Neural Network based Modulation & Channel Coding Identification for SATCOM Systems

4th & Last Milestone's Performance Evaluation

Principal Investigator:

Syed Ali Hassan, Ph.D.

Electrical Engineering

Department of Electrical Engineering

School of Electrical Engineering and Computer Science (SEECS)

National University of Sciences and Technology (NUST)

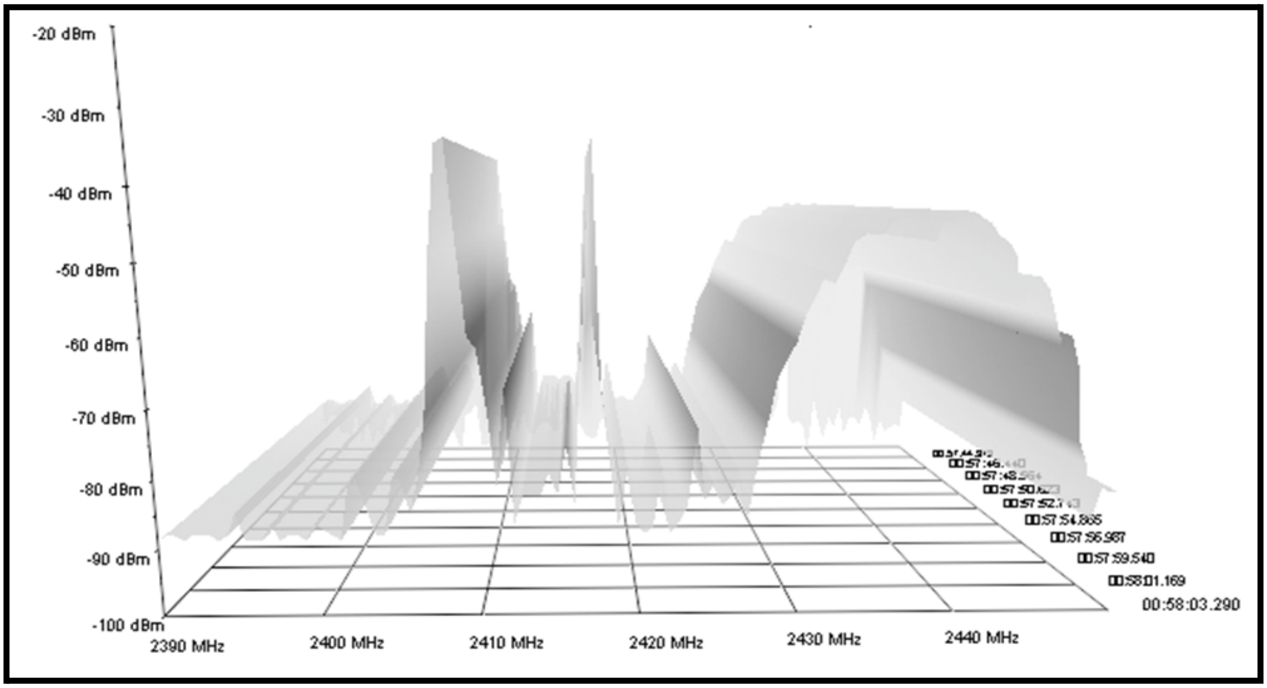
Islamabad, Pakistan

2022

Lab prototype of SDR-based AMR for demonstration of proposed Neural Network model

Project report

Neural Network based Modulation & Channel Coding Identification for SATCOM Systems



Prepared by:
Research Assistant: **Saad Iqbal, BE**
Electrical Engineering

Principal Investigator: **Syed Ali Hassan, Ph.D.**
Electrical Engineering
Department of Electrical Engineering
School of Electrical Engineering and Computer Science (SEECS)
National University of Sciences and Technology (NUST)
Islamabad, Pakistan

Table of content

Preface

This document is penned down in reference to the 4th & concluding milestone of the research project “Neural Network based Modulation and Channel Coding Identification for SATCOM Systems". The author has touched every aspect encountered by him in implementing the Proof of Concept (PoC) of Real-time AMR. Along with that, the author has elaborated the whole project sequence with the quick recap of 8-months efforts placed for the success of the project.

Prior to this final report, following documents are prepared according to each milestone of this research project:

- Saad Iqbal and Syed Ali Hassan, "*Literature Review & Feasibility Report of Neural Network based Modulation and Channel Coding Identification for SATCOM Systems*".
- Saad Iqbal and Syed Ali Hassan, "*Proposed Neural Network for MODCOD report of Neural Network based Modulation and Channel Coding Identification for SATCOM Systems*".
- Saad Iqbal and Syed Ali Hassan, "*Lab prototype of AMR for demonstration of proposed NN model report of Neural Network based Modulation and Channel Coding Identification for SATCOM Systems*".

This document contains three sub-sections. To ensure that the reader fully understands, sections are interlinked. The first section discusses the extension of AMR testbed design. The author has performed real-time signal processing enabling the existing design to perform modulation classification at real-time. In the next section, every minute detail of designing AMR specific graphical user interface has been discussed. The last section has elaborated the performance metrics that have been observed during real-time AMR testing.

The report concludes on the conclusion & future work, which will elaborate the overall achievements gained through-out the project along with the identified research gaps. This document would be treated as a final report in regard to this research project.

Introduction & Approach

The project "Neural Network based Modulation and Channel Coding Identification for SATCOM Systems" crux is to design a real-time system to classify the modulation & coding scheme embedded in the incoming broadcasted signal via emerging technology of neural network. In literature, the communication equipment equipped with such intelligence is known as *cognitive radio*.

Our final milestone is to demonstrate a lab prototype of NN-based cognitive radio, which can predict the modulation of incoming I-Q data stream with ~95% confidence. The sub tasks of our final milestone are:

1. SDR based data transmission and reception.
2. NN based MODCOD identification using SDR transmissions in a lab environment.

From the beginning of the project, the indigenous design for AMR is proposed whose implementation is kept in a modular way to ensure scalability in future. The development of SDR-based testbed for AMR has already been discussed [3]. This document would continue from where we have left, and would end our discussion on the analysis on real-time AMR performance.

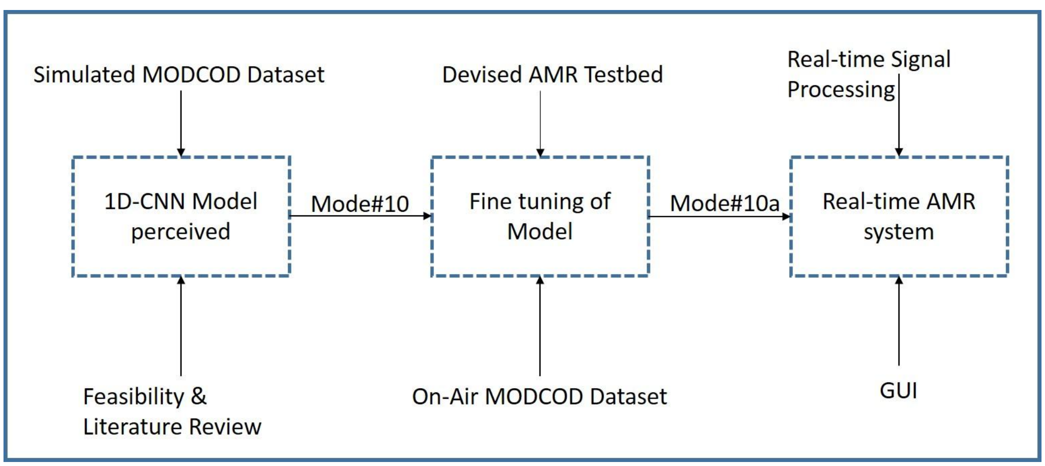


Figure no. 1 Chronological order of tasks being carried out during Project

The figure no.1 illustrates the chronological order of tasks being carried out throughout the project timeline. The approach for the development of the user-friendly real-time AMR using state of the art SDRs is stated below:

1. The pre-trained model#10a developed during the third milestone [3], passed through rigorous testing with an I-Q stream fetched from Rx USRP.
2. The algorithm designed for real-time signal processing required prior to prediction model.
3. GUI implementation of designed AMR to provide smooth interaction for users.
4. The evaluation of indigenous AMR's performance along with a demo video of its working sequence.

The same SDR-based testbed, used in for our previous milestone as shown in figure no.2, is utilized for the tasks in hand.

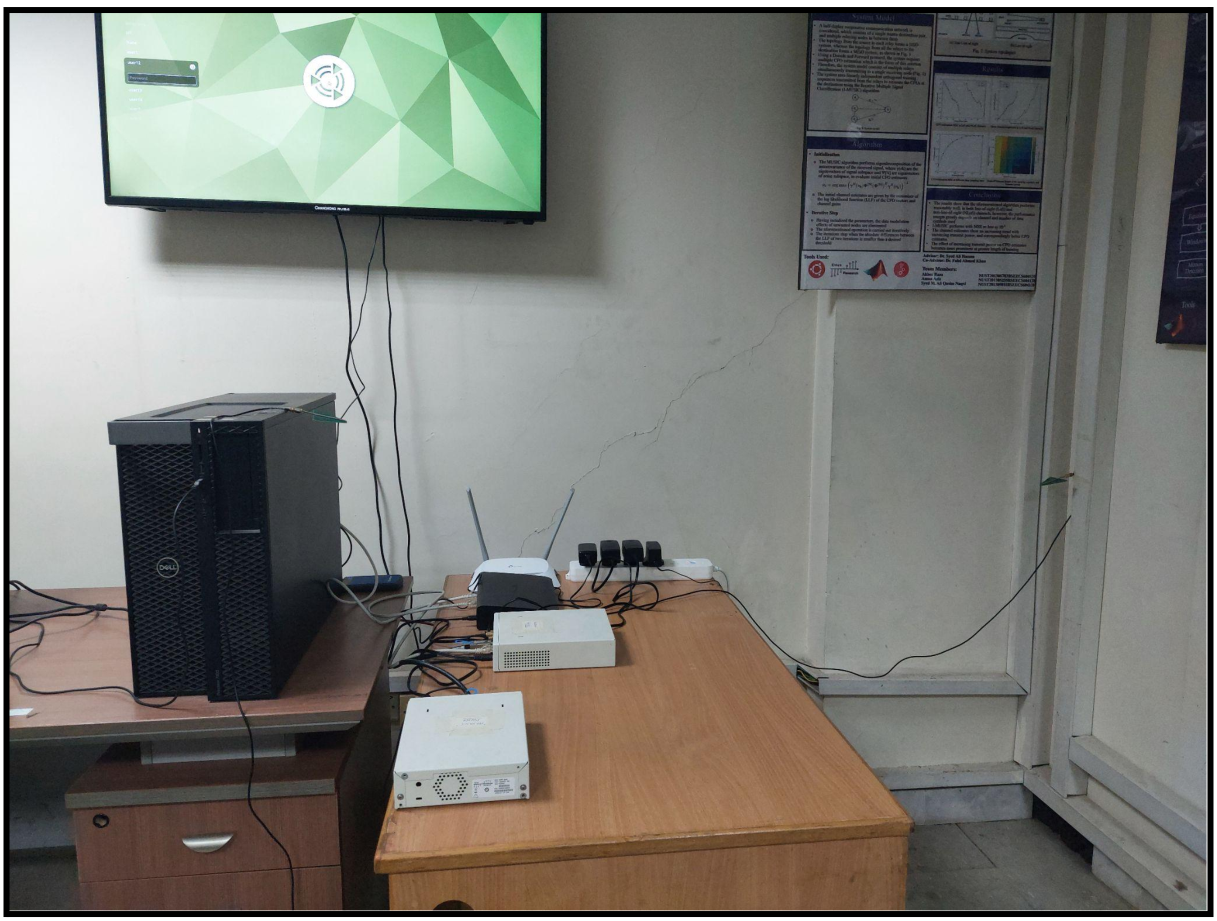


Figure no. 2 Indigenous SDR-based testbed for realtime AMR application

Real-time SDR-based AMR design for proposed NN model

In the previous report [3], the proposed neural network 1D-CNN Model#10a has been designed which out-performed the rest of models specifically for OnAir dataset. It's time to extend our efforts to develop the real-time SDR-based AMR which utilizes the proposed NN model for classification purposes. The glimpse of "SDR-based AMR hardware design" can be viewed in a short video titled "*system.mp4*" (provided in resource CD). The software design of real-time SDR-based AMR consists of two modules i.e. GNU radio application and modulation classifier interfaced together through the ZMQ protocol as shown in Figure no. 3.

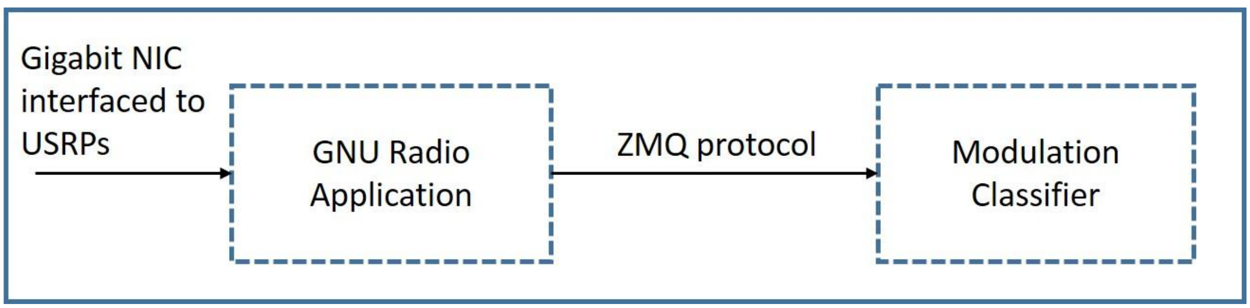


Figure No. 3 Flow diagram of AMR Software Design

GNU Radio Application

The GNU Radio application of AMR design is interface to SDRs and responsible for following operation [3]:

- Data broadcasting using modulation scheme among the selected ones i.e. BPSK, QPSK, PSK8, PSK16, QAM16 and QAM32.
- Capturing the incoming signal and passing it to the data processing window via ZMQ protocol for training and testing of NN-models.
- The real-time SNR monitoring.
- Phase correcting at the receiver end.
- Real-time spectrum monitoring using QT based GUI
- The real-time controlling the Tx and Rx gains.

All these features are accessible on the QT-GUI window of the *AMR GNU Radio application*, and assists the user to tune the SDRs and out-going I-Q stream setting as per their requirement. For the lab prototype, Both Tx & Rx USRPs are tuned at 3G Hz with a sample rate of 1M Samp/s. The broadcasted I-Q signal could be any of six modulations. The Rx USRP captures the broadcasted signal and passes it to the *Modulation Classifier* after downsampling it with a factor of 25.

ZMQ Protocol

ZMQ also known as *ZeroMQ protocol* is transport layer protocol for exchanging messages between two peers over a connected transport layer such as TCP. *GNU Radio application* is using ZeroMQ PUB service to broadcast outgoing data on the network. On the other end, *Modulation Classifier* utilizes the ZeroMQ SUB service as shown below for successful retrieval of broadcasted data. The nature of broadcasted data is *complex64* at *40 kHz* baud-rate.

```
context = zmq.Context()
socket = context.socket(zmq.SUB)
socket.connect("tcp://127.0.0.1:55555") # connect, not bind, the PUB will bind, only 1 can bind
socket.setsockopt(zmq.SUBSCRIBE, b'') # subscribe to topic of all (needed or else it won't work)
```

Modulation Classifier

M*odulation Classifier Application* of AMR is developed in python scripting language, and it has three main functions. These are given below in their execution sequence:

- Fetch the incoming I-Q data via ZMQ SUB protocol and write it over ring buffers.
- Real-time signal processing on the I-Q block data.
- Draw the inference using the proposed NN-model.

Fetch

The *socket.poll()* function keeps a tap on the network for incoming messages. Upon success, the raw data is translated back in its original data-type and is transferred to *buffer_NIC* as shown below:

```
    while(not(Pause_switch)):
      if socket.poll(10) != 0: # check if there is a message on the socket
        msg = socket.recv() # grab the message
        data = np.frombuffer(msg, dtype=np.complex64, count=-1) # make sure to use
correct data type (complex64 or float32); '-1' means read all data in the buffer

        for values in data:
          buffer_NIC[pointer_NIC] = values
          pointer_NIC = np.remainder(pointer_NIC +1, 512)

         ##
         # Rest of code omitted for avoid confusion
         ##
      else:
        time.sleep(0.1) # wait 100ms and try again
```

Pre-processing

As soon as the inference job is activated. The snapshot of the buffer is relayed to the pre-processing routine, which transforms the I-Q data into the input block of the NN-model as shown below:

```
if (data_updated and not(give_me_new_data)):
  test_samp[0,:,0] = np.real(sent_pkt[:])
  test_samp[0,:,1] = np.imag(sent_pkt[:])
  test_samp[0,:,2] = np.abs(sent_pkt[:])
  test_samp[0,:,3] = np.angle(sent_pkt[:])
```

Note: *The nature & dimension of input layer of model#10a is unnormalized and [512,4] respectively.*

Inference

The pre-trained model is loaded through the *Load_model0()* function at the beginning. Later *keras.models* native routine is called upon the input block data for inference as shown below:

```
import tensorflow as tf
from keras.models import Sequential, load_model
from tensorflow.keras.models import Model

def Load_model0():
  global Load_model0
```

```
  Load_model0 = load_model(text0.get(1.0, "end-1c"))
pred = Load_model0.predict(test_samp)
```

The inference results are then translated in classification of modulation type used in the input block data. As long as new data keeps coming, the model would keep predicting the modulation scheme of incoming data.

Graphical User Interface (GUI) of AMR

General Overview

The Graphical User Interface (GUI) for Modulation Classifier is shown in figure no.4. There are seven prominent elements, which are as follow:

- **Project title**, states the project title of the application which is "Modulation Classification Program Developed at Nust".
- **Model name**, is the output textbox, it contains the name & path of the selected NN-model.
- **Browse**, button opens up the home directory and allows the user to select the model from his/her local directory.
- **Load,** button loads the selected model into the prediction script.
- **CLI space**, prints the prediction results along with a timestamp.
- **Analysis/Stop**, button starts and stops the prediction model from making a decision on the incoming I-Q stream.
- **Exit**, button close the modulation classifier application in a smooth manner.

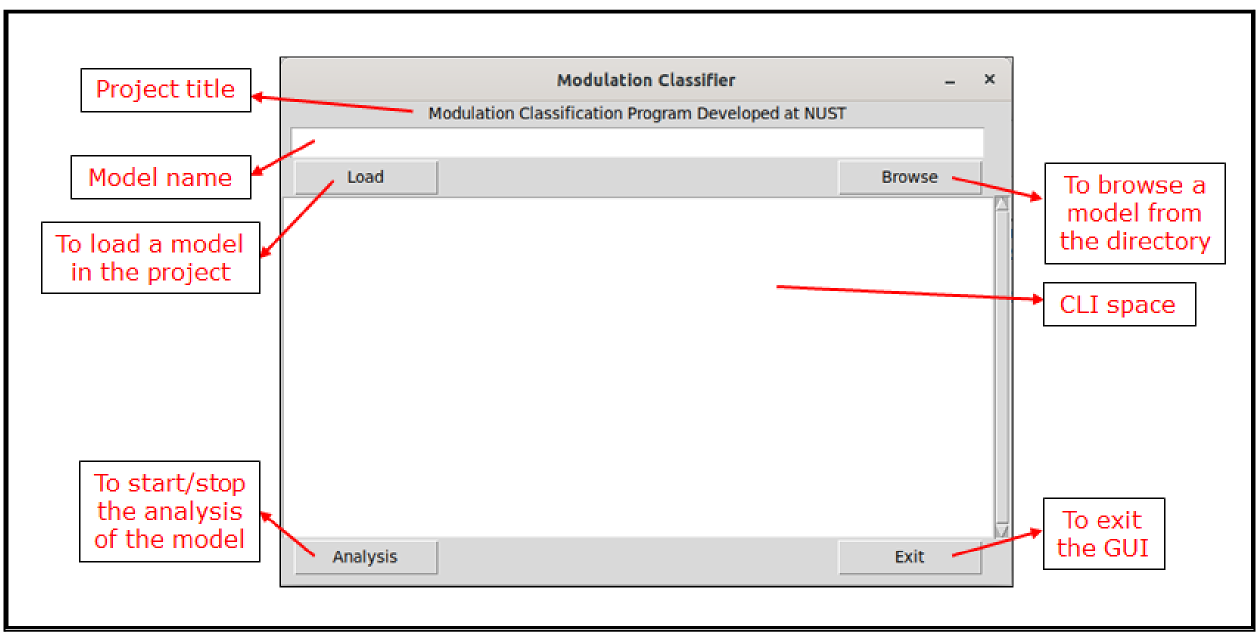


Figure No. 4: GUI of the Modulation Classifier

Application's Frontend

The Frontend of classifier application is developed in Tkinter. *Tkinter is considered the de facto Python GUI framework*. The grid manager is used for managing the items on the main application window. The GUI buttons action is configured with their respective backend tasks as shown in the figure no. 05 and run-time errors are avoided by efficient coding practices. For more, please consult the resource CD for the source file "MODCOD_GUI.py" of the GUI application.

```
label0 = Label( top, text="Modulation Classification Program Developed
at NUST", anchor=CENTER) # Description

button0 = Button(top, text="Browse", command=open_file, height=1, width
=12)  # Browse the model file from the directory
text0 = Text(top, height=1, width=49, font=('Times New Roman',14),
state='disabled')  # Selected_file

text1 = Text(top, height=15, width=75, font=('Times New Roman',12),
state='disabled')  # Output Window

button1 = Button(top, text="Analysis", command=analysis_function,
height=1, width=12) # Start Analysis

button2 = Button(top, text="Exit", command=decent_close, height=1,
width=12) # Exit the program

scrollb = Scrollbar(top, command=text1.yview)

label0.grid(row = 0, column = 0, columnspan = 2, padx=120)
button0.grid(row = 1, column = 1, sticky = W, pady = 2, padx = 15)
text0.grid(row = 1, column = 0,sticky = W, padx = 5, pady = 2)

text1.grid(row = 2, column = 0, columnspan = 2)
scrollb.grid(row=2, column=2, sticky='ns')
text1['yscrollcommand'] = scrollb.set

button1.grid(row = 3, column = 0, sticky = W, padx = 10)
button2.grid(row = 3, column = 1, sticky = E, padx = 10)

top.geometry("640x400")
top.resizable(False,False)
```

Figure no. 05 Snapshot of "MODCOD_GUI.py" script

GUI Tutorial

Although, *modulation classifier* application is very straight-forward. For the ease of the user, following are the tutorial steps of the modulation classifier:

1. Firstly, the model is browsed from the directory using the ‘Browse’ button, which ends up showing the full model path location in the ‘Model name’ field as shown in the figure no. 4.
2. The ‘Load’ button is then pressed to load the browsed model in the project.
3. Afterward, the ‘Analysis’ button is pressed to start the analysis of the loaded model. This action starts showing the CLI information in the ‘CLI space’ shown in the figure.
4. Once the analysis is started, the ‘Analysis’ button is turned into a ‘Stop’ button, which can be used to stop the analysis.

Furthermore, the **DEMO** video is provided in the resource CD, which would give insight on the run-time behavior of the UI application.

Performance Analysis

The real-time AMR prototype has been developed and the *Demo video* is attached in the resource CD. Author has witnessed enormous benefits by adding the real-time prediction feature. The main benefit is in terms of testing of our product. During our system testing, we have observed following key factors:

- Channel properties matter most
- Phase correction is necessary
- 100% bifurcation between PSK and QAM
- Reasonable better performance at high Tx & Rx gains

Channel properties matter most

It has been observed that NN models are sensitive to channel properties. As the variation in channel occurs i.e. different forms of Rayleigh and Rician fading, the behavior of models start producing anomalies. However, this has been sorted out by transfer learning of the model in the respective environment prior to its use.

Phase correction is necessary

The received I-Q stream always has phase error which rotates symbols in constellation diagrams. Because PSK modulation symbols can only be distinguished based on their absolute phase, this can seriously impair AMR performance. It has been observed that AMR works efficiently if the cognitive radios are phase locked and the proper phase correction is implemented on the receiving end.

100% bifurcation between PSK and QAM

During real-time AMR testing, 1D-CNN models showed 100% bifurcation between the PSK and QAM modulation schemes even in the worst scenarios i.e. no phase correction, worse SNR and limited symbols per model's input block.

Reasonable better performance at high Tx & Rx gains

1D-CNN models have demonstrated better performance at high gain settings of Tx and Rx. The "SBX 400-4400 MHz Rx/Tx (40 MHz)" daughter board of Ettus family is being used in the USRP to enable the transmission at ~3GHz carrier frequency. It provides the accessibility of tuning gain of Tx and Rx via GNU Radio application in real-time. The internal gain feature of SDRs is directly proportional to the SNR of the communication link. In order to simulate the variation in SNR in real time, a 22 dB to 28 dB window has been adopted, and the AMR with model#10a has shown excellent results at 28 dB gain settings.

Conclusion

This comprehensive document covers all aspects of the 4th milestone of the research project "Neural Network based Modulation and Channel Coding Identification for SATCOM Systems". Both sub-tasks given below of the 4th milestone are achieved.

- [x] ~~SDR based data transmission and reception.~~
- [x] ~~NN based MODCOD identification using SDR transmissions in a lab environment.~~

Demo has been submitted along with the report which demonstrates the real-time SDR-based modulation classification using the proposed NN model in the lab environment. Moreover, the approach in system design of AMR was modular from day one to ensure scalability in future. This SDR-based design now can be treated as universal to start research on any aspect related to AMR. For example:

- Adding more modulation types
- Model's performance analysis with SNR deterioration.
- Evaluate multiple type of NN models i.e. ResNet, inception, 2D CNN, LSTM, etc for AMR application.
- Crunch the On-Air datasets for AMR problems in different channel states.
- Apply enormous preprocessing techniques including the state-of-the-art feature extractions.

Future Work

The foundation stone has been installed in the progression of AMR related research with the successful accomplishment of this project. During the project, we have uncovered enormous research gaps and prominent one’s are listed below:

- QAM-64, OQPSK, and other more modulation schemes could be added to the dataset to increase the model's adaptability.
- The investigation of a novel, logical method for classification of coding schemes.
- The creation of modulation and coding schemes that are difficult for automatic modulation and coding recognition (AMR) systems to recognise.
- MODCOD recognition in OFDM signals to exploit the Wi-Fi, 3GPP, WiMAX, etc.
- Designing of End-to-End Neural Network model for MODCOD recognition application.
- Real-time identification of modulation schemes in the SATCOM systems.

References

1. Saad Iqbal and Syed Ali Hassan, "Literature Review & Feasibility Report of Neural Network based Modulation and Channel Coding Identification for SATCOM Systems", December, 2021.
1. Saad Iqbal and Syed Ali Hassan, "Proposed Neural Network for MODCOD report of Neural Network based Modulation and Channel Coding Identification for SATCOM Systems", May, 2022.
2. Saad Iqbal and Syed Ali Hassan, "Lab prototype of AMR for demonstration of proposed NN model report of Neural Network based Modulation and Channel Coding Identification for SATCOM Systems", August, 2022.