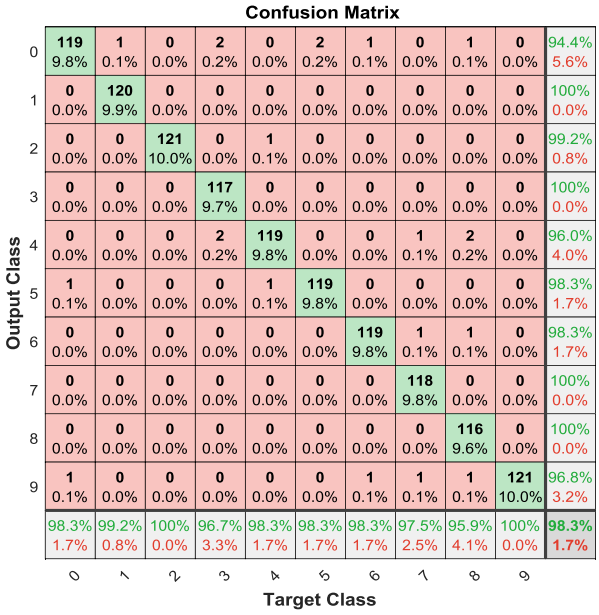

Fig. 11. A Confusion Matrix of the LFM Signals by Using 2D-CNN.

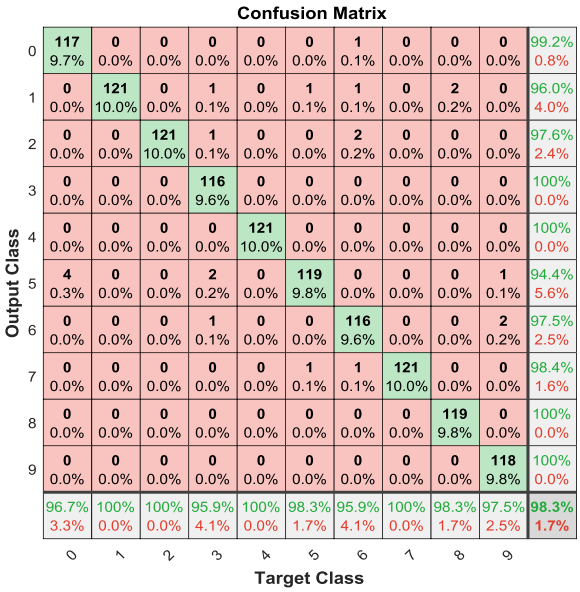

Fig. 12. A Confusion Matrix of the QFM Signals by Using 2D-CNN.

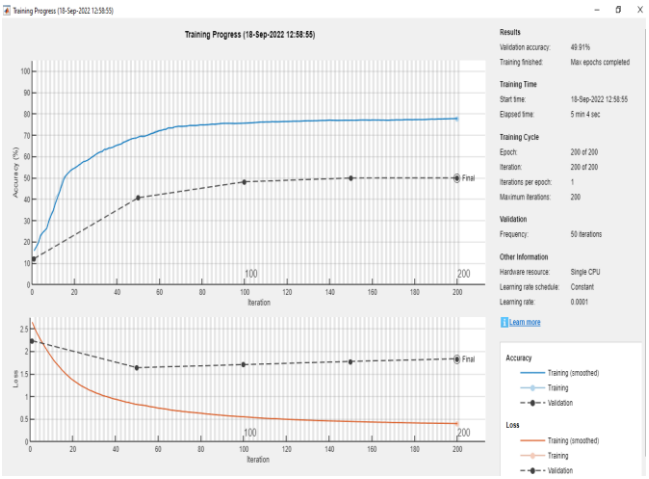

Fig. 13. Accuracy and Loss Rate for Noisy ST Signals by GRU.

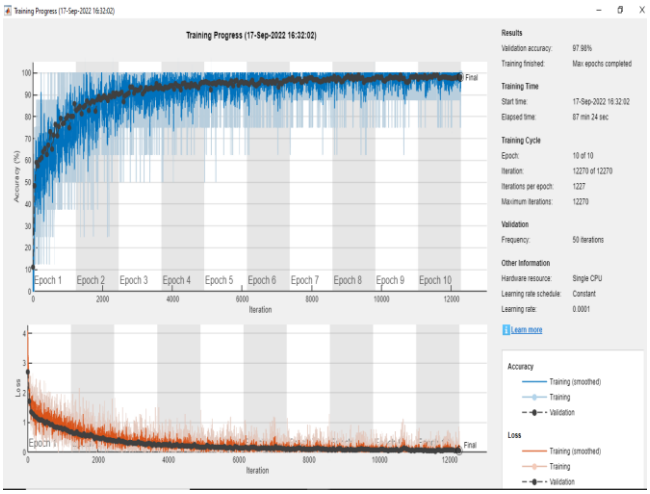

Fig. 14. Accuracy and Loss Rate for Noisy LFM Signals by 2D-CNN.

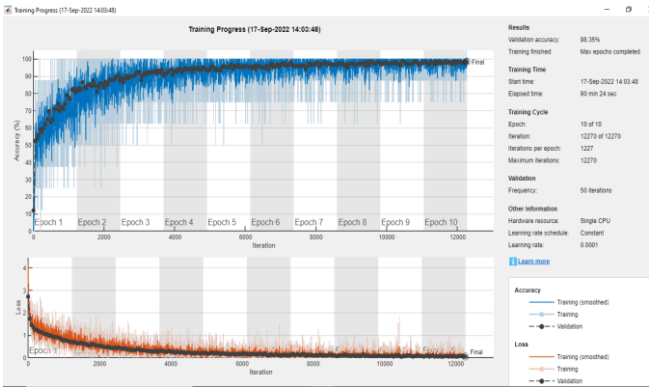

Fig. 15. Accuracy and Loss Rate for Noisy QFM Signals by 2D-CNN.

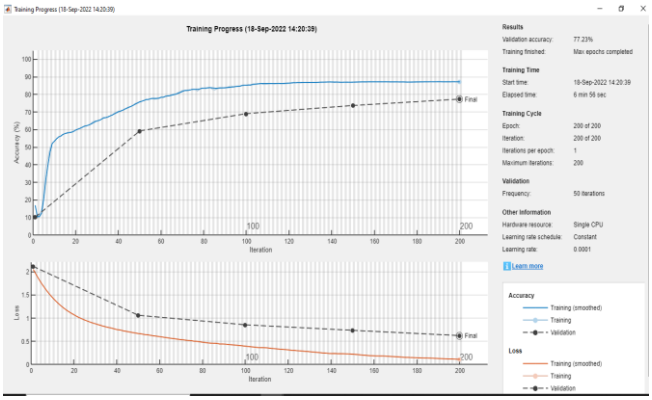

Fig. 16. Accuracy and Loss Rate for Noisy LFM Signals by LSTM.

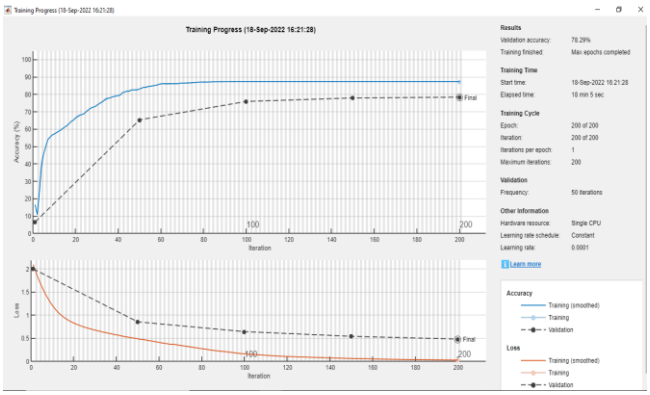

Fig. 17. Accuracy and Loss Rate for Noisy QFM Signals by BiLSTM.

XIV. CONCLUSION

This research provided performance deep-learning algorithms for estimating the IF, LCR, and QCR for noisy single-tone, linear, and non-linear frequency-modulated signal. Under additive white Gaussian noise and symmetric stable noise, the simulation is an important signal (impulsive model). The geometric SNR range belongs to [-50 50] dB, which examines the IF, LCR, and QCR under various Geometric Signal Noise Ratios (GSNRs). The RNN and CNN deep learning methods have many layers. The GRU layers include 5 layers sequence input layer, one GRU layer, a full connect layer, a softmax layer, and a classification layer. The LSTM layers include 5 layers sequence input layer, one LSTM layer, a full connect layer, a softmax layer, and a classification layer. The BiLSTM layers include 5 layers sequence input layer, one BiLSTM layer, a full connect layer, a softmax layer, and a classification layer. The 1D-CNN layers include

19 layers image input layer, four convolution layers, five batch normalization layers, four ReLU, two full connect layers, a dropout layer, a softmax layer, and a classification layer. The 2D-CNN layers include 21 layers of the image input layer, four convolution layers, three batch normalization, four ReLU activation functions, four Max-pooling layers, a dropout layer, two fully connected layers, a Softmax layer, and a classification layer. The simple structure built for the RNN or CNN model serves to reduce the communication system's complexity, power consumption, and cost. The simulation results demonstrate that alpha is more detrimental than beta, even if it has a tiny disability, and it has a substantial effect on guess parameters. The result shows the high accuracy of the estimation of the parameters by the 2D-CNN method for LFM and QFM signals and the GRU method for ST signals. The measurements are also used to select the best models including accuracy, precision, recall, F-measure, and ROC is a useful tool for assessing the performance of a machine learning or deep learning model.

REFERENCES

[1] B. Boashash, "Estimating and interpreting the instantaneous frequency of a signal – part 1: Fundamentals," Proc. IEEE. 1992.

[2] A. Papandreou-Suppappola, "Applications in TimeFrequency Signal Processing", CRC Press, 2002.

[3] Huang B., Lin, C. L., Chen, W., Juang, C. F., & Wu, X., "Signal Frequency Estimation Based on RNN"., In Chinese Control And Decision Conference, IEEE., 2020.

[4] Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and Hinrich Schutze, "Combining recurrent and convolutional neural networks for relation classification", In Proceedings of NAACL HLT, 2016.

[5] Yin W., Kann, K., Yu, M., & Schütze H., "Comparative study of CNN and RNN for natural language processing", 2016.

[6] Wu, H., Yang, M., Yang, S., Lu, H., Wang, C., & Rao, Y., "A novel DAS signal recognition method based on spatiotemporal information extraction with 1DCNNs-BiLSTM network", IEEE Access, 2020.

[7] Huda Saleem Razzaq, and Zahir M. Hussain. "Instantaneous Frequency Estimation for Frequency-Modulated Signals under Gaussian and Symmetric α-Stable Noise." 2021 31st International Telecommunication Networks and Applications Conference (ITNAC). IEEE, 2021.

[8] ElMoaqet, H., Eid, M., Glos, M., Ryalat, M., & Penzel T.," Deep recurrent neural networks for automatic detection of sleep apnea from single channel respiration signals", Sensors, 20(18), 5037, 2020.

[9] Al-Khafaji, Safaa D., Zahir M. Hussain, and Katrina Neville, "Frequency estimation of FM signals under non-Gaussian and colored noise.", International Journal of Applied Engineering Research, 2017.

[10] Sajedian I., Rho J., "Accurate and instant frequency estimation from noisy sinusoidal waves by deep learning.", Nano Convergence 6, 27, doi: 10.1186/s40580-019-0197-y, 2019.

[11] Boashash B., "Time-Frequency Signal Analysis and Processing: A Comprehensive Reference", Elsevier, Oxford, UK, 2016.

[12] Milczarek H., Leśnik C., Djurović I., Kawalec A., "Estimating the Instantaneous Frequency of Linear and Nonlinear Frequency Modulated Radar Signals—A Comparative Study". Sensors, 21(8), 2840, 2021.

[13] Roy, Subhrajit, Isabell Kiral-Kornek, and Stefan Harrer. "ChronoNet: a deep recurrent neural network for abnormal EEG identification." Conference on artificial intelligence in medicine in Europe. Springer, Cham, 2019.

[14] Riaz A., Nabeel M., Khan M., & Jamil H., " SBAG: a hybrid deep learning model for large scale traffic speed prediction", International Journal of Advanced Computer Science and Applications, 2020.

[15] Vinze, M., Danve, J., Shankar, S., Khan, M., & Kulkarni, N., "Automatic Music Generation using Long Short-Term Memory Neural Networks". Neural Computation, 9(8), 1735-1780., 1997.

[16] Selvin, S., Vinayakumar, R., Gopalakrishnan, E. A., Menon, V. K., & Soman, K. P., "Stock price prediction using LSTM, RNN, and CNN-sliding window model". In 2017 international conference on advances in computing, communications, and informatics (ICACCI). IEEE. 2017.

[17] Arslan Serdar, "A hybrid forecasting model using LSTM and Prophet for energy consumption with the decomposition of time series data." PeerJ Computer Science, doi:10.7717/peerj-cs.1001, 2022.

[18] Graves, A., & Schmidhuber J., "Framewise phoneme classification with bidirectional LSTM networks. In Proceedings. 2005 IEEE International Joint Conference on Neural Networks, IEEE. 2005.

[19] Maddu R., Vanga A. R., Sajja J. K., Basha G., Shaik R., "Prediction of the land surface temperature of major coastal cities of India using bidirectional LSTM neural networks", Journal of Water and Climate Change, doi: 10.2166/wcc.2021.460, 2021.

[20] Moons B., Bankman D., Verhelst M., "Embedded Deep Learning: Algorithms, Architectures, and Circuits for Always-on Neural Network Processing". Springer, 2018.

[21] Phil K., "Matlab deep learning with machine learning, neural networks, and artificial intelligence". Apress, New York, 2017.

[22] Schilling, Fabian. "The effect of batch normalization on deep convolutional neural networks." (2016).

[23] Hind R. Almayyali, Zahir M. Hussain, "Deep Learning versus Spectral Techniques for Frequency Estimation of Single Tones: Reduced Complexity for Software-Defined Radio and IoT Sensor Communications," Sensors, 2021.

[24] E. Swiercz, "Automatic Classification of LFM Signals for Radar Emitter Recognition Using Wavelet Decomposition and LVQ Classifier," ACTA PHYSICA POLONICA A, 2011.

[25] Zhiping Yin, Weidong Chen, "A New LFM-Signal Detector Based on Fractional Fourier Transform," EURASIP Journal on Advances in Signal Processing, 2010.

[26] Powers D. M., "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation", 2011.