MOD3NN: A Framework for Automatic Signal Modulation Detection Using 3D CNN

Vishal Perekadan
University of Alabama
Huntsville, AL
vp0023@uah.edu

Chaity Banerjee
University of Alabama
Huntsville, AL
cm0282@uah.edu

Tathagata Mukherjee
University of Alabama
Huntsville, AL
tm0130@uah.edu

Eduardo Pasiliao
Air Force Research Labs
Shalimar, FL
eduardo.pasiliao@us.af.mil

Hovannes Kulhandjian
California State University
Fresno, CA
hkulhandjian@csufresno.edu

Michel Kulhandjian
Rice University
Houston, TX
michel.kulhandjian@rice.edu

Abstract

In this work, we present an application of a three-dimensional convolutional neural network for the task of automatic modulation recognition from raw I/Q signal data. Raw I/Q signal data exhibits a special "helical" structure that can be exploited with three-dimensional convolutions (3D convolutions) to learn spatio-temporal features from the signal for the problem of modulation recognition. By tweaking the convolutional filters to learn the helical symmetry of the data, we can design a shallow network for automatic modulation recognition (AMR). We present the results of our experiments with raw I/Q signal data collected in an uncalibrated radio frequency (RF) environment using several different modulation schemes. We show that with our methods and implementation, we can achieve around 99% accuracy for automatic modulation recognition, for a variety of practical modulation techniques without the need for explicit feature engineering.

Introduction

Large-scale autonomous deployment of communication systems has become commonplace with the proliferation of 4G/5G cellular networks, the wide availability of Internet-of-Things (IoT) devices, and the advent of low-cost commercial off-the-shelf software-defined radios. The hardware underlying such devices is not standardized, nor is the software stack deployed on such hardware. This has resulted in a need for heterogeneous systems to coexist and communicate in an increasingly congested radio frequency (RF) environment (Weber and Weber 2010). The ubiquitous use of large-scale peer-to-peer communication and the nature of the underlying networks and systems brings forth the challenge of accurately identifying and adapting to the communication protocols being used. An important part of any such protocol is the modulation scheme used for modulating the data onto the signal. The modulation scheme used for data transfer can vary from one device to the other in a heterogeneous communication network. Hence automatic recognition of the modulation scheme is important for any autonomous deployment of large-scale communication networks involving heterogeneous devices and protocols.

Modulation detection is a receiver-end function and forms an important step between signal reception and demodulation. Typically, in the case of peer-to-peer communication between homogeneous systems, the transmitter, and the receiver agree on a modulation scheme. This information is then used at the receiver for demodulation. However, in the case of large-scale heterogeneous wireless networks, it is not possible for the communicating devices to have an agreed-upon modulation scheme because of the nature of the underlying applications. For example, in the case of large-scale wireless sensor networks (WSN), sensor nodes might have nonuniform hardware platforms and support heterogeneous communication protocols. However, they might still need to be able to communicate with each other. Being able to automatically detect the modulation scheme used by a received signal and then adjusting one's parameters accordingly allows for seamless communications across such heterogeneous platforms. Various methods for automatically detecting the modulation from received signal data are available (Digulescu et al. 2013; O'Shea, Corgan, and Clancy 2016). Early methods for this task used intricate knowledge of the signal space to devise features that can be used for inferring the modulation scheme using classical machine learning techniques. For example in (Hsue and Soliman 1990), the authors use a zero-crossing sampler with constant envelope signals to extract features that represent signal parameters useful for identifying the modulation method. These features are finally used with a classifier to detect the modulation. Such methods, due to their reliance on expert-generated features do not scale to large heterogeneous networks. Thus, though methods based on expert-engineered features work well for small homogeneous networks, there has been a pivot towards automatic feature learning for large scale heterogeneous networks due to solubility and scaling issues for systems designed with hand engineered features.

Automatic feature learning systems for modulation detection (O'Shea, Corgan, and Clancy 2016) are mostly implemented using deep neural networks (LeCun, Bengio, and Hinton 2015). These methods are usually end-to-end and can scale for large communication networks due to their inherent ability to learn discriminative features automatically from the raw input signal data. However, many of these methods, though promising, have been shown to work only with simulated signal data. This makes them unsuitable for use in real-world environments where the underly-

Copyright © 2023 by the authors. All rights reserved.

ing RF characteristics are unknown and often beyond our control. Finally, many of these "end-to-end" methods treat the time-varying signal as a time series and use recurrent neural networks with either gated recurrent units (GRU) or long-short-term-memory (LSTM) (Dey and Salemt 2017; Hochreiter and Schmidhuber 1997) for inference. However, these structures are known to be computationally complex and are sometimes hard to train and optimize as they do not converge easily due to problems with vanishing/exploding gradients. This in turn makes such methods impractical for large-scale deployments in uncharacterized heterogeneous RF environments. In this work, we address these problems by designing and implementing an "end-to-end" system that implicitly exploits the inherent "helical" symmetries of the raw I/Q data for inferring the underlying modulation scheme. In our framework, a three-dimensional (3D) convolutional neural network (CNN), in short a 3D CNN, is used for processing the time series I/Q data. A 3D CNN uses three-dimensional convolutional filters as opposed to the two-dimensional ones used in standard convolutional neural networks (CNNs). The 3D convolutional filters are used to learn the spatio-temporal properties of the raw I/Q time series data. Succinctly, they are used to implicitly learn the helical symmetries from the raw signal without the need for using a recurrent structure. Our implementation is simple to train and can be used in uncharacterized RF environments. We show the efficacy of our implementation with over-the-air signal data from an uncharacterized RF environment using five different modulation schemes namely, BPSK, QPSK, 8PSK, 16QAM and 64QAM.

Our contributions in this paper can be summarized as follows: (1) Design and implement a shallow 3D convolutional neural network for the task of learning spatio-temporal features of a signal for modulation detection. Being shallow, the network can be used for almost real-time inference in a communication pipeline; (2) We demonstrate the efficacy of the system through the results of experiments with over-the-air signal data collected in uncharacterized RF environments using software-defined radios. Going forward, in Section we describe the details of the proposed framework followed by where we provide a description of the collected datasets, experiments, and results. We conclude this work in Section .

Proposed Approach

Time series data like raw I/Q signal information, is in general processed with Recurrent Neural Networks (RNN). However CNN architectures are easier to train as compared to RNN architectures, which are notoriously hard to train and optimize as pointed out before. As a result, the use of 3D CNN-based networks provide an efficient way to learn intrinsic signal characteristics for end-to-end modulation identification using raw I/Q signal data. In order to use 3-dimensional convolutions, the input to the network should be "volumetric data" and not standard 2D images. In general, a single I/Q data point is represented as a tuple (I, Q). It must be noted that a receiver "reads" M I/Q samples at a time from the transmitter, and one of the simplest ways of representing this data is in the form of a vector of M I/Q tuples. In this section, we describe how we convert the 2D I/Q data to 3D, to be used as input into a 3D CNN. Going forward let us represent the M I/Q samples corresponding to a time stamp t as a vector $X_t = [I_1, Q_1, \ldots, I_M, Q_M]^t$. Note that the I/Q time series data can then be represented as $\{X_t\}; t = 1, 2, \ldots$. Next, we describe the creation of the 3D volume data for use with the 3D CNN. Our approach involves creating an I/Q image from the raw I/Q data and then stacking them together to create the volume. Note that creating an I/Q image is lossy as it involves binning the I/Q values. However, since the modulation imposes a repeating pattern on the signal, the loss thus incurred should not affect the extraction of discriminative features to differentiate between the various modulation schemes. In fact, our experimental results support this statement as we point out later.

The 3D volume is obtained by "stacking" up δ 2D I/Q "images," where each 2D I/Q image corresponds to a time stamp t and is obtained from X_t as follows: given X_t, the I/Q data corresponding to a time stamp t, we first find the range of the I and the Q values. Let the maximum and minimum values corresponding to the I and Q components be (I_{min}, I_{max}) and (Q_{min}, Q_{max}), respectively. Let us suppose that we want to create an image of size $k \times k$. In order to do this, we divide the range of the I values namely $I_{max} - I_{min}$ into k "bins" and do the same for the range of the Q values, thus obtaining a total of $k \times k$ "bins" between the I and Q values. Finally, we assign each member of X_t into one of these "bins" based on the value of its I component and Q component. Let us denote the image created from X_t by $\mathcal{I}_t$. In order to create the volumetric representation of the I/Q data, we stack δ of these I/Q images together and use them as input to the 3D CNN. Thus, the volume corresponding to time window $(t, t + \delta)$, denoted by $V(t, t + \delta)$ can be represented as $[\mathcal{I}_t | \ldots | \mathcal{I}_{t+\delta}]$, where "|" denotes the stacking operation along the temporal axis. Note that $V(t, t+\delta)$ of dimension $\mathbb{R}^{k \times k \times \delta}$ is the input to our 3D CNN. Figure 1 portrays the I/Q volume corresponding to a given time window, for data generated using 16-QAM. Next, we briefly discuss the idea of 3D convolutions and describe their usage in the context of our network.

3D CNNs are used to extract features from volumetric data in order to establish a relationship between the three input dimensions of the dataset. The primary difference between a standard 2D convolution and a 3D one is that in 3D convolutions the kernel being convolved can move in three directions, whereas in 2D convolutions it can only move along two directions. This causes the network to better learn the dependencies within the three directions in case of volumetric data and a difference in output dimensions post-convolution. It must be noted that the kernel for the convolution will only move in three dimensions if its depth is less than the depth of the input feature map. If the kernel cannot move in three dimensions then the resulting convolution will not be three-dimensional. In this case, the dependence between the three dimensions will not be captured by the convolution filter, which will only capture the features of the underlying 2D subspace. Based on this, the window size δ in our case must be at least 3, otherwise, the resulting convolution will not be able to capture the spatio-temporal

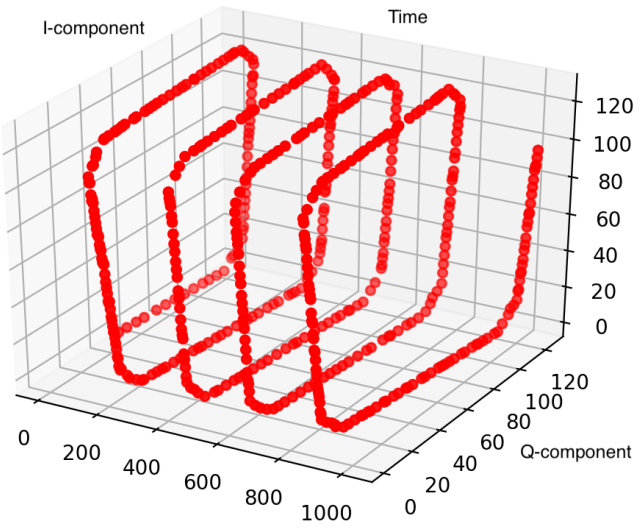

Figure 1: 3D Volume from I/Q Data. Note the Helical Structure of the Data. The Short Term Spatio-Temporal Dependencies Exhibited Here Characterize the Transmitter.

features as desired. Furthermore, note that δ is a parameter in our system that can be optimized. Apart from δ, which is an intrinsic parameter of the volumetric input, there is another parameter that controls what is learned by the CNN. This parameter, denoted by β, is the number of 2D images that overlap between two consecutive volumes that are used as input to the 3D CNN. Note that this is not a parameter that characterizes the 3D stack corresponding to a time interval but rather this parameter controls how much continuity the CNN observes from one 3D stack to another. If this is too small the 3D CNN will not be able to pick up enough spatio-temporal dependency. If it is too large there is redundancy which may lead to large training and inference times. Hence there is a need to optimally select this parameter as well. However, since the value of δ and β are integers, the underlying optimization problem is an integer programming problem and hence solving them is NP-hard. As a result, we take an approach where instead of trying to find the optimal value of δ and β, we settle for a value obtained through heuristic methods such as multiple trials that provided us an acceptable accuracy for the modulation detection problem. Next, we describe the data we have utilized and the results we obtained from our experimental studies.

Experiments & Results

We begin by describing the datasets used for our experiments. We used custom datasets collected in uncharacterized RF environments using universal software-defined radios (USDRs). The dataset is summarized below:

Data & Implementation

We collected data across three locations in Eglin AFB, Shalimar Florida. The data consists of five different modulations:

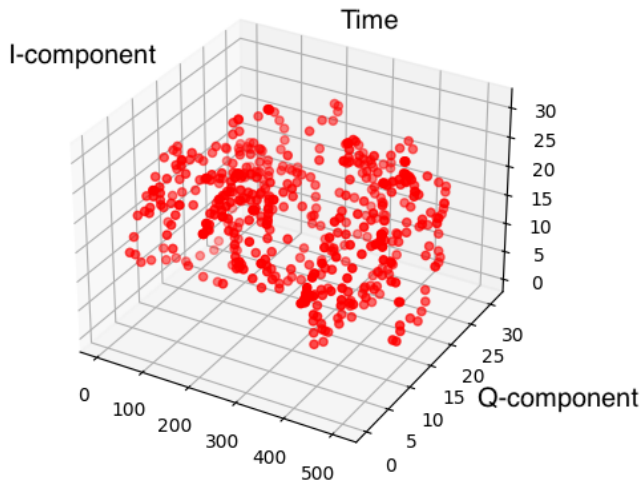

Figure 2: I/Q Data from 64-QAM

BPSK, QPSK, 8PSK, 16QAM, 64QAM, with 100,000 samples per modulation for a total of 1.5 Million (500K @ 3 locations) samples. Each sample for a timestamp has 1024 IQ values (512 I and Q each). We used a USRP B200 receiver with USRP B210 transmitter and GNU Radio is used for programming the USDRs. Transmit data is generated using a Random Uniform Source block from GNURadio and raw dump of IQ data is obtained without any processing at the receiver end.

All the data collection were conducted using a system running MacOS Catalina, with 2.5 GHz Dual-Core Intel Core i7 processor and 16 GB RAM. Figure 2 shows the plot of the I/Q time series data obtained from 64QAM, whereas Fig. 3 shows a single I/Q image obtained from the data. Note that the raw I/Q plot is a plot of the time series data of I/Q values sampled over time, whereas the corresponding I/Q image is a binned two-dimensional image obtained from the time series (see Section). We refrain from showing all the images here due to space constraints. We must point out that for all the cases, irrespective of the modulation scheme used, the time series data showed a distinct helical symmetry, with different parameters. Our goal is to exploit this helical structure and learn the helical parameters using a relatively shallow neural network architecture that uses 3D CNN to learn the spatio-temporal features which implicitly capture the helical symmetry of a given modulation scheme. We describe our neural network implementation next.

CNN Architecture

We designed and implemented a shallow CNN architecture with a total of six layers for the task of modulation detection. Our architecture has two convolutional layers, one pooling layer, one flatten layer, and two fully connected layers. The overall CNN architecture is shown in Fig. 4. Note that we use a kernel of size $(9, 9, 9)$ in the first convolutional layer, a kernel size of $(4, 4, 4)$ in the 3D max-pooling layer and finally a kernel size of $(5, 5, 5)$ in the second convolutional

Figure 3: I/Q Image from 64-QAM

layer. In the first convolutional layer, we have 16 filters while in the second convolutional layer we have 64 filters. We employed the ReLU activation and the ADAM optimizer with a learning rate of 0.0001 with the cross entropy loss.

The input to our neural network was a stack of 2D I/Q images obtained by following the method outlined in Section . Each I/Q image was obtained by binning the I/Q values into 64 bins resulting in 64×64 2D images, which were then piled into a 3D stack. Each such stack of I/Q images had $\delta = 50$ images and an overlap of $\beta = 10$ images. Note that δ and β are parameters that can be optimized. However, optimizing these parameters leads to an integer programming problem that is NP-hard. As discussed in Section , we have utilized a trial and error approach to find the values of these parameters that work best for our data. Based on our experiments, we find that $\delta = 50$ and $\beta = 10$ provides good results. Using this paradigm, for each modulation scheme we have 7498 3D stacks. Finally, we use a confusion matrix to measure the accuracy of our implementation.

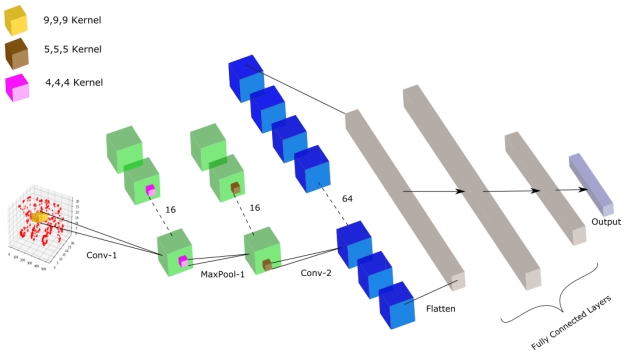

Figure 4: 3D CNN Architecture

Results

We ran several experiments to gauge the performance of our system. The 3D CNN network was trained on a system running Ubuntu, having 64 GB RAM, a GTX 1080 Ti graphics card, and tested on a Macbook with Intel i7 with 16 GB RAM. We report our results for the best (δ, β) pair which is $(50, 10)$, and refrain from reporting all experimental results due to space constraints. We used a $(80, 20)$ split for the training and validation sets. The confusion matrix is shown in Fig. 5. We observe that the system provides the best results for 8-PSK whereas the worst performance for 64 QAM. In fact, for 64 QAM it confuses all the misclassified samples as if they were 16 QAM, which is most likely due to the uncharacterized noise added to the signal from the multi-path present in the environment. We also note that for BPSK, most of the misclassified samples are confused with QPSK and for QPSK it is with BPSK. We believe that this is due to the same reason; the nature of the channel and multi-path in the uncharacterized environment. Overall we see that the total number of samples that are misclassified is small compared to the ones classified correctly. This also shows that the lossy binning of the I/Q values does not hurt the overall performance of the system as hypothesized earlier. In fact, the binning leads to faster execution through data compression while providing acceptable accuracy.

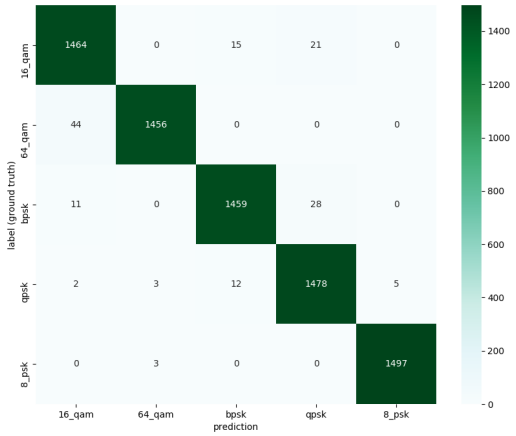

Figure 5: Confusion Matrix for Five Modulation Schemes

Conclusions

In this work, we have discussed the design and implementation of a shallow 3D CNN for the task of AMR. We demonstrated the efficacy of our methods using over-the-air signal data collected in an uncalibrated RF environment. Our results demonstrate the possibility of using the system for real time modulation detection in large scale heterogeneous networks operating in uncharacterized radio environments. Going forward we plan to extend this work by automatically optimizing the parameters δ and β that characterize this framework. We also plan to further study this framework by understanding its relationship with the SNR both theoretically as well as empirically.

References

[Dey and Salemt 2017] Dey, R., and Salemt, F. M. 2017. Gate-variants of Gated Recurrent Unit (GRU) neural networks. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 1597–1600.

[Digulescu et al. 2013] Digulescu, A.; Celik, E. T.; Serbanescu, A.; Fratu, O.; and Halunga, S. 2013. Modulation detection in wireless sensor networks using recurrence plot analysis. In *2013 16th International Symposium on Wireless Personal Multimedia Communications (WPMC)*, 1–5. IEEE.

[Hochreiter and Schmidhuber 1997] Hochreiter, S., and Schmidhuber, J. 1997. Long Short-term Memory. *Neural computation* 9(8):1735—1780.

[Hsue and Soliman 1990] Hsue, S.-Z., and Soliman, S. S. 1990. Automatic modulation classification using zero crossing. In *IEE Proceedings F (Radar and Signal Processing)*, volume 137, 459–464. IET.

[LeCun, Bengio, and Hinton 2015] LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *Nature* 521(7553):436–444.

[O'Shea, Corgan, and Clancy 2016] O'Shea, T. J.; Corgan, J.; and Clancy, T. C. 2016. Convolutional Radio Modulation Recognition Networks. In *Engineering Applications of Neural Networks*, 213–226.

[Weber and Weber 2010] Weber, R. H., and Weber, R. 2010. *Internet of Things*, volume 12. Springer.