Generation of Stereo Audio from a Mono Source Using a Nearest-Neighbor Algorithm

Project Members: Ethan James

Motivation

I will attempt to reproduce the result of a research paper [1] that proposes a model used for generating a left-right (stereophonic) music audio file from a single-channel (monophonic) source. This model allows a to input a single-channel music recording and receive a stereo recording that gives an idea of what the original source may have sounded like if it were recorded in stereo. I chose to pursue this project for a couple of reasons:

- Most historical music recordings, including almost all music recorded before the 1960s when stereo audio became standard, were recorded as mono. While no perfect solution will ever exist to generate a stereo track from these mono sources, a well-engineered solution may produce a stereo track with reasonable accuracy.
- I personally plan to pursue the field of Digital Signal Processing (DSP) as a career. This project involves extensive transformation of the mono audio track before applying any machine learning algorithm, and the robustness and quality of the signal processing directly correlates with the quality of the algorithm's output. I'm hoping that this project will allow me to improve my DSP skills and background.

Methodology

The research paper gives no specific details on signal processing or machine learning frameworks used; it only contains the mathematical formulas used. I'll need to take a deeper look at each step to produce an identical implementation, but the basic outline of the work is as follows:

Training:

- For a single training data point (A stereo audio track), take a spectrogram of the audio at a random timestamp and use the left and right tracks to generate the "Parametric Stereo Coding" (PS Coding) of the track - the PS encoding maps some frequency components of the audio to a position in the left-right stereo field.
- Repeat approximately 500,000 times to train a nearest-neighbor model.

Evaluation:

- Take spectrograms of the validation audio track at all timestamps, and then find the PS enoding of each spectrogram. Then, find the K closest PS encoding data points in the nearest-neighhbor model, and use the known left-right stereo position of those K points to predict the stereo position to estimate the stereo position of each PS encoding.
- Decode the PS encoding to stereo spectrograms, then use some IFFT (Inverse Fast Fourier Transform)-based algorithm to convert the spectrograms back to a time-domain audio signal.

This overview essentially outlines my work: I'll need to do some signal processing to encode/decode the audio to allow the nearest-neighbor model to work on the audio (likely with python signal processing libraries like

`scipy.signal`, and of course `numpy`). I'm less sure about what machine learning framework may be used for a nearest-neighbor model of this scale, but `sklearn.neighbors` seems promising.

Datasets are extremely readily available, as almost all modern music recordings are in stereo. I'll likely use this dataset [2] that contains a million music audio tracks.

Evaluation

The paper proposes an evaluation metric for a mono-to-stereo conversion model. It's my understanding that this evaluation metric simply computes the distance between each PS coding location in the left-right stereo field and reports the sum of the distances. I'll implement this metric for training and evaluating my model.

However, the quality of a stereo track is highly subjective, as is music itself. In my final report, I will provide a subjective evaluation (my opinion) of the model's performance using various validation audio tracks. I'll also compare the outputs of the model for different parameters if time permits for tweaking of the model.

Timeline

By midterm report:

- Implement Parametric Stereo Encoding/Decoding algorithm in Python
- Train a nearest-neighbor model using a fraction of the song dataset (possibly %5 of the available data) and produce an output that provides some evidence that the model is operational (i.e. the output isn't garbled noise)

By final report:

- Fully train the model using 500,000 data points - this is the amount of data used in the original paper
- Provide a detailed and quantitative explanation of how my model performs similarly to the model evaluated in the original paper

References

[1] - MONO-TO-STEREO THROUGH PARAMETRIC STEREO GENERATION - https://arxiv.org/pdf/2306.14647

[2] - Million Song Dataset - http://millionsongdataset.com/