EECE 571 Course Project: Modulation Classification Using Neural Networks

Akshay Viswakumar (Student# 32971665),
Department of Electrical & Computer Engineering,
The University of Birtish Columbia

I. Introduction

THE goal of Automatic Modulation Classification (AMC) is to be able to infer the type of modulation technique by observing samples of received signals. This is a pattern recognition task and a good classifier would need to base its decision solely on key features extracted from the received signal and no other prior knowledge. A good classifier must also be robust and capable of making inferences from real world signals which are subject to degradation due to effects of channel fading and noise.

An intuitive choice of a solution to the problem of AMC would be to design a classifier that is trained using supervised learning techniques. This notion is intuitive because: (1) Supervised learning techniques continue to demonstrate time and again, their superiority over conventional algorithms and techniques for popular classification problems, given enough labeled training data. (2) At present, there is no dearth of labeled training data for this problem with plenty of datasets which may be sampled from actual over-the-air recordings or synthetically generated using tools like GNU Radio. The dataset that has been provided for this project, "RADIOML 2016.10A" is a synthetic dataset (with simulated effects of channel conditions) that was created and made available to the community by DEEPSIG [1].

The goal of this project was to design an Artificial Neural Network (ANN) solution and a Convolutional Neural Network (CNN) solution to the AMC problem. This report documents the work carried out in pursuit of the aforementioned goal. The remainder of this report is organised in the following manner. Section 2 is an analysis of the available dataset and a discussion of choices (common to both solutions) made to pre-process and segment the data. Section 3 and Section 4 deal with the ANN and CNN solutions respectively. Each of these sections start with a small background sub-section, a discussion of related work followed by design choices, observations and results. Finally, both solutions are compared in Section 5. The choice to include related work and background within the sections they are discussed in may be a little unconventional, but I believe this will greatly improve the way this report reads.

II. RADIOML Data

A. Description of Data

The dataset consists of 220,000 labelled signals. Signal samples are split and their real and imaginary components are stored separately. In this way, each signal is a 2x128 array representing 128 μs of a received waveform sampled at 10^6 samples/second. Each signal is labelled based on both the modulation technique as well as the signal-to-noise ratio (SNR) value. There are 11 different classes based on modulation techniques (8PSK, AM-DSB, AM-SSB, BPSK, CPFSK, GFSK, PAM4, QAM16, QAM64, QPSK, WBFM). There are 20 different classes based on the SNR (-20, -18, -16, -14, -12, -10, -8, -6, -4, -2, 0, 2, 4, 6, 8, 10, 12, 14, 16, 18).

B. Pre-Processing

The dataset is available as a pickle file that stores a python data structure in a serialized manner. Data from the pickle file were loaded into Numpy Arrays. The loaded label data consisted of binary strings which aren't otherwise meaningful to a neural network. The modulation labels were converted into on-hot encoded vectors where each vector was 1x11 and each bit represented one of the 11 modulation techniques.

Normalization is carried out on feature vectors before they are fed into the respective neural networks. This will be discussed in subsequent sections since the kind of feature vectors are different between the ANN and CNN networks.

C. Partitioning via Stratified Sampling

The available dataset was split in half to form training and testing datasets consisting of 110,000 samples each. The training set was then partitioned once again where 90% of the samples went on to form the actual Training set and 10% of the samples used to form a Validation set.

Each partition of the data set was formed in a way that there was adequate representation from all modulation classes. This was to make sure that the neural network could observe nearly the same number of samples from all possible classes and that there wouldn't be any over-representation or bias for one class or the other. Refer to Figure 1 which displays the histogram of the partitioned dataset based on Modulation Techniques. This sort of Stratified Sampling [2] was possible because the original dataset has been designed in order to ensure that there is equal representation from all classes.

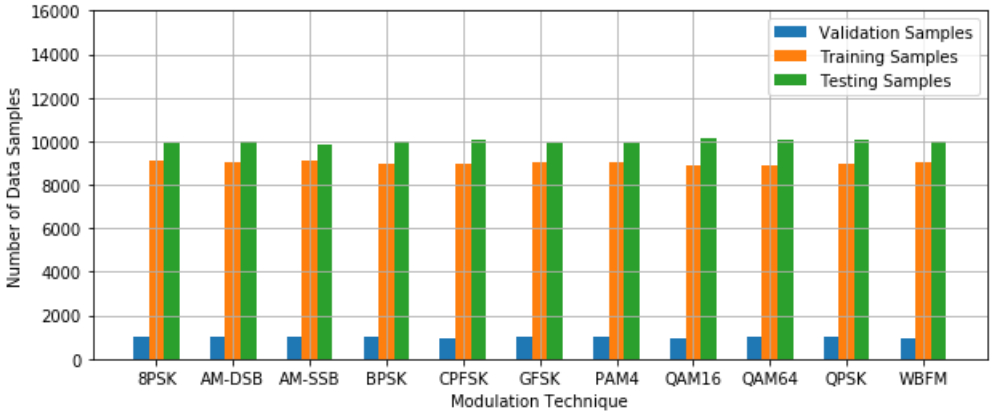


Fig. 1: Histogram of Partitioned Data, grouped by Modulation Technique

The Keras framework is able to reserve a portion of the training set for validation purposes at the time of training. However, the source of entropy for this is not within the user's control. Data was manually partitioned (using seeded pseudo-random number generators) in an effort to ensure that datasets stayed the same across different tests.

III. Artificial/Deep Neural Network (A/DNN) Solution

A. Background

Artificial Neural Networks (ANNs) are models inspired by the nervous system. Artificial neurons can get "activated" by certain inputs and an arrangement of such neurons connected together using weighted links can be used for classification or regression tasks. These networks can be trained by updating connection weights via feedback from outputs. The earliest neural networks only had a single layer which connected the input and output layers. Deep Neural Networks (DNNs) are ANNs with multiple layers in between the input and output layers. DNNs have proved to be useful in dealing with complicated problems that conventional ANNs could not otherwise solve. These days, the term DNN and ANN both refer to fully connected feed forward neural networks and in the context of this work they mean one and the same. These two terms may be used interchangeably but they refer to the same thing unless stated otherwise.

B. Related Work

DNNs are ubiquitous in regression and classification tasks. For the AMC task, certain features that can provide sufficient separation between modulation classes. The most commonly used features are cumulants. Cumulants are statistical quantities determined using the moments of a probability distribution. Identical distributions will therefore have identical cumulants. In [3], Lee et al conducted an analysis of several second-order, fourth-order and sixth-order cumulants in order to determine which combination of these cumulants would help with AMC for five modulation schemes. They demonstrate that C_{20}, C_{41},

C_{42} and C_{63} can help discriminate between PSK and QAM modulation schemes. These cumulants are some of the features that have been used in my DNN.

Narendar et al [4] suggest the use of a normalized version of the fourth order cumulant, C_{42}, such that $C_{42}^N = \frac{C_{42}}{(C_{21})^2}$. They are able to use this normalized C_{42} to achieve good classification results in their AMC system. It should be noted that while their approach to AMC does not make use of neural networks, it is still based on pattern recognition. Neural networks are all about identifying patterns and so, their idea will no doubt be useful in this project.

In [5], Kim et al propose a DNN based solution to AMC that makes use of 21 features extracted from the input signal. Unfortunately, their rationale behind choosing these features isn't clear. Most of these 21 features are based on distribution-based statistics and are either cumulants or other quantities derived from moments (Kurtosis and Skewness). The remaining features are derived from statistics related to characteristics of the input signal such as instantaneous amplitude, phase and power. I found 7 of these features: C_{40}, C_{63}, C_{80}, Kurtosis, Skewness, Peak-to-Rms Power Ratio and Peak-to-average Power Ratios to be useful and have made use of the same in my DNN implementation.

C. Feature Vector

The input to the DNN is a vector that made up of eleven features that are extracted from the input signal. These features have been selected based on their popularity in the AMC domain as well as for their ability to discriminate between the different modulation classes. These eleven features are defined below.

Let $a = a_I[n] + j a_Q[n]$ be an input signal, whose samples contain the in-phase and quadrature phase components, represented in rectangular notation.

Let $E[.]$ denote the Expectation operator

Let $M_{x,y} = E[(r[n])^{x-y}(r^*[n])^y]$ denote the $(x+y)^{th}$ Order Moment of $r[n]$

1) Cumulant, $\mid C_{20} \mid = \mid E\left[a^2(n)\right] \mid$

2) Cumulant, $\mid C_{21} \mid = \mid E\left[a(n)^2\right] \mid$

3) Cumulant, $\mid C_{40} \mid = \mid M_{40} - 3M_{20}^2 \mid$

4) Cumulant, $\mid C_{41} \mid = \mid M_{40} - 3M_{20}M_{21} \mid$

5) Cumulant, $\mid C_{42}^N orm \mid = \mid \frac{M_{42} - M_{20}^2 - 2M_{21}^2}{C_{21}^2} \mid$

6) Cumulant, $\mid C_{63} \mid = \mid M_{63} - 9M_{21}M_{42} + 12M_{21}^3 - 3M_{20}M_{42} - 3M_{22}M_{41} + 18M_{20}M_{21}M_{22} \mid$

7) Cumulant, $\mid C_{80} \mid = \mid M_{80} - 35M_{40}^2 - 28M_{60}M_{20} + 420M_{40} - 630M_{20} \mid$

8) Kurtosis, $K = \mid \frac{E[a - E[a]]^4}{E\left[(a-E[a])^2\right]^2} \mid$

9) Skewness, $S = \mid \frac{E[a - E[a]]^3}{E\left[(a-E[a])^2\right]^{\frac{3}{2}}} \mid$

10) Peak-to-RMS Power Ratio, $PR = \frac{max\left(|a|^2\right)}{\frac{1}{N}\sum_{i=1}^{N}(a[i])^2}$

11) Peak-to-average Power Ratio, $PA = \frac{max(|a|)}{\frac{1}{N}\sum_{i=1}^{N} a[i]}$

TABLE I: Architecture of DNN-AV

Layer	Layer Type	Description	Activation
1	Dense	4096 Units	ReLU
2	Dropout	Dropout Rate = 0.5	-
3	Dense	2048 Units	ReLu
4	Dropout	Dropout Rate = 0.5	-
5	Dense	1024 Units	ReLu
6	Dropout	Dropout Rate = 0.5	-
7	Dense (Output)	Units = 11	SoftMax

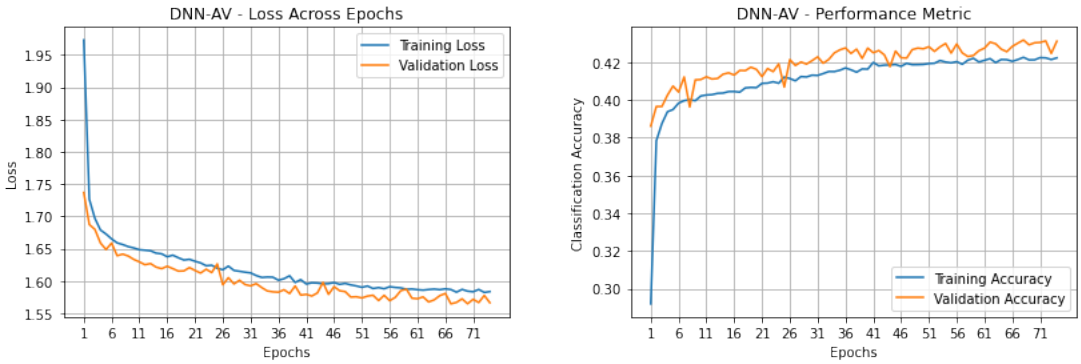
TABLE II: Other Design Choices

Parameter	Description
Loss Function	Categorical Cross-Entropy
Weight Optimizer	Adam
Activation Function (Hidden Layer)	ReLU
Activation Function (Output Layer)	SoftMax
Regularization	Dropout, Early Stopping
Weight Initialization	Xavier Initialization ('Glorot_Uniform')
Batch Size	1024

D. Network Design

A number of DNN architectures were constructed and trained. The performance of each was then evaluated based on their performance on the validation set. DNN-AV was the best performing network that had the lowest validation loss of **1.5647** (which occurred at the 71st Epoch). It also had the highest prediction accuracy of **43.02%** (average across all modulation classes). The loss and accuracy curves for DNN-AV have been displayed in Figure 2.

The architecture of DNN-AV has been detailed in TABLE I. Other design specifics have been detailed in TABLE II. The initial design was a much simpler two-layer network which was grown in stages. Increasing the number of neurons and layers did increase performance somewhat but extending the network either in width or length beyond the final design of DNN-AV tended to yield diminishing returns as the performance gain was not justified when considering in the amount of parameters to be computed. Larger networks are prone to overfitting and Dropout regularization as well as early stopping helped ensure that the network would not just memorize the training data.

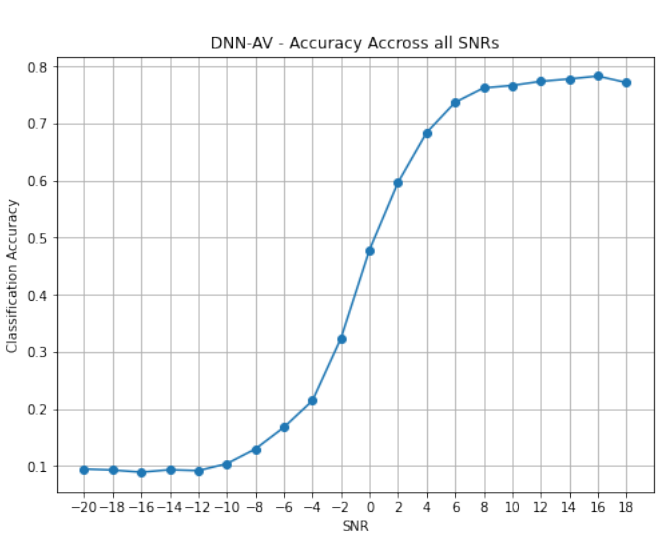


(a) Training Performance (b) Performance Metric

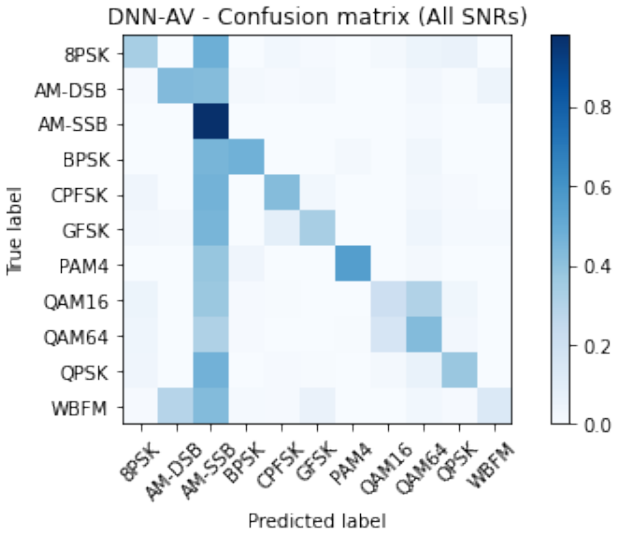
Fig. 2: CNN-AV - Accuracy and Loss During Training

E. Observations

Figure 3a displayes the average classification accuracy (across all modulation classes) that DNN-AV achieves at each SNR level. The corresponding confusion matrix is shown in Figure 3b. Figure 4 shows confusion matrices at different SNR values. There's not a lot of difference between the confusion matrices at -20 dB and -8 dB. The DNN does not perform well for input signals having a low SNR. This isn't all that surprising since the features extracted from a noisy signal will no doubt be contaminated. Observing the confusion matrix at 0 dB, we can start to see the diagonal form. At this stage, AM-SSB and PAM-4 have good classification accuracies. Note how the DNN is unable to distinguish between QAM-16 and QAM-64. At

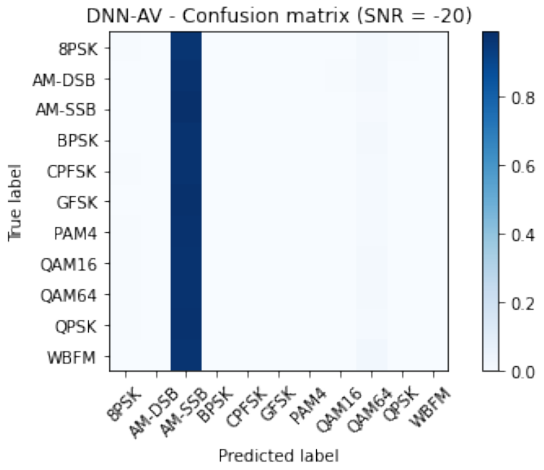


(a) Performance Metric

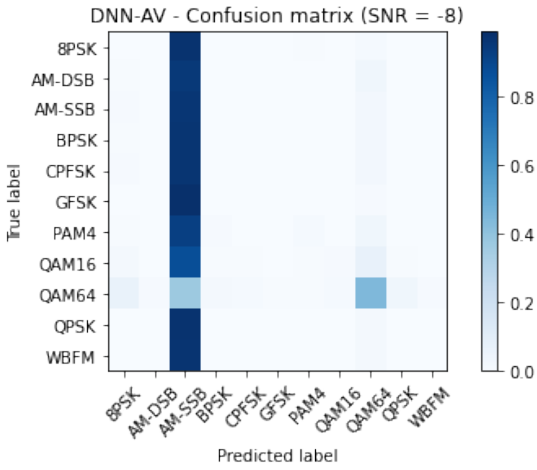


(b) Confusion Matrix

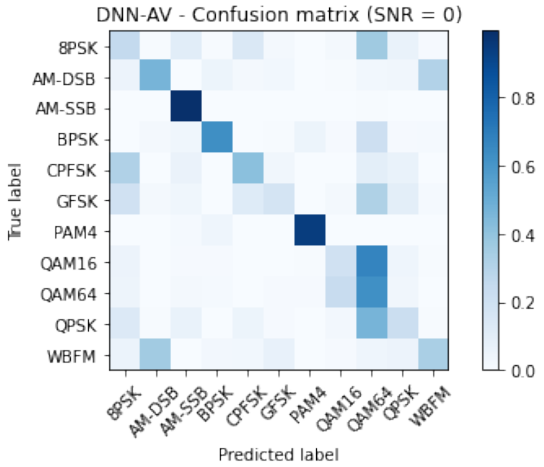
Fig. 3: Performance of DNN-AV on Test Data



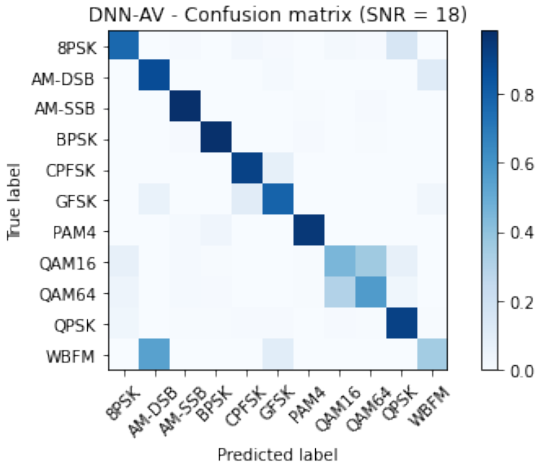
(a)



(b)



(c)



(d)

Fig. 4: DNN-AV Confusion Matrices at Different SNRs

higher SNRs (4 dB and above 3a), the DNN starts performing much better. At 18 dB we can observe a nearly clean diagonal. Pay attention to the confusion between the QAM-16 and QAM-64 classes is a little better.

IV. Convolutional Neural Network (CNN) Solution

A. Background

CNNs take inspiration from the mammalian visual system. In particular, the way how the brain processes visual information hierarchically often starting with simple structures such as oriented edges [6]. In the late 70s, Fukushima applied this idea to create the Neocognitron [7], a predecessor to CNNs, which was built to detect handwritten characters. The Neocognitron introduced the notion of a convolutional layer which was a spatially invariant filter that could be used to detect features irrespective of location in an image. The coefficients of the convolutional filter still had to be trained via some supervised learning algorithm. Fast forward to 1998, when LeCun et al [8] applied error backpropagation to train convolutional filter coefficients leading to the birth of CNNs as we now know them.

While CNNs, like UNet [9] may be built using convolutional layers alone, they often make use of fully connected neurons when performing classification. In a CNN designed for classification, the convolutional layers essentially perform feature extraction in a hierarchical manner starting with low-level features in the first convolutional layer.

B. Related Work

CNNs are commonly used in problems where the input data consist of images. However, they can be useful in any task where data has features that are spatially relevant such as in the case of text classification or speech recognition. Along the same lines, CNNs can be useful in AMC where the input data is made up of finite number of symbols that are unique for a given modulation scheme.

There are a lot of papers that present the solutions to the AMC problem. Each paper proposes one type of network design or the other and claims it to be best suited to the task. However, there's often nothing more than empirical evidence to back their claims. This sort of situation is also commonplace in other domains, computer vision for instance, where neural networks are rampant.

That said, a number of recent publications were reviewed in an effort to identify "best-practices" that have proven helpful for designing a CNN for AMC. O'Shea et al [10] achieve reasonably good performance on the RADIOML 2016.10A dataset with two CNN based networks. These networks are able to get over 10% accuracy for the lowest SNRs and above 80% accuracy for higher SNRs. Both networks consist of two convolutional layers and a fully connected layer (excluding the final classification layer). They differ only in that one design has more filters in the convolutional layers that the other. Both networks make use of the Adam optimizer as well as dropout for regularization. A lot of details are left to interpretation. Ramjee et al [11] make use of a deeper CNN design with four convolutional layers based on the more complex CNN proposed in [10]. While they report a higher accuracy at high SNRs (over 80% at 18dB), their network does not do well at the lower SNRs.

Lee and colleagues propose an interesting solution in [12] wherein they first extract a number of statistical features (such as higher order cumulants) which are then used to compose a "feature image" which is then fed into a CNN. Their claim is that feature images are distinct for a given modulation technique. The results they report indicate an improvement in the accuracies at high SNRs (nearly 100% over 4dB). The performance at lower SNRs are not very remarkable.

In [13], Wang and Yang highlight a problem that I have observed during my experiments. It is difficult for most CNNs that operate on the input signal in rectangular form to differentiate between QAM-16 and QAM-64 modulation even at the highest SNR. The authors use an additional CNN used exclusively to differentiate between QAM-16 and QAM-64. This secondary network uses constellation diagrams as input.

C. Network Design

Several CNN designs were constructed and trained using the same training set. These designs were then compared based their performances on the same validation set. CNN-AV was the best performing network with the lowest validation loss of **1.2095** (which occurred at the 45th Epoch). It also achieved the highest Average Classification Accuracy (across all modulation

TABLE III: Architecture of CNN-AV

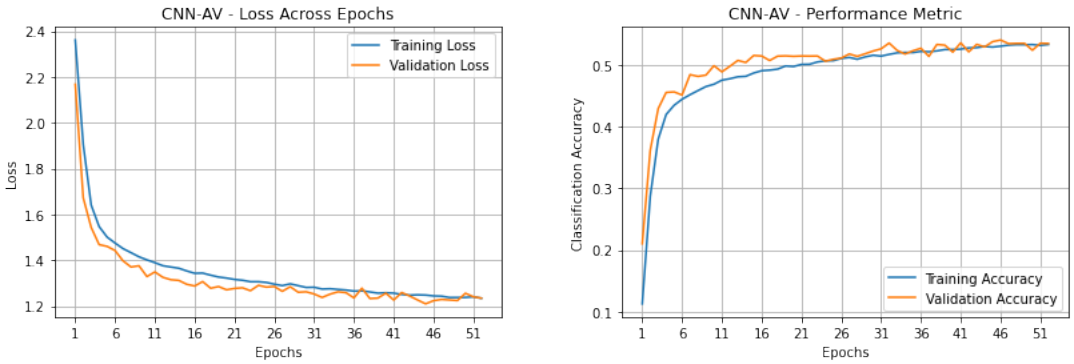
Layer	Layer Type	Description	Activation
1	Convolution	512 Filters, Size = (1,3), Stride = (1,1)	ReLU
2	MaxPooling	Window = (1,2), Stride = (1,2)	-
3	Dropout	Dropout Rate = 0.5	-
4	Convolution	256 Filters, Size = (1,3), Stride = (1,1)	ReLu
5	MaxPooling	Window = (1,2), Stride = (1,2)	-
6	Dropout	Dropout Rate = 0.5	-
7	Convolution	128 Filters, Size = (1,3), Stride = (1,1)	ReLu
8	MaxPooling	Window = (1,2), Stride = (1,2)	-
9	Dropout	Dropout Rate = 0.5	-
10	Convolution	64 Filters, Size = (1,3), Stride = (1,1)	ReLu
11	MaxPooling	Window = (1,2), Stride = (1,2)	-
12	Dropout	Dropout Rate = 0.5	-
13	Dense	Units = 128	ReLu
14	Dense (Output)	Units = 11	SoftMax

TABLE IV: Other Design Choices

Parameter	Description
Loss Function	Categorical Cross-Entropy
Weight Optimizer	Adam
Activation Function (Hidden Layer)	ReLU
Activation Function (Output Layer)	SoftMax
Regularization	Dropout, Early Stopping
Weight Initialization	Xavier Initialization ('Glorot_Uniform')
Batch Size	1000

techniques) of **53.81%**. The loss and accuracies across the number of training Epochs have been plotted in Figure 5. The architecture of CNN-AV has been detailed in TABLE III. Other design choices for CNN-AV have been detailed in TABLE IV.

Like in the case of the DNN, the design process for the CNN based network was highly empirical. The existence of Pooling layers significantly increased the number of hyper parameters that could be tuned. While the final design for CNN-AV is large, training was still manageable on account of these pooling layers. They helped reduce the dimensionality of outputs from a layer meaning lesser connections and trainable weights to subsequent layers.

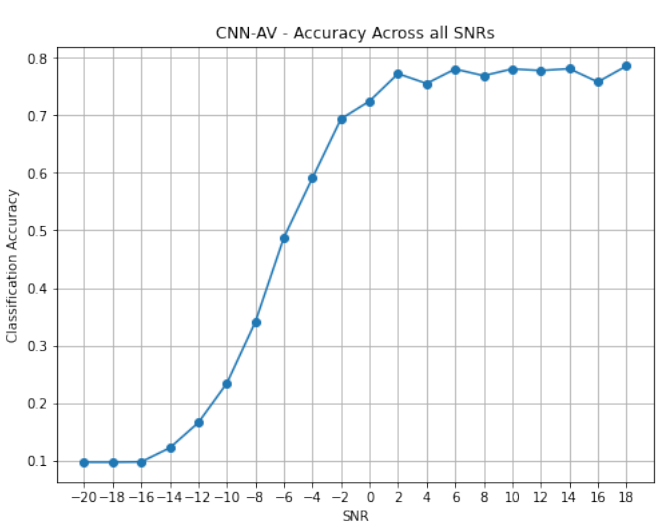


(a) Training Performance (b) Performance Metric

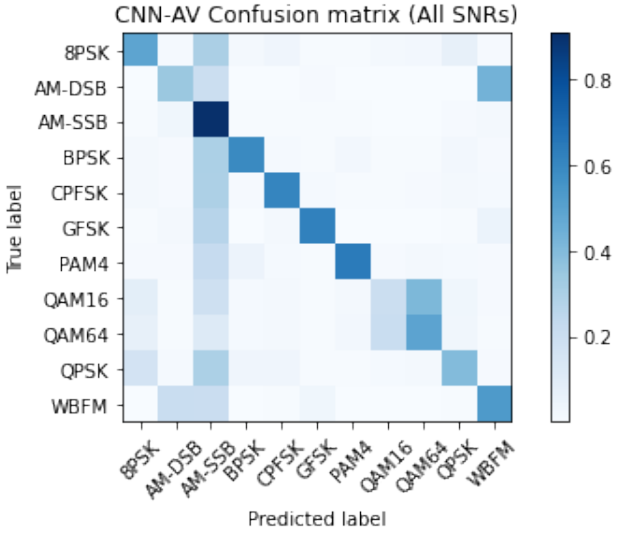
Fig. 5: CNN-AV - Accuracy and Loss During Training

D. Observations

Figure 6a displays the average classification accuracy (across all modulation classes) that CNN-AV achieves at each SNR level. The corresponding confusion matrix is shown in Figure 6. Figure 5 shows confusion matrix plot for specific SNR values. At -20 dB, CNN-AV has poor performance with all inputs signals classified as AM-SSB. The performance gets better as SNR increases. At SNRs greater than 0 dB, the performance improvement is minimal as seen from Figure 7 (c) and (d). While there is a slight improvement, QAM-16 signals are still misclassified as QAM-64. These two modulation schemes are essentially the same and differ only by the number of bits used to encode symbols. It appears that the CNN is unable to discern between the two just based on the wave envelope.

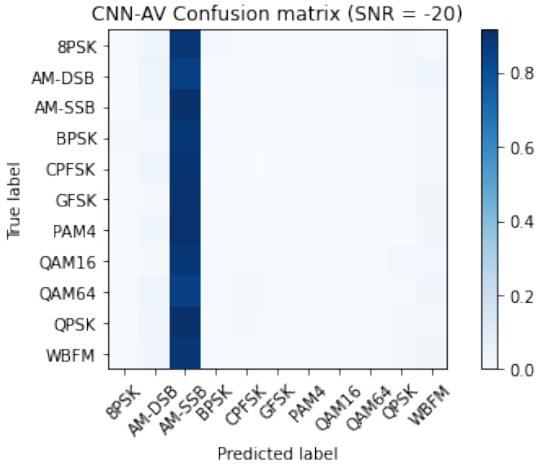


(a) Performance Metric

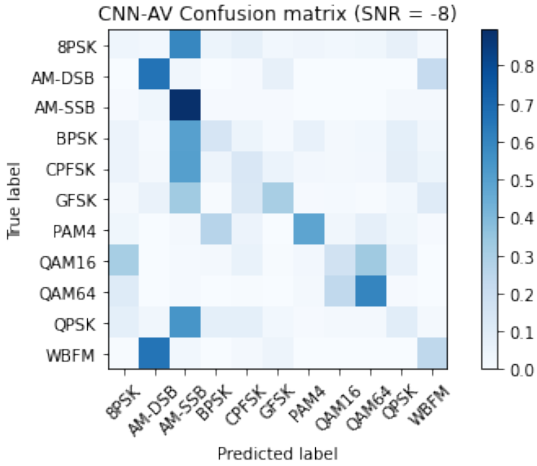


(b) Confusion Matrix

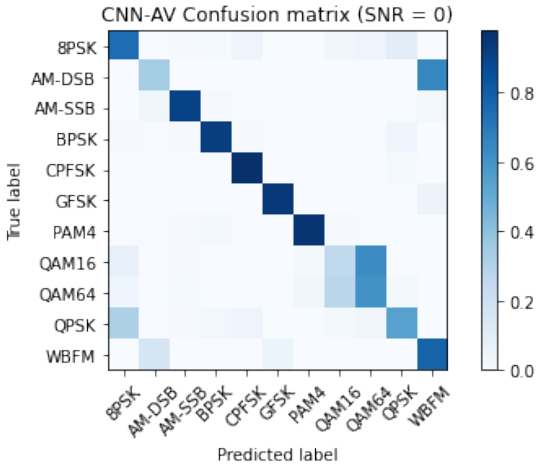
Fig. 6: Performance of CNN-AV on Test Data



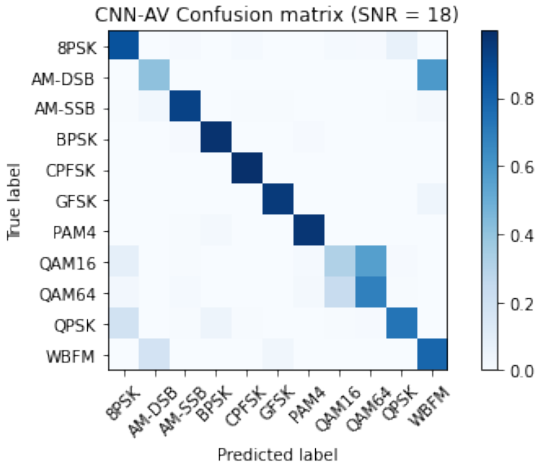
(a)



(b)



(c)



(d)

Fig. 7: CNN-AV Confusion Matrices at Different SNRs

V. Discussion

The average prediction accuracy across modulation classes has been the performance metric used in this project. The performance metric across the training, testing and validation sets have been reported in the TABLE V. Both networks perform nearly identically when the input signal has a high SNR. For inputs signals at lower SNR levels, the CNN based architecture performs relatively better as it is able to reasonably distinguish between modulation classes at SNR values as low as -4 dB (60% accuracy). The difference in performance at low SNRs is most likely due to the nature of how each architecture fundamentally functions.

TABLE V: Average Prediction Accuracy Across All Modulation Classes (Performance Metric)

	Network	
	DNN-AV	CNN-AV
Training Set	42.23%	53.07%
Validation Set	43.02%	54.07%
Test Set	42.64%	53.04%

The CNN processes the input signal by looking at signal amplitudes. This is more or less equivalent to observing the signal waveform. Convolutional layers are then trained such that they activate when specific features are identified. It does seem then that even at lower SNRs the CNN is still able to discern some of these features unique to different modulation classes.

In case of the DNN, feature extraction is a separate process and the network sees only what it is fed as input. Unfortunately, the onus is on the designer to ensure that the right features are passed to the network. Without a sufficient understanding of the problem, incorrect features fed into the network will not allow the network to learn. It does seem then that the features I chose as input to DNN-AV were easily affected by noise. Perhaps identifying and extracting other features which are more resilient to noise can help improve the performance.

Another point of note relates to the ambiguity between QAM-16 and QAM-64. While CNN-AV could not make the distinction even at the highest SNR levels, DNN-AV performed somewhat better at 18 dB (refer to Figure 4 (d)).

VI. Conclusion

In conclusion, both DNNs and CNNs are viable approaches when attempting to tackle the task of Automatic Modulation Classification. While CNNs are great at extracting features from input, they come at a cost of increased computation. DNNs on the other hand are much simpler but are only efficient if they're given the right kind of inputs.

References

[1] T. O'Shea and N. West, "Radio machine learning dataset generation with gnu radio," *Proceedings of the GNU Radio Conference*, vol. 1, no. 1, 2016. [Online]. Available: https://pubs.gnuradio.org/index.php/grcon/article/view/11

[2] "Encyclopedia of survey research methods au - lavrakas, paul," Thousand Oaks, California, 2008. [Online]. Available: https://methods.sagepub.com/reference/encyclopedia-of-survey-research-methods

[3] S. H. Lee, K.-Y. Kim, and Y. Shin, "Effective feature selection method for deep learning-based automatic modulation classification scheme using higher-order statistics," *Applied Sciences*, vol. 10, no. 2, p. 588, Jan 2020. [Online]. Available: http://dx.doi.org/10.3390/app10020588

[4] M. Narendar, A. P. Vinod, A. S. M. Kumar, and A. K. Krishna, "Automatic modulation classification for cognitive radios using cumulants based on fractional lower order statistics," in *2011 XXXth URSI General Assembly and Scientific Symposium*, 2011, pp. 1–4.

[5] B. Kim, J. Kim, H. Chae, D. Yoon, and J. W. Choi, "Deep neural network-based automatic modulation classification technique," in *2016 International Conference on Information and Communication Technology Convergence (ICTC)*, 2016, pp. 579–582.

[6] D. H. HUBEL and T. N. WIESEL, "Receptive fields of single neurones in the cat's striate cortex," *The Journal of physiology*, vol. 148, no. 3, pp. 574–591, Oct 1959, 14403679[pmid]. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/14403679

[7] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980. [Online]. Available: https://doi.org/10.1007/BF00344251

[8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[9] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[10] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," 2016.

[11] S. Ramjee, S. Ju, D. Yang, X. Liu, A. E. Gamal, and Y. C. Eldar, "Fast deep learning for automatic modulation classification," 2019.

[12] J. H. Lee, K. Kim, and Y. Shin, "Feature image-based automatic modulation classification method using cnn algorithm," in *2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, 2019, pp. 1–4.

[13] Y. Wang, M. Liu, J. Yang, and G. Gui, "Data-driven deep learning for automatic modulation recognition in cognitive radios," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 4074–4077, 2019.