Journal of Theoretical and Applied Information Technology
15th February 2024. Vol.102. No 3
© Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

PRUNING-BASED HYBRID NEURAL NETWORK FOR AUTOMATIC MODULATION CLASSIFICATION IN COGNITIVE RADIO NETWORKS

NADIA KASSRI[1], ABDESLAM ENNOUAARY[2], SLIMANE BAH[3], IMAD HAJJAJI[4], HAMZA DAHMOUNI[5]

[1,2,4,5] National Institute of Posts and Telecommunications, Rabat, Morocco
[3] Mohammadia School of Engineers, Rabat, Morocco
E-mail : [1]n.kassri@usms.ma, [2]abdeslam.ennouaary@gmail.com, [3]slimane.bah@emi.ac.ma, [4]imad.hajjaji@gmail.com, [5]dahmouni@inpt.ac.ma

ABSTRACT

Today, there is a greater need than ever to focus on realistic solutions to efficiently manage the available frequency spectrum, due to the scarcity of this limited resource. Thanks to its opportunistic communication paradigm allowing fair access for all users, Cognitive Radio (CR) has been commonly recognized as an enabling technology to mitigate the spectrum scarcity problem and ensure optimal use of the available spectrum. Automatic Modulation Classification (AMC) is an indispensable function for carrying out various CR tasks, including link adaptation and dynamic spectrum access. Recently, Deep Learning (DL) networks have shown promising results in dealing with AMC compared to traditional methods. However, most DL-based approaches proposed so far have high computational and memory requirements, which make them impractical for resource-constrained IoT applications. To address this challenge, we introduce in this work a novel lightweight hybrid Neural Network for AMC, consisting of a combination of Gated Recurrent Unit (GRU) and Convolutional Neural Network (CNN) structures, in which the fully connected layers are omitted. Our proposed design outperforms the baseline models in terms of recognition accuracy across various Signal-to-Noise Ratios (SNRs), while maintaining a lightweight structure and low computational complexity. To further enhance model compression and resource utilization, we have introduced an Iterative Magnitude-Based Pruning approach combined with Quantization Aware Training (QAT). Simulation results using the RadioML 2018.01A dataset validate the efficacy of our proposed model. With a sparsity of 0.7, the pruned variant, comprising 80 GRU cells, attains an average accuracy of 60.88% across all SNRs. Remarkably, it achieves a highest accuracy of 95.95% at 20 dB, while utilizing only 8 210 parameters.

Keywords: *Automatic Modulation Classification, Convolutional Neural Network, Gated Recurrent Unit, Cognitive Radio Networks, Spectrum Sensing, Resources-Constrained Objects, Pruning, Quantization Aware Training.*

1. INTRODUCTION

The frequency spectrum is currently in short supply due to several factors, including the inherent scarcity of spectral resources, the exclusive use of allocated frequency bands by specific communication systems derived from the static spectrum assignment policy of governmental agencies, and the extreme growth of wireless technologies, which has resulted in high demand for spectrum [1].

Cognitive Radio (CR) has been widely considered as a promising technology to deal with the aforesaid challenges by introducing the opportunistic use of the spectral bands that are under exploited by authorized users.

Operating as a Software-Defined Radio (SDR), CR dynamically monitors its surroundings, decides on spectrum occupancy, and adjusts its operating parameters to avoid interference with licensed users [1]. One of the main functional phases of the cognitive cycle is spectrum sensing. In this step, CR explores the radio environment, identifies the free channels, and gathers other meaningful information, such as modulation types of sensed signals. The latter is of paramount importance in detecting physical layer attacks and performing several CR tasks, including link adaptation and dynamic spectrum access [2].

Automatic Modulation Classification (AMC) involves the process of blindly acquiring knowledge about the modulation format of a detected signal.

As illustrated in figure 1, a typical AMC classifier involves two primary phases: "Signal preprocessing" and "Application of a classification algorithm". In the initial phase, preprocessing tasks are applied to extract useful signal features such as symbol period, signal power, carrier frequency, and noise power [3,4]. In the second phase, a suitable classification method is employed to classify the modulation schemes present in the sensed signals [2].

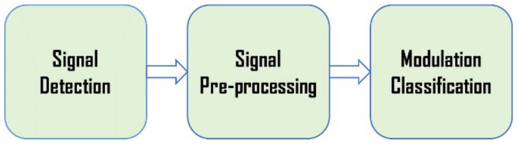


Figure 1: General AMC Scheme.

Generally, traditional AMC methods can be segregated into two primary classes: Likelihood-based (LB) and Feature-based (FB) methods [5]. The former treat modulation recognition as a multiple-hypothesis testing challenge: the likelihood function is first calculated for each modulation hypothesis based on the incoming signal samples, and then the ultimate classification decision is established by evaluating the likelihood ratio tests against a predefined threshold. Whereas the latter addresses modulation classification by taking the following steps: first, retrieving features from intercepted signals and subsequently applying appropriate classifiers, such as K-Nearest Neighbor (KNN) and Support Vector Machines (SVM), trained with specific handcrafted features to classify different modulation types [6]. Cyclic statistics, wavelet transform, high-order cumulants and are the most commonly used features in FB schemes [6].

Both LB and FB techniques have their own advantages and limitations. While LB methods can guarantee optimal classification accuracy, they face challenges related to computational complexity and the necessity for prior information about intercepted signals, which is generally not afforded in real-world situations. On the other hand, while FB solutions are easy to deploy and implement and less dependent on prior information about captured signals, they have limited performance under severe channel impairments [5–7].

Driven by the impressive achievements of Deep Learning (DL) in diverse fields like computer vision and image recognition, many papers have explored the use of DL networks to overcome the shortcomings of traditional techniques in dealing with AMC problem [8]. DL-based schemes revolutionize conventional approaches by operating as end-to-end learning systems, in which feature extraction and classification are seamlessly integrated. This joint learning framework enables the system to automatically extract discriminative features from raw data, eliminating the reliance on manually engineered features that often lack robust characterization capabilities [6].

As discussed in Section II, prior works on AMC techniques based on DL architectures can ensure high classification accuracy by leveraging the intelligence learned from bulk data and stored in the network. However, most DL-based approaches require a lot of energy, processing power, and storage space, which makes them impractical for resource-constrained scenarios like Internet-of-Things (IoT) networks.

In this paper, we aim to address the AMC challenge from the standpoint of resource-limited devices by proposing a lightweight DL architecture for modulation recognition. It is important to highlight that building a lightweight model is potentially possible by minimizing the quantity of trainable parameters within the layers. To make this reduction more significant, we propose, in this work, a hybrid DL scheme without dense layers. The latter, also known as fully-connected layers, refer to the layers whose inner neurons receive input from all neurons of their preceding layers and accordingly induce a lot of trainable parameters [5].

Moreover, it has been noted that after the completion of the training process for a neural network, a considerable proportion of units remain inactive, functioning as dormant neurons [9-10]. Additionally, certain weights demonstrate minimal impact on the final model inference. To leverage this notable remark, researchers have explored the use of various pruning approaches to reduce the complexity of neural networks. These approaches are typically categorized into two main types: structured pruning and unstructured pruning. In structured pruning, entire channels or filters are pruned, whereas in unstructured pruning, only selected neurons are pruned. However, it has been observed that overly aggressive pruning may result in a reduction of accuracy [9-11]. To address this issue, we have introduced an Iterative Magnitude-Based Pruning approach to compress our model efficiently without compromising accuracy. This approach has been used in conjunction with Quantization Aware Training (QAT) [12]. The latter aims to further optimize the model's performance by quantizing the weights and activations, enabling more resource-efficient inference and ensuring the preservation of accuracy during the pruning process. [12].

On the other hand, IoT objects are known for their limited signal acquisition capabilities, particularly in noisy environments [13]. Therefore, to validate our model, we utilized the well-known RadioML 2018.01A dataset, which comprises wireless signals with 24 distinct modulations, each available at 26 SNRs spanning from -20 dB to +30 dB in increments of 2 dB [14].

The primary contributions of the current work are outlined as follows:

- Proposing a highly accurate GRU-CNN-based AMC scheme, in which GRU block is employed to retrieve temporal dependencies and CNN block is set up to capture additional relevant features.
- The suggested solution is designed to be computationally efficient by minimizing the number of trainable parameters and substituting fully connected layers with a Global Average Pooling (GAP)-based scheme. This design enables its deployment in objects with limited computational resources.
- Introducing an Iterative Magnitude-Based Pruning approach along with Quantization Aware Training (QAT) for efficient Model compression and resource utilization.
- Investigating and testing our model using the RadioML 2018.01A dataset.
- Comparing our model with some conventional DL frameworks and contemporary approaches, addressing the same problem, our work demonstrates superior performance across different SNRs.

The following sections of this paper are organized as follows: Section II provides an overview of pertinent literature. Section III details the methodology applied in this study. In Section IV, the simulation results are showcased along with a thorough performance analysis. Ultimately, Section V serves as the conclusion of the paper.

2. RELATED WORK

As previously stated, the AMC endeavors to recognize the modulation format of intercepted signals, which is often regarded as an Artificial Intelligence (AI) multi-class decision-making task. Typical feature engineering-based approaches may be used to develop classification schemes using supervised or unsupervised learning to extract the underlying radio properties, including modulation type information [15]. However, AMC must deal with various challenging problems such as the growth of modulation formats, the difficulty in distinguishing between higher-order digital modulation formats, and the significant degradation of channel conditions.

In light of the aforementioned considerations, a multitude of approaches for modulation recognition has been put forward, incorporating a range of DL networks like Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs), to enhance classification performance in respect of accuracy, computational complexity, and processing speed. It is worth noting that DL approaches eliminate the need for handcrafted features and decrease the computational complexity associated with feature extraction.

The application of CNNs to address AMC challenges has been explored in numerous research studies. For instance, the study in [16] presented a two-phase approach that merges a CNN-based structure with the Continuous Wavelet Transform (CWT) for multi-class modulation recognition. The first phase entails utilizing the CWT to effectively extract the time-frequency information. In the second phase, the two-dimensional time-frequency knowledge obtained from the first phase is used as input for the CNN to classify the various modulation schemes. This work exhibits superior performance under conditions of varying noise levels, as compared to previous research.

Similarly, RNN-based techniques have been utilized in several AMC methods discussed in the literature. For example, authors of [17] proposed an effective AMC approach based on recurrent neural networks. The latter outperforms previous CNN-based approaches, particularly at high SNR levels. Specifically, the proposed scheme attains a recognition accuracy improvement from 80% to 91%.

Furthermore, some studies have investigated the use of hybrid DL-based AMC methods that integrate CNNs and RNNs. These hybrid networks are designed to capture both the spatial and temporal characteristics of modulated signals and have demonstrated better performance than using either CNNs or RNNs independently. For example, in [18], a method for AMC using a dual-stream architecture based on a fusion of CNNs and LSTM networks was proposed. The method accounts for the interaction between features and leverages the strengths of both CNNs and LSTMs. Firstly, the signals are pre-processed and transformed into Amplitude/Phase representation and I/Q format to obtain more relevant features for classification. Then, CNN and LSTM are fully integrated to capture the temporal and spatial characteristics from each signal representation, taking full advantage of CNNs in spatial feature extraction and LSTMs in handling sequential data. The proposed method demonstrated superior performance compared to the baseline methods under various SNR conditions.

On the other hand, although several studies have investigated the use of DL schemes for AMC, only a limited number of publications have focused on their use for resource-limited objects. Indeed, most developed DL-based AMC methods are complex solutions that require a large memory footprint and high computational complexity, rendering their implementation impractical for resource-limited devices.

To develop lightweight deep learning schemes intended for end-devices with limited resources, two well-known methods have been used, namely pruning and quantization techniques. Pruning DL models involves examining the weight values of neural connections to eliminate redundant or unnecessary ones with minimal impact on accuracy. As for quantization, it aims to reduce the computational cost by decreasing the precision of the numerical representation of the weights and activations of a neural network.

An example of research using quantization is reported in [19], where the authors introduced a quantized neural network model for AMC. The model subjected to quantization exhibited a recognition accuracy of 75% at 10 dB SNR, mirroring the performance of the unquantized model.

In [20], a hybrid solution that combines pruning and quantization approaches was developed to reduce the model's complexity. The proposed approach attained superior accuracy compared to the baseline model, utilizing only 40% of the hardware resources. It attained a real-time throughput of 527 000 classifications per second with a latency of 7.5 microseconds on a Field-Programmable Gate Array (FPGA) platform.

Another study that merges both approaches is discussed in [21]. The authors presented a rapid and effective neural network to deal with AMC problem. They introduced a Multiscale Convolutional (MSC) layer to learn important features and employed Separable Convolution Blocks (SCB) to decrease network complexity. To further reduce complexity, pruning, quantization, and factorization techniques were employed. As a result, the network achieved high classification accuracy while utilizing fewer parameters.

A recent study in [22] proposed a methodology to assess the effect of quantization on accuracy and inference cost in DL networks. The findings suggest that independently quantizing each layer according to the compromise in performance can achieve similar accuracy to non-quantized models at a reduced inference cost.

Despite their effectiveness in reducing the complexity of DL-based AMC models, pruning and quantization techniques present some limitations, including potential accuracy reduction, loss of information, and difficulty in addressing all modulation schemes.

To address the above challenges, less complex CNN-based and RNN-based AMC approaches were proposed. To improve cost-effectiveness while preserving classification accuracy, Chen et al. presented a single-layer LSTM method integrating a random erasing-based test time augmentation mechanism (RE-TTA) [23]. This method recovered the spatial correlations between multiple antenna polarization (AP) samples in the input signal by attaching an attention scheme between the LSTM layer and the dense layer. Numerical analysis of the RadioML2016.10A dataset revealed the efficacy of the proposed approach, which outperformed basic RNN and LSTM-based methods in respect of classification accuracy.

Additionally, in [24], a cost-effective CNN-based was proposed for AMC in cognitive radio networks. The network architecture features various convolutional blocks which utilize asymmetric convolution kernels to learn the spatiotemporal signal correlations. The blocks are equipped with skip connections to maintain residual information. The proposed model, MCNet, achieves an overall accuracy of 93.59% at 20 dB SNR on the widely used RadioML 2018.01A dataset. Similarly, the authors of [25] proposed a novel CNN design for AMC using bottleneck and asymmetric convolution structures to reduce computational complexity and incorporating skip connection technique to enhance classification accuracy by resolving the vanishing gradient issue. Experimental findings on the RadioML 2018.01A dataset indicate that the proposed model surpasses the traditional MCNet model with regard to classification accuracy over a SNR span of -4 dB to 20 dB.

In the same context in this work, we propose a fast and cost-efficient method that offers the best balance between computational complexity and accuracy compared to baseline models.

3. METHODOLOGY

3.1. Signal Model

A modulated signal emitted in the radio environment may experience several changes, such as multipath fading and shadowing effects, due to its reflection, refraction, and scattering as it propagates through the radio environment. These effects can cause variations in the signal strength and can lead to signal distortion or signal loss. Thus, for a transmitted signal x(t), we can express the received signal, y(t), as follows [13]:

$$y(t) = x(t) * h(t) + n(t) \tag{1}$$

where h(t) represents the channel gain and n(t) denotes the Additive White Gaussian Noise (AWGN).

The channel gain h(t) involves all the effects undergone by the signal during propagation. For instance, in the case of a frequency non-selective fading channel, the channel gain can be expressed as [26]:

$$h(t) = \alpha(t)e^{j\phi(t)} \tag{2}$$

where α(t) is a random variable with a Rayleigh distribution and ϕ(t) is the phase.

In CR systems, radio receivers can provide intercepted signals in an I/Q format. The I/Q format separates a signal into two components, known as the in-phase (I) and quadrature (Q) components. These components can be mathematically represented as [13]:

$$I = A\cos\theta \tag{3}$$
$$Q = A\sin\theta \tag{4}$$

where A and θ are the instantaneous amplitude and phase of y(t).

The I and Q components encompass valuable information about the signal, such as its frequency, phase, and amplitude, enabling the recognition of modulation format used in a given communication signal.

3.2. Recurrent Neural Networks

Among the different types of DL networks, RNNs are the best suited for performing predictive operations on sequential or time-series-based data due to their capacity to capture temporal dependencies [27]. Standard RNNs suffer from the vanishing gradient problem. In this problem, as layers are added to the RNNs, the gradient becomes exponentially smaller, making the update of weights almost negligible and, therefore, the network hard to train. Furthermore, this issue leads to a significant disadvantage for RNNs, namely short-term memory, where the network fails to memorize data for an extended period and starts to forget its previous inputs, which results in poor performance and low accuracy, especially when dealing with longer data sequences.

LSTM and GRU, modified versions of RNNs, alleviate this issue by introducing internal mechanisms called "gates" to control the flow of information. During training, these gates decide which information is relevant to include or exclude to the hidden state that serves as the neural networks' memory. Compared to LSTM, the GRU necessitates a smaller number of trainable parameters, resulting in reduced memory usage and increased processing speed.

On the other hand, BiGRU and bidirectional LSTM (BiLSTM) capture features from data in both the forward and backward passes, enabling them to acquire a more comprehensive understanding of context-dependency, thereby enhancing the performance of the learning process. However, this advantage comes at the expense of longer training times and heightened computational complexity [27].

In line with our goal of proposing a DL-based AMC model with fewer parameters, we have chosen the Gated Recurrent Unit to endow our model with the ability to handle long-term dependencies, as it is the lightest variant of RNNs. As shown in figure 2, the GRU cell architecture involves two gates, namely the reset gate and the update gate. The update gate decides on the amount of prior information necessary for transitioning to the subsequent state, and the reset gate establishes what information should be disregarded.

The governing equations for GRUs are [28,29]:

$$h^t = (1 - z^t) * h^{t-1} + z^t * \tilde{h}^t \quad (5)$$
$$\tilde{h}^t = tanh\,(r^t * (U^h h^{t-1}) + W^h x^t + b^h) \quad (6)$$

where h^t, h^{t-1}, and $\tilde{h}^t$ denote the hidden state of the network at the current time step, the previous time step, and the new state at the current time step, respectively. The hidden layer at each time step is calculated as a combination of the previous hidden state and the current input at that time step, represented by x^t.

The weight matrices and bias vectors of the network are represented as U, W, and b, and are learned during the training phase. The hidden layer of the GRU contains memory cells that perform the main functions of the network, such as retaining and forgetting information over time.

The amount of previously disregarded information and newly introduced data is determined by z^t, while the proportion of this information that is incorporated into the new state is controlled by r^t . The equations below can be used to calculate $z^t$and r^t:

$$z^t = \sigma(W^z x^t + U^z h^{t-1} + b^z) \quad (7)$$
$$r^t = \sigma(W^r x^t + U^r h^{t-1} + b^r) \quad (8)$$

where x^t represents the input at step t. The computations involve the use of two mathematical functions, namely the logistic sigmoid function, denoted by σ, and the hyperbolic tangent function, denoted by tanh.

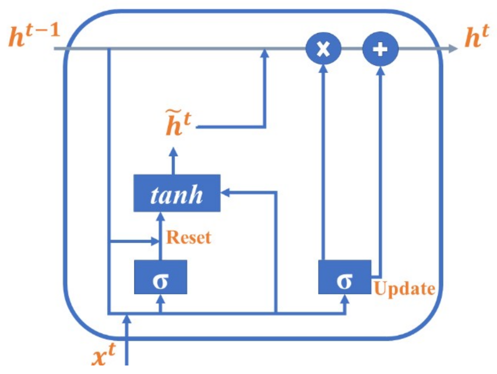


Figure 2: Structure of a GRU Cell [28]

3.3. Convolutional Neural Networks

CNNs are DL neural networks that employ convolution and pooling operations to capture complex features from the data. These networks are the best and most widely used for image analysis, where they have resulted in valuable progress. Although computer vision and image analysis have been the most widespread uses of CNNs, many researchers have also investigated the use of CNNs for other data analysis and classification problems, such as the AMC problem.

CNNs are generally built with three types of layers: convolutional layers, pooling layers, and fully connected layers [30]. Convolution layers are the main building blocks of CNNs, where the majority of computations (convolutional operations) takes place to extract fundamental features from data. Concerning pooling layers, they are used to reduce the dimensionality of the feature maps and, consequently, the number of parameters to be learned and the number of calculations to be performed.

Preceded by a flattening layer, which converts the multidimensional data into a one-dimensional format, the fully connected layers are set up to wrap up the CNN architecture. As mentioned earlier, fully connected layers significantly increase the number of trainable parameters and the complexity of the network due to their high connectedness. In our work, we have omitted these layers and replaced them with GAP-based scheme to ensure the lightness of our proposed model intended for resource-constrained objects.

Based on the dimensionality of the input data, CNNs can be classified into three types: 1D

CNNs (Conv1D), 2D CNNs (Conv2D), and 3D CNNs (Conv3D).

- 1D CNNs are designed to operate on one-dimensional data, such as sequences or time series data. The convolutional filters in these networks are applied by sliding them over the input data and performing element-wise multiplications with the data, followed by a summation operation.
- 2D CNNs are the most widely used type of CNNs, and they are used to process two-dimensional data, such as images. The convolutional filters, which are two-dimensional arrays of weights, are applied to the input data in a sliding window fashion: At each position, the filter weights are multiplied by the corresponding input data and summed up to produce a single output value.
- 3D CNNs are used to process three-dimensional data, such as videos. The convolutional filters in conv3D operate similarly to those in conv2D, with the main distinction being the number of dimensions involved in the convolution process.

Due to their low computational complexity and suitability for processing sequential data, we have chosen one-dimensional convolutional layers (conv1D) to enable our model to extract underlying patterns and features from the data.

Given an input channel number of N, the m^{th} feature map obtained at the output of a conv1D layer can be expressed as follows [25]:

$$y^m = \sum_{n=1}^{N} K^n * W^m + b^m \qquad (9)$$

$$z^m = \alpha \frac{y^m - \mu_b}{\sqrt{{\sigma_b}^2 + \beta}} + \gamma \qquad (10)$$

where y^m is the input of the BN layer, μ_b and ${\sigma_b}^2$ are the mean and variance of the batch of input data, α and γ are learnable parameters, and β is a constant used to avoid division by zero.

To introduce non-linearity and sparsity to our model, we have used The Rectified Linear Unit (ReLU) activation function after the BN layer. ReLU function has the following form:

$$f(z^m) = max\ (0, z^m) \qquad (11)$$

where z^m is the input to the activation function.

3.4. Proposed Approach

To take advantage of the complementary capabilities of CNNs and RNNs, we propose a model that combines both networks using a fusion architecture. The use of this hybrid model leverages the ability of CNNs to capture relevant features of the data, as well as the potential of RNNs to process sequential information (see figure 3).

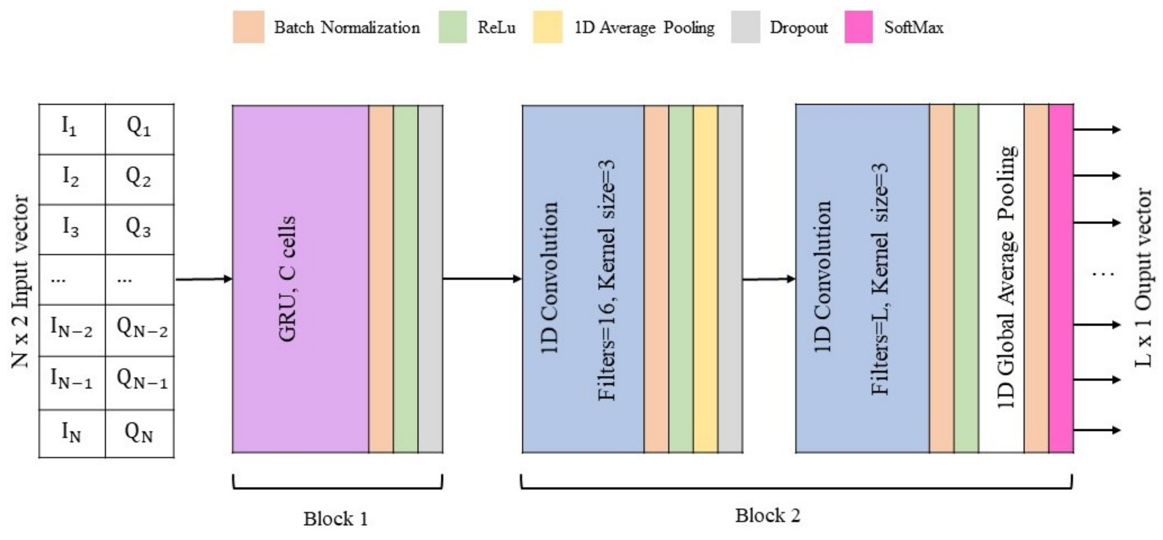


Figure 3: Architecture Of The Proposed Model.

As shown in figure 3, our proposed model contains two blocks. The first block includes a GRU layer with C cells, a BN layer, a ReLU layer, and a dropout layer. As explained in the previous subsections, the use of the GRU layer, to handle sequential dependencies, aligns perfectly with our purpose of conceiving a DL model with fewer parameters, given that GRU is the lightweight variant of RNNs. The input to the GRU layer is a Nx2 vector, where N represents the number of samples and 2 refers to the "I" and "Q" components of each sample. The output of the GRU layer is then processed through a BN layer, a ReLU layer, and a dropout layer (0.2). The latter is a regularization technique widely employed in neural networks to prevent overfitting during training.

The second block of the model comprises a 1D CNN scheme, which consists of two 1D-Convolutional layers, batch normalization layers, activation layers, a dropout layer, and pooling layers. The peculiarity of this structure lies in its utilization of a GAP-based scheme, instead of the typical dense layers, to wrap up the model. The first Conv1D layer involves 16 filters with a kernel size of 3 and processes an input vector of shape (N, C), where N is the number of samples in a signal instance and C is the number of GRU cells. The output of this layer is then passed through a BN layer and a ReLU layer. Subsequently, an Average Pooling layer with a pool size of 2 is applied to reduce the spatial dimension of the feature map and decrease the number of trainable parameters in the network, while also reducing the computation required in the forward and backward passes. A dropout layer (0.2) is incorporated to further mitigate overfitting, the output of this dropout layer is fed into the second Conv1D layer which contains L filters with a kernel size of 3 (L is the number of modulation schemes), followed by a BN layer and a ReLU layer.

Finally, to map the signal characteristics captured from the previous layers to each of the modulation types, a GAP layer is applied, followed by a BN layer and a SoftMax layer to generate the final classification.

3.5. Model Pruning Approach

As already mentioned, pruning is a technique used in Machine Learning (ML) to reduce the size of a neural network by eliminating unnecessary connections or parameters without significantly affecting its performance. This process helps reduce model size, memory footprint, and inference time, making the model more efficient and suitable for resource-constrained objects [10][26]. Although pruning typically leads to a decrease in accuracy, when implemented gradually and with careful consideration, it can yield favorable outcomes. Indeed, by methodically pruning and subsequently retraining the model, it can gradually adapt to the removed parameters, potentially mitigating the accuracy loss and ultimately achieving desirable results.

The pruning process in this study employed magnitude-based pruning [11] with an iterative approach to achieve the desired sparsity level. Initially, the original model is trained using a specific architecture, dataset, and optimization algorithm until convergence. Once the model has converged, magnitude-based pruning is applied, removing weights below a determined threshold. The pruned model is then retrained to allow it to recover its weights and restore its performance. To increase the sparsity further, the pruned model is selected as the new "original model" for subsequent pruning iterations. This iterative process of pruning and retraining is repeated until the target sparsity level is reached. The criteria for stopping the iterative process are based on the desired sparsity level.

As discussed in the following section, the experimental findings demonstrate the efficacy of the proposed pruning approach in reducing model size and computational requirements while maintaining satisfactory performance levels.

It's worth noting that pruning has been applied along with Quantization Aware Training (QAT). Instead of using 32 bits, the weights are quantized to 8 bits, yielding a more compressed and cost-efficient model.

4. IMPLEMENTATION DETAILS AND RESULTS

4.1. Dataset and data preprocessing

In our experiments, we have used the RadioML 2018.01A dataset with input dimensions of 1024 ×2 as benchmark to evaluate the performance of our proposed model.

The RadioML 2018.01A [14] is a publicly available dataset that comprises 2 555 904 frames in the form of I/Q data with 24 distinct modulations, each available at 26 SNRs (Ranged from -20 dB to +30 dB in 2 dB increments). Each modulation/SNR combination contains 4096 instances, each of which is 1024 samples long.

Figure 4 shows a structural example of the RadioML 2018.01A dataset and table 1 summarizes its parameters and content.

In addition, to avoid the accuracy paradox and ensure the reliability of our proposed classifier, we have used balanced training and test sets for performance analysis. Indeed, providing a classifier with imbalanced data may lead to a bias towards the majority class due to insufficient data on the minority class. Therefore, we have used a random undersampling technique with a fixed seed to maintain the reproducibility of results. This approach preserved the class balance of the original datasets, ensuring equal representation of all classes in both the training and test sets.

In all experiments conducted, the data was divided into a 4:1 ratio, with 80% of the examples used for training and 20% used for testing.

It's worth noting that in our simulation works, we have used only half of the RadioML 2018.01A dataset because of hardware limitations.

Table 1: RADIOML 2018.01A Dataset Description.

Parameter	Value or description
Number of modulation schemes	24
Modulation formats	Digital: GMSK, OOK, 256QAM, 128QAM, 64QAM, 32QAM, 16QAM, 128APSK, 64APSK, 32APSK, 16APSK, 32PSK, 16PSK, 8PSK, 8ASK, 4ASK.
	Analog: FM, AM-DSB-SC, AM-DSB-WC, AM-SSB-SC, AM-SSB-WC.
SNRs(dB)	-20: 2: 30
Signal format	I/Q format
Vector shape	1024x2
Total number of examples	2 555 904
Channel characterization	AWGN Doppler shift Non-impulsive delay spread Symbol rate offset Carrier frequency offset Selective multipath Rician fading

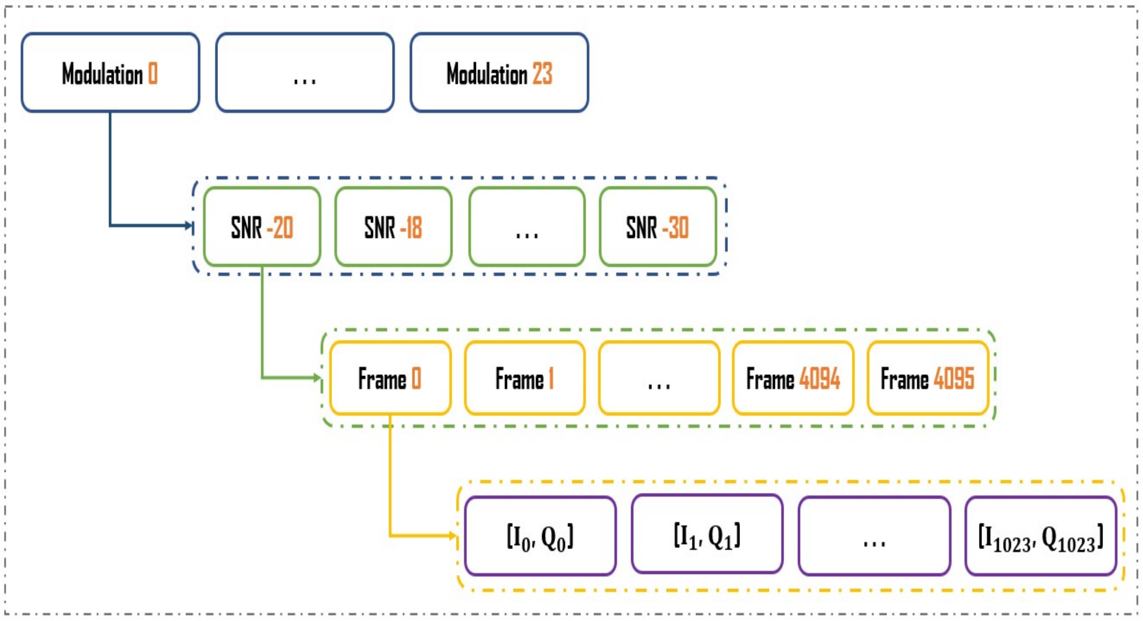


Figure 4: Structural Example Of The RadioML 2018.01A Dataset.

4.2. Results and Analysis

Training and testing were conducted using Python 3.9.7, Keras 2.7 and TensorFlow 2.7, on a workstation with an Ubuntu 18.04.6 LTS OS, an Intel® Xeon(R) CPU E5-2660 v4 @ 2.00GHz × 28 Processor, 32 GB of RAM, and an NVIDIA Quadro M6000/PCIe/SSE2 GPU with CUDA, providing enhanced processing speed.

The categorical cross-entropy loss function and Adam optimizer were used in all experiments, employing a learning rate of 0.001. The latter was decreased by 90% every 20 epochs, and the training was performed for 100 epochs. To avoid overfitting, a callback was implemented to halt training when the validation loss did not exhibit any improvement for 12 consecutive epochs. The batch size was set at 128.

The performance of our model is assessed by conducting a comparative analysis with several well-known conventional models, namely GRU [31], PET-CGDNN [26], MCNet[24], ResNet[14]and a baseline model (Asymmetric-CNN) referenced in [25]. The performance evaluation metric used for this purpose is classification accuracy [32].

In addition, in the realm of CR networks, the computational complexity of models is a critical factor that needs to be considered, especially in real-time communication scenarios. To accurately evaluate this complexity, we utilize three essential metrics: trainable parameters, memory footprint, and inference time.

4.2.1. Impact of the number of GRU cells on recognition accuracy

In our experiments, the performance of several versions of the proposed model, which are constructed with different numbers of GRU cells is evaluated. This allows to investigate the effect of the number of GRU cells on recognition accuracy.

From table 2, it is apparent that as the number of GRU cells is increased, the model's capacity to learn complex patterns in the data increases, leading to improved performance. However, with a number of GRU cells greater than 80, the marginal gain in the performance of our model begins to be less important due to the principle of diminishing marginal returns [13]. This means that the additional benefit of adding more cells becomes less significant beyond a certain point, leading to a plateau in classification accuracy or even a degradation in performance if the model becomes too complex. Furthermore, the disparity in performance among the various versions of our model is minimal when the number of GRU cells is equal to or greater than 80. However, there is a substantial decline in classification performance for our model with 40 cells. Specifically, the overall classification accuracy decreases by over 5% when downgrading the number of GRU cells from 50 to 40.

Table 2: Performance Results Of The Proposed Model.

Number of GRU cells	Average accuracy (%)	Highest accuracy (%)	Trainable parameters	Memory footprint (kBytes)
40	57.36	90.72	8 632	34
50	60.38	95.35	11 920	48
60	60.77	95.79	15 840	63
70	60.85	95.93	20 360	81
80	61.66	96.65	25 480	102
90	61.56	96.45	31 200	125
100	62.04	96.78	37 107	148

4.2.2. Impact of pruning on Model performance

In light of the above results, retrieving more GRU cells to reduce complexity is no longer efficient as it is accompanied by a high decrease in accuracy. For this reason, we have applied an Iterative Magnitude-Based Pruning approach, explained in the fifth subsection of section 3, to further compress our model without compromising the performance. To that end, we have selected the variant of our model with 80 GRU cells as a base of the pruning process.

In our simulation works, the pruning process commences at the beginning step with a sparsity level of 0.2. It spans a duration of 5 epochs to reach the specified end step, at which point the target sparsity level is achieved. Throughout this process, the learning rate is set to 0.0001, and the batch size is configured to 128. Subsequently, the pruned model is retrained until convergence. Table 3 shows the performance of different pruned versions of our model.

By employing a number of parameters between 4509 and 13144, we have generated higher performing models that exhibit an overall accuracy surpassing 58.5% and a highest accuracy exceeding 94%.

Table 3. Effect Of Pruning On Model Performance.

Pruned Model	Original Model	Target Sparsity	Parameters	Average accuracy (%)	Highest accuracy (%)	Memory footprint (kBytes)
Model_Sparsity 0.5	Proposed Model (80 cells)	0.5	13 144	61.31	96.43	13.1
Model_Sparsity 0.7	Model_Sparsity 0.5	0.7	8 210	60.88	95.95	8.2
Model_Sparsity 0.8	Model_Sparsity 0.7	0.8	5 742	60.18	95.27	5.7
Model_Sparsity 0.85	Model_Sparsity 0.8	0.85	4 509	58.7	94 .01	4.5

It's important to highlight that the substantial reduction in memory footprint is attributed to two main factors: pruning and quantization. Pruning reduces the number of parameters, while quantization reduces the memory required to store each parameter from 4 bytes to just one byte.

4.2.3. Comparison with conventional models

It is noteworthy that the subsequent analysis in this paper pertains to our model with 80 GRU cells and sparsity 0.7.

As depicted in table 4, our proposed model (80 GRU cells), featuring a CNN/GRU architecture with a sparsity of 0.7, outperforms the conventional models in terms of both average accuracy and highest accuracy.

The average accuracy of 60.88% achieved by our model surpasses that of other architectures, including GRU (60.26%), MCNET (58.3%), Lightweight Backbone Network (60.29%), and PET-CGDNN-sparsity 0.5- (59.6%).

Our model's effectiveness to learn intricate patterns and dependencies within the data using the combination of convolutional and recurrent layers contributes to its superior performance in correctly classifying radio signals.

Additionally, our model achieves the highest accuracy of 95.95% among all models considered, showcasing its proficiency in correctly identifying specific radio signal modulations. This result indicates the effectiveness of our proposed approach in handling diverse and complex modulation schemes present in the dataset.

A notable advantage of our model lies in its remarkably low memory footprint of 8.2 kilobytes. In contrast, conventional models such as MCNET (604.9 kBytes), Lightweight Backbone Network (185.9 kBytes), and ResNet (944 kBytes) require significantly larger memory footprint. This demonstrates the efficiency of our model in terms of memory utilization, making it well-suited for deployment in resource-constrained environments or on edge devices.

Furthermore, our model showcases the shortest inference time of 0.073 milliseconds per sample compared to other models in the experiment. This rapid processing speed enhances its potential applicability in real-time scenarios and time-sensitive applications, such as wireless communication systems and signal analysis in dynamic environments.

Table 4: Performance Comparison Of Our Proposed Model With Some Conventional Models (Best Values Are Bold).

Model	Basic structure	Average accuracy (%)	Highest accuracy (%)	Parameters	Memory footprint (kBytes)	Inference time (ms/sample)
GRU	GRU	60.26	95 .2	39 224	156.9	0.233
MCNET	CNN	58.3	92.63	151 224	604.9	0.144
Lightweight Backbone Network	CNN	60.29	95.7	46 472	185.9	0.14
PET-CGDNN (sparsity 0.5)	CNN+GRU	59.6	94.9	37K	148	0.219
ResNet	CNN	47.47%	76.04	236K	944	0.164
Our Model (sparsity 0.7)	CNN+GRU	**60.88**	**95.95**	**8 210**	**8 .2**	**0.073**

The performance comparison results affirm our model's superiority in accuracy, memory efficiency, and inference speed. The combination of convolutional and recurrent layers, along with pruning and QAT, proves to be a successful strategy for achieving state-of-the-art results in radio signal modulation classification. The findings from this study highlight the practical advantages and potential applications of our proposed model in the field of radio signal processing and communication systems.

4.2.4. Classification accuracy of 24 modulation formats

Figure 5 presents the complete classification accuracy of 24 modulation formats, which are categorized into three classes for improved visualization: APSK+PSK in figure 5a, QAM+ASK in figure 5b, and analog+low-order in figure 5c. As reported in this figure, our model achieved 100% classification accuracy in identifying some modulation types, namely FM, AM-DSB-WC, OOK, and 32PSK over the SNR range of +0 dB to 30 dB, with an exception of 99.27% accuracy at 0 dB SNR for 32PSK.

The remaining schemes can be classified into two groups: the low-performance class and the high-performance class. For example, the former may include 256QAM and AM-SSB-WC which are predicted with accuracy rates of 85.12% and 34.64%, respectively, at 20 dB SNR. Whereas the latter may involve 16QAM and GMSK which are recognized with an accuracy rate greater than 99%

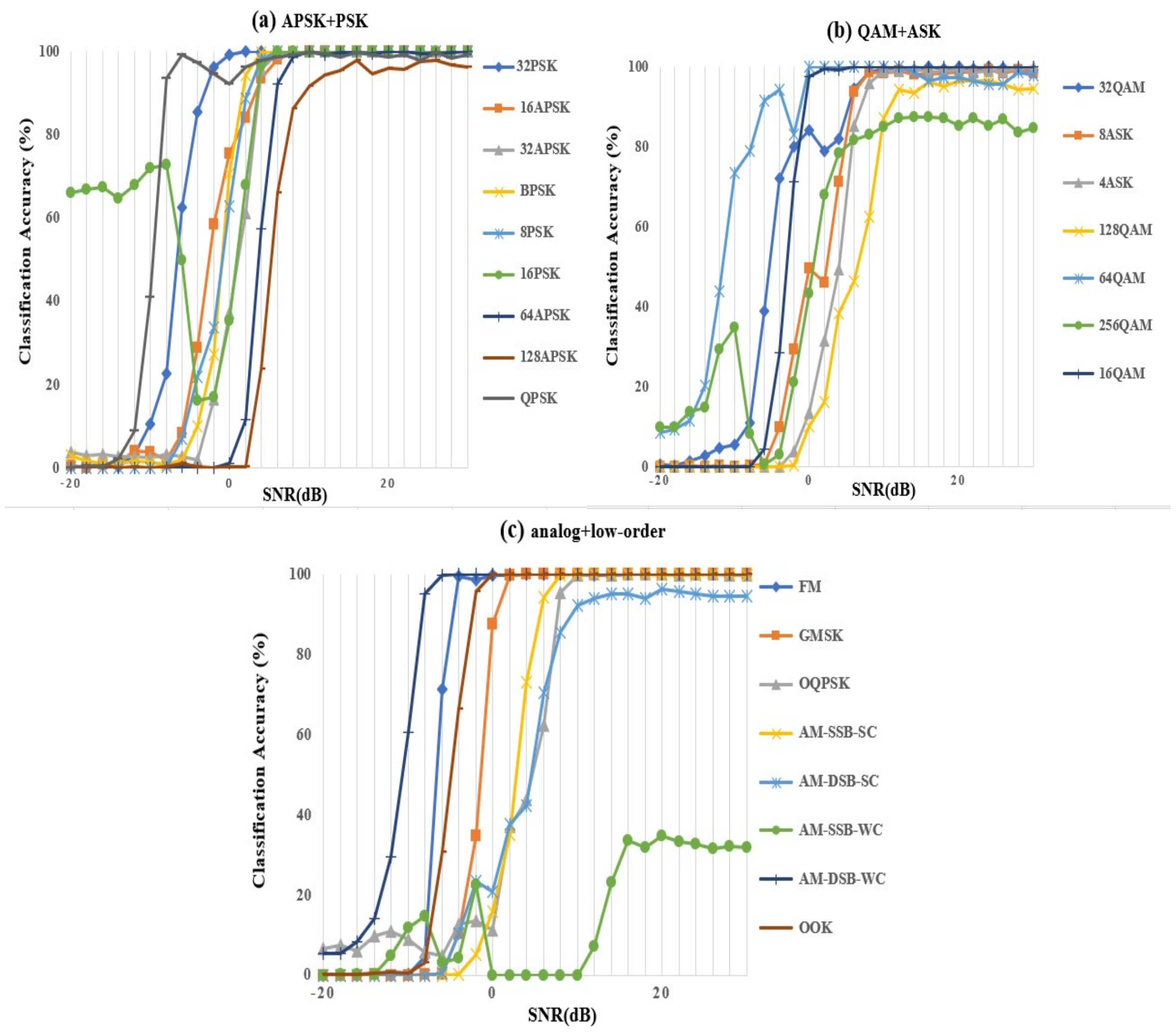


Figure 5: Classification Accuracy Of 24 Modulations Divided Into Three Categories: (a) APSK+PSK, (b) QAM+ASK, and (c) Analog+ Low-Order.

over the SNR range of +2 dB to 30 dB.

Furthermore, the vulnerability of high-order modulation signals to different channel impairments results in a reduction in the recognition rate as the APSK order increases. In particular, the accuracy experiences a notable decrease of more than 40% when elevating the APSK order from 32 to 64 at SNR levels below 6 dB.

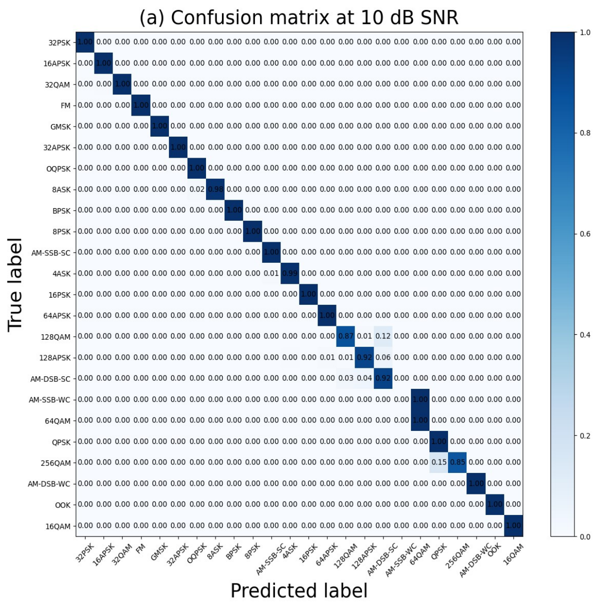
For more insights into the effectiveness of our model in recognizing different modulation schemes, the confusion matrices of our proposed model 80 GRU cells (sparsity 0.7) at 10 dB SNR and 0 dB SNR are shown in figure 6. For comparison purposes, the confusion matrices of the Lightweight Backbone Network at the same SNRs are also given in figure 7.

Figure 6a shows that our model achieves high accuracy rates exceeding 99% for most modulation formats. However, it fails to differentiate between AM-SSB-WC and 64QAM, all AM-SSB-WC signals are detected as 64 QAM signals. Additionally, to a lesser degree, our model cannot differentiate AM-DSB-SC from 128APSK and 128QAM, and 256QAM from QPSK.

In addition, figure 6b clearly illustrates the impact of noise on classification accuracy, particularly on higher-order modulation types. Furthermore, as the signal-to-noise ratio decreases, the previously identified confusions become more pronounced.

Besides, figure 7 illustrates that the Lightweight Backbone Network shares two common confusions with our proposed model. These confusions specifically involve AM-SSB-WC with 128APSK and 128QAM, as well as 256QAM with QPSK. Furthermore, the baseline model fails to distinguish 64QAM from AM-SSB-SC.

Also from the confusion matrices, it can be concluded that our model (with 80 GRU cells and a sparsity of 0.7) exhibits superior performance compared to the baseline model across the majority of modulation schemes, while maintaining a lightweight structure and low computational complexity.



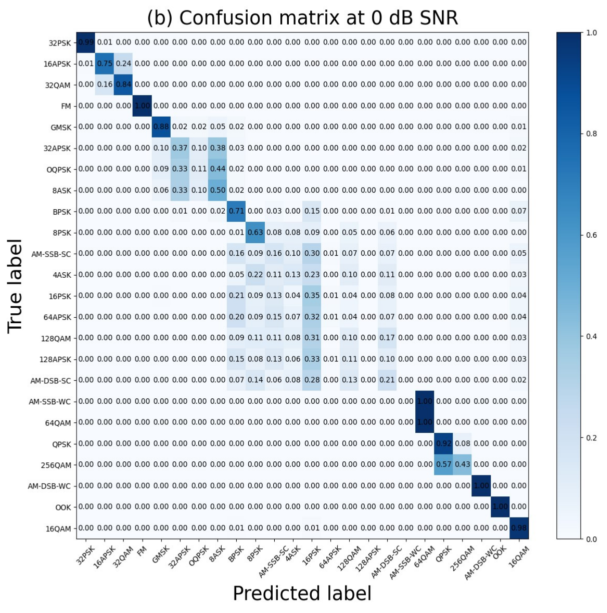


Figure 6: Confusion Matrices Of The Proposed Model With 80 GRU Cells (Sparsity 0.7). (a) At 10 dB SNR, Accuracy: 93.83%. (b) At 0 dB SNR, Accuracy: 54.49%.

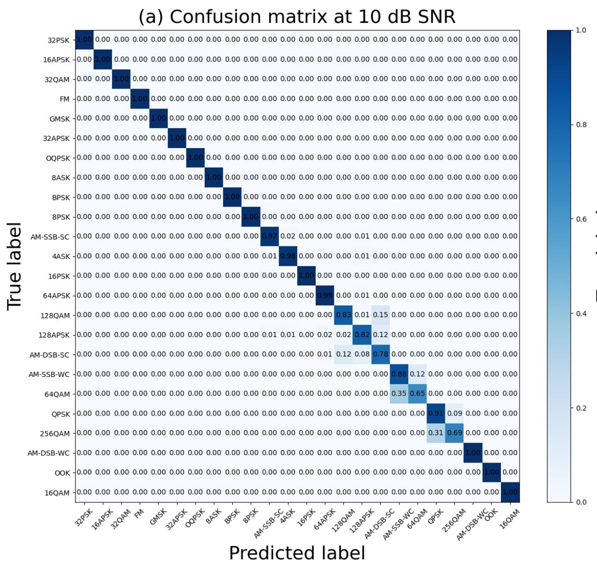
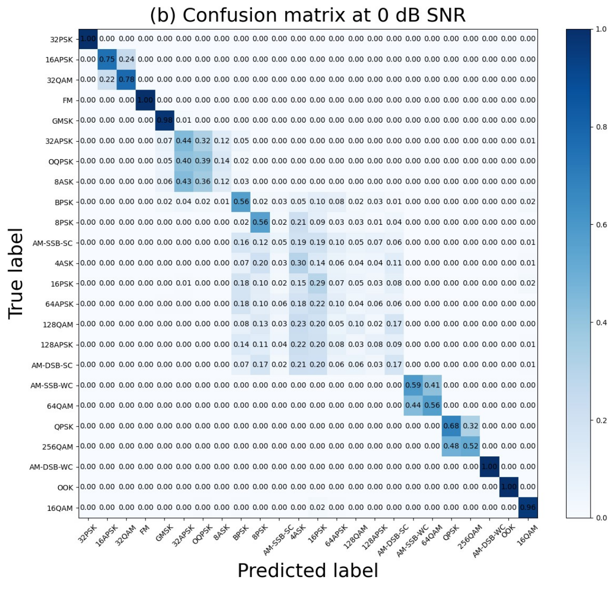



Figure 7: Confusion Matrices Of The Baseline Model. (a) At 10 dB SNR, Accuracy: 93.71%. (b) At 0 dB SNR, Accuracy: 53.98%.

5. CONCLUSION

This paper proposes a highly accurate and lightweight hybrid DL method for effectively classifying the modulations of radio signals across diverse channel impairments in cognitive radio networks. The proposed model utilizes a GRU block to retrieve temporal dependencies and a CNN block to capture higher-order features from the data. The model's computational efficiency is achieved by reducing the number of trainable parameters and substituting fully connected layers with a GAP-based scheme.

In addition to compress further our model, an Iterative Magnitude-Based Pruning approach along with Quantization Aware Training (QAT) was applied. The superiority of the proposed model is illustrated through experiments conducted on the RadioML 2018.01A dataset, showing higher performance across different SNRs compared to selected benchmark models.

In future work, multi-stream feature extraction using different types of data representation will be introduced to enhance the model's ability to classify modulations accurately and avoid all confusions existed between them while maintaining its lightweight design.

REFERENCES:

[1] N. Kassri, A. Ennouaary, S. Bah, H. Baghdadi, A Review on SDR, Spectrum Sensing, and CR-based IoT in Cognitive Radio Networks, International Journal of Advanced Computer Science and Applications. 12 (2021) 100–121. https://doi.org/10.14569/IJACSA.2021.0120613.

[2] T. Huynh-The, Q.V. Pham, T. Van Nguyen, T.T. Nguyen, R. Ruby, M. Zeng, D.S. Kim, Automatic Modulation Classification: A Deep Architecture Survey, IEEE Access. 9 (2021) 142950–142971. https://doi.org/10.1109/ACCESS.2021.3120419.

[3] Z. Zhu, A. Nandi, Automatic modulation classification: principles, algorithms and applications, 2015. https://books.google.com/books?hl=fr&lr=&id=AZtUDwAAQBAJ&oi=fnd&pg=PR11&dq=Automatic+Modulation+Classification:+Principles,+Algorithms+and+Applications&ots=ZdVUeXTLnW&sig=M3wy4yWYZMPjFCY6xYT5hkcB-JM (accessed April 26, 2022).

[4] O.A. Dobre, A. Abdi, Y. Bar-Ness, W. Su, Survey of automatic modulation classification techniques: classical approaches and new trends, IET Communications. 1 (2007) 137. https://doi.org/10.1049/iet-com:20050176.

[5] N. Kassri, A. Ennouaary, S. Bah, Lightweight Hybrid Deep Learning Scheme for Automatic Modulation Classification in Cognitive Radio Networks, in: 2022 9th International Conference on Future Internet of Things and Cloud (FiCloud), IEEE, 2022: pp. 113–118. https://doi.org/10.1109/FiCloud57274.2022.00023.

[6] D. Zhang, W. Ding, B. Zhang, C. Xie, H. Li, C. Liu, J. Han, Automatic Modulation Classification Based on Deep Learning for Unmanned Aerial Vehicles, Sensors. 18 (2018) 924. https://doi.org/10.3390/s18030924.

[7] Zhechen. Zhu, A.Kumar. Nandi, Automatic modulation classification: principles, algorithms, and applications, (n.d.).

[8] F. Zhang, C. Luo, J. Xu, Y. Luo, F.-C. Zheng, Deep learning based automatic modulation recognition: Models, datasets, and challenges, Digit Signal Process. 129 (2022) 103650. https://doi.org/10.1016/j.dsp.2022.103650.

[9] Yang, Z., Zhang, H. (2022). Comparative Analysis of Structured Pruning and Unstructured Pruning. In: Hung, J.C., Yen, N.Y., Chang, JW. (eds) Frontier Computing. FC 2021. Lecture Notes in Electrical Engineering, vol 827. Springer, Singapore. https://doi.org/10.1007/978-981-16-8052-6_112https://doi.org/10.1109/TVT.2018.2868698.

[10] M. Zhu, S. Gupta, To prune, or not to prune: exploring the efficacy of pruning for model compression, (2017).

[11] S. Vadera, S. Ameen, Methods for Pruning Deep Neural Networks, IEEE Access. 10 (2022) 63280–63300. https://doi.org/10.1109/ACCESS.2022.3182659

[12] "Quantization aware training." Accessed: Jan. 29, 2024. [Online]. Available: https://www.tensorflow.org/model_optimization/guide/quantization/training.

[13] R. Utrilla, E. Fonseca, A. Araujo, L.A. Dasilva, Gated Recurrent Unit Neural Networks for Automatic Modulation Classification with Resource-Constrained End-Devices, IEEE Access. 8 (2020) 112783–112794. https://doi.org/10.1109/ACCESS.2020.3002770.

[14] T.J. O'Shea, T. Roy, T.C. Clancy, Over-the-Air Deep Learning Based Radio Signal Classification, IEEE J Sel Top Signal Process. 12 (2018) 168–179. https://doi.org/10.1109/JSTSP.2018.2797022.

[15] A. Swami, B.M. Sadler, Hierarchical digital modulation classification using cumulants, IEEE Transactions on Communications. 48 (2000) 416–429. https://doi.org/10.1109/26.837045.

[16] A.M. Abdulkarem, F. Abedi, H.M.A. Ghanimi, S. Kumar, W.K. Al-Azzawi, A.H. Abbas, A.S. Abosinnee, I.M. Almaameri, A. Alkhayyat, Robust Automatic Modulation Classification Using Convolutional Deep Neural Network Based on Scalogram Information, Computers. 11 (2022) 162. https://doi.org/10.3390/computers11110162.

[17] D. Hong, Z. Zhang, X. Xu, Automatic modulation classification using recurrent neural networks, in: 2017 3rd IEEE International Conference on Computer and Communications (ICCC), IEEE, 2017: pp. 695–700. https://doi.org/10.1109/CompComm.2017.8322633.

[18] Z. Zhang, H. Luo, C. Wang, C. Gan, Y. Xiang, Automatic Modulation Classification Using CNN-LSTM Based Dual-Stream Structure, IEEE Trans Veh Technol. 69 (2020) 13521–13531. https://doi.org/10.1109/TVT.2020.3030018.

[19] S. Budgett, P. de Waard, Quantized neural networks for modulation recognition, in: T. Pham, L. Solomon, M.E. Hohil (Eds.), Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications IV, SPIE, 2022: p. 44. https://doi.org/10.1117/12.2617678.

[20] S. Kumar, R. Mahapatra, A. Singh, Automatic Modulation Recognition: An FPGA Implementation, IEEE Communications Letters. 26 (2022) 2062–2066. https://doi.org/10.1109/LCOMM.2022.3184771.

[21] Mohammad Chegini, Pouya Shiri, Amirali Baniasadi, RFNet: Fast and efficient neural network for modulation classification of radio frequency signals, ITU Journal on Future and Evolving Technologies. 3 (2022) 261–272. https://doi.org/10.52953/XBPT2357.

[22] D. Góez, P. Soto, S. Latré, N. Gaviria, M. Camelo, A Methodology to Design Quantized Deep Neural Networks for Automatic Modulation Recognition, Algorithms. 15 (2022) 441. https://doi.org/10.3390/a15120441.

[23] Y. Chen, W. Shao, J. Liu, L. Yu, Z. Qian, Automatic Modulation Classification Scheme Based on LSTM With Random Erasing and Attention Mechanism, IEEE Access. 8 (2020) 154290–154300. https://doi.org/10.1109/ACCESS.2020.3017641.

[24] T. Huynh-The, C.-H. Hua, Q.-V. Pham, D.-S. Kim, MCNet: An Efficient CNN Architecture for Robust Automatic Modulation Classification, IEEE Communications Letters. 24 (2020) 811–815.

https://doi.org/10.1109/LCOMM.2020.2968030
.

[25] S.-H. Kim, J.-W. Kim, V.-S. Doan, D.-S. Kim, Lightweight Deep Learning Model for Automatic Modulation Classification in Cognitive Radio Networks, IEEE Access. 8 (2020) 197532–197541. https://doi.org/10.1109/ACCESS.2020.3033989
.

[26] F. Zhang, C. Luo, J. Xu, Y. Luo, An Efficient Deep Learning Model for Automatic Modulation Recognition Based on Parameter Estimation and Transformation, (2021).

[27] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, S. Valaee, Recent Advances in Recurrent Neural Networks, (2017).

[28] H. Lin, A. Gharehbaghi, Q. Zhang, S.S. Band, H.T. Pai, K.-W. Chau, A. Mosavi, Time series-based groundwater level forecasting using gated recurrent unit deep neural networks, Engineering Applications of Computational Fluid Mechanics. 16 (2022) 1655–1672. https://doi.org/10.1080/19942060.2022.2104928.

[29] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, (2014).

[30] R. Yamashita, M. Nishio, R.K.G. Do, K. Togashi, Convolutional neural networks: an overview and application in radiology, Insights Imaging. 9 (2018) 611–629. https://doi.org/10.1007/s13244-018-0639-9.

.

[31] D. Hong, Z. Zhang, X. Xu, Automatic modulation classification using recurrent neural networks, in: 2017 3rd IEEE International Conference on Computer and Communications (ICCC), IEEE, 2017: pp. 695–700. https://doi.org/10.1109/CompComm.2017.8322633.

[32] S.A. Hicks, I. Strümke, V. Thambawita, M. Hammou, M.A. Riegler, P. Halvorsen, S. Parasa, On evaluation metrics for medical applications of artificial intelligence, Sci Rep. 12 (2022) 5979. https://doi.org/10.1038/s41598-022-09954-8.