Deep Networks for Equalization in Communications

Laura Brink, Nipun Ramakrishnan, Nikhil Shinde, and Anant Sahai

***Abstract*— In this paper, we explore how neural networks can make more robust and adaptable equalization processes in communication systems. We address the effects of inter-symbol interference and carrier frequency offset and show that ....**

I. INTRODUCTION

With the rise of internet of things, more and more users will need to access the wireless spectrum. As the spectrum becomes saturated with users, not only must our systems adapt to changing environments but also be able to coexist and thrive in the presence of unknown neighbors. We must design a robust communications system that will give us the freedom to.

Most communications systems have three main processes; equalization, demodulation, and error-correction. While we will need to design robust forms of all of these processes, we will focus on equalization for the remainder of this paper.

A. Inter-symbol Interference and Equalization

What is equalization and ISI?

Inter-symbol interference occurs when we are transmitting over a channel that has some echos. These echos cause the receiver to hear a garbled signal instead of the original signal from the transmitter. This is called inter-symbol interference because the receiver is hearing a combination of symbols across time.

Let $\vec{x} = [x_0, x_1, \dots x_n]$ be the set of n complex symbols that the transmitter sends over the channel that connects the transmitter to the receiver. Each channel will have different characteristics. Some channels may have echos, others may have delays, often channels will have both. When a channel has echos, this is called a multipath channel because there are multiple paths to reach the receiver. Each path is called a tap. We can characterize a channel by characterizing the taps.

What can cause these echos? How prevelant are they?

Channel taps: Let $\vec{a} = [a_0, a_1, \dots a_l]$ be the set of characteristic for a multipath channel that has l taps. When a sequence of symbols like $\vec{x}$ is transmitted over this channel, the channel taps are convolved over the sequence. Additionally, there is noise in the system denoted by η_i.

$$\tilde{x}_m = \sum_{i=0}^{l} a_i x_{m-i} + \eta_i$$

The receiver will hear a signal that is corrupted by inter-symbol interfence and noise; $\tilde{\vec{x}} = [\tilde{x}_0, \tilde{x_1}, \dots \tilde{x}_{n+l}]$.

Receivers must be able to handle garbled signals in order to transmit data in the real world. The process of removing the inter-symbol interference is called equalization. The goal of equalization is to take in a garbled signal and output a signal without any inter-symbol interference.

INSERT IMAGE OF 2 TAP CHANNEL EFFECTS ON QPSK

Engineers have built processes to remove inter-symbol interference. First, let's go into the case when the channel characteristics are known.

1) Equalization for a known channel: If you know the channel characteristics, $\vec{a}$, perfectly, then there are a few different methods that can be used.

Zero-forcing

While it's important to consider how well a receiver can equalize with a known channel, this is rarely the case. Usually, we do not know the channel characteristics.

2) Equalization for an unknown channel: When the receiver does not know the channel characteristics, the process of equalization essentially has two jobs; first, identify the channel, second, remove the inter-symbol interference. If the receiver did not identify the channel first, there would be no way to remove the affects of it on the received signal.

In order to do channel estimation, most systems require that packets begin with a known sequence called a preamble.

Channel estimation: least squares

Minimum mean squared error equalizer.

3) How do real systems handle equalization?: OFDM does not have this problem!

B. Carrier Frequency Offset and Correction

Now, if we were to implement our minimum mean squared error algorithm on a physical receiver, we would find some problems with our equalization process. Our equalizer will equalize the first symbols very well. However, as we equalize end parts of our sequence, we will encounter a physical phenomonen called carrier frequency offset, CFO.

Carrier frequency offset occurs when ???

When there is a significant CFO present, our symbols will gradually start rotating. CFO will effect our received symbols like

$$\tilde{x}_m = x_m e^{mj\omega}$$

The effect will look like something like this

INSERT IMAGE OF CFO OCCURRING on QPSK

1) How do real systems handle CFO correction?: There are a few ways to handle CFO, some are more elegant than others.

The first solution is to try to remove the problem. Since CFO is dependent on the length of a packet, one solution is to make packets so short that

A more elegant solution is using phase-lock loops (costas loops).

What must a modern day receiver handle? What does it look like when we have both CFO and ISI?

$$\tilde{x}_m = (\sum_{i=0}^{l} a_i x_{m-i}) e^{mj\omega} + \eta_i$$

INSERT IMAGE OF AFFECTS OF BOTH CFO AND EQUALIZATION

INSERT IMAGE OF SAME THING WITH NOISE!

II. Related Works

related works!

III. Deep Networks for Equalization

A. Replicate Results

B. Channel Estimation

- compare least squares and how KNN did with pure deep nn based architecture
- NN did better than least squares
- hyperparam search over general 1-layer to 4-layer dense layers, and number of nodes and activations
- plots: how error changed with respect to data points, as number of data points increased, NN outperformed LS
- preamble: 100
- want: QPSK, plot of preamble length
- want? how to visualize that NN does better than LS

C. Channel Equalization

- compare to MMSE
- re run with the new RNN architecture
- backprop length of 3. crude search from 1-10. 2 tap channel
- added channel preprocessing: didn't seem to make too much of a difference
- plot log/ log scale to find converging in error

1) Learning an inverse:

- can a NN learn to do division?
- given a one tap channel and data sequence, output is equalized data sequence
- with and without log feature scaling
- without log errors = 10^-6
- MMSE gets error = 10^-32
- with log errors =
- straight inversion, without log feature scaling 10^-7 - with dense layers
- inversion with log feature scaling with error of 10^-14
- plots: inversion, but as a function of beta for both non-log and log

2) Learning to multiply two inputs: NN given x, y - output x*y

D. Channel Est + Equal

re run that

IV. Deep Networks Carrier Frequency Offset

A. Recurrent Neural Network Follows a Circle

for a constant rate for a given rate

B. Deep Network Carrier Frequency Offset Estimation

- complex gradients problems
- act like the real and imaginary parts are separate
- plots: one tap channel plots, without equalization problems
- plots: two tap channel plots, with equalization problems

C. Deep Network Carrier Frequency Offset Correction

Program a Costas loop for comparison

V. Combine CFO and equal?

TABLE I
AN EXAMPLE OF A TABLE

One	Two
Three	Four

We suggest that you use a text box to insert a graphic (which is ideally a 300 dpi TIFF or EPS file, with all fonts embedded) because, in an document, this method is somewhat more stable than directly inserting a picture.

Fig. 1. Inductance of oscillation winding on amorphous magnetic core versus DC bias magnetic field

Figure Labels: Use 8 point Times New Roman for Figure labels. Use words rather than symbols or abbreviations when writing Figure axis labels to avoid confusing the reader. As an example, write the quantity Magnetization, or Magnetization, M, not just M. If including units in the label, present them within parentheses. Do not label axes only with units. In the example, write Magnetization (A/m) or Magnetization A[m(1)], not just A/m. Do not label axes with a ratio of quantities and units. For example, write Temperature (K), not Temperature/K.

VI. CONCLUSIONS

We built a system.

[1]
[2]
[3]

APPENDIX

Appendixes should appear before the acknowledgment.

References are important to the reader; therefore, each citation must be complete and correct. If at all possible, references should be commonly available publications.

REFERENCES

[1] S. Dorner, S. Cammerer, J. Hoydis, and S. t. Brink, "Deep learning-based communication over the air," 2017.

[2] H. Kim, Y. Jiang, R. Rana, S. Kannan, S. Oh, and P. Viswanath, "Communication algorithms via deep learning," *International Conference on Learning Representations*, 2018.

[3] N. Farsad and A. Goldsmith, "Neural network detection of data sequences in communication systems," *IEEE Transactions on Signal Processing*, vol. PP, 2018.