

Automatic Modulation Recognition Using Generative Adversarial Networks

Jiawei Yin

University of California, San Diego
j5yin@eng.ucsd.edu

Jinglong Du

University of California, San Diego
jid020@ucsd.edu

Ziwen Li

University of California, San Diego
zil021@eng.ucsd.edu

Abstract

In this paper, we explore the automatic modulation recognition problem using neural networks. Classic neural networks such as CNN, RNN and numerous pretrained networks have been widely applied in classification tasks. However, A recent popular network framework called Generative Adversarial Network(GAN) shows great potential in solving semi-supervised learning problems. We implement this feature into the modulation classification field. Using the Tensorflow and Keras package, we successfully implement our GAN network and compare the results with other models. Our overall model accuracy is up to 50% with all kinds of SNR data. The best modulation classification accuracy is more than 73% using GAN.

1. Introduction

Modulation recognition plays an extremely important role in cognitive radio(CR) technology. The traditional modulation recognition is accomplished by expert's manual work. But this approach has several shortcomings which limits its performance under certain conditions. First, it cannot adapt to unknown tasks and signals, which is common when performing CR work. Besides, it requires a lot of hand-craft work to be done by the experts. Thus, we put forward a new method which can overcome the problems mentioned above. Our approach is based on generative adversarial network(GAN) of semi-supervised deep learning, and it is able to discriminate between real signal and fake signal. What's more, our model can classify the different types of signals. If performing well, our model is able to replace the traditional method and save a lot of hand-craft work.

2. Background

In this section, we will review the motivations and related works that have been done in the modulation classification field. Modulation is a kind of technique that changes the original signal including amplitude, frequency

and phase into another another form of expression. The history of modulation classification dates back to the development of signal detection problems in 1950[9]. At that time, people use binary classifications to judge whether it's a signal or not. Later on, modulation classification extends the signal detection of yes-no questions into multi-classification problems. This marks people are able to distinguish what types of the original signals are in the receiver part.

Nowadays, using computer software and hardware becomes an integral part in classification problems. They use different datasets to train deep neural networks. Typical models are CNN, RNN and numerous pretrained networks such as Resnet, GoogleNet, VGG16 and so on. Thanks to the rapid development of deep learning, several researches have been done on the Deep learning to automatic modulation recognition (AMR). Azzous et al is a pioneer in this field. According to [1], He proposed to use an artificial neural network (ANN) to classify the analog modulation communication signals. This is a very significant attempt which inspires the idea of applying deep learning to AMR. O'Shea et al proposed a method to apply the convolutional neural network (CNN) to the modulation recognition field in [6]. It should be noted that they use time-domain in-phase orthogonal (IQ) signal as the input of the network which is a good start. We also use the dataset generated by O'Shea's research which will be mentioned in the later section. Recent research focuses more on the supervised learning and the most distinctive one is proposed by Tang Bin et al [8]. He applies GAN as an approach of data augmentation to do the classification which is quite novel and eye-catching.

In this paper, we introduce a recent popular neural network called Generative Adversarial Networks (GAN) to do the classification tasks. This concept was firstly put forward by Goodfellow et al. [2] in 2014. The GAN network plays min-max games which greatly improved the generalization ability of the model although the initial edition is not very stable in different scenarios. After that, hundreds of improvements on the GAN emerged in the recent years' study such as CGAN [4], DCGAN [7].

3. Dataset and Features

In wireless communication, the true signal can be expressed as:

$$S[n] = S_I[n]\cos(2\pi f_c n) - S_Q[n]\sin(2\pi f_c n) \quad (1)$$

Where f_c is the carrier frequency. Then the S_I and S_Q will be extracted and stored into two rows to form the original data for each data point. The dataset that we used has the same logic. In our project, we will analyze based on the electronic signal data with different signal to ratio rates and different preferred modules. The dataset that we evaluate is a synthetic RF data set [5] generated by researchers at Virginia Tech. 8 digital and 3 analog widely used modulations are included in the dataset. These include 16QAM, 64QAM, BFSK, CPFSK, PAM, BPSK, QPSK and PSK for digital signals and AM-SSB, AM-DSB and WB-FM for analog signals. There are 220 thousand signal sets, and for each signal data, it comes with 2×128 float numbers as features and we wish to classify them into 11 different modules. As mentioned we extracted In-Phase and Quadrature samples to form this 2-wide dimension. The label of each datapoint includes both modulation types and SNR. The visualization of 11 modulations are shown below:



Figure 1. Visualization of 11 different modulation types

4. GAN Model

Figure 2 is the model that we referenced[3] and make some changes on it. Our model consists of 4 parts, the encoder(E), generator(G), discriminator(D) and classifier(C).

Figure 3 is the detailed design in the feature extraction part.

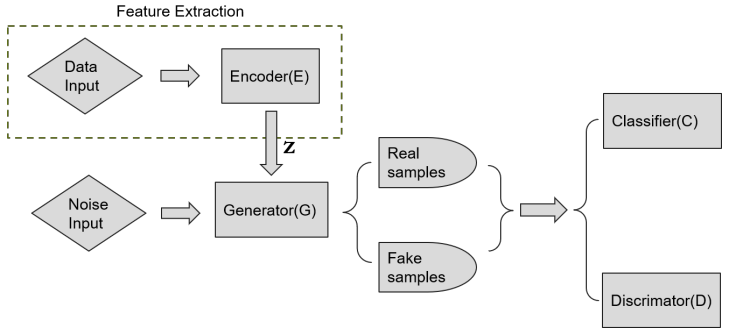


Figure 2. General Structure of GAN[3]

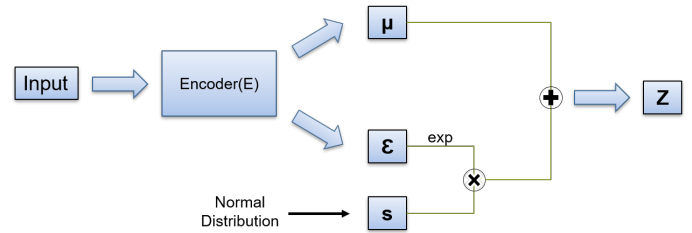


Figure 3. Detailed design in feature extraction

The encoder is responsible for extracting the features from the input real data. We first extract the mean and variance of the latent variables, then combine them using the Gaussian distribution equation to get the latent variables which will then be passed to the generator. In order to better utilize the temporal connection of the input signal, we use LSTM as the first layer of E. Then the two separately dense layers are followed to get the mean and variance. At the end, we use a self-build layer to combine them to get the extracted features.

In terms of the generator, it combines the extracted features from the encoder and the real labels of the input samples. It then generates the real samples and passes them to the classifier and discriminator. We also provide the generator with random latent variables from gaussian distribution and random label. The generator utilizes them to generate fake samples, which are also passed to the classifier and discriminator. In detail, the generator consists of an embedding layer and a hadamard product function which combines the latent variables and the input label. Then a dense layer and a reshape layer are applied to convert the input into a 3-dimensional variable. After that, three transpose convolution layers and a reshape layer are used to generate the signal with the same shape as the real samples.

For the classifier and the discriminator, they have the exact opposite structures as the generator except the last dense layer. The last dense layer of the classifier has 11 nodes

followed by a softmax activation function in order to predict the label of the input signal. On the other hand, the last dense layer of discriminator only has 2 nodes which are used to differentiate the fake sample from the real ones.

5. Implementation

In this section, we will introduce the experiment environment and parameter settings in coding level. The detailed model layer and configuration will be shown in the following figures.

We use Python 3.6 as our programming language. And we apply Tensorflow and Keras packages as our platform to implement deep neural networks. Also we use GPU programming to speed up our model training and testing. All of the results can be tested again.

5.1. MLP

MLP	
Layer Type	Input shape
Dense	256
Dense	256
Dense	256
Dense	256
Dense	11

Figure 4. MLP Layer Structure

The structure of MLP is quite Intuitive. We add 5 Dense layers to train our model. In fact, we use this structure as our baseline in our project. The output layer is using Softmax.

5.2. CNN

In this part, we design a standard CNN structure. It starts with a reshape layer of $2*128$ and then adds a zero-padding layer. After that we add the convolution layer and drop layer. Finally we add three Dense layers to output our results. The optimizer of CNN is Adam with learning rate of 0.001 and weight decay of 10^{-6} .

5.3. GAN

The detailed configuration of 4 sub-models are shown in the figure. Encoder receives $2*128$ original signal inputs

CNN	
Layer Type	Input Shape
Reshape	$2*128$
ZeroPadding 2D	$2*128*1$
Conv2D	$2*132*1$
Dropout	$2*130*256$
ZeroPadding 2D	$2*130*256$
Conv2D	$2*134*256$
Dropout	$1*132*80$
Flatten	$1*132*80$
Dense	10560
Dense	256
Dense	11

Figure 5. CNN Layer Structure

Encoder(E)		Generator(G)	
Layer Type	Input Shape	Layer Type	Input Shape
Reshape	$2*128$	Dense	100
LSTM	$128*2$	Reshape	8192
Dense	10	Conv2DTranspose	$2*128*32$
Dense	100	Conv2DTranspose	$2*128*256$
Custom Layer	100	Conv2DTranspose	$2*128*80$
		Reshape	$2*128*1$

Figure 6. Layer Structures of Encoder and Generator

and output the encoded 100 neurons. This process compresses and extracts useful information. Then the Generator receives two inputs: the real one and the fake one generated by the random Gaussian variable. After layer computation, it outputs $2*128$ data point which generates the real data and fake data. The next step is to let classifier(C) and discriminator(D) to distinguish the real one from the fake one. We can see that C and D have similar structures and the difference is the output part which is not shown in the figure. The function of classifier is to give a label for the input using one-hot encoding so the output layer is Softmax. The function of discriminator is to give a score based on the input to see whether it's real or fake.

We use the Adam to be the optimizer of all our mod-

Classifier(C)		Discriminator(D)	
Layer Type	Input Shape	Layer Type	Input Shape
Reshape	2*128	Reshape	2*128
Conv2D	2*128*1	Conv2D	2*128*1
AveragePooling2D	2*128*32	AveragePooling2D	2*128*32
Conv2D	2*64*32	Conv2D	2*64*32
AveragePooling2D	2*64*32	AveragePooling2D	2*64*32
Flatten	2*32*32	Flatten	2*32*32
Dense	2048	Dense	2048

Figure 7. Layer Structure of Classifier and Discriminator

els(E,G,C,D). The learning rate of C is 0.0004, while the learning rate of E,G,D is 0.0001, which comes from the fact that C is much harder to train than the others. We have trained for 300 epochs and the batch size is 500, for each iteration within an epoch, we first use E to get the latent variables of real samples, then generate random latent variables of the batch size, then concatenate them and use generator to reconstruct the real samples and fake samples. The classifier and discriminator are then updated according to the samples before.

Another data result is, we use the discriminator to distinguish between generated fake signals with Gaussian Distribution and the real signals, the model gives more than 99.9% accuracy, which offered better performance on improving the classification model.

6. Results and Analysis

In this section, we will compare the results among three different models with different SNR. It can be seen from

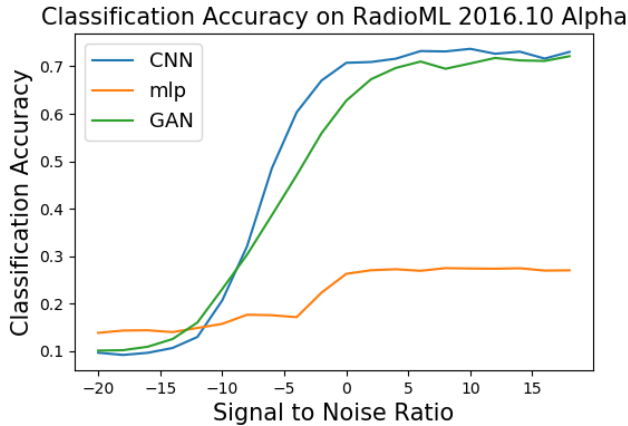


Figure 8. Classification Accuracy on three different models with different SNR

the figure that MLP has very bad results in all SNRs. How-

ever, larger SNR gives a little bit of improvement on the classification accuracy. CNN and GAN actually have similar results. GAN has a little bit higher accuracy in low level SNR. On the other hand, CNN has a little bit better accuracy in high SNR.

The result for the accuracy of GAN is not very ideal. We think the reasons are: (1) we only run 200 epochs to do the classification. But the training and testing accuracy are still increasing. With more epochs, we may have better results. (2) the model itself still has potential to improve in terms of some layer structures and hyper-parameter tuning. (3) GAN do not have absolute advantages in modulation classifications. Instead, the generalization ability and unsupervised learning features are the direction that we need to focus on in the later work.

7. Conclusions and Further Work

For fully connected multi-layer perceptrons, we use it as a baseline mode and the result is shown above. The overall accuracy for validation is 21.63%. The MLP is very easily overfitted in the middle of the 120 epochs of training, and it clearly shows that we could have better performance with other architectures instead of a simple MLP.

For CNN, we use different optimizers to do the image classification. The overall accuracy for validation is 50.22%. It shows that large momentum value will accelerate the convergence speed and the performance of adding momentum will be better than the simple SGD optimizer. In the future, we try to add the layer to the current CNN network. Now the best error rate is around 11%. So it has great potential to reduce the error rate in comparison with the up-to-date pretrained networks.

The GAN based semi-supervised model provides more potential for the accuracy of classifying the signals in overall result, we achieved up to 46.29% accuracy in all SNR rates together, and for different SNR results, the result is shown in the figure above. It is noticeable that the convergence speed for our model is slower than the lightweight simple CNN classification model, but we believe our model's potential is higher than the CNN model, which provides a good chance for further improvement with a reasonable level of computation resources and times in the future. We have spent around 20 hours on training the GAN model, and it continuously gets the better performance. We hope that in the future we could use more time to train the model and the overall performance could reach the expected level, which is higher than the CNN model.

8. Contributions

Jiawei Yin: (1)Prepare for the dataset and do the pre-processing step. (2)Develop CNN model and plot accuracy and loss graph with different SNRs. (3)Write the final paper (4)Wrote some critics to peer review.

Jinglong Du: (1)Work together on analysing data and model building. (2)Develop a basic MLP baseline model for baseline comparison and help on other models. (3)Wrote some critics to peer review.(4)Working together with the final report

Ziwen Li: (1)Develop GAN network using Keras package.(2)Debug the core part of GAN network (3)Wrote some critics to peer review. (4)Working together with the final report

References

- [1] A. Azzouz, E.E.; Nandi. Automatic modulation recognition of communication signals. *IEEE Trans.Commun.*, 46:431—436, 1996.
- [2] M. M. B. X. D. W. S. O. A. C. Y. B. Ian J. Goodfellow, Jean P. Generative adversarial nets. *arXiv:1406.266*, 2014.
- [3] L. G. Z. C. Li M, Li O. Generative adversarial networks-based semi-supervised automatic modulation recognition for cognitive radio networks. *Sensors (Basel)*.
- [4] S. Mirza, M.; Osindero. Conditional generative adversarial nets. *Neural Information Processing Systems (NIPS), Montreal, QC, Canada8-13*.
- [5] T. O’Shea and N. West. Radio machine learning dataset generation with gnu radio. *Proceedings of the GNU Radio Conference*, 1(1), 2016.
- [6] J. C. T. O’Shea, T.J.; Corgan. Convolutional radio modulation recognition networks. *International Conference on Engineering Applications of Neural Networks, Aberdeen, UK, 2–5*.
- [7] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015.
- [8] B. Tang, Y. Tu, Z. Zhang, and Y. Lin. Digital signal modulation classification with data augmentation using generative adversarial nets in cognitive radio networks. *IEEE Access*, 6:15713–15722, 2018.
- [9] T. B. W. Peterson and W. Fox. The theory of signal detectability. *Transactions of the IRE Professional Group on Information Theory*, 4(4):171–212, 1954.

9. Reply to review

9.1. Group 38

1. The loss plot shows that the model is evolving continuously. The accuracy plot is important indeed and we have added our accuracy results in our paper. The axis have been adjusted to proper size. The representation of X-axis has changed. And it's just a visual problem and each epoch is integer.
2. The purpose of using GAN is when the label of one signal is lost, we need to use previous experience to judge this signal. Basically, GAN belongs to the unsupervised learning field. And it should have good generalization ability.
3. We will change our recording equipment next time to have better recording effects.
4. We will do more practice before recording next time.
5. We have added this part in our paper. Thank you for your advice.
6. Good suggestions. We will add markdown next time.

9.2. Group 63

1. Analog and digital are just different modulation types. In the training dataset, they are just different IQ values in the matrix. We have drawn visualizations in our paper to see the difference.
2. The function of the classifier is to do the classification among 11 modulation types. One-hot encoding is just a representation to see the final result after the softmax layer.
3. Yes, the core idea of Generator is to confuse Discriminators using fake signals(generated by Gaussian Random noise).
4. Due to limited time, we only trained 25 epochs. In our epoch we trained 120 epochs.
5. We just need to plot the accuracy value stored in the fitting function.
6. We will change our recording equipment next time to have better recording effects.

9.3. Group 76

1. We have added this part in our paper. Thank you for your suggestion.
2. It's a typo in our slide. It's -20 dB to 18 dB.
3. Good Suggestion. We will change the size next time.
4. We have added our results in our paper.
5. Actually, GAN is not very stable in different datasets. We still have lots of work do to modify our codes.
6. The preprocessing has limited improvements. We do think that AM-DSB modulation dataset has some problems when plotting.
7. Good suggestions. We will add it in our code.