Emotional Classifier using Audio Data

PHYS4606 Final Project in Spring 2023

Manami Kanemura

Contents

1 Introduction

Audio classification and speech recognition using deep learning models have been one of major topics in computer science. When you call an airline company, they will not let you talk to a human first but will give you some guidance in automated voice instruction. If you speak an unclear word or sentence, it asks you to say it again. This is possible because the voice instruction is programmed to take inputs of your voice, compare them with the expected answers, and output if your question has an answer or not. As voice is converted into signals in computers, signal processing has been one of the primary research fields since the early 1990's. Moreover, voice and emotions are deeply connected. When you give a birthday gift to your friend and your friend checks it and says 'Thank you...!' in a lower voice, you should guess that your friend may not be happy with the choice of your gift. Or if your friend says 'Thank you...!' in a higher tone, you gain some confidence in your sense of choosing gifts.

The goal of this project is to study the signal processing for the human voice and to build deep learning models to classify a speaker's emotion based on voice data (angry or happy). Given in the example of a phone call to the airline company, computers linearly transfer voice data, namely automatic lady in the voice instruction does not perceive your aggressive voice tone even if you are upset due to a flight delay. On the contrary, as given in the example of your friend's reaction after giving a birthday present, if your friend's changes the voice tone as soon as she/he sees the gift, you would immediately know whether your friend likes it or not. In this project, the differences in signal preprocessing between how computers detect our voice and how humans detect our voice by applying Fast Fourier Transform (FFT) and Mel-scale spectrums. After the analysis of voice data, convolutional neural networks (CNN) with different pooling methods are built to classify the binary emotion from vocal data. All codes, audio data, and additional figures are available in my GitHub repository.

This project uses custom voice data of my boyfriend calling my name ('mana') with angry or happy tones. While communication plays a significantly important role in a relationship, it gets arduous in some cases. Hence, my motivation behind this project is to reduce the amount of mental work to guess my next action based on the voice tone when my boyfriend calls my name. I can peacefully reply to him if he is happy. Otherwise, I have to be ready to argue. Additionally, it is my first attempt to use audio data, which I expect to learn a lot of new ideas and techniques.

2 Related Work

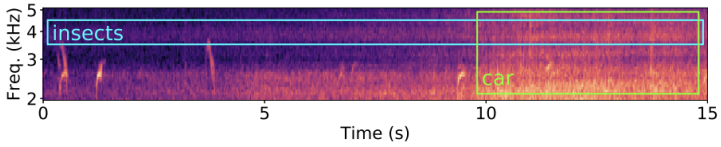
2.1 Audio Data Scaling

Speech classification has been one of the most popular fields in computer science and related fields in recent years due to the rapid emergence of chatbots and automatic voice recognition. Audio data or sound is wave and converted into signals in computers. Hence, it is built upon signal processing. Sine Fast Fourier Transform (FFT) was invented in the early 1990s, the signal conversion from a time domain into a frequency domain has established tactic in signal processing. However, signals are readable to computers but voice or sound is human-readable

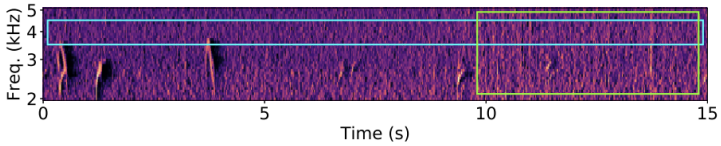
data, which makes it slightly different in how computers and humans detect 'signals'. As covered in Sec. 4.1.1, FFT does not detect sounds as we detect. Therefore, mel scale was invented in 1987 which weights heavier on lower frequencies as human ears detect sounds in lower frequencies better than in higher frequencies. Mel scale is in the unit of Hz, but it is a non-linear frequency transformation. In this project, both FFT and mel-scale will be applied to the dataset and observe how they are different.

2.2 Background Noise Reduction Method

Noise removal is essential in audio datasets, which could disturb the analysis. Cited from [2], PCEN is a recently emerged technique to reduce background noise. It is expected to detect small noise and make clearer audio data. In Sec. 4.1.3, PCEN will be applied to the dataset to see if the expected effect is observed.



(a) Logarithmic transformation.



(b) Per-channel energy normalization (PCEN).

Figure 1: Background Noise Reduction using Log-Mel and PCEN [2]

3 Methodology

3.1 Data Collection

This project uses a custom audio dataset collected by the author. I recorded 83 audio data of an early-20's male participant (initial: *TG*) calling my name (*mana*) with different emotional conditions (angry or happy). The data was collected over four days to prevent the data from being biased. The audio was recorded with an iPhone 12 internal microphone, whose internal microphone samples data at the rate of $f_s = 48000$ Hz [1]. The data is stored in the phone as m4a extension, so it was converted to wav extension using **pydub** version 0.25.1, a python package specialized in audio data[1].

Using wav files, each data was expanded so that all data has the same length. That is, take the longest data and add 0 padding to all other data to insert extra 0 columns. This is required to construct deep learning models as the size of the data has to be consistent.

[1]The environment for pydub requires **ffmpeg** and **ffprobe**. Please refer to my post on GitHub if anything goes wrong with the environment.

3.2 Implementation

This project flows from data collection, data preprocessing, and classification. In the section on classification, I constructed convolutional neural networks with the batch norm, pooling, and dropout. Due to the size of each data (∼10,000), it is highly recommended to use GPU. I trained models in google colaboratory, saved the trained model to my local directory, and loaded them to analyze on my laptop. Also, python is a primary programming language throughout this project, and all neural networks are written in PyTorch. The 83 data (angry - 44 data, happy - 43 data) was split into 45 training dataset, 19 validation dataset, and 19 test dataset using skit-learn library in python.

4 Data Preprocessing

4.1 Denoising

An important requirement for audio data is to denoise. In this segment, I focus on denoising by applying several methods in the dataset including high-pass filter (Sec. 4.1.2), and Per-Channel Energy Normalization (PCEN) (Sec. 4.1.3).

4.1.1 Data Exploration

Before diving into several methods of denoising, the original data is explored. Fast Fourier Transform (FFT) provides us with a nice power spectrum showing how loud the sound is. Shown in Fig. 2, the log-scaled power spectrum does not clearly describe the differences between an angry tone and a happy tone.

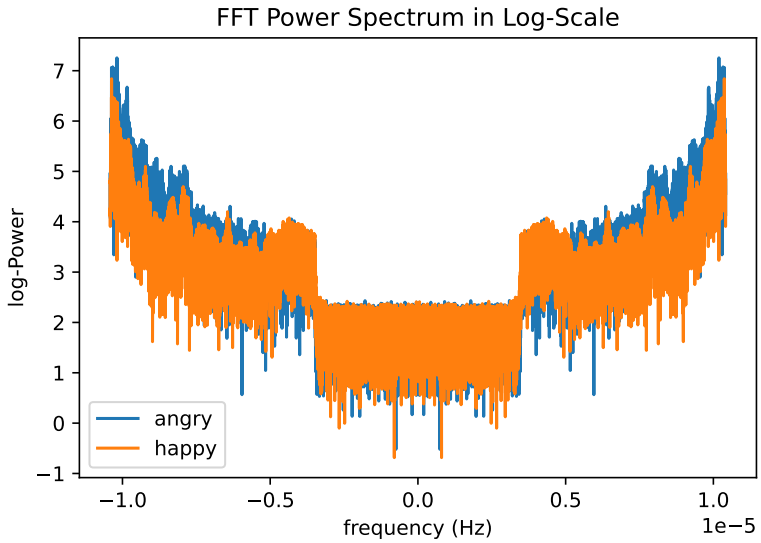


Figure 2: Log-Scaled Power Spectrum after Applying FFT

This is due to the nature of our human ears. Unlike 'machines', our ears detect sound on different scales, which is called Mel Scale. Mel scale is computed as in Eqn. 1.

$$m = 2595 \log_{10}(1 + \frac{f}{700}) \tag{1}$$

where m is the sound in mel-scale and f is the sound in frequency-scale in Hz. It should be noted that m is still a frequency unit but on a different scale. The Mel-Scale is distinguishable from FFT in that it emphasizes more on the lower frequencies than the higher frequencies because human ears detect better on lower frequencies.

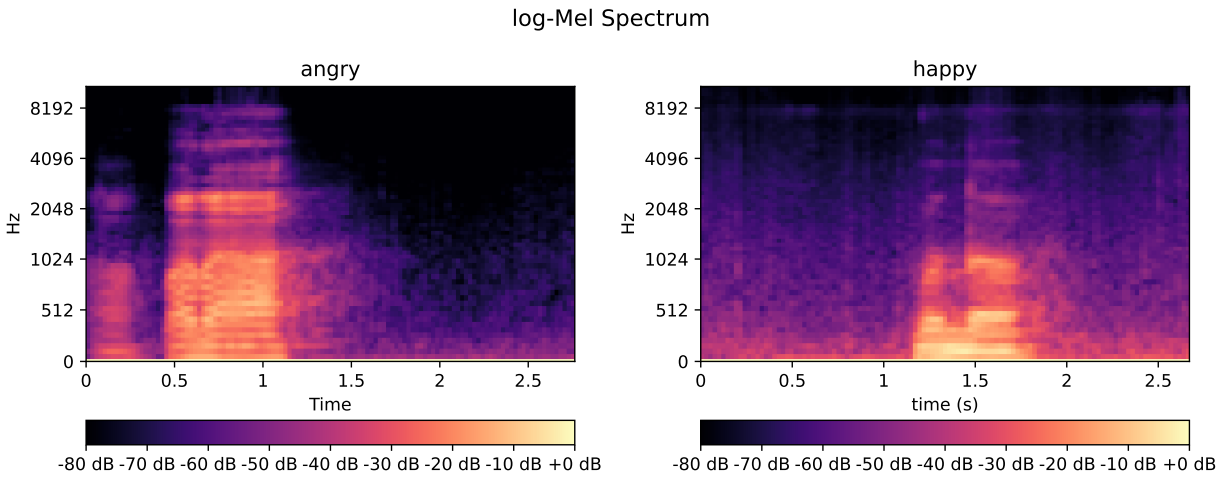


Figure 3: Log-Scaled Mel Spectrum

Fig. 3 illustrates examples of angry and happy tones in log-mel scale. This visually confirms that the angry tone adds more emphasis on lower frequencies than the happy tone as the yellow region ($\sim$0 dB) has wider. The finding in Fig. 3 is intuitive, yet not visible through the power spectrum of FFT.

4.1.2 High-Pass Filter

Given that the lower frequency is dominant in angry tone, data is distinguishable if it is more dependent on higher frequencies.

A high-pass filter is a type of filter that, for a given threshold W_n, a signal is passed if it is higher than W_n and cut off if it is lower than W_n. In this project, the threshold W_n was examined based on Fig. 3 where $W_n = 1000(Hz)$ is the line that distinguishes between angry and happy.

Shown in Fig. 4, the high-pass filter successfully exported the characteristics of each data. For example, the left plot compares the waves of angry voices between the original and the high-pass filtered data. Since the lower frequencies were cut off, the width of the frequency was shrunk, which shows a clearer separation between 'ma' and 'na'. Additionally, the center plot compares the happy voice. While the word separation was unrecognizable from the original waveform, it is now visible after applying the filter. Finally, the right plot illustrates the filtered data in an angry and happy tone. The width of frequency is so much different as the 'angry' tone dominates in a wider range of low frequency[2]

[2]The participant enjoyed saying my name with the angry tone for some reason.

The data in this project has uniqueness on lower frequencies, thus the high-pass filter was applied. However, if data is dominant in higher frequencies, the low-pass filter can be applied.

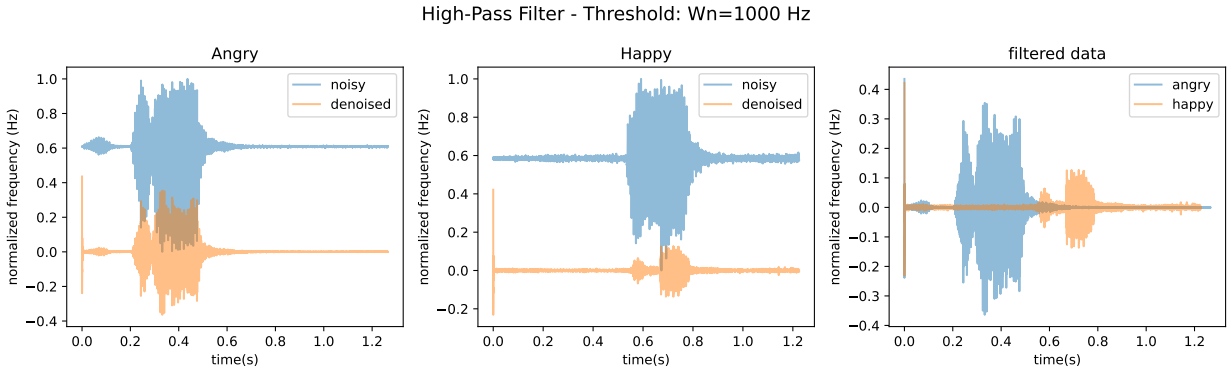


Figure 4: High-Pass Filter

4.1.3 PCEN

Next, Per-Channel Energy Normalization (PCEN) was applied to remove further noises. PCEN is an alternative denoising method that outperforms log-mel spectrum by combining dynamic range compression (DRC) and adaptive gain control (AGC). PCEN is widely used to remove noises from automatic speech and acoustic event detection due to its robustness. The robustness is rooted in the combination of DRC and AGC in that DRC lowers the variance of target sound and AGC reduces the background noise by applying low-pass filter [2].

Shown in Fig. 5, both plots are dark whereas Fig. 3 were colorful. This is due to noise reduction by PCEN. Interestingly, the lower frequencies in the happy tone is more dense than that in the angry tone, which seems to be a contradiction to our observation in Sec. 4.1.1 and 4.1.2.

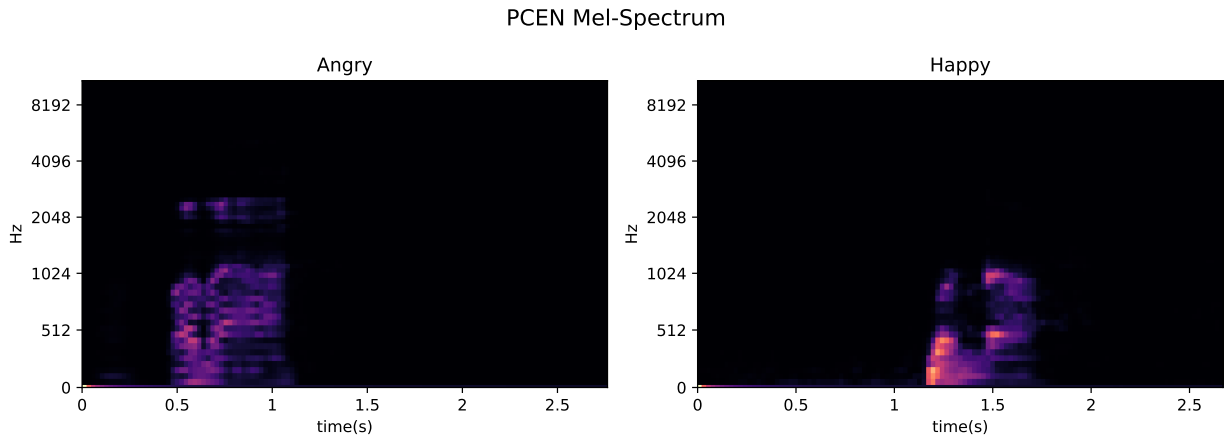


Figure 5: PCEN Mel Spectrum

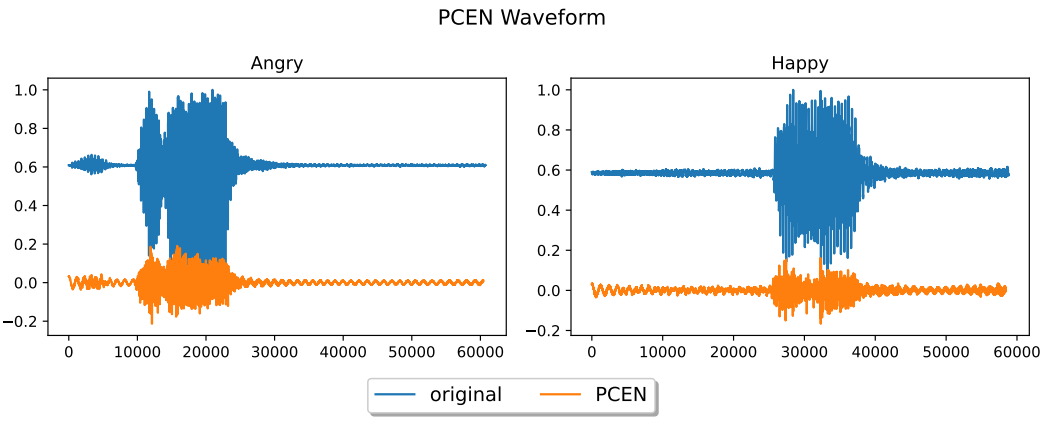


Figure 6: PCEN Waveform

However, the PCEN-ed data do not sound like a human voice, it is a robotic sound[3]. This may make sense if Fig. 6 is shown. Fig. 6 have very similar shapes of waves except for the tiny vibrations in silence. Due to the recording settings, the silence was not completely silent, where very slight noise was happening in the background. By normalizing the data, such small changes caused the vibration-like sound.

5 CNN Results

In this section, the audio data was fed into convolutional neural networks (CNN) to classify the emotion based on the tone. CNN was chosen because waves are spatial data. And Audio data is essentially patterns where the waveform varies as people open and close their month. Therefore, CNN can identify the characteristics of waves if it learns from spatial information, which CNN is suitable for. Area Under Curve (AUC) score was used as a metric to measure how well the models classify the emotion. AUC is the area under the ROC curve and takes a value from 0 to 1, where 1 is the maximum. Binary cross entropy loss (BCE loss) is used to compute the loss throughout the training, and Adam was applied as an optimizer. The initial learning rate was set to $lr = 0.0003$ and OnceCycleLR was used to schedule learning steps.

In this project, the effect of pooling on audio data was also determined by comparing the results of max pooling and average pooling. Fig. 7 describes the architecture of CNN used in this project. Feeding the data in waveform into the model, the data goes through the three blocks of Conv1d, ReLU, BatchNormalization, and Dropout and passes through linear and sigmoid layers to omit the binary outputs. The CNN model with MaxPooling was trained for 100 epochs and the model with AvgPooling was trained for 80 epochs.

Once the training was completed, the loss functions during training and validation were plotted to see if the models were overfitted or underfitted. Shown in the left and center plots in Fig, 8, both training and validation loss decrease as the training proceeds, which is a great sign that the models were properly trained. According to the best AUC scores

[3]The original sound and PCEN-ed sound are available in GitHub

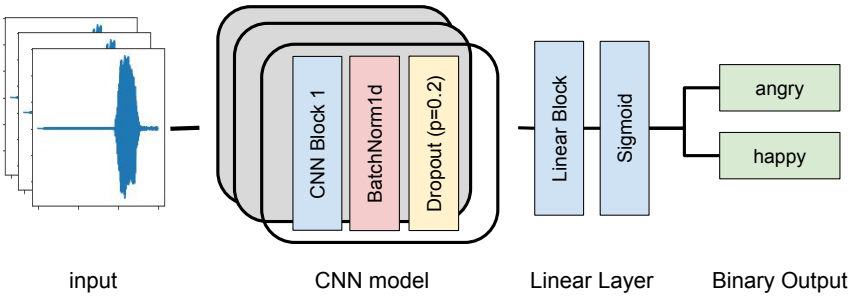


Figure 7: CNN Model Architecture

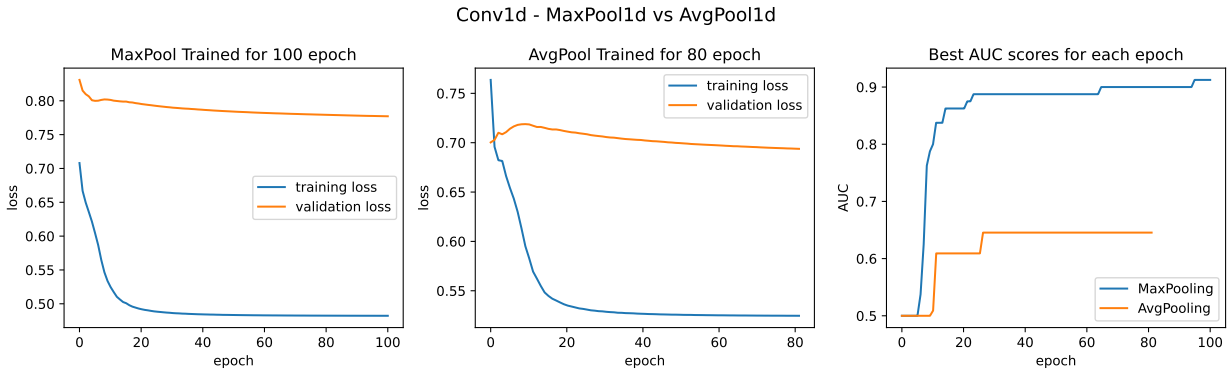


Figure 8: CNN Training / Validation Results

for each epoch in Fig, 8, the model with MaxPooling measured the higher AUC score trend than the model with AvgPooling. Indeed, the highest AUC score marked for MaxPooling was 0.913, whereas the model with AvgPooling scored 0.645 (Table 1). Therefore, the CNN model with MaxPooling classified the angry-vs-happy emotions better than the model with AvgPooling.

AUC score	
MaxPool1d	0.913
AvgPool1d	0.645

Table 1: Conv1d AUC scores

Pooling is taking a piece of data and applying an operation to extract information. In the case of MaxPooling (kernel size = 2), the larger data out of two data points is kept. Similarly, AvgPooling (kernel size=2) computes the average of two data points and keeps the average value. From the experiments, it was found that MaxPooling worked better than AvgPooling in the dataset since MaxPooling preserves the characteristics of waveforms whereas AvgPooling vanishes them.

6 Discussion

This project aims to classify the speaker's emotion, angry or happy, based on the voice recording of a single word with 44 angry audio data and 43 angry audio data. From experiments, I found that CNN with MaxPooling outperforms CNN with AvgPooling as it maintains the overall shape of waveforms scoring 0.913 AUC score. However, there is some space to improve the models and implement thorough examinations. First, Fig. 8 shows that the training and validation losses are still decreasing, namely the training could last longer. By implementing early stopping to CNN with MaxPooling, it is feasible to aim for a higher AUC score. Another potential improvement could be the number of datasets. In this project, I recorded a data provider's voice over a couple of days to collect 87 data. However, only 45 data were used to train the model. While 0.913 AUC score is considered to be high, it is feasible to reach a higher AUC score if more data is available.

7 Conclusion

The goal of this project was to study the waveform and denoising methods and to classify the emotion based on the audio data using deep neural networks.

In Sec. 4.1.1, we observed that the Mel spectrum in log scale exposed some characteristics of voice tone which were not clearly visible from FFT. From Fig. 3, it was examined that angry tone is dominated by lower frequencies. Then, the high-pass filter was applied to filter out lower frequencies (Sec.4.1.2). By filtering out some lower frequencies, the hidden shape of waveforms was exposed in Fig. 4 such that the waveform of angry has two blocks (one corresponds to 'ma' and another corresponds to 'na') and the waveform of happy has two distinguishable blocks. Finally, PCEN was applied to remove further background noise (Sec. 4.1.3). Although PCEN was intended to denoise small yet noisy background, it did not work well for this dataset as it converted the human voice to robotic sound.

Then, CNN models with MaxPooling and AvgPooling were constructed to classify if the speaker is angry or happy from the voice tone (Sec. 5). By applying BatchNorm and dropout layers, the models were trained for 80 - 100 epochs. Based on the AUC scores, the CNN model with MaxPooling, which scored 0.913, performed better than the model with AvgPooling, which measured 0.645.

For simplicity, this project uses voice data of saying one word and aims to classify binary emotions. However, it is feasible to apply a more complex dataset to classify various classes. While many applications could be thought of, one application could be anomaly detection using the similar algorithms used in this project. In this project, I fed both angry and happy voices into the model, but I could also feed only happy data into the model. If the model finds data that is never seen, then the model can immediately detect that something wrong with the speaker. Because the voice highly reflects our emotions, such an application might be useful for nurse call systems in hospitals where patients are in serious, life-threatening situations or need some random errands.

Last not but least, recalling my motivation behind this project, this classifier is not yet enough to reduce my mental effort to detect if my boyfriend is happy or angry based on his voice tone since it is still in the model-based. If time allows, I would like to improve the

model with MaxPooling and create a chatbot or web application that gives instant emotional classification from his voice message. I hope this will enhance our healthy relationship.

8 Acknowledgement

I respect and express profound gratitude to Professor Mauricio Santillana, for giving great lectures in signal processing which created the base of this project in PHYS4606 at Northeastern University. I am also grateful to *TG* for calling my name 83 times with different emotions over a couple of days.

References

[1] iPhone 12 - Technical Specifications — support.apple.com. https://support.apple.com/kb/SP830?locale=en_US. [Accessed 26-Apr-2023].

[2] Vincent Lostanlen, Justin Salamon, Mark Cartwright, Brian McFee, Andrew Farnsworth, Steve Kelling, and Juan Pablo Bello. Per-channel energy normalization: Why and how. *IEEE Signal Processing Letters*, 26(1):39–43, 2019.