Improvements to Modulation Classification Techniques using Deep Learning

Ankush Jolly
Electrical Engineering
University of California, San Diego
ajolly@eng.ucsd.edu

Payam Khorramshahi
Electrical Engineering
University of California, San Diego
pkhorram@eng.ucsd.edu

Tabish Saeed
Electrical Engineering
University of California, San Diego
tasaeed@eng.ucsd.edu

***Abstract*—Modulation Classification of the received wireless signals serves as a useful tool in both military and civilian applications. The recent advancements in the field of Deep Learning (DL) has surged the interest of researchers in the wireless community to apply DL to the task of modulation classification. Recently in [1], a GNU Radio based generation of the dataset (RadioML2016.10a) was introduced which imitates the channel imperfections of a real wireless channel. In this work, we have investigated the performance of three DL models: Convolutional Neural Network (CNN), Residual Network (ResNet) and Convolutional Long Short-term Deep Neural Network (CLDNN). The maximum accuracy achieved for CNN, CLDNN and ResNet is 88.4%, 85.9% and 84.1% respectively. We have used RadioML2016.10a and RadioML2016.10b for training and testing our DL models. We conclude by comparing the performance of the implemented three models and propose future work for further research.**

I. Introduction

Signal modulation is a technique used in wireless communications that changes the signal characteristic before transmission. There are three properties of a carrier signal that can be modulated: amplitude, frequency and phase. The amplitude represents the intensity or the power of the signal, the frequency tells how often the signal is repeated while the phase tells the location of the signal waveform (in one cycle) with respect to time. There are two categories of modulation: analog and digital. In analog modulation, frequency (FM) and amplitude (AM) of the carrier signal is varied to transmit an analog signal. Digital modulation allows to transmit a digital signal (1s and 0s) by either varying the phase (PSK) or both the phase and amplitude of the analog carrier signal (QAM). The signal wave-forms in digital modulation can also be represented by the constellation diagrams which represent the complex data (IQ) samples of the signal. Fig 1 shows the different digital modulation constellation diagrams. It also represents the number of bits that can be transmitted in one symbol. For example, QPSK (Quadrature Phase Shift Keying) modulation transmits 2 bits and hence the signal waveform can be represented in 4 distinct ways.

In a typical communication scenario, both the transmitter (TX) and the receiver (RX) agree to a particular modulation scheme ahead of time. This implies that the RX knows the modulation scheme present in the received signal. Interestingly, there could be some use cases where the receiver doesn't know the modulation scheme of the received signal. Hence, the detection of modulation scheme without any apriori knowledge of received IQ samples is known as Automatic Modulation Classification (AMC). In military applications, if one hopes to recover the message from a piece of intercepted and possibly adversary communication signal, a modulation classifier is needed to determine the modulation type used by the transmitter. In civilian applications, it can be used in spectrum monitoring and surveillance. This can ensure efficient usage of the expensive spectrum.

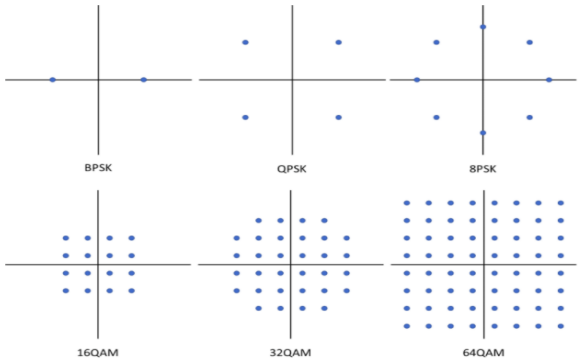

Fig. 1: Constellation Diagrams [2]

II. Related Work

The earliest works in AMC can be traced back to 1980s where hand-crafted features were created from raw temporal signals like zero crossing locations, square law classifiers, statistical moment classifiers. Hence these traditional methods mainly relied on likelihood based [3], feature based [4] and artificial neural network [5] based methods. These methods used to work only on specific modulation schemes and SNR levels. The recent advancements in the DL model architectures, ease of access to open source software libraries like PyTorch, TensorFlow, etc., advent of Graphics Processing Unit (GPUs) and Tensor Processing Units (TPUs) have made DL a lucrative tool to solve complex problems in a much faster time. This has caused the researchers in the wireless community to apply DL in the field of wireless communications. Machine Learning (ML) models are data driven models that use the information of the real-world conditions more efficiently that statistical models can't do. Hence, the recent works [6]–[11] have been applying DL to the task of modulation classification. Convolutional Neural Networks (CNNs) have been used in image classification and voice signal processing tasks. Due to its superior performance in feature extraction, CNNS were applied in [9] for modulation classification. However, CNNs have also been challenged with problems like vanishing gradients, accuracy degradation after certain network depth and over fitting. Residual networks (ResNet) and Convolutional Long Short-term Memory Networks (CLDNNs) have also been recently introduced to strengthen the feature propagation in neural networks. In this work, we have analysed and compared the performance of three models: Robust CNN [11], ResNet [7] and CLDNN [7]. Section III gives an overview of the data preprocessing, simulation setup and the implemented model architectures. Section IV presents the

results and discussion followed by conclusions and future work in Section V.

III. Method

A. Data Preprocessing

In this work, we implemented several experiments over two datasets. One of the dataset contains IQ samples from 11 modulation classes (8 digital and 3 analog) over 20 SNR values ranging from -20 dB to 18 dB (RADIOML 2016.10A). Another larger dataset (RADIOML 2016.10B) contains IQ samples for 10 modulation classes in the same range of SNR values. Each stream of IQ sample has the shape of $[2, 128]$. The total 11 modulation classes are as follows: ["8PSK", "AM-DSB", "AM-SSB", "BPSK", "CPFSK", "GFSK", "PAM4", "QAM16", "QAM64", "QPSK", "WBFM"]. The bigger dataset excludes the "AM-SSB" class. The smaller dataset contains 11,000 stream of IQ samples for each class whereas the bigger dataset contains 60,000 stream of IQ samples for each class. Both dataset are stored in a dictionary with keys representing tuples of (modulation class, SNR value) and values providing the corresponding IQ samples.

In the first step of data preprocessing, we rearranged the dataset into a dictionary composed of a single key representing the entire dataset. Initially, all labels (modulation classes) were digitized (a number was assigned to every individual class). Then, the labels were transformed into a one-hot encoding vector, $v_i \in \mathbb{R}^n$, where n is the total number of classes present in the dataset. In this vector, the element whose index value is equal to the class number is set to 1 while the rest of the elements are set to 0. The reason behind performing this transformation is to have a compatible input for the objective function which will be later discussed in detail. The new dictionary contains all data points from every class and SNR value where they are stored in a list of [one-hot encoded labels, IQ samples].

B. Models

CLDNN: CLDNN has been widely used in recognition tasks involving time domain signals like speech, images and video due to its inherent memory property that leads to recognizing temporal correlations in the input signal. This inherent property is because of the Long Short-term Memory (LSTM) layer in between the convolutional layers and the dense layers. Table I shows the architecture of the implemented model. The original architecture has been taken from [7] which uses a dropout rate of 0.6. Softmax is used at the output of the last dense layer to get the probability scores of the modulation classes. This is also applicable to the other two models that we have implemented. We modified the model by adding maxpooling layers after the convolutional layers and changed the dropout rate to 0.3.

ResNet: We implemented the residual network architecture [7] that has convolutional layers, dense layers and a skip connection. This skip connection, between input layer and third convolutional layer, helps in reducing gradient vanishing problem in this task. In order to improve model performance, we modify the model with upsample and maxpool layers. The purpose behind adding these additional layers was to reduce the training time for this model and also effectively increasing the accuracy achieved. A dropout of 0.6 was also added between the layers to avoid overfitting. The model design for the updated architecture is shown in Table. II.

Robust CNN: We implemented the architecture of a robust and fast CNN mentioned in [11]. The model design is shown in Table III. The model was initially trained over two different dropout rates. Then the model with the better performance was further modified and trained by adding Batch Normalization layer after each convolution layer.

TABLE I: CLDNN ARCHITECTURE

Layer	Output Dimension
Input	2 ×128
Conv1	2 ×128 × 256
MaxPool1	2 ×64 × 256
Dropout1	2 ×64 × 256
Conv2	2 ×64 × 256
MaxPool2	2 ×32 × 256
Dropout2	2 ×32 × 256
Conv3	2 ×32 × 80
MaxPool3	2 ×16 × 80
Dropout3	2 ×16 × 80
Conv4	2 ×16 × 80
MaxPool4	2 ×8 × 80
Dropout4	2 ×8 × 80
Reshape	2 ×640
LSTM	50
DropOut5	50
Dense1	128
DropOut6	128
Dense2	10
Trainable Par.	6,214,43

TABLE II: RESNET ARCHITECTURE

Layer	Output Dimension
Input	2 ×128
Conv1	2 ×128 × 256
Upsample1	2 ×256 × 256
Dropout1	2 ×256 × 256
Conv2	2 ×256 × 256
MaxPool2	2 ×128 × 256
Dropout2	2 ×128 × 256
Add1	2 ×128 × 256
Conv3	2 ×128 × 80
MaxPool3	2 ×64 × 80
Dropout3	2 ×64 × 80
Conv4	2 ×32 × 80
MaxPool4	2 ×32 × 80
Dropout4	2 ×32 × 80
Flatten	5120
Dense1	128
Dropout4	128
Dense2	11
Trainable Par.	1,132,203

C. Training

Before starting the training procedure, the data was divided into three separate sets. 90% of the data was assigned to the training set and the other 10% percent assigned to test set. 10% of training set was extracted as validation set to track the performance of model at every epoch to check for overfitting during the training. Once the model was completely trained, test data were used to evaluate the model performance regarding accuracy. Two of the most crucial components of the training procedure are the loss function and the optimizer. Since the problem addressed in this work is a classification task, we decided to use the cross-entropy loss function. Moreover, the ADAM optimizer was used to estimate the parameters of the model with a learning rate of 0.0001. All

TABLE III: ROBUST CNN ARCHITECTURE

Layer	Output Dimension
Input	2 ×128
Conv1	2 ×128 × 256
MaxPool1	2 ×64 × 256
Dropout1	2 ×64 × 256
Conv2	2 ×64 × 128
MaxPool2	2 ×32 × 128
Dropout2	2 ×32 × 128
Conv3	2 ×32 × 64
MaxPool3	2 ×16 × 64
Dropout3	2 ×16 × 64
Conv4	2 ×16 × 64
MaxPool4	2 ×8 × 64
Dropout4	2 ×8 × 64
Flatten	1024
Dense1	128
Dense2	11
Trainable Par.	6,595,94

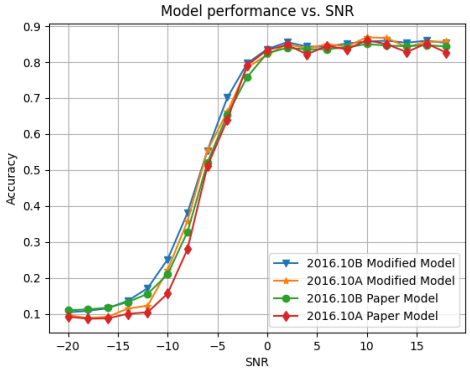

Fig. 2: CLDNN Performance

models were trained for 100 epochs with a batch size of 128. We used Keras Tensorflow for training our models.

IV. Results

For the CLDNN model, we tried out four experiments to understand the effect of the dataset and the modifications (maxpooling layers) that we made on the classifier's performance and training time. Fig. 2 shows the performance results of the classifier. We trained the original model specified in [7] using both the datasets. We modified the existing model by adding maxpooling layers after convolutional layers and changed the dropout rate from 0.6 to 0.3. This modified model was trained on both the datasets. The peak accuracy of 86.9% was achieved at SNR of 16 dB on modified model with bigger dataset. We did notice a better performance using a bigger dataset as the model tends to generalize better with a bigger dataset. Apart from that, adding maxpooling layer and changing the dropout rate not only improved the accuracy slightly but also resulted in a faster training time. For instance, for 50 epochs, the modified model's training time on bigger dataset was ~3 hours while it was ~3.6 hours for the modified model without maxpooling layers on the same dataset. Additionally, the modified model is performing better than the original model at low SNR values.

For ResNet, we trained the model for three different scenarios. Initially, we implemented the original architecture [7] and

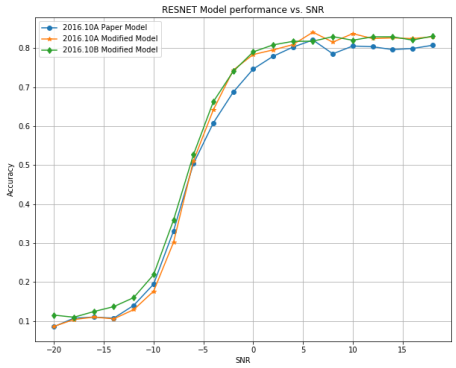

Fig. 3: ResNet Performance

trained it with the smaller dataset. After making modifications to the paper model as discussed in Section III, we trained it on both small and large datasets. In essence, these two models differ by maxpooling and upsampling layers; therefore, comparing these models provided us insight on the effect of Maxpooling on the neural network performance. The results for these three experiments are shown in the Fig. 3. Modified model trained on the large dataset works finest, followed by the modified model trained on small dataset. The original paper model gives us the least accuracy. These results reaffirm our intuition that training on larger dataset increases the accuracy. Since the larger dataset has 10 modulation classes compared to smaller dataset which has 11 modulation classes, classification on the larger dataset should be more effective and reliable as shown in 3. Moreover, our modifications to the original model increased accuracy since upsampling and maxpooling layers increase the network depth. Apart from this, the training times were effectively reduced due to the modifications. The original model took 3.5 hours to train on small dataset, whereas the modified model took almost 2 hours to train on the same dataset.

For Robust CNN model, the first experiment was to train the original model with two different dropout rates and the performance was evaluated. Fig. 5 shows that increasing the dropout rate from 0.3 to 0.5 would negatively impact the accuracy. However, higher dropout reduces the model complexity by removing more parameter's connections. Thus, the training process took less time (~2.5 hrs) with the rate of 0.5 compared with the rate of 0.3 which took ~3.7 hrs. In the second experiment we added a batch normalization layer after each convolution layer and kept the dropout rate at 0.3. This modification elevated the accuracy for all SNR values by approximately 3%. Batch Normalization layers kept the distribution for the output of each layer constant which led to faster convergence. The accuracy boost led by the Batch normalization layers shows that the original model could also have the tendency to reach this accuracy but with the cost of more computation time. In Figure 7, the robust CNN model shows lot of fluctuations in the loss. This means that the function has local minima. Thus, every time that gradient descent converges toward local minimum, the loss decreases. With a good learning rate, the model learned to jump between these points till reaching the global minimum. This is the reason for observing a lot of noise in the plot.

Fig. 6 shows the final performance plot for the three models

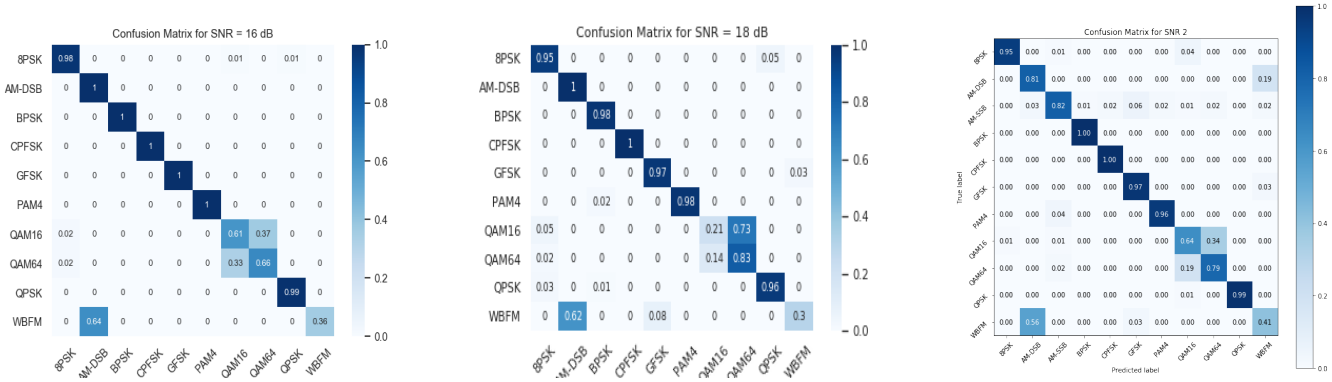

Fig. 4: Confusion Matrices for (a) CLDNN (b) ResNet (c) Robust CNN

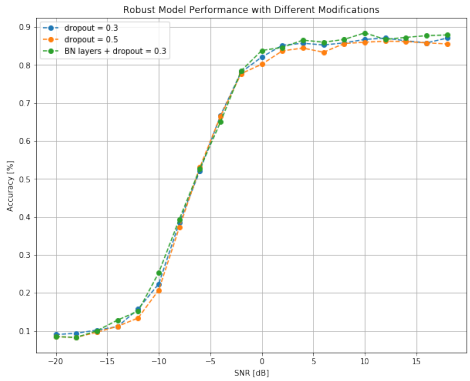

Fig. 5: Robust CNN Performance

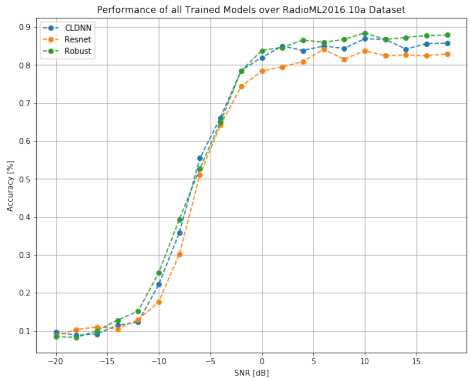

Fig. 6: Accuracy Performance Comparison

that we experimented with. To make the comparison fair, all the model results are of smaller dataset. Robust CNN performs the best amongst the three models with a peak accuracy of 88.4% while CLDNN and ResNet achieve a peak accuracy of 85.9% and 84.1% respectively. Fig. 7 depicts the validation loss of the three models. CLDNN achieves the least validation loss after 50 epochs. Fig. 4 shows the confusion matrices for the three models. Confusion matrices depict how accurately a predicted class aligns with its true class in the form of a probability score. For CLDNN, there are 10 modulation classes because the best performing model was the one trained on bigger dataset while there are 11 modulation classes for the best performing Robust CNN and ResNet. QAM16 and QAM64 are misclassified due to their similar constellation diagrams. Hence, even a small amount of noise makes the differentiation vulnerable to misclassification. WBFM and AM-DSB have a very high misclassification rate which needs to be further worked upon.

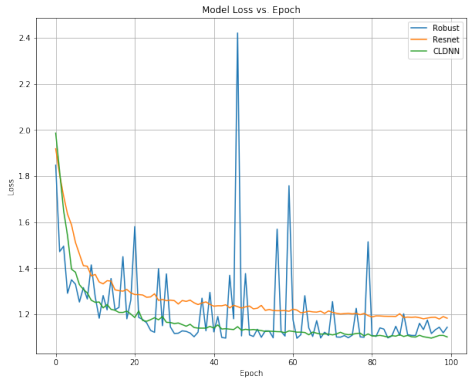

Fig. 7: Validation loss plots

V. Conclusions and Future Work

We applied three state-of-the-art deep neural networks for the radio modulation recognition task. Using various convolutional layers, residual layers and recurrent layers, we tried to extract different features from signals to classify modulations. The Robust CNN gives the best peak accuracy of classification. The ResNet and CLDNN models are also able to perform the classification task quite well, since there is only a slight difference in achieved classification accuracy. The reason behind this slight difference is the innate nature of the layers used in each architecture. The classification accuracy might also vary between these models due to their different depths. Nonetheless, all these models achieve satisfactory accuracies for classification task. In future, we plan to explore more models that are known to be successful in modulation recognition task and also try to further understand the effect of different layers and skip connections. Additionally, both the datasets use synthetic data generated from GNU Radio. We would like to explore our model's performance with an over-the-air (OTA) captured data [12], RadioML 2018.10A.

REFERENCES

[1] Timothy J. O'Shea and Nathan E. West. Radio machine learning dataset generation with gnu radio. 2016.

[2] https://www.nuwaves.com/constellation-diagrams.

[3] James A. Sills. Maximum-likelihood modulation classification for psk/qam. *MILCOM 1999. IEEE Military Communications. Conference Proceedings (Cat. No.99CH36341)*, 1:217–220 vol.1, 1999.

[4] S. S. Soliman and S. . Hsue. Signal classification using statistical moments. *IEEE Transactions on Communications*, 40(5):908–916, May 1992.

[5] A. K. Nandi and E. E. Azzouz. Modulation recognition using artificial neural networks. *Signal Process.*, 56(2):165–175, January 1997.

[6] T. J. O'Shea, T. Roy, and T. C. Clancy. Over-the-air deep learning based radio signal classification. *IEEE Journal of Selected Topics in Signal Processing*, 12(1):168–179, Feb 2018.

[7] X. Liu, D. Yang, and A. E. Gamal. Deep neural network architectures for modulation classification. In *2017 51st Asilomar Conference on Signals, Systems, and Computers*, pages 915–919, Oct 2017.

[8] Timothy J. O'Shea, Johnathan Corgan, and T. Charles Clancy. Convolutional radio modulation recognition networks. In Chrisina Jayne and Lazaros Iliadis, editors, *Engineering Applications of Neural Networks*, pages 213–226, Cham, 2016. Springer International Publishing.

[9] Tim O'Shea, Latha Pemula, Dhruv Batra, and T. Clancy. Radio transformer networks: Attention models for learning to synchronize in wireless systems. 05 2016.

[10] K. Karra, S. Kuzdeba, and J. Petersen. Modulation recognition using hierarchical deep neural networks. In *2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pages 1–3, March 2017.

[11] Kürşat Tekbiyik, Ali Ekti, Ali Görçin, Gunes Karabulut Kurt, and Cihat Keçeci. Robust and fast automatic modulation classification with cnn under multipath fading channels. 11 2019.

[12] https://www.deepsig.ai/datasets.

Response to critiques

Critical Review from Group 38:

Why didn't the group develop a single neural network model for all SNR data?
We did consider this part, we have rectified this mistake because we realize that measuring SNR before passing the signal to the classifier would be an overhead that we want to avoid. In fact, we have got really good results after making this change. Thanks for the suggestion.

There was a clear sign of overfitting in the Robust_CNN model as the training loss and validation loss diverged after around 20 epochs.
We have improved upon that part and there is some noise in the plot and we have explained our intuition behind that in the report.

We however, didn't notice a modification to the Resnet model. Maybe they could try to make improvements to the ResNet they developed.
We have made some modifications to the model for some improvements. We have mentioned that in the report.

Critical Review from Group 4:

You mention why using DL in the slides which is good. It's better to also explain more on why using DL in the wireless communication fields.
Because of time constraints, we couldn't. We have explained it in the report.

For ResNet, you can do some improvements on the hyper-parameters or adding some layers to make your own model.
We have tried doing that by adding some additional layers. It has been mentioned in the report.

The current three models are a bit old-fashioned. Maybe next time, you can try some update-to-date neural networks.
We may agree that they are a bit old fashioned but they have been well known to have a robust performance. In fact, CLDNN in modulation classification is relatively new. We may try GaNs in future work.

In the project you implement three models. However, it's better to explain the relationship among these three models.
We have done comparisons between the three models in the report by explaining the peculiar property of each model.

Critical Review from Group 63:

Seem like why DL became popular nowadays doesn't have to be explained.

- It could be better if why DL should be or is effective to be applied to Wireless Communications.

Given the time constraints of the presentation, we had to avoid that. We did mention why DL is becoming an important tool. We have mentioned that in the report.

The reason why these DL models are chosen is not explained.

In fact, we explained the benefit of the model briefly in the presentation. We couldn't delve deeper into it because of time constraints.

The comparison among the results would make work seem structured.

We have made a comparison plot of the three best models in the report.

The analysis could make the work better.

We have done a thorough analysis in the report.

ECE 228 - Group 76
Individual Contributions

GitHubLink:
https://github.com/pkhorram/Optimizing-Modulation-Classification-with-Deep-Learning

Ankush Jolly - A53304870

- Modelling and Optimization of CLDNN on (RadioML 2016.10a)
- Modelling and Optimization of CLDNN on (RadioML 2016.10b)
- Presentation and Report

Payam Khorramshahi - A12719367

- Data Pre-Processing
- Training Procedure
- Modelling and Optimization of Robust CNN model on (RadioML 2016.10a)
- Presentation and Report

Tabish Saeed - A53312389

- Modelling and Optimization of ResNet on (RadioML 2016.10a)
- Modelling and Optimization of ResNet on (RadioML 2016.10b)
- Presentation and Report