SMART EQUALIZER: Utilizing Machine Learning To Automate Audio Circuit Equalizing

Hagan Beatson, Alex Chacko, Ryan Lenihan, Michael Richards

CAPSTONE Senior Design 2020-2021, Department of Electrical and Computer Engineering, University of Delaware

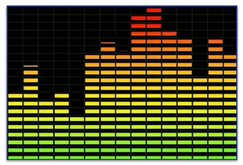
GitHub **Report**

PURPOSE and PROBLEM

Audio mixing is, in simple terms, the art of controlling & manipulating frequencies in an audio recording to achieve a desired effect for the listener. Mixing is what allows for dialogue in movies to be as clear as day even during busy action sequences, or what allows for the bass in a hip hop song to blow out the subwoofer of a cheap car.

Controlling the various frequencies that go into a song, or any other piece of audio, can prove to be extremely influential on the overall experience

Fig 1: A conventional analog equalizer



We saw an opportunity in analog audio hardware with digital preprocessing. Our project, ***Smart Equalizer***, aims to use genre classification to automatically equalize an output circuit.

By using a neural network to identify the genre of a currently playing song, a microcontroller can be instructed to adjust power levels on different frequency ranges. This automatic equalizing should be fast and reliable.

This is where Smart Equalizer comes into play, to find a common ground between the audiences of audiophiles and casual listeners.

Fig 2: Visual representation of audio signals

- Professional sound mixing can vary wildly in quality
- Many people feel that self-equalizing can yield more desirable audio mixing
- Tuning equalizers yourself can be daunting and lead to lots of unnecessary testing and steps to help ensure optimal experience

GOALS and REQUIREMENTS

- Lower the bar for entry for audiophiles
- Reduce the learning curve involved with equalization
- Give users a simple and customizable experience to optimize their music
- Support multiple genres and custom genre profiles
- Have a seamless end-to-end system integrating a neural network and hardware-level specifications
- **Make listening to music a more enjoyable experience for everyone**

APPROACH and METHODOLOGY

Since this project focuses on both hardware and software components, we decided to split into three teams to handle each section (software, firmware, hardware).

- *Software* focused on the machine learning/classification pipeline.
- *Firmware* focuses on using data from the PC neural network to control potentiometers on the output circuit using an Arduino and its associated libraries
- *Hardware* focused on creating a circuit to divide the sections of an audio signal (low, mid, high) for equalization.

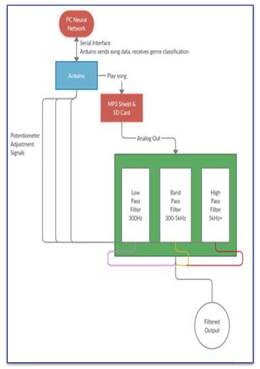


Fig 3: Flow chart of proposed workflow

DESIGN

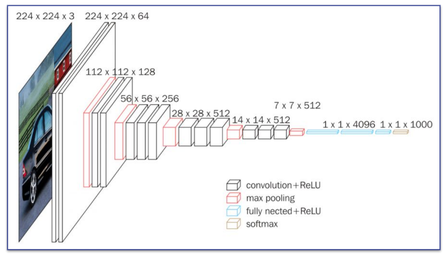
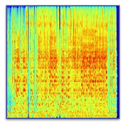


Fig 4: Diagram of VGG16 CNN architecture, backbone of our predictive pipeline

Our pipeline utilizes a modified *VGG-16 convolutional neural network* in predicting the genre of a given audio file. The file is pre-processed into a **spectrogram**, which visualizes the songs various frequencies and amplitudes. Then, based on this image, a deep learning prediction is created to be fed into the circuit itself.



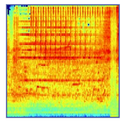


Fig 5: Generated spectrograms

The Arduino controls an array of MAX5411 logarithmic potentiometers. Each chip has two resistors, so we use one resistor for the left and right channels each. An SPI interface allows us to set up to 10kΩ in dB.

The design of the circuit is based on the Sallen-Key Butterworth filter. This schematic was chosen based on its ability to cut off quickly at the desired frequency, in addition to not having extensive ringing on the ends of the signal. We separated the signal with a low-pass, band-pass, and high-pass filter. We determined the values of the passive components using manual calculation, and simulated the output in EAGLE to confirm its accuracy.

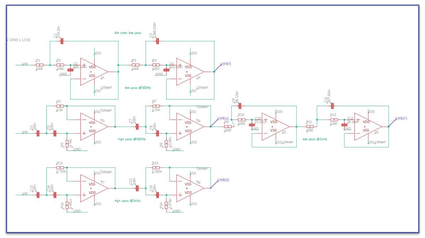


Fig 6: Butterworth filter design for each filtering circuit

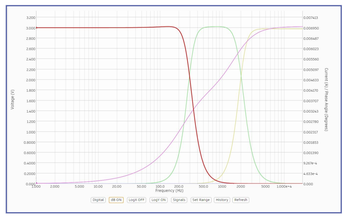


Fig 7: Frequency cutoff for each circuit in simulation

COMPONENT and SYSTEM TESTING

Results of comparative testing with the modified VGG16 network against other deep learning approaches shows robust performance.

This means that the CNN used in our pipeline is **5% more accurate** than the next most accurate solution across all 8 supported genres.

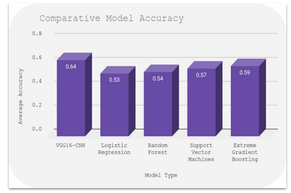
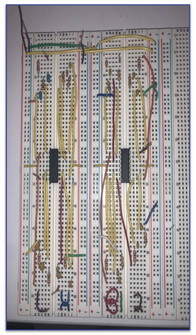


Fig 8: Comparative benchmark of deep learning approaches to classification

Circuit Demo



To prove the circuits would work accurately, we made breadboards of each as a proof of concept. Due to the limited availability of passive through hole components, we had to adjust values for the build. After doing so, each filtering circuit worked a expected, and we moved to the PCB layout.

Fig 9: Each filtering circuit breadboarded

PROJECT STATUS

- Software - Finish the prediction pipeline, add support for command-line user interface, add support for setting custom profiles and editing existing profile EQ values to maximize usability
- Hardware - Get PCB from JLC PCB, and test to verify they work properly
- Firmware - Using new breakout boards, test capability of playing signal from Arduino to speakers.

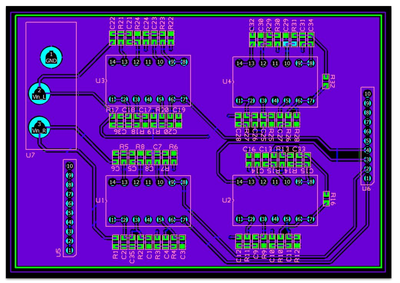


Fig 10: PCB layout for filtering circuits