Signal Processing for a Cause

Emily Prentice
Bethel Tech Data Science Student

BETHEL SCHOOL
OF TECHNOLOGY

A Little About Emily

- Current Employment
  - Parkview Christian Academy, 1st Grade Teacher / Worship Team Leader

- Prior Education
  - Bethel School of Supernatural Ministry
  - Bethel Music College, Musician Track
  - Carthage College, B.A. Biology

- Volunteer Experience
  - Harvest Chapel, Keyboardist & Audio Engineer
  - Convoy of Hope, Missions Trips to Haiti & Honduras
  - Paw Paw IL Fire Department, EMT-Basic

Honduras, 2018

Project Background - The Audio Engineer Experience

- Much of what audio engineers do is clean up audio signals before they are outputted to the audience
- The cleaner the audio, the more enjoyable the listening experience
- Microphones often pick up more than the desired signal, creating muddy audio and unpleasant listening experiences
- Noise gates can be used to counteract this, but they are far from perfect

Source: harmonycentral.com

Project Background - Problem to be Solved

- If a microphone could detect the desired signal at the source, it would create less work for the audio engineers to detect the desired audio signal at the source.
    - A singer’s microphone would only capture their voice
    - Drum mics would only capture drum sounds
- Through the use of machine learning and deep neural networks, microphones could be implanted with A.I. that could detect the desired audio signal
- From there, only the desired audio signal would be sent on to audio interface and eventually onto the audience

Project Background - Implications

- Immediate
  - Faster and easier clean up of various audio signals
  - Clearer, more concise sounds
  - Overall improved listening enjoyability for the listener, especially for music and/or speaking engagements

- Long-Term Thinking
  - These microphones could be used by / donated to those in less than ideal circumstances who need to get a clear message across
    - Missionaries, doctors in third-world clinics, natural disaster relief efforts

Methods - Data

- **Data used to train the model:**
  - **Male Vocal Audio - 250 wav file clips**
    Average Clip Length: 8.42 sec
    Source: VocalSet: A Singing Voice Dataset | Zenodo
  - **Female Vocal Audio - 250 wav file clips**
    Average Clip Length: 7.07 sec
    Source: VocalSet: A Singing Voice Dataset | Zenodo
  - **Drum Kit Audio - 250 wav file clips**
    Average Clip Length: 2.39 sec
    Source: Kaggle: Drum Kit Sound Samples

Methods - Data

- Data ran through the model:
    - Recordings from Worship Team Rehearsal at Harvest Chapel
        - Shayne Audio (Male Vocal)
        - Chayce Audio (Male Vocal)
        - Jess D Audio (Female Vocal)
        - Jess O Audio (Female Vocal)
        - Drum Kit Audio
    - Each type of recording consisted of 5 samples, 183 sec (3.05 mins) in length

Methods - Workflow

1. Convert audio data to waveform
    - This is where the audio data is converted into a numerical representation

2. Transform waveform to spectrogram
    - A spectrogram can be defined as a, “picture of sound”
    - This conversion will allow for the use of a convolutional neural networks
    - This will allow for the training of a convolutional neural networks

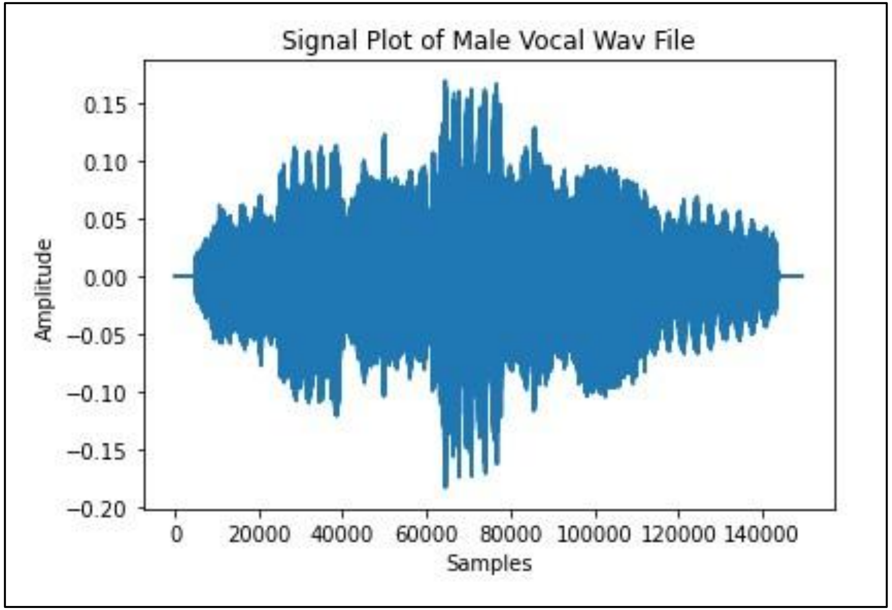
3. Classify the desired signal
    - Once the model is trained, sliding window classification will be used
    - The larger audio clips will be ran through the neural network and specific times the desired audio signal is present will be recorded

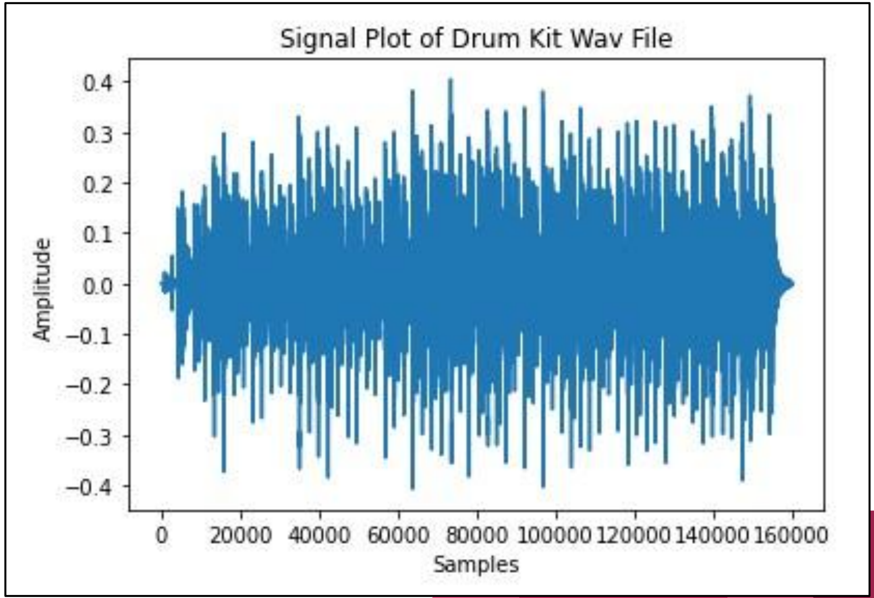
Methods - Languages & Packages Used

- Language:
  - Python via Google Colab

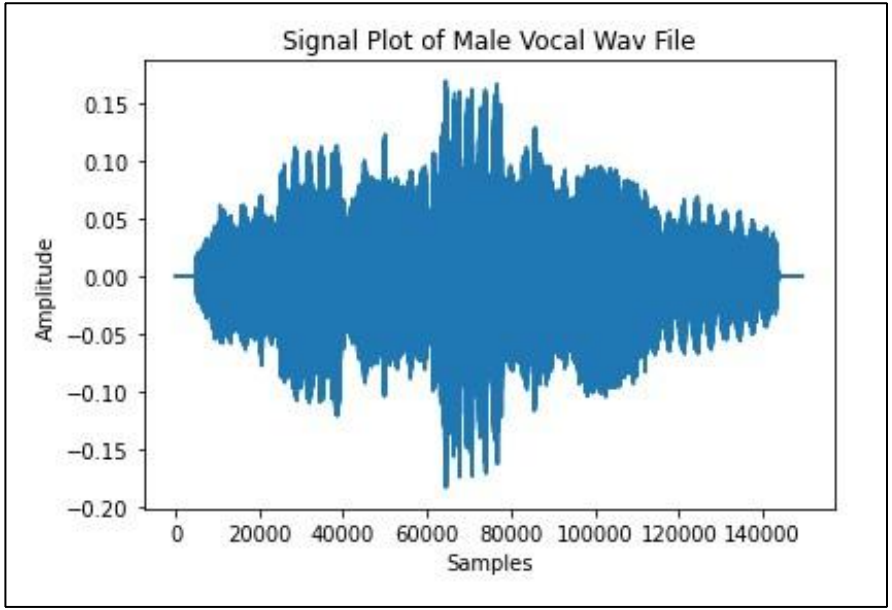
- Packages Used:
  - Tensorflow
    - Audio processing
    - Running Cuda enabled GPU
    - Building our deep learning model
  - Matplotlib
    - Visualizing results
  - Os
    - Directory navigation

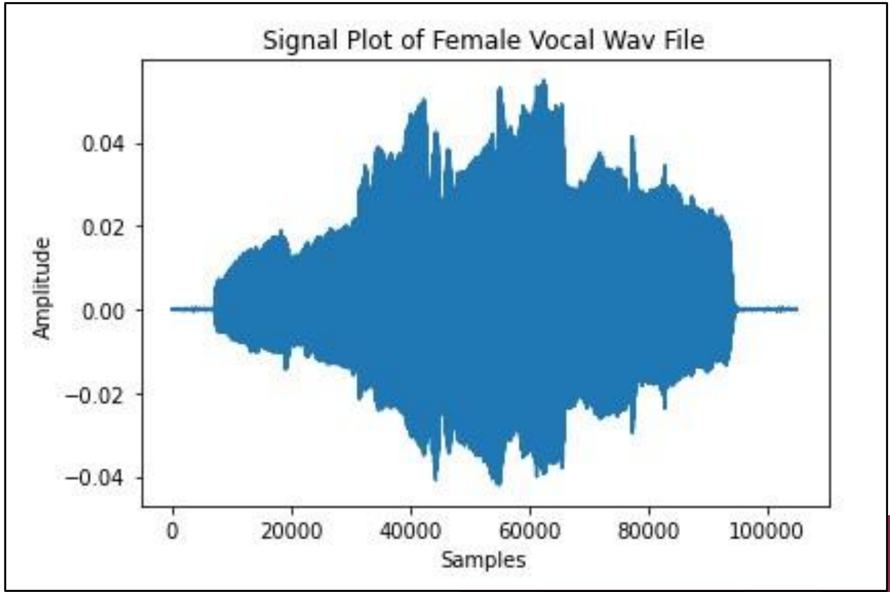
Results - Signal Plots of Male Vocal & Drum Kit Wav Files



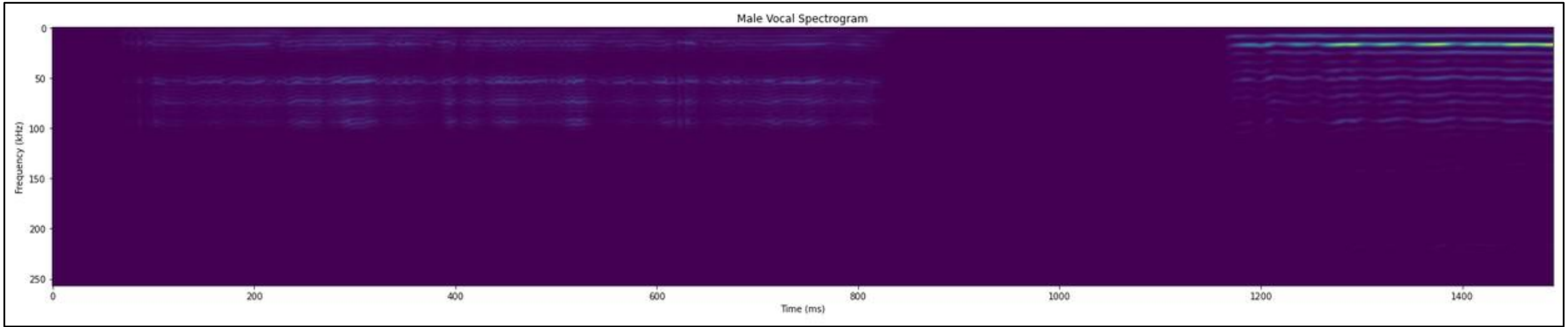


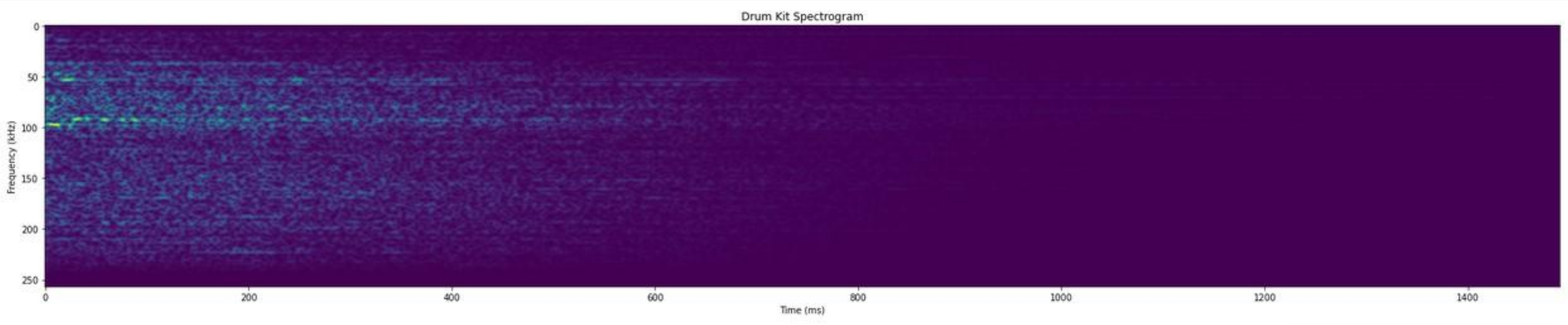
Results - Signal Plots of Male Vocal & Female Vocal Wav Files



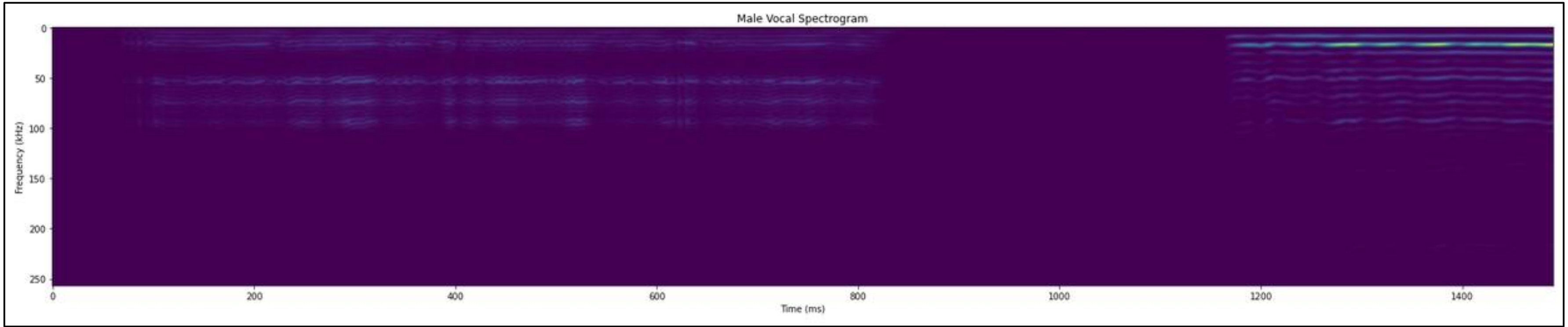


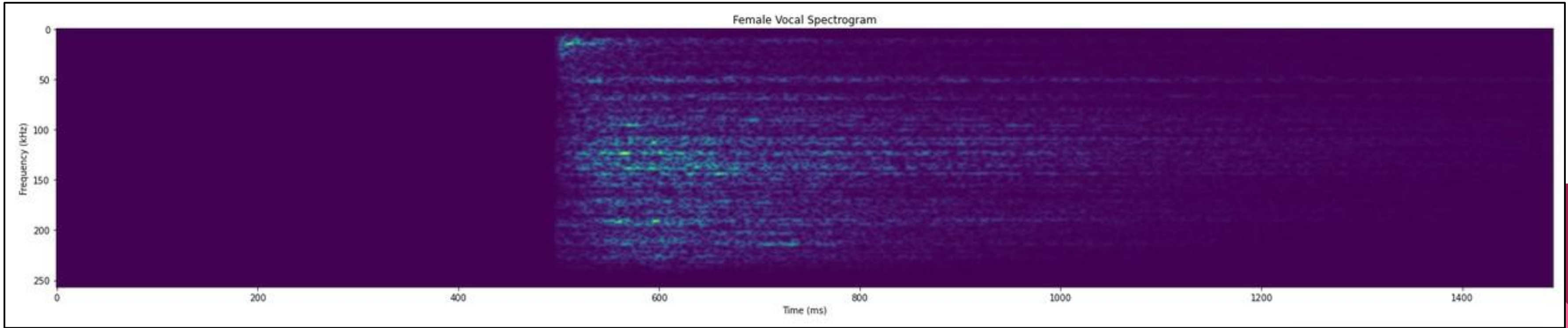
Results - Male Vocal & Drum Kit Spectrogram





Results - Male Vocal & Female Vocal Spectrogram

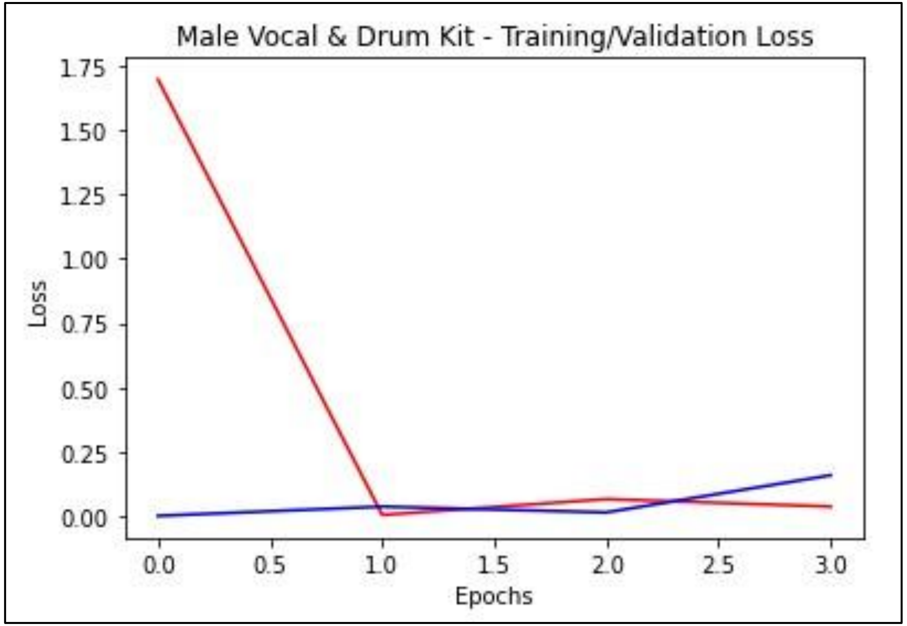


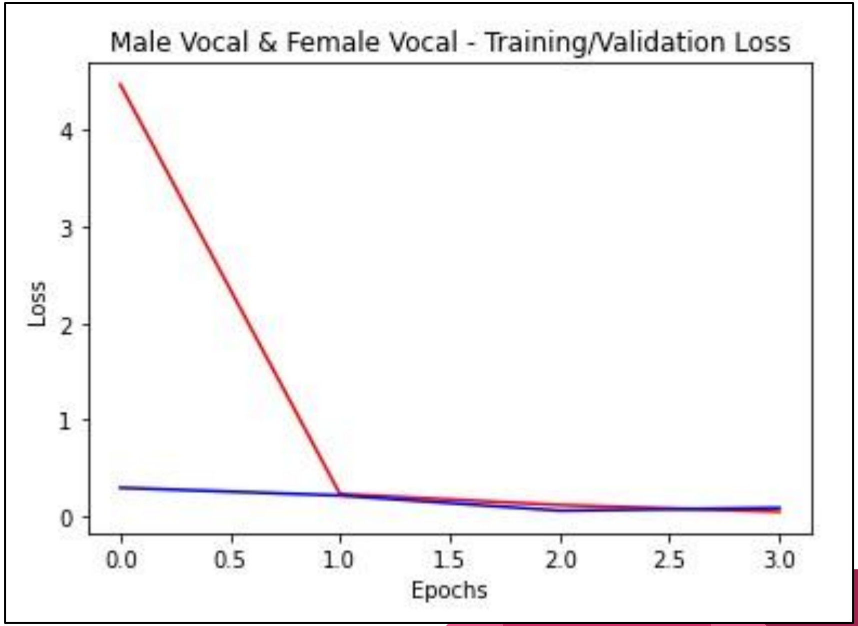


Results - Training & Validation Model Epoch Results

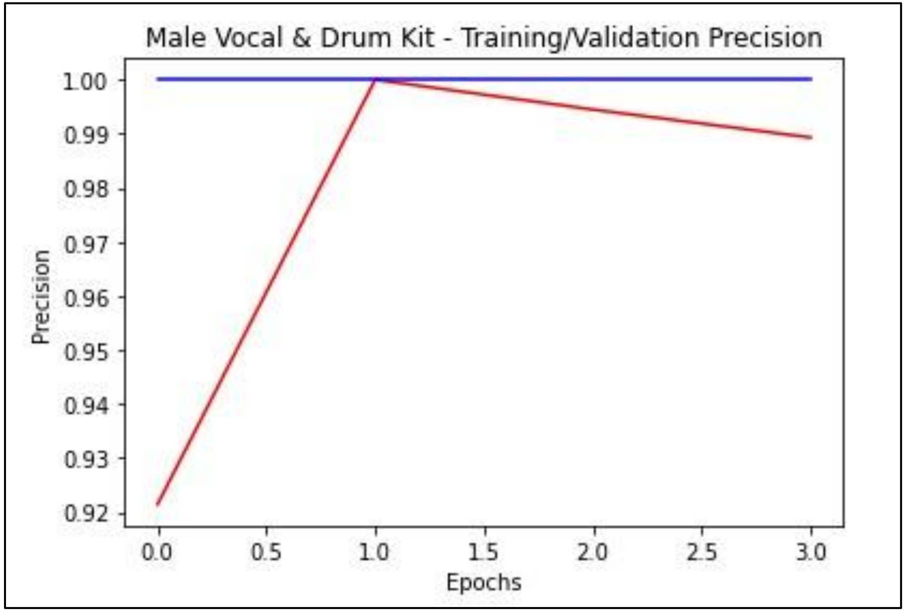
Male Vocal & Drum Kit	Training Loss	Training Precision	Training Recall	Validation Loss	Validation Precision	Validation Recall
Epoch 1	1.694	0.922	0.893	0.002	1.000	1.000
Epoch 2	0.005	1.000	1.000	0.038	1.000	0.968
Epoch 3	0.067	0.995	0.984	0.015	1.000	0.984
Epoch 4	0.038	0.989	0.995	0.159	1.000	0.986
Male Vocal & Female Vocal	**Training Loss**	**Training Precision**	**Training Recall**	**Validation Loss**	**Validation Precision**	**Validation Recall**
Epoch 1	4.474	0.790	0.811	0.292	0.949	0.849
Epoch 2	0.228	0.966	0.918	0.212	0.909	0.938
Epoch 3	0.114	0.984	0.953	0.056	1.000	0.968
Epoch 4	0.045	0.995	0.979	0.084	0.960	1.000

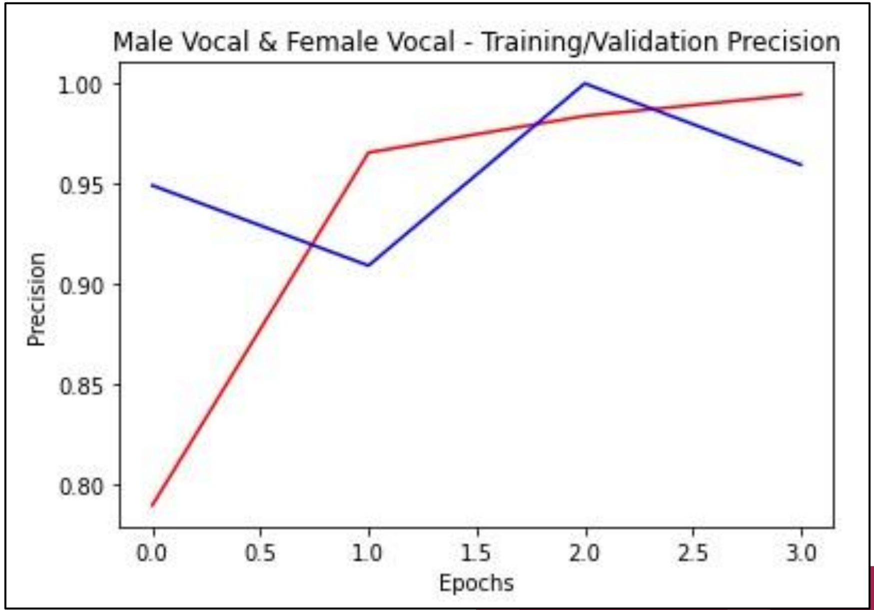
Results - Model Training Loss Plots



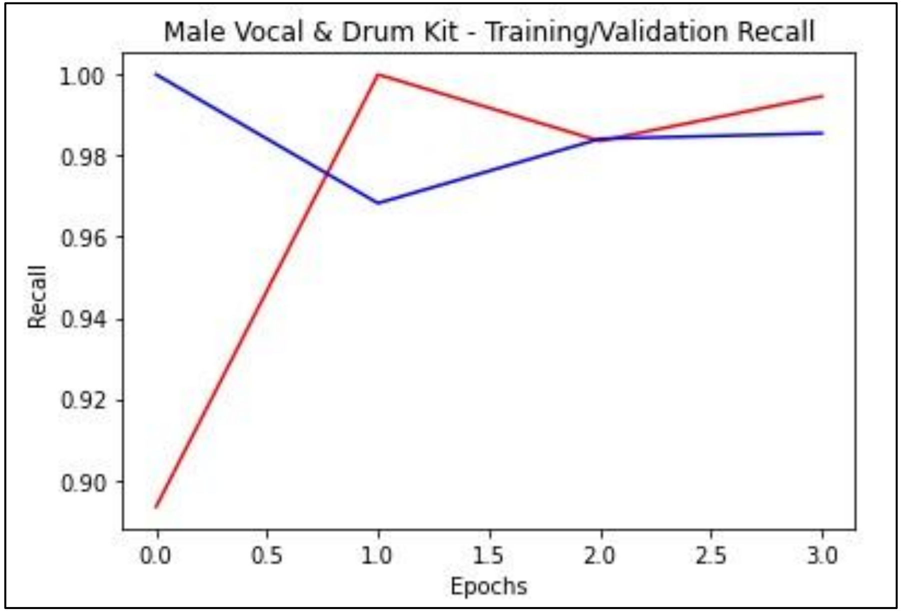


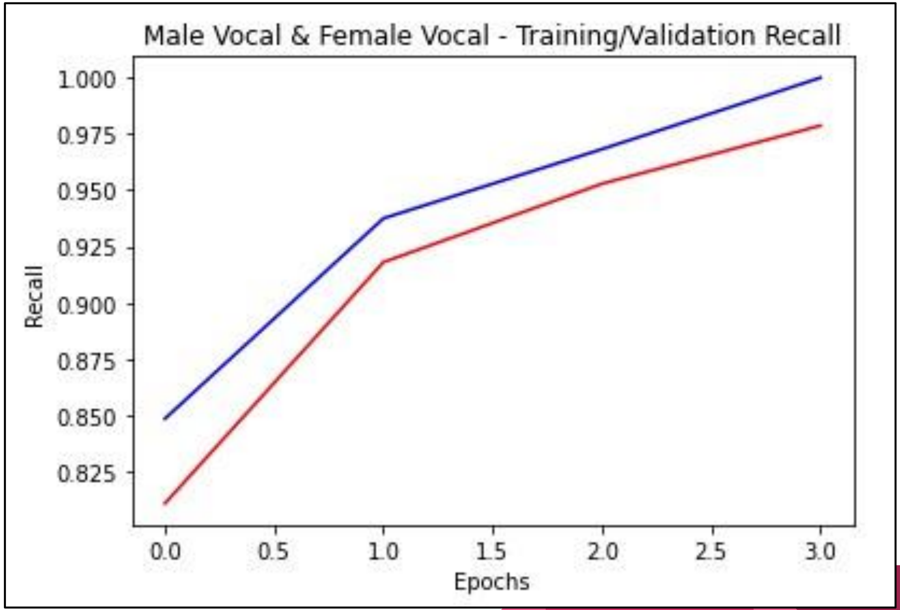
Results - Model Training Precision Plots





Results - Model Training Recall Plots





Results - Predictions on a Single Clip

Drum Kit & Male Vocal

```
yhat
[0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1]

y_test.astype(int)
array([0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1])
```

Female Vocal & Male Vocal

```
yhat
[0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1]

y_test.astype(int)
array([0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1])
```

Results - Signal Density

Female Vocal Signal Density

Recording	Female_Vocal_Sign
Chayce Audio (1) 1.r	11
Chayce Audio (2) 1.r	10
Chayce Audio (3) 1.r	7
Chayce Audio (5) 1.r	10
Chayce Audio (4) 1.r	5
Shayne Audio (1) 1.r	13
Shayne Audio (2) 1.r	16
Shayne Audio (3) 1.r	7
Shayne Audio (4) 1.r	11
Shayne Audio (5) 1.r	11
Jess D Audio (1) 1.m	9
Jess D Audio (2) 1.m	6
Jess D Audio (3) 1.m	6
Jess D Audio (4) 1.m	4
Jess D Audio (5) 1.m	1
Jess O Audio (1) 1.m	8
Jess O Audio (2) 1.m	12
Jess O Audio (3) 1.m	6
Jess O Audio (4) 1.m	4
Jess O Audio (5) 1.m	7

Drum Kit Signal Density

Recording	Drum_Kit_Signal
Chayce Audio (1) 1.r	2
Chayce Audio (2) 1.r	7
Chayce Audio (3) 1.r	4
Chayce Audio (4) 1.r	5
Chayce Audio (5) 1.r	1
Drum Kit (2) 1.mp3	1
Drum Kit (1) 1.mp3	1
Drum Kit (3) 1.mp3	4
Drum Kit (4) 1.mp3	4
Drum Kit (5) 1.mp3	2
Shayne Audio (1) 1.r	2
Shayne Audio (2) 1.r	0
Shayne Audio (3) 1.r	7
Shayne Audio (4) 1.r	7
Shayne Audio (5) 1.r	5

Summary

- Both deep learning models are working with precision and accuracy
  - Drum Kit & Male Vocal:
    - Training Partition - 99% recall, 99% precision
    - Validation Partition - 99% recall, 100% precision
  - Female Vocal & Male Vocal:
    - Training Partition - 98% recall, 99% precision
    - Validation Partition - 100% recall, 96% precision

- One model is able to accurately detect the amount of drum kit signal density

- The other model accurately detects the amount of female audio signal.

Conclusion

- Both models and their data can be used to engineer A.I. for the microphone
- As more data is collected, more models will be able to be created, which will lead to greater efficiency in detecting the desired signal
- This will, inturn, increase the A.I.'s ability within the microphone to detect the desired audio signal and only allow that signal through

Questions?

References

- Nicholas Renotte - “Build a Deep Audio Classifier with Python and Tensorflow” (April 16, 2022)
  - Youtube Video
  - Github
- Seth Adams - Audio Classification (November 8, 2020)
  - Github
- Valerio Velardo - Audio Signal Processing for Machine Learning (June 18, 2020)
  - Youtube Playlist