Exploring the Cocktail Party

A review of

Héctor M. de la Rosa Prado

A thesis presented for the degree of
Bachelor in IT in Science

Department Name
National Autonomous University of Mexico
Mexico
November 12, 2019

Contents

Thesis Title

Thesis Subtitle

Héctor M. de la Rosa Prado

Abstract

Part I

Introduction

Chapter 0

Authors Notes

Imagine being in a public area with one of your friends. As you talk about your work and what you learned in school there are people around you making noise. They're also talking about their lives, what they plan on doing in the mall or the movie that just came out. Kids laughing with their parents talking, nearby traffic with car horns flaring in the distance. If you wanted to, you could hear them, but instead you ignore it.

You continue to listen to your friend, regardless of the noise in the background. You have no problem paying attention until suddenly you hear your name being mentioned in the crowd. For some reason, somewhere, someone said your name. You look around to see a familiar face walking towards you...

What we described in the short and imaginary story above is what is called the "Cocktail Party Effect". This effect was first coined by Cherry Colin[1] where he proposed a series of tests that would measure the limits of a human's ability to listen to a specific voice under different circumstances. The problem of getting a machine to do this same task was called the Cocktail Party Problem by Cherry.

The cocktail party problem is one of the biggest unsolved problems in computation. The short and imaginary story above gives us an example of how we, as humans, solve this problem in a seemingly effortless way. Thanks to evolution, humans can it so effortless, in fact, that most people don't even stop to appreciate how complicated the task actually is.

Its difficulty, however, is not the reason it is so widely studied. In fact, this small problem seems to be the barrier that has kept us from advancing in the automation of automation, or at least in the way we would like. The cocktail party is not only present in sound, but in just about any signal processing problem, from medical scanning to telecommunications[2], noise always seems to find its way into our sensors.

That is why a general solution to this problem will not only allow us to improve greatly in audio related tasks, such as speech recognition, transcriptions, audio classification, and audio/speech enhancements, but also in various fields like medical analysis and seismology. The reach of these advancements also promises a wide range of new technologies shortly after.

This is why, in this work, we will attempt to take a dive into how we can leverage the current improvements in artificial intelligence in tackling the problem. Ever since the latest boom of deep learning [3]

Even though, as we saw before, the problem generalizes beyond speech, we will limit our research to stick with voiced data. This is due to the complexity in examining signals as a whole, and because of the focus that most methods currently have on the subject.

Given how the difficulty of the problem lays much beyond my current limits as a researcher we will not show. We will accept limiting our problem greatly for practical reasons, in an attempt to get insight on . This also includes a review of how traditional methods attempt to solve the problem

0.1 History

Chapter 1

The Cocktail Party Problem

Speech separation has been a problem that researchers have been interested in for years. So much so that the problem was formulated decades ago by Colin Cherry[1]. In his famous paper he gives an example of the task with a conversation in a Cocktail party, giving the problem its name. Over 65 years later one could argue that progress is just now being made in the area, mostly due to the advancements in general learning algorithms, like deep learning, giving an edge in unstructured data analysis.

1.1 Segmentation vs Attention Problem

Many studies*** have been used to divide the problem into more manageable tasks, many of which were included in the experiments made by Cherry himself. Thanks to this, the problem has evolved quite a bit since it was first formulated, deviating from its original definition and including all types of noise, not just speech. As McDermott mentions in his article[2], the problem has been split in two parts, which together engulfs the problem as a whole. Usually both parts are frequently referred to as the Cocktail Party problem interchangeably.

The first part addresses who or what to pay attention to, and when to change targets. For clarity in this paper, we will refer to this specific part as the Attention problem. To solve this problem we need background knowledge about the person paying attention as well as the context to know what the listener wants to listen to. What's more is that the listener might want to listen to two speakers at the same time, given that they are both saying something important. The difficulty of the problem lies in how subjective it is. Because of this, it is practically impossible to model and automate the process.

The second part refers to actually separating the sounds, or also known as "sound segregation". This part will be referred to as the "segregation problem. This differs from the previous task because here we look to separate the audio by different sources. We have no need to know which is more important than the other, as long as we can listen to each source independently. This is the process that has received more attention in the last couple of years, since it is easier to model. That said, it is still an ill-posed problem (more on this later) which means that traditional methods have a hard time.

In this thesis we will only be addressing the segregation problem, trying to separate an audio by the different sources of noise it contains. This problem has quite a bit of research *** but still has a long way before we can actually consider it to be solved. One of the reasons was just mentioned, but it's also due to the fact that the problem, under certain conditions, becomes impossible to model. This leaves us with a few options, we either limit the problem to certain use cases, or we facilitate the problem adding more information. We will consider both of these actions to be "constraints", as one limits the conditions that could be solved while the other limits possible solutions, requiring more information.

1.2 Inverse Problems

An inverse problem refers to reconstructing the source from a generated set of components. In this case, we want to reconstruct the separated audios from a generated mixed audio. Inverse problems are usually classified by how "Well-Posed" it is, which is also generally used as an indicator of how difficult the problem could be. Many examples of Inverse problems that extend to signal processing include medical imaging, oceanography, astronomy and geophysics. [InverseProblems]

An inverse problem has 4 components, this includes a measurement operator (MO), injectivity or regularization of the MO, knowledge on noise and the model and finally a numerical implementation. Some of these components are then separated into different sub-catagories, but we will only mention the ones relevant to this work. We address the components based on Guillaume Bal's definition [InverseProblems].

Measurement Operator

- 1
- 2
- 3

Injectivity

- 1
- 2
- 3

Noise and Model

- Noise is practically inevitable in audio, extending from white to brown noise and most times even "structured" noise. This could come from background music or oncoming traffic. When we record in a crowded room the noise changes too, leaving a sound that we can relate with a busy street.

- The measurement will also have errors, but these are less relevant to us. In audio compression, there is always a slight error from the original source to what we hear later on. This, however, is such a small change that we as humans cannot perceive it, at least not without the help of special tools.
- ***

Numerical Implementation For this project we will be using an optimization method, comparing it to previous linear systems. Although these methods were usually considered fairly inexpensive, we will be using neural networks, which are both expensive computational as well as in the data sense.

1.3 Ill-Posed Problems

Let's imagine that we have an audio with two sources that generate noise. One is a dog, and the other is a human. Both audio have different structures and can be separated by a person with quite little effort. We could even separate the audios using certain statistical methods***. Sadly the Cocktail Party includes problems much more complicated than the one we just mentioned. A wide variety of audios include at least two people talking. This is harder to separate, sometimes even for humans, especially if the two individuals have similar voices and speak in a similar manner.

The difficulty of the Cocktail Party problem is that it is Ill-Posed. The terms Ill and Well-Posed problems was first coined by Hadamard, Jacques, a French mathematician born in the mid 1800's. According to his definition, a problem is considered to be Well-Posed if:

- For any input there is a solution
- That solution is unique
- The solution is "stable" in the input space (continuous)

If one of these three conditions were not met, the problem was considered Ill-Posed. At the time, it was considered necessary for any mathematical problem in physics and tech to be formulated as a well-posed problem. This meant that, for practical uses, studying Ill-Posed problems was useless[Ill-Posed].

Now, however, Ill-Posed problems are very common, in fact it is considered rare that a inverse-problem should be well-posed. It has come to the point where researchers seemingly compare how "Ill-Posed" a problem may be[InverseProblems]. These extends to Well, mildly Ill and severely Ill-Posed problems.

1.4 Constraints

If we continue with our thought experiment from before, what would happen if instead we had two machines that emit static noise of the same type, same volume similar position etc. This would be practically impossible to separate without knowing more about the noise being emitted to try and find some structural difference in the two. Not only that, but the problem wouldn't be useful in this broader sense. Why would you want to separate noise from noise? The situations where we want to solve the problem don't require us to tackle a perfectly generalized version of it. This is why it is common to set constraints to the problem before beginning to train a model.

As mentioned before, this work is about speech separation, so it is fair to assume that the audio will have at least two people talking. Another constraint we will add is that we will only focus on english speakers, leaving a more general model for another project. As we will see below we will also be using the Common Voice dataset. This dataset is publicly available to make similar models and consists of almost 40000 voices in more than 1000 hours of recorded english voice in different accents.

We could also consider the number of speakers as a constraint, since having only 2 speakers simplifies the problem greatly. During these tests*** we will try to make a general model that can split the different speakers even when mixed with an arbitrary amount of voices. Clearly there will be a limit given how 100 voices will more than likely be worse than gibberish, but this arbitrary model could help us separate audio even when unexpected speakers enter the conversation.

Another constraint that could be considered is that we only need to listen to one voice. If you use a hypothetical "perfect" model to separate voices and listen to your friend in a crowded cocktail party, you have little to no interest in listening to other people's conversations. Although this could definitely be easier and give better results, we go back to what we mentioned before when we introduced the cocktail party. This constraint in reality mentions a problem with attention, which for now we are considering a different problem.

Part II

Methodology

Chapter 2

Literature Review

2.1 Current SoTA with AI

2.1.1 Speech Recognition

2.1.2 Speech Separation

Looking to Listen

2.2 Feature Extraction and Normalization

2.2.1 Do Spectrograms belong in AI?

2.3 Data Augmentation

2.4 Targets and Masks

Chapter 3

Methodology

3.1 Computer Requirements

3.2 Dataset

3.3 Preprocessing

3.4 Model

3.5 Targets

3.6 Error

Chapter 4

Implementation

4.1 Libraries

4.1.1 Librosa

4.1.2 PyTorch

4.1.3 TorchAudio

4.2 Source Code

4.2.1 Running Instructions

Part III

Results

Chapter 5

Results

5.1 Initial Dataset

5.1.1 Quantified Losses

5.1.2 Quality Tests

5.1.3 Samples (images and audios)

5.2 Transfer learning

5.2.1 Quantified Losses

5.2.2 Quality Tests

5.2.3 Samples (images and audios)

5.3 Transfer Learning with fine tuning

5.3.1 Quantified Losses

5.3.2 Quality Tests

5.3.3 Samples (images and audios)

Chapter 6

Discussion

6.1 Future Work

Appendices

Appendix A

Time Series

A.1 Signal Processing

A.2 Speech Properties

Appendix B

Transforms

B.1 Fourier Transform

B.1.1 Spectrums

B.1.2 Discrete Fourier Transform

B.1.3 Short-Time Fourier Transform

B.2 Wavelet

Appendix C

Spectrograms

As we saw before, audio can be better represented as mix of different frequencies that define the sound we are hearing. The problem lies in the fact that, in speech, these frequencies change drastically during each conversation, sentence, word and syllable! This means that the perfect spectrogram would need to be a continuous representation of each moment of time and for every frequency.

This is impossible for various reasons, one being the limitations of computational representations, which forces us to make a discrete representation instead. But even if that wasn't the case, the fourier transform doesn't work on single points in time, instead it works on entire signals. That said, we can use the short time fourier transform to give us a spectrogram that is discrete in time. Combined with the discrete fourier transform, we will also have a discrete representation in the frequency axis.

Spectrograms have 3 dimensions, similar to how images are represented, but not exactly. In any given image there are two dimensions that represent space and one that represents intensity. Meanwhile Spectrograms have one dimension for time, one for frequency and the last represents the strength and sometimes the phase of the frequency signal at that time. Although some writers, the standard tends to place frequency as height, time as width and color as signal intensity in a given point.

Although this is a great step forward to giving us a better representation of audio, there are still too many parameters that play a role. Each one of these parameters affects how well the spectrogram can correctly contain the information in the audio. Although there are already well established defaults for most human tasks, it still hasn't been well studied in great part of automated and machine learning tasks.

The exception to this being Speech Recognition, but even so they haven't been heavily studied. There are few papers and datasets that can be used to prove when some parameters are better than others or if there are conservitive settings that tend to work consistently in all problems. In the end, some authors question if spectrograms are even the representation that we are looking for for various reasons that we will see in more detail later in Chapter 7.

For now we'll hide our skepticism and focus on the advantages of using spectrograms.

The spectrograms we have been referring to so far are what are called linear-spectrograms. Although these are the simplest and easiest to implement, it is not the only one. We will mention the difference between Amplitude and Decibel Spectrograms as well as Frequency and Mel-Bin Spectrograms.

C.1 Linear and Log

The values in amplitude are represented in the “color” dimension, giving us the intensities that we see in the spectrogram image. As we saw in the Second Chapter, the amplitude tells us the strength of the signal. In the spectrogram, however, there may also be a phase value. This is because the signals tend to be represented as a Complex number, which can be expressed in cartesian or polar coordinates.

C.1.1 Amplitude vs Decibels

C.1.2 Mel-Bins

C.2 The Phase Problem

C.2.1 Linear

C.2.2 Decibel

C.2.3 Mel-bin

C.3 Phase Retrieval Techniques

C.3.1 Phase Storage

C.3.2 Griffin-lin Algorithm

C.3.3 Vocoders

Appendix D

Machine Learning

D.1 Supervised vs Unsupervised Learning

D.1.1 Supervised Learning

D.1.2 Unsupervised Learning

D.2 Advancements

D.2.1 AREAS**

D.3 Problems

D.3.1 Data

D.3.2 Interpretability

D.3.3 Overfitting

D.3.4 Hyper-parameters

D.3.5 Computation

Appendix E

Deep Learning

E.1 Neural Networks

E.1.1 Structure

Inputs

Weights

Bias

E.1.2 Back-Propagation Algorithm

Forward Propagation

Gradients

E.2 Image Processing

E.2.1 Convolutional Neural Networks

E.2.2 Attention

E.3 Segmentation

E.3.1 Medicine

E.3.2 U-Net

Bibliography

[1] C. E. Colin, "Some Experiments on the Recognition of Speech, with One and with Two Ears," *The Journal of the Acoustical Society of America*, vol. 25, no. 5, pp. 975–979, 1953.

[2] L. Marchegiani, S. Karadogan, T. Andersen, J. Larsen, and L. Hansen, "The Role of Top-Down Attention in the Cocktail Party: Revisiting Cherry's Experiment after Sixty Years," in *Proceedings of the tenth International Conference on Machine Learning and Applications (ICMLA'11)*, (United States), IEEE, 2011.

[3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.