Synthetic Wireless Signal Generation for Neural Network Algorithms

Tina L. Burns*, Richard P. Martin*, Jorge Ortiz*, Ivan Seskar*, Dragoslav Stojadinovic,*, Ryan Davis*, Miguel Camelo†
* WINLAB, Rutgers University, North Brunswick, NJ, USA
† University of Antwerp - imec, IDLab - Department of Computer Science Sint-Pietersvliet 7, 2000 Antwerp, Belgium
tld95@scarletmail.rutgers.edu, jorge.ortiz@rutgers.edu, seskar, dragos, ryan, rmartin@winlab.rutgers.edu,
miguel.camelo@uantwerpen.be

ABSTRACT - As the use of wireless communication devices increases, so does the need for effective signal generation techniques. We show how different methods of waveform generation with the same signal-to-noise ratios result in varied accuracy performance for modulation recognition when using neural networks (NNs). Our research indicates that two aspects of waveform generation significantly change NN behavior with equivalent SNRs. First, generating purely random IQ constellation points adversely impacts classification accuracy, as real radios do not have completely random constellation points. We illustrate that NNs can use the intermediate IQ samples to improve classification accuracy. Second, we exhibit that adding pure additive white Gaussian noise (AWGN) results in different accuracies compared to using channel distortion models, which include fading, multipath, and phase shifts. We show that relying on SNR alone to characterize signals can result in misleading and incorrect performance evaluations of NNs when applied to radio performance. Our work also demonstrates that care must be taken when applying channels models to simulate noise, as NNs can use specific channel distortion features to identify a modulation.

Index Terms: *Neural Networks, Deep Learning, CNN, LSTM, Fully Connected Neural Network MATLAB, GNU Radio, Synthetic Data Generation, AWGN, Modulation Recognition, Wireless Radio Signals*

I. Introduction

Technology trends have been increasing transistor count and decreasing cost per transistor, which in turn has enabled wireless radio communications to be added to nearly every digital device. The increase in the accessibility of radio communications has led to a rapid development cycle. Recent years have thus seen an explosion of interest in applying machine learning techniques in radio designs

Evaluation of novel radio designs that use machine learning, and in particular neural networks, requires methods for both generating waveforms and creating a noise environment. On one end of the spectrum, waveforms and environments can be purely synthetic, where the waveforms are generated in software and the designs are simulated in software. The other end of the evaluation spectrum uses real devices in real environments. One example of this is using WiFi devices in an urban street intersection. There are many intermediate points that use a mix of synthetic waves, emulated signals and real devices. While using real devices and environments is optimal from the perspective of representativeness, their drawbacks are that they are not always reproducible or controllable. This limits the ability to understand the impact of a single factor [7].

Varying the Signal-to-Noise Ratio (SNR) is the classic approach for evaluating radio system designs [5]. This metric is commonly defined as the ratio of the signal power to 'all other' signals, where other signals can originate from a variety of causes, such as multipath interference, other transmitters, etc. We show that SNR alone is an incomplete characterization when evaluating Neural Networks (NNs) used for signal recognition. Because NNs are excellent at pattern recognition, they recognize systemic distortions as signal features, impacting recognition accuracy. Two signals can have equivalent SNRs but vary widely in the predictability of the noise.

We demonstrate that two critical aspects of synthetic waveform generation must be considered when using NNs. The first aspect pertains to signal generation. Digital signals use a stream of symbols that correspond to points on the IQ plane. The resulting points can be chosen purely randomly or by using constraints imposed by underlying analog waveforms. We reveal that choosing IQ symbol positions purely randomly introduces more randomness than real radio signals, as real signals have intermediate IQ points which can be recognized by a NN to aid classification accuracy. The second aspect to consider is noise generation. Noise can be added as a pure additive white Gaussian noise (AWGN), or signal distortions can be applied using a channel model. We show the choice of noise generation greatly impacts NN behavior, even when the AGWN and channel models SNRs are equivalent. Although AWGN is not a realistic channel model, as it over-applies randomness to a signal compared to a real radio, it prevents an NN from using any distortions as recognizable features.

Our work is novel because it compares the synthetic wireless generation and noise characteristics of two popular platforms, MATLAB and GNU Radio. The remaining sections of this paper are organized as follows: Section II reviews some related work in the area. A description of the simulation techniques is provided in section III. Details about the neural networks can be found in section IV. Section V discusses our results. The paper culminates with our conclusion and future work in sections VII and VIII.

II. Background and Related Work

Recall a digital baseband signal is represented by a stream of IQ samples as received from an analog to digital converter (ADC). A transmitted symbol is represented by the position of the sample on the IQ plane. Both the IQ symbol and IQ streams can be synthetically generated using computer simulations. Two common programs to do this are GNU Radio and MATLAB. Often, researchers implicitly assume (1) that the signals generated are representative of the IQ patterns in real radio systems and (2) that the AWGN added in these simulation techniques are sufficient for characterizing the noise of radio systems. However, our research indicates that these implications may be flawed and that appropriate attention should be allotted towards determining the best method for both signal and noise generation.

GNU Radio is an open-source software containing several tools to model radio communication signals. [10] Researchers have used GNU Radio to model OFDM systems, test algorithms for wireless smart grids, and generate signals for software defined radios. [10, 4] The GNU Radio channel models are extremely useful because they allow users to adjust variables such as frequency offsets, multipath distortions, and timing errors [4]. Additionally, the channel models enable users to introduce and adjust aspects of additive white Gaussian (AWGN) noise.

Tim O'Shea and Nathan West used the benefits of GNU Radio's AWGN features as they analyzed neural networks for radio signal processing. Similar to our work, their research involved using neural networks to classify synthetically generated signals. Unfortunately, they do not compare their results to signals created in MATLAB. Additionally, their work does not examine the impact of the different noise channels in GNU Radio. [12] Our studies indicate that the AGWN of some GNU Radio channel models do not provide adequate variation and are detectable by neural networks thus reducing its effectiveness as a source of random noise.

MATLAB/Simulink is another software tool with a variety of radio signal generation features, including the ability to simulate modulation schemes, add different noise characteristics, and adjust signal features such as gain, frequency, and bandwidth. These tools are advantageous and allow researchers to simulate cognitive radio systems, build software defined radios, evaluate spectrum sensing algorithms, and explore spectrum management techniques. [6, 15]

A recent study by Yu Wang et. al. used both MATLAB and GNU Radio to generate wireless signals for AMR classification. Similar to our research this work used IQ samples to examine the performance of various DL networks. [16] However, their work lacks a direct comparison of datasets generated separately on each platform. Also, their work does not discuss how GNU Radio and MATLAB differ in IQ pattern placements.

Our research shows that some of the tools used in MATLAB simulations may produce IQ patterns that are too randomized and ignore some of the filtering techniques applied in real radios. This in turn creates IQ patterns that are harder to classify for some prediction algorithms which may falsely represent how neural networks respond to real radio signals.

III. Methodology

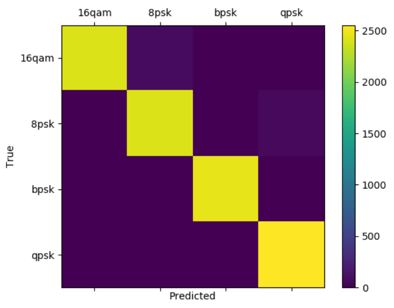
Our observations on the signal generation's impact on NN accuracy are motivated by our work testing different neural networks for use in modulation recognition [3]. A goal of that work was to develop neural networks that given an unknown signal, could classify the signal's modulation type, for example, BPSK or 16QAM. This section describes how the signals were generated, and how noise was applied to them to emulate challenging radio environments. We then describe which neural networks were used.

A. Signal Generation

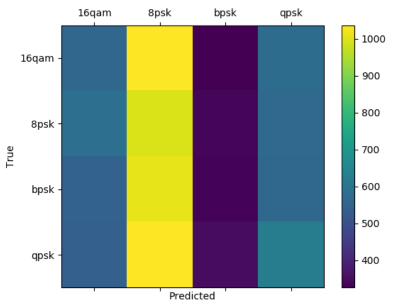
We employed both MATLAB and GNU Radio for signal generation, which we use for the training and testing sets to our neural networks. We generated several sets of signals: (1) the MATLAB set, (2) the GNU Radio sets, and (3) the GRID set. The MATLAB dataset was formed by using MATLAB as its signal generator; while the GNU Radio and GRID sets used the GNU Radio for signal generation. Within the GNU Radio sets, we used the same signal generator, but applied noise using different approaches. These approaches lead to separate sets, which we describe below.

We generated signals with BPSK, QPSK, 8PSK, and 16QAM modulations. IQ samples were output to binary files in 32-bit floating-point format, with interleaved I and Q values. The final output files of the GNU Radio and MATLAB dataset were normalized to values between -1 and 1. The output files of the Grid dataset were retrieved directly from the USRP receiver and were not normalized.

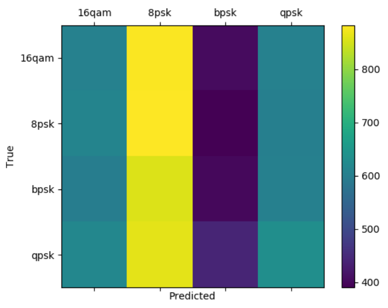
MATLAB We select a symbol stream that corresponds to IQ constellation points at random, where N is the number of constellation points. The IQ value of the constellation point is appended to the signal stream and then modified by adding an AWGN to attain the appropriate SNR. Because successive IQ points are randomized, the constellation points are not necessarily adjacent to one another; moving to the next IQ point often involves crossing the origin. To mitigate the distance between successive IQ samples, we implemented gray-scale binary ordering for the mapping. It ensures that the binary representation of the adjacent modulation symbols only differ by one bit. Mathematically this is beneficial because it minimizes the bit error rate [2]. However, even with grayscale mapping, the constellation points show more variation in movement from point to point than that of the GNU Radio data. This is discussed more in section V.



(A) Noise and Channel
(SNR of -40 dB)



(B) Additive Noise
(SNR of -40 dB)



(C) Noise Only

Fig. 1. Feature Detection at low SNR levels

GNU Radio GNU Radio allows users to adjust several aspects in constellation generation including variables such as the number of constellation points, types of encoding, and samples per symbol (SPS). [1, 17] Setting a high SPS results in oversampling, and the constellation plot shows intermediate transition points between the constellation symbols. Additionally, GNU Radio may use low pass filtering on the baseband signal which assists in smoother transition between constellation points. [1]. The smoother transitions prevent the temporally adjacent constellations from jumping across the origin in the IQ plane.

GRID For this set, the GNU Radio *Additive Noise* signal output files were transmitted over the air from a USRP x310 transmitter to a USRP B210 receiver 3 feet apart. The recorded signals from the receiver become the GRID set. This set provides a test set that shows how the neural networks respond to the impact of real radio hardware and a multipath environment.

B. Noise Generation

For the MATLAB set, we applied AWGN to the IQ samples directly. For the GNU Radio sets, we applied three noise models, resulting in 3 signal sets: (1) Noise and Channel, (2) Additive Noise, and (3) Noise Only.

Noise and Channel This signal set is built using three GNU Radio building blocks: the Modulation Tx, Multiply Const, and Channel Model blocks. The *Modulation Tx* block generates the constellation type as specified by the user. *Multiply Const* is the variable for adjusting the gain of the constellation signal. The *Channel Model* block combines the signal with random Gaussian noise. The inputs for the Channel Model block includes the noise voltage, which adjusts the gain of the noise. Additionally, this block contains a variable for the SNR input which adjusts the noise voltages to achieve the appropriate signal to interference ratio.

Additive Noise For the additive noise generation set, the channel model block was removed and a separate GNU Radio block for the noise was included. The GNU Radio blockchain contains many of the same variable inputs from the *Noise and Channel*.

If we consider s[n] as the noiseless signal, w[n] the AWGN noise, and y[n] as the output signal, the profile can be represented by equation1.

$$y[n] = s[n] + w[n] \tag{1}$$

Noise Only The Noise Only GNU Radio signal set contains a fast noise block without a modulation source block. For each signal set, the files were arbitrarily labeled as different modulation schemes.

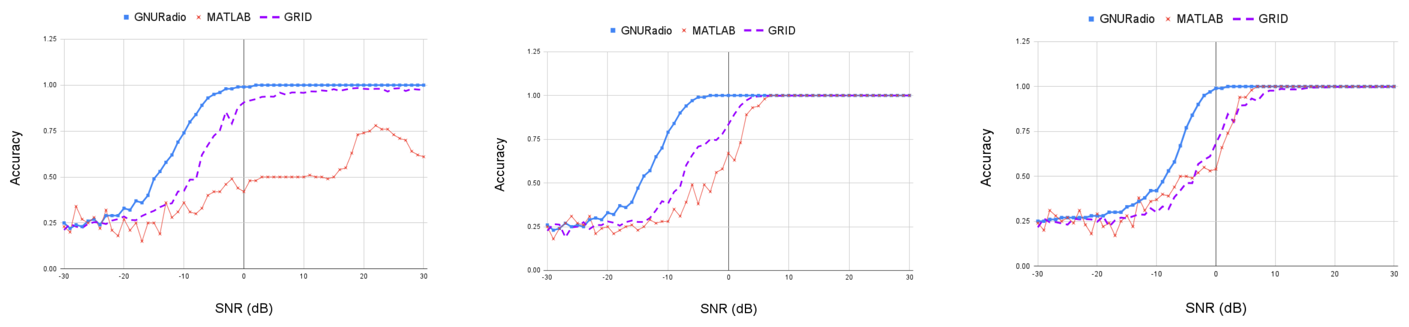
IV. Neural Network Description

For this project we explored 3 types of neural networks: (1) Fully Connected Neural Network (FCNN), Convolutional Neural Network (CNN), and the Long Short Term Memory (LSTM) network.

Fully Connected Neural Network (FCNN) Fully connected networks contain multi-layer perceptions (MLPS) in which all nodes are connected to the nodes in the previous layers.[9] Our FCNN network is comprised of 4 fully connected hidden layers which employ a softsign activation function, one dropout layer, and one fully connected layer at the output that uses a relu activation function.

Convolutional Neural Network (CNN) Convolution Neural Networks are extremely popular and have been implemented in several areas including image classification, SAR image segmentation, computer recognition, speech identification, and automatic modulation recognition. [14, 11] The CNN Network contains 6 convolutional layers, 3 hidden fully connected layers, a fully connected output layer, and some additional upsampling and max-pooling layers. The CNN uses categorical classification.

Long Short Term Memory (LSTM) LSTM's are time recursive deep learning networks that have been used in a variety of applications including speech synthesis, handwriting recognition, audio analysis, video analysis, and language modeling and translation. [8] Our LSTM model contains 2 LSTM layers with 32 filters, one dropout layer, and one fully connected output layer. This network employs a rsmprop optimizer.



FCNN CNN LSTM

Fig. 2. SNR vs Accuracy (2.4 Million Samples)

V. Experiments and Discussion

Our experiments can be divided into the following sections: (1) noise analysis, (2) accuracy analysis, and (3) temporal clustering. In noise analysis, we describe how adding noise using channel models can influence neural networks. In accuracy analysis we show how the MATLAB approach using AWGN results in accuracies that are too low compared to the GNU Radio signal sets. The temporal clustering illustrates how smoothing the IQ samples results in better NN performance. All experiments contained data with the following modulation schemes BPSK, QPSK, 8PSK and 16QAM.

A. Noise Analysis

Since SNR significantly impacts radio performance, it is essential to develop effective noise models when generating wireless signals. One of the challenges experienced during this project was developing signal sets with noise and distortion characteristics that could not be “learned” by the neural networks. For the *Noise and Channel* set, the networks recognized distortions in the GNU Radio channels as features. This resulted in high modulation classification accuracy even when the signal was highly attenuated and the gain of the noise channel dominated. For example, the neural networks were able to classify the correct modulation scheme of an unknown signal with over 95% accuracy even when the signal was attenuated. Part (a) of figure 1 demonstrates this high classification accuracy in the confusion matrix.

Confusion matrices are powerful tools that allow researchers to visualize the precision, accuracy, and recall of deep learning networks in a graphical format[13]. For our confusion matrices, the predicted and true modulation classes are on the X-axis and Y-axis, respectively. The more densely populated predictions are in yellow, while the less densely populated distributions are a magenta color. A descending yellow diagonal line from the upper left to the lower right corner of the graph indicates high accuracy. From part (a) of this graph, we see that the DL network predicts with high accuracy even at an SNR of -40 dB where noise dominates the channel.

Upon further investigation, we discovered that this was caused by the neural networks recognizing the distortion characteristics in the Channel Model block of *Noise and Channel* GNU Radio Profile. So, as discussed in section III, we removed the channel model and added the noise to the signal externally in the *Additive Noise* profile. For this profile, we attained approximately 25% classification accuracy at an SNR of -40 dB. This low accuracy is also confirmed by the distribution allocation represented in part (b) of figure 1.

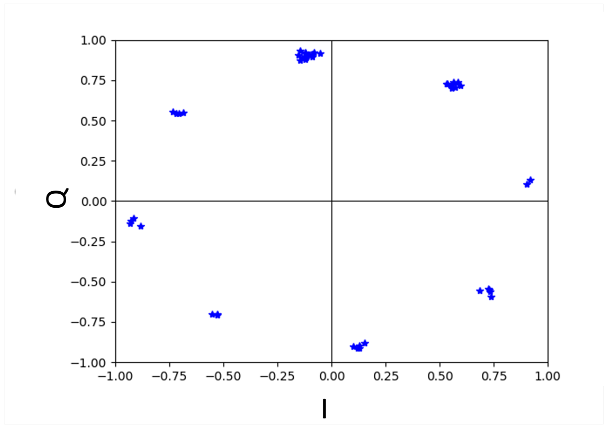
	GNURadio	**MATLAB**	**GRID**
CNN	100%	67%	83%
FCNN	99%	42%	91%
LSTM	99%	54%	68%

TABLE I
Accuracy Comparison at an SNR of 0dB

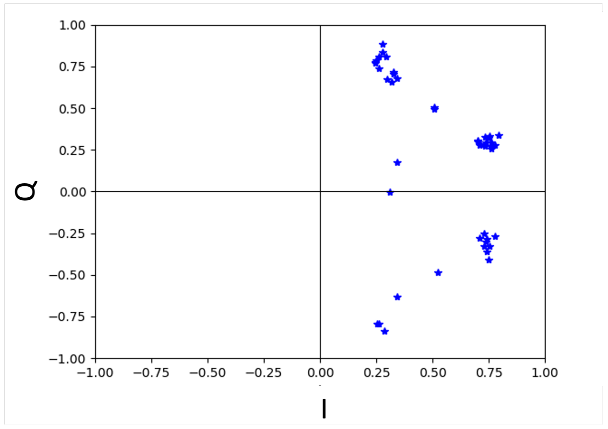
B. Accuracy Analysis

For the accuracy analysis, we observe classification accuracy from SNR levels of -30dB to 30dB in to get a comprehensive view of how the NN performs over a broad SNR range. Figure 2 shows the SNR vs. classification accuracy of the FCNN, CNN, and LSTM networks for the MATLAB, Additive Noise GNU Radio, and GRID signal sets. All three DL neural networks perform much better using the GNU Radio set than the MATLAB set. For instance, at an SNR of 0 dB the accuracy of the FCNN network is 99% for GNU Radio and only 42% with the MATLAB set. For the LSTM network, the MATLAB set has a 40% lower accuracy rate than the GNU Radio set at an SNR of 0 dB. This is summarized in Table I The CNN shows a similar response with the MATLAB signal set at 50% below the GNU Radio accuracy at an SNR of -10 dB. The highest accuracy attained for the FCNN with the MATLAB set is 78%. We hypothesize that the difference in accuracy is caused by the effect of temporal clustering. Overall, we found that the CNN network had the highest accuracy while the FCNN network had the lowest. We also observed that the LSTM network has the smallest gap in accuracies between the three radio generation platforms.

Analysis of the GRID set shows that, as expected, the networks’ classification accuracies are slightly lower when the data is transmitted over the air with actual radios. At an SNR of 10 dB, the accuracy of the GRID data for the FCNN and CNN networks are 40% the accuracy of the GNU Radio datasets.



(A) MATLAB



(B) GNU Radio

Fig. 3. 8PSK Constellation Plots for 100 IQ Pairs

This indicates that the noise and distortions of the environment and of the radios themselves will impact the classification accuracy, even if the systems are modeled well.

VI. Temporal Clustering

Upon examining the data we observed that although static IQ constellation plots look very similar, the temporal progression of the IQ points are very different. For example, for an 8PSK modulation in GNU Radio, the points travel to adjacent IQ locations. The impact of this motion is that the shift in IQ points does not involve crossing the origin. Instead, the IQ constellation points remain in one half of the IQ plane for a large chunk of points before moving to the next quadrant. In the MATLAB data however, the points are more random and do not move to the next adjacent point. This effect is observed in figure 3 which displays the IQ constellation plots for 100 IQ pairs for the MATLAB and GNU Radio sets.

VII. Conclusions

Our research demonstrates that different signal generation techniques have widely different impacts on the classification accuracy of neural networks, even with equivalent SNRs. The results from the noise generation analysis indicate that some synthetic data generation methods may have very high accuracy results if the neural networks can distinguish unique channel characteristics that contribute to noise. Thus, it is imperative that the noise added to the system be appropriately randomized.

Our work also shows that it is also possible to over-randomize the signal. For example, some signal generation methods result in low accuracy because of differing approaches to mapping IQ constellation points. As shown from the MATLAB SNR vs Accuracy results, if the output of the IQ constellation is purely randomized, the constellation will follow a more randomized temporal allocation pattern than what happens in realistic radio signals. This in turn makes it more difficult for the networks to characterize the modulation.

It is important to understand these biases when modeling radio communication signals and evaluating the neural networks' performance. Understanding these bias will help designers create synthetic data that is more characteristic of actual radio signals. It will also ensure that neural networks have better performance in real-world environments.

VIII. Future Work

Future work for this project should include testing the GNU Radio data with channel models containing more randomized distortions rather than adding AWGN from an external block. Additionally, further work could include evaluating the MATLAB data with additional functions, such as a root cosine filter (RRC), that could smooth out the IQ transition points. To add to this, testing could be performed on more realistic training sets that include signals from live devices such as WiFi and Bluetooth devices in harsh multipath environments. This would allow us to assess how the networks perform on classifying real-world signals.

IX. Acknowledgements

We would like to thank all those at Rutgers University and WINLAB that helped with this project. Funding for this work was provided by NSF awards OAC-1925482 and CNS-1836901. Future financial support of this research will be from NSF award RU-NET project 826336.

Due to space limitations, some details were excluded from this paper. Additional information about the Methodology and Neural Networks discussed here can be found in [3].

References

[1] Gnu radio - psk mod block - unexpected constellation diagram, accessed May 2021. https://dsp.stackexchange.com/questions/26403/gnu-radio-psk-mod-block-unexpected-constellation-diagram.

[2] Gray-coded binary ordering, accessed May 2021. www.mathworks.com/help/comm/ug/gray-coded-binary-ordering.html.

[3] Tina Burns, Richard P. Martin, Dragoslav Stojadinovic, Ryan Davis, Ivan Seskar, Jorge Ortiz, and Miguel Camelo. Evaluating deep learning networks for modulation recognition. *IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2021.

[4] P. Ferrari. E. Sisinni. A. Flammini. A. Depari. Adding accurate timestamping capability to wireless networks for smart grids. *Computer Networks*, pages 1056–1063, July 2014. doi:10.1016/j.comnet.2014.03.005.

[5] electronicnotes. Radio signal to noise s/n ratio, snr. https://www.electronics-notes.com/articles/radio/radio-receiver-sensitivity/signal-to-noise-ratio-s-n-snr-formula.php.

[6] Subhajit Chatterjee Goutam Ghosh, Prasun Das. Simulation and analysis of cognative radio system using matlab. *International Journal of Next-Generation Networks (IJNGN)*, 6, June 2014. 10.5121/ijngn.2014.620331.

[7] Muhammad Imran, Abas Md Said, and Halabi Hasbullah. A survey of simulators, emulators and testbeds for wireless sensor networks. In *2010 International Symposium on Information Technology*, volume 2, pages 897–902. IEEE, 2010.

[8] Greff K. Srivastava RK. Koutnik J. Steunebrink BR. Schmidhuber J. Ltsm: A search space odyssey. *IEEE transaction on neural networks and learning systems*, 28, 2017. doi:10.1109/TNNLS.2016.2582924.

[9] J Janke, M Castelli, and A Popovič. Analysis of the proficiency of fully connected neural networks in the process of classifying digital images. benchmark of different classification algorithms on high-level image features from convolutional layers. *Expert Systems with Applications*, 135:12–38, 2019. doi.org/10.1016/j.eswa.2019.05.058.

[10] C. Anjana. S. Sundaresan. Tessy Zacharia. Gandhiraj Rajendran. Amrita Vishwa Vidyapeetham. Soman Kp. An experimental study on channel estimation and synchronization to reduce error rate in ofdm using gnu radio. *Procedia Computer Science*, 46:1056–1063, December 2015. doi:10.1016/j.procs.2015.01.017.

[11] VJ Lawhern, AJ Solon, NR Waytowich, SM Gordon, CP Hung, and BJ. Lance. Eegnet: a compact convolutional neural network for eeg-based brain–computer interfaces. *Journal of Neural Engineering*, 15:05613, July 2018. doi:10.1088/1741-2552/aace8c.

[12] Tim O'Shea and Nathan West. Radio machine learning dataset generation with gnu radio. In *Proceedings of the6thGNU Radio Conference*. https://pubs.gnuradio.org/index.php/grcon/article/view /11/10.

[13] Towards Data Science. Understanding confusion matrix. https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62.

[14] F Sharifzadeh, Akbarizadeh G, and Y. Seifi Kavian. Ship classification in sar images using a new hybrid cnn–mlp classifier. *Journal of the Indian Society of Remote Sensing.*, 47:551–562, 2019. doi:10.1007/s12524-018-0891-y.

[15] Ahmad Ali Tabassam, Farhan Azmat Ali, Sumit Kalsait, and Muhammad Uzair Suleman. Building software-defined radios in matlab simulink - a step towards cognitive radios. In *2011 UkSim 13th International Conference on Computer Modelling and Simulation*, pages 492–497, 2011.

[16] Miao Liu Jie Yang Wang, Yu and Guan Gui. Data-driven deep learning for automatic modulation recognition in cognitive radios. *IEEE Transactions on Vehicular Technology*, 68:4074–77, 2019. https://doi.org/10.1109/TVT.2019.2900460.

[17] Wikipedia. Guided tutorial psk demodulation, accessed May 2021. https://wiki.gnuradio.org/index.php/Guided_Tutorial _PSK_Demodulation .