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 and SNR 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 and Figure 2 which display the histograms of the partitioned dataset based on SNR and Modulation Techniques. This sort of Stratified Sampling [2] was possible because the original dataset has been designed in such a way that there is equal representation from all classes.

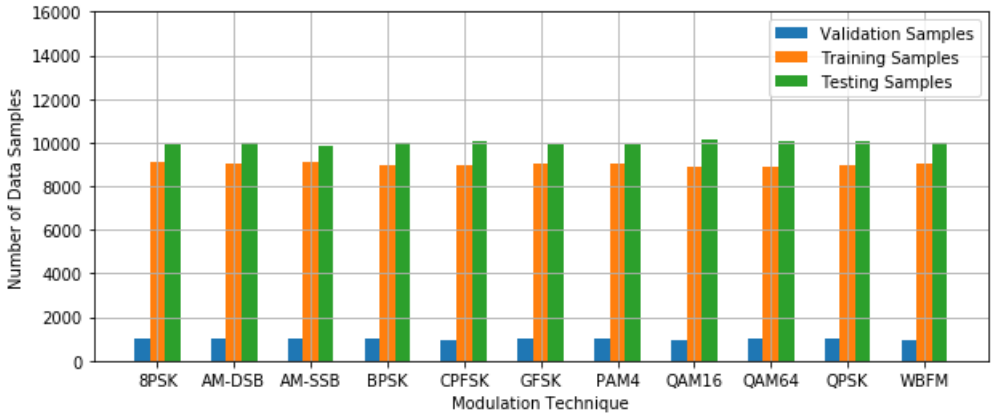


Fig. 1: Histogram of Partitioned Data, grouped by Modulation Technique

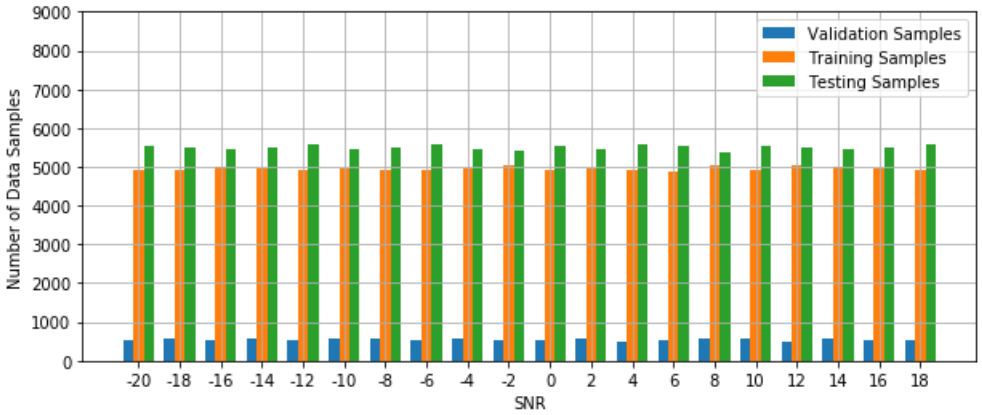


Fig. 2: Histogram of Partitioned Data, grouped by SNR

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 Neural Network (ANN) Solution

A. Background

B. Related Work

C. Network Design

D. Observations

E. Results

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 [3]. In the late 70s, Fukushima applied this idea to create the Neocognitron [4], 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 [5] applied error backpropagation to train convolutional filter coefficients leading to the birth of CNNs as we now know them.

While CNNs, like UNet [6] 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 initial convolutional layers essentially perform feature extraction.

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 [7] 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 [8] make use of a deeper CNN design with four convolutional layers based on the more complex CNN proposed in [7]. 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 [9] 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 [10], 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

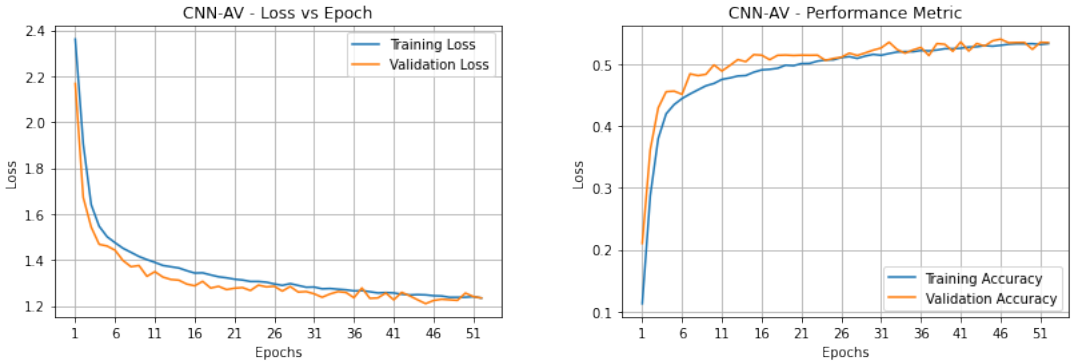
TABLE I: Design of CNN-AV

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	Units = 11	SoftMax

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

D. Observations



(a) Training Performance (b) Performance Metric

Fig. 3: CNN-AV - Accuracy and Loss During Training

Testing

CONF MATRIX

E. Results

V. Conclusion

References

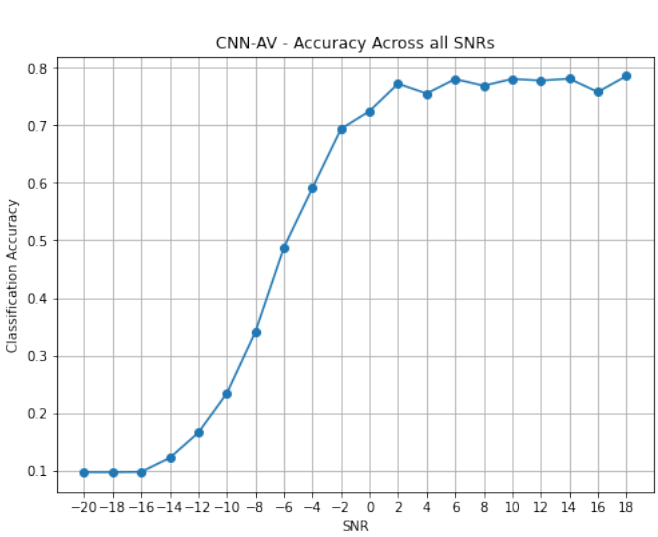
[1] T. O'Shea and N. West, "Radio machine learning dataset generation with gnu radio," *Proceedings of the GNU Radio Conference*, vol. 1, no. 1, 2016. [Online]. Available: https://pubs.gnuradio.org/index.php/grcon/article/view/11

[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

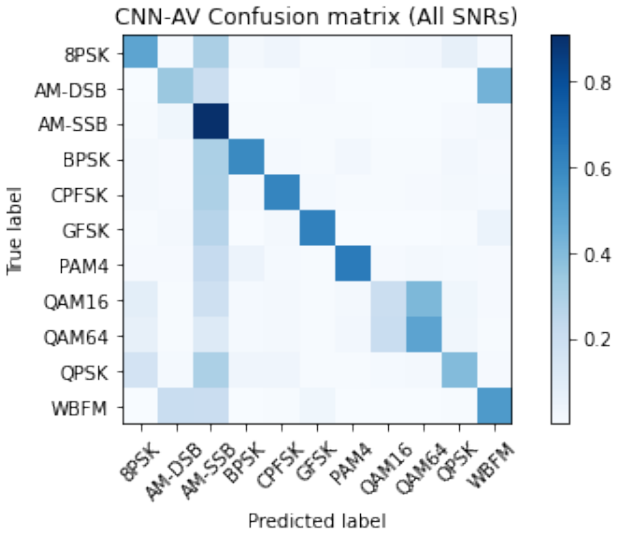
[3] D. H. HUBEL and T. N. WIESEL, "Receptive fields of single neurones in the cat's striate cortex," *The Journal of physiology*, vol. 148, no. 3, pp. 574–591, Oct 1959, 14403679[pmid]. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/14403679

[4] 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

[5] 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.

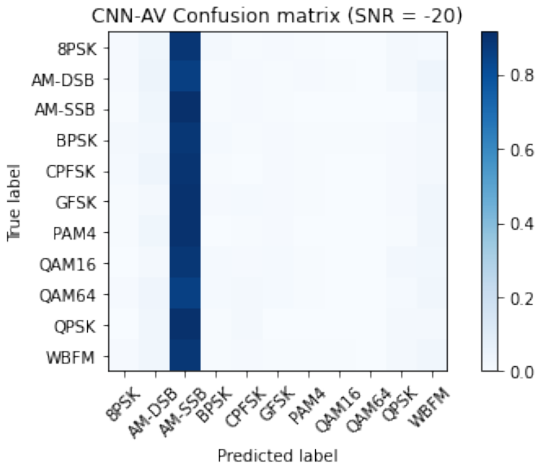


(a) Performance Metric

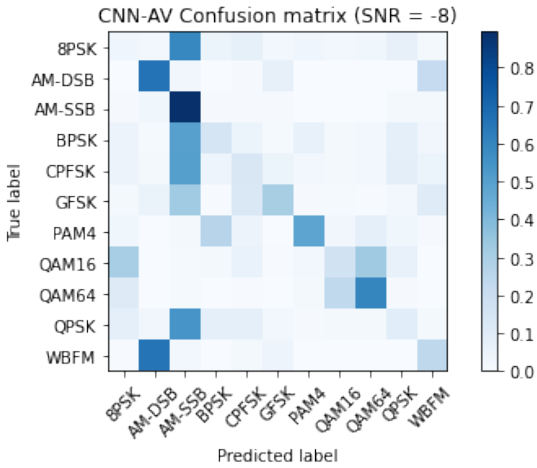


(b) Confusion Matrix

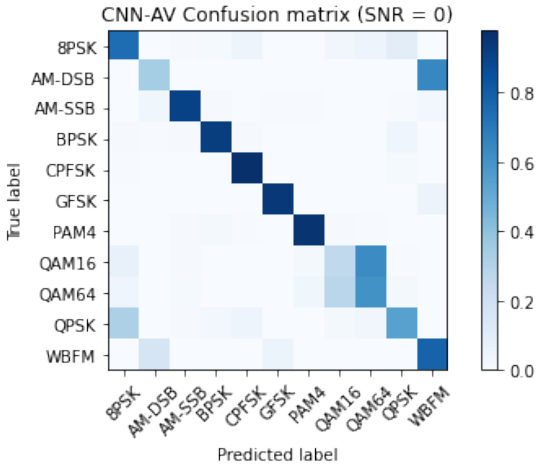
Fig. 4: Performance of CNN-AV on Test Data



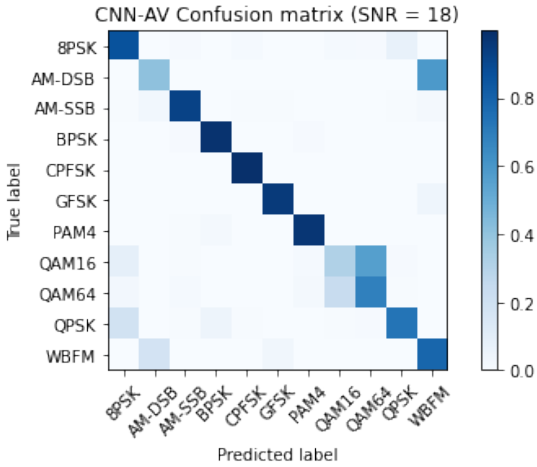
(a)



(b)



(c)



(d)

Fig. 5: CNN-AV Confusion Matrix at Different SNRs

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.
[7] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," 2016.
[8] S. Ramjee, S. Ju, D. Yang, X. Liu, A. E. Gamal, and Y. C. Eldar, "Fast deep learning for automatic modulation classification," 2019.
[9] 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.
[10] 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.