Machine Learning Model for Instrument Identification

APS360 Final Report

Abhay Verma	1005866966
Javiera Bao	1007571735
Kieran Kasha	1005910392
Rishik Kumar	1004069214

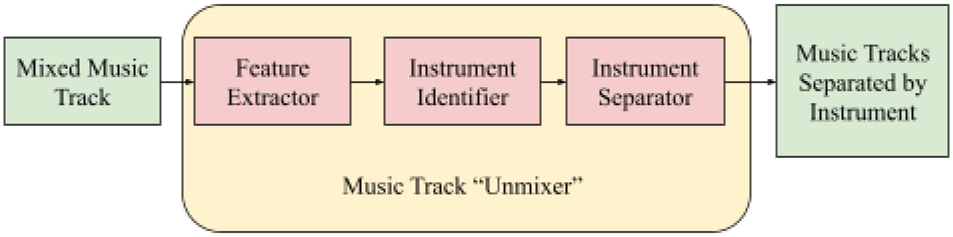
April 9th, 2021

Word Count: 2496/2500

1.0 Introduction

The goal of the project is to identify the instrument featured in a piece of music. The motivations behind this project are for automatic instrument track separation for music production, mixing, and karaoke. Automatic instrument identification has become increasingly important as computing has advanced in the past few decades. Until recently, music mixers would have to consciously listen to hours of audio to identify sections that contained certain instruments. Automating this process could reduce the cost of music production and allow creators to focus on the creative aspects of their work. Additionally, instrument classification is also difficult to accomplish accurately for those with little musical experience. Machine learning can optimize this task, allowing artists to focus on the creative parts of the music production process. Thus, this project will focus on the classification task as attempting to automate all parts of the track separation process (Figure 1) is a graduate-level problem [1].

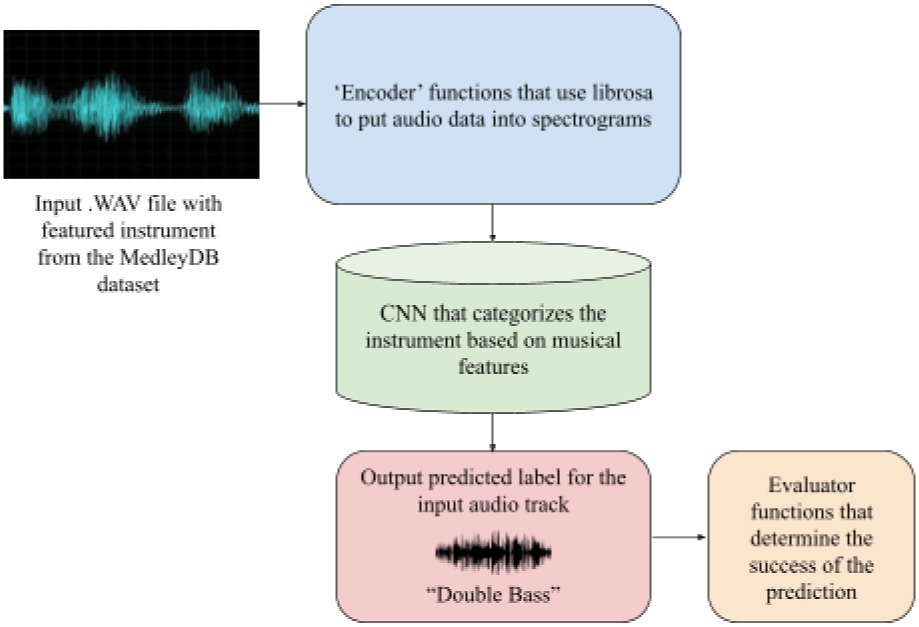
Figure 1: The Instrument Track Separation Process [2]



2.0 Diagram of Project Core Idea

Figure 2 shows machine learning to be an appropriate tool for instrument classification. Audio files are converted into spectrogram images whose features can be extracted to train a classifying CNN.

Figure 2: High-Level Model Diagram


'Encoder' functions that use librosa to put audio data into spectrograms
Input .WAV file with featured instrument from the MedleyDB dataset
CNN that categorizes the instrument based on musical features
Output predicted label for the input audio track
"Double Bass"
Evaluator functions that determine the success of the prediction

3.0 Background and Related Work

Automatic instrument classification has been a prevalent challenge since the early 1990s [3]. Some of the earliest work in the field was done with computational auditory scene analysis (CASA), a non-machine learning approach utilizing filters and grouping algorithms to make predictions [4]. These methods suffered in terms of limited algorithmic complexity and an inability to analyze data with any amount of background noise [4]. More recent work has utilized CNNs to tackle this problem due to the propensity of these models to efficiently learn the nuanced features of audio files. A 2019 project by Nadim Kawwa utilized a CNN to identify instruments in the NSynth dataset [5]. This dataset consists of single-note audio files from synthesized MIDI instruments [6]. The sufficient performance of this model is what led us to develop our own CNN for instrument classification.

4.0 Data Processing

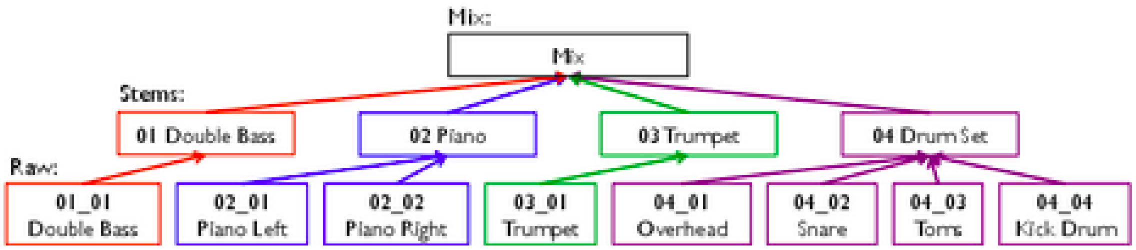
The majority of projects in this field utilize single-note audio files separated by instrument. Although these databases could have been used to develop our model, they were not sufficient for the real-life problem we were trying to solve. Music producers would most likely be using tracks with melodies consisting of multiple notes played by physical instruments. Resulting in increasing the difficulty of our project, a more complex dataset required more work to preprocess data and train our model but would allow our model to better generalize on new, realistic data.

4.1 Database

We believed the MedleyDB database would be sufficient for this task [7]. MedleyDB is a restricted access database, containing 196 songs in .wav format. It was necessary to request access to its version 1.0 (122 songs) and 2.0 (74 songs) through the Zenodo platform [8].

For each song, there is a folder with files of three types: mix, stem, and raw. A mix is composed of a set of stems, and each stem is composed of a set of raw audio files. For our project, we decided to use stem files, which represent an instrument. For example, a "drum set" stem contains tracks for each drum in the drum set, a more realistic label for music production purposes [2].

Figure 3: MedleyDB Dataset Composition [7]

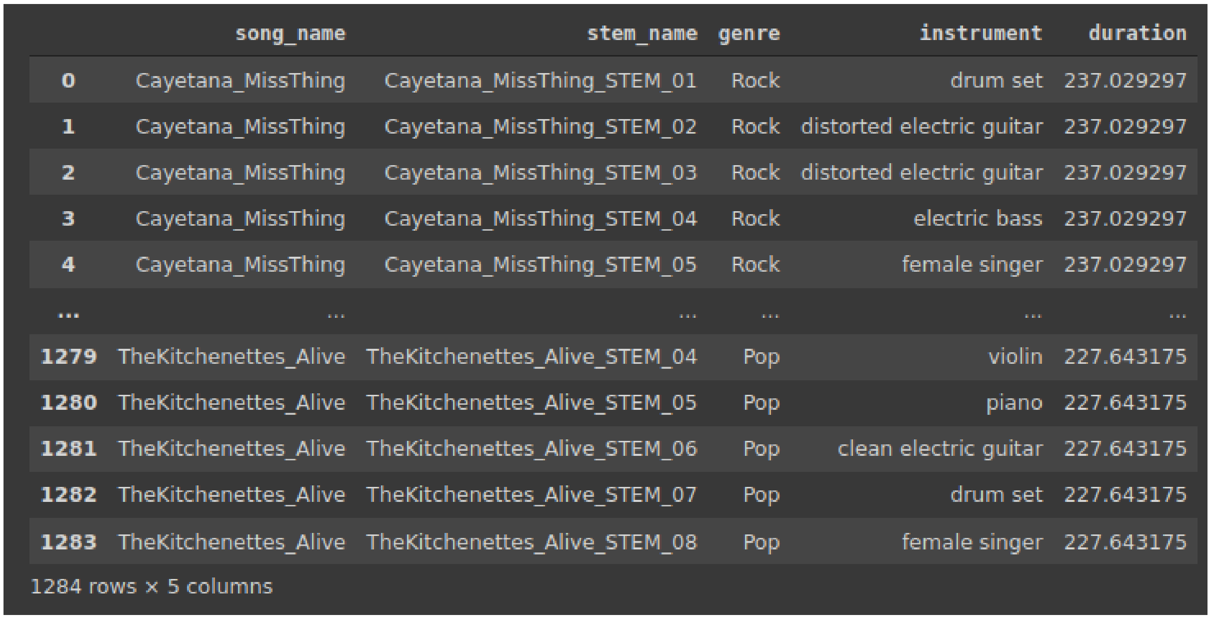


4.2 Data processing

Preprocessing consisted of a series of steps. Since our model is a CNN, we believed using images would be the best method to train it. Thus, we needed to convert the stem WAV files into spectrograms using the librosa library. After this, we had 1284 spectrograms to work with.

Then, we created a dataframe with every stem name, original song name, genre, instrument, and duration. Using this dataframe, we plotted the stem durations, deciding to keep the stems with the most common lengths. These songs were between 15 seconds and 7 minutes long, leaving us with 1100 spectrograms.

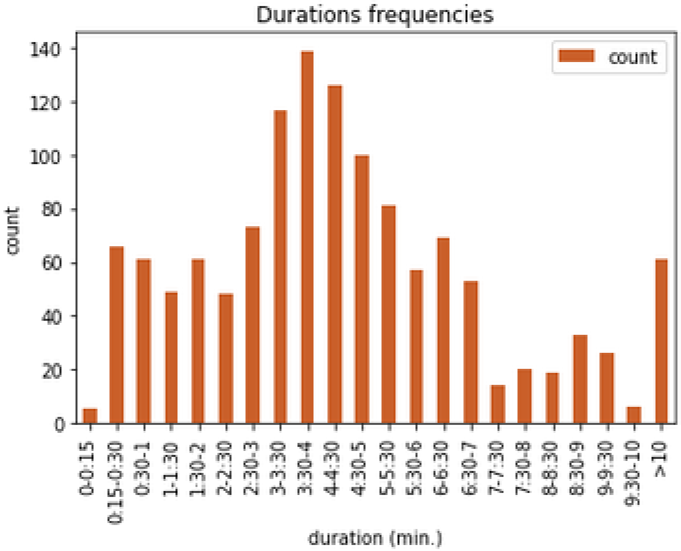
Figure 4: Initial Dataset Dataframe



	song_name	stem_name	genre	instrument	duration
0	Cayetana_MissThing	Cayetana_MissThing_STEM_01	Rock	drum set	237.029297
1	Cayetana_MissThing	Cayetana_MissThing_STEM_02	Rock	distorted electric guitar	237.029297
2	Cayetana_MissThing	Cayetana_MissThing_STEM_03	Rock	distorted electric guitar	237.029297
3	Cayetana_MissThing	Cayetana_MissThing_STEM_04	Rock	electric bass	237.029297
4	Cayetana_MissThing	Cayetana_MissThing_STEM_05	Rock	female singer	237.029297
...	...	...	...	...	...
1279	TheKitchenettes_Alive	TheKitchenettes_Alive_STEM_04	Pop	violin	227.643175
1280	TheKitchenettes_Alive	TheKitchenettes_Alive_STEM_05	Pop	piano	227.643175
1281	TheKitchenettes_Alive	TheKitchenettes_Alive_STEM_06	Pop	clean electric guitar	227.643175
1282	TheKitchenettes_Alive	TheKitchenettes_Alive_STEM_07	Pop	drum set	227.643175
1283	TheKitchenettes_Alive	TheKitchenettes_Alive_STEM_08	Pop	female singer	227.643175

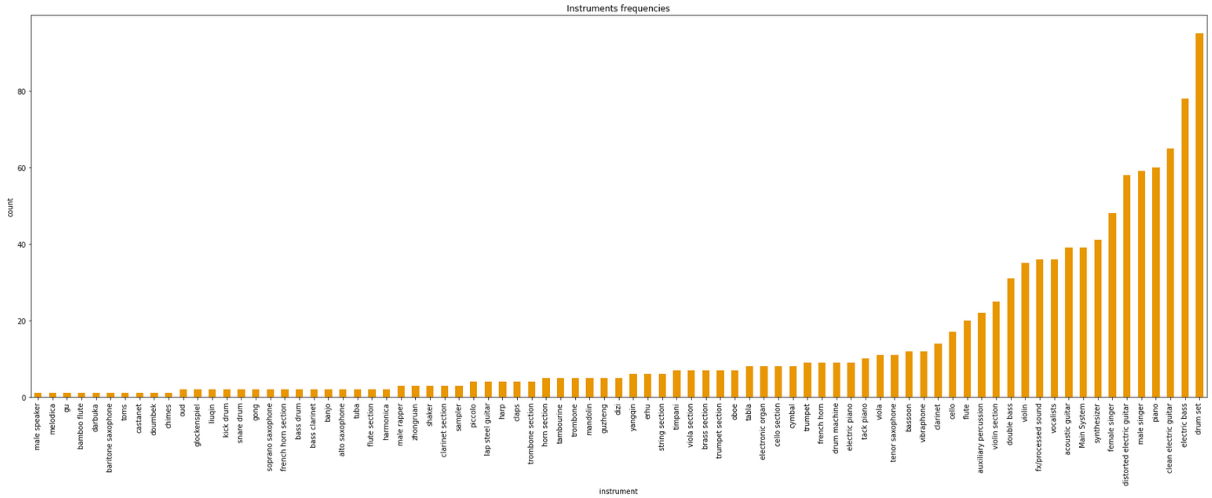
1284 rows × 5 columns

Figure 5: Stems Duration Frequency Graph



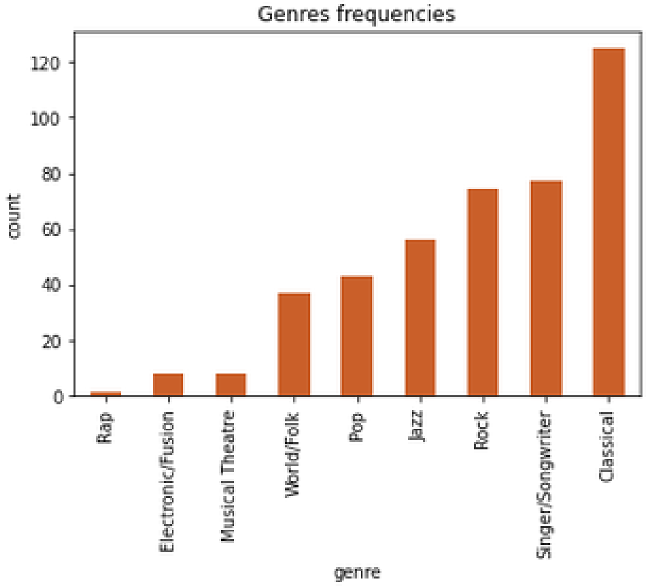
We did the same with the instrument labels, keeping the most common of them, to assist in making balanced instrument distributions. We discarded tracks with full instrumental ensembles, since this did not align with the project of classifying individual instruments. Thus, the chosen instruments were: drum set, piano, acoustic guitar, violin, double bass, flute, cello, clarinet, viola, female singer and male singer. We decided to combine the female and male singer labels into just one singer label, so the human voice did not get over represented in the dataset. That process left us with 429 spectrograms.

Figure 6: MedleyDB Instrument Frequency Graph



To continue balancing our dataset, we removed the less common genres. We decided to remove the Rap, Musical Theatre and Electronic/Fusion genres since they were less prevalent and contained less instrument classes. This left us with 412 spectrograms.

Figure 7: Genre Frequency Graph

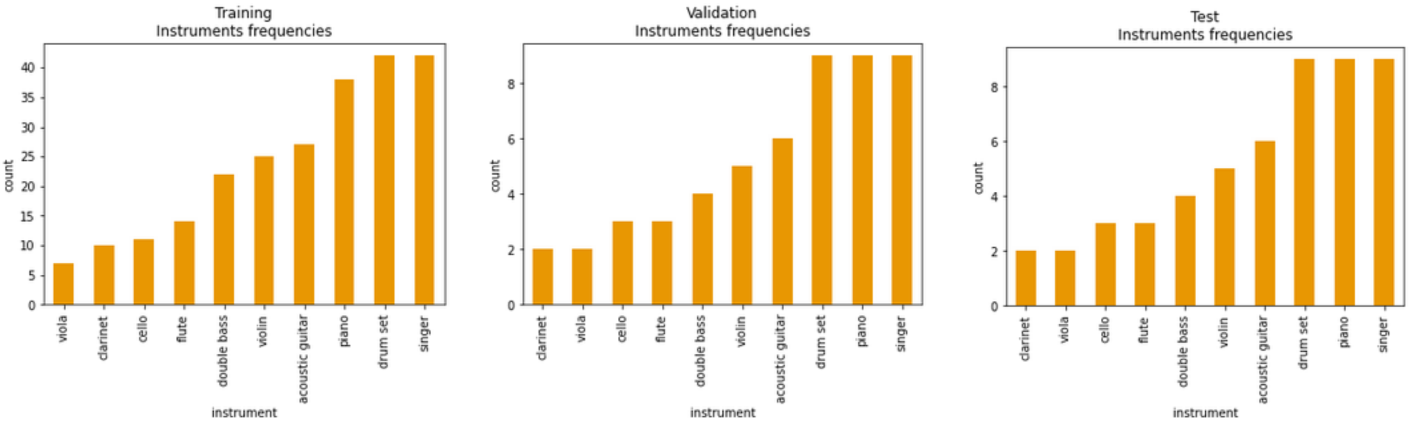


4.3 Dataset Split

To reduce bias, we randomly split the remaining spectrograms into training, validation, and test sets. We started filtering the songs by genre, and distributing them equally among the sets. We split the dataset into 70% training, 15% validation, and 15% testing. Then we redistributed the songs, to achieve a consistent distribution of instruments within the different sets, balancing over represented instruments. As the singer and drum set labels were more common than the others, we limited their quantity to 60, leaving 342 spectrograms.

The splits were documented in three .csv files, so the model could parse these files, and locate the corresponding spectrograms in a Drive folder.

Figure 8: Final Dataset Composition

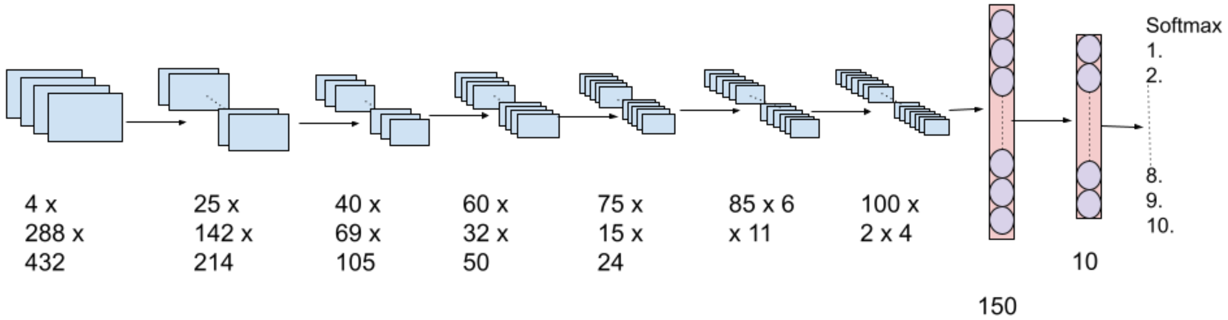


5.0 Architecture

Understanding the nuanced features of an input is one of the most attractive abilities of a CNN. Preliminary research on successful models such as Spleeter by Deezer indicated the success of CNNs in music recognition [9]. The reason for making a six-layered model is due to the small number of spectrograms per instrument. Since our model had to detect very minute details to differentiate between classes, a deep CNN with many trainable kernels made sense.

Our hand-coded deep CNN primary model (Figure 9) consists of six convolutional layers and two linear layers with ReLU and softmax activation functions. The softmax is applied after the second fully connected layer to find probabilities of each instrument. Max Pooling of kernel size 2 and stride 2 is applied after every convolutional layer to consolidate information. The kernel size for the first three layers is five and the last three layers is three.

Figure 9: Primary Model Architecture

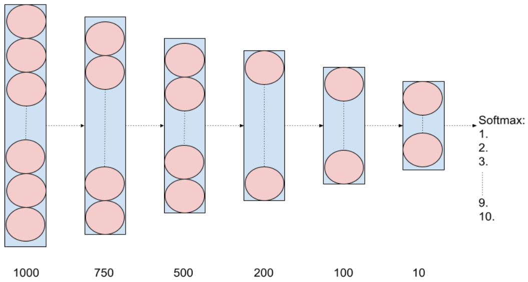


Our final model was trained on a dataset of 238 music files containing 10 instrument classes. After more than two days of tuning the hyperparameters, we decided upon a learning rate of $0.65 \times 10^{-4}$with a batch size of 30 over 25 epochs.

6.0 Baseline Model

Different instruments generate spectrograms that look unique with even single instruments generating somewhat different spectrograms for various melodies. Such patterns are very hard to identify by a human but machine learning models can pick up these patterns relatively quickly. Since ANNs are modelled after the human brain, it made sense to use a simple ANN as the baseline model for our project. We chose a six-layered model (Figure 10) with 1000, 750, 500, 200, 100 hidden units for each layer respectively.

Figure 10: Baseline Model Architecture

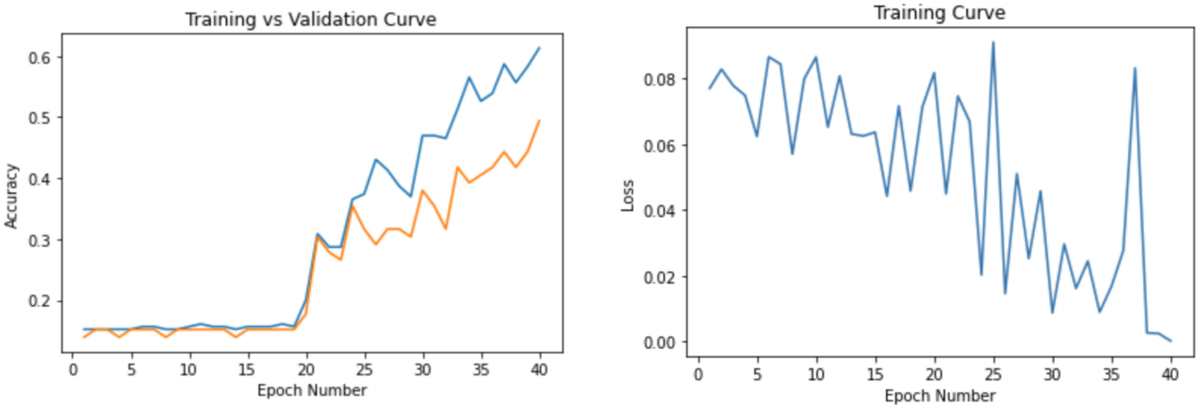


The final learning rate for this model was 0.8×10^{-4} for a batch size of 30 over 24 epochs. The final testing accuracy for the baseline model was 34.6% with an F1-score of 0.23. These results are further elaborated upon in section 10.0.

7.0 Quantitative Results

Figures 11 and 12 show the final training and validation accuracy as well as the loss. Although some overfitting on the training set can be gleaned, the validation accuracy continued rising while loss fell.

Figure 11 & 12: Training vs Validation Accuracy Curve and Loss for our Primary Model

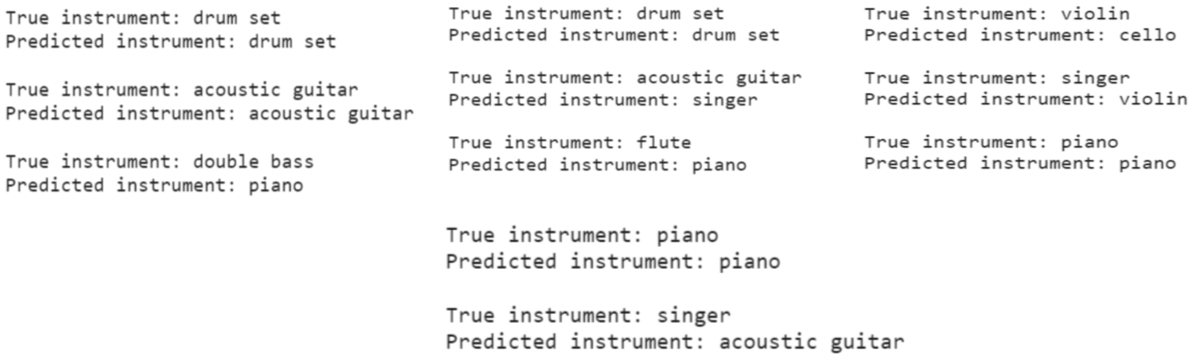


The final testing accuracy for our primary model was exactly 50%. However, since the dataset was slightly unbalanced over the instrument classes, a better indication of model performance is the F1-score, which was 0.453.

8.0 Qualitative Results

To ensure our model worked, we performed a sanity check, attempting to overfit on a smaller dataset containing only 3 songs for each instrument. The accuracy after ~100 epochs was 100%, showing our model to be qualitatively viable.

Figure 13: First Eleven Predictions Made by Our Model



```
True instrument: drum set
Predicted instrument: drum set

True instrument: acoustic guitar
Predicted instrument: acoustic guitar

True instrument: double bass
Predicted instrument: piano
```

```
True instrument: drum set
Predicted instrument: drum set

True instrument: acoustic guitar
Predicted instrument: singer

True instrument: flute
Predicted instrument: piano
```

```
True instrument: violin
Predicted instrument: cello

True instrument: singer
Predicted instrument: violin

True instrument: piano
Predicted instrument: piano
```

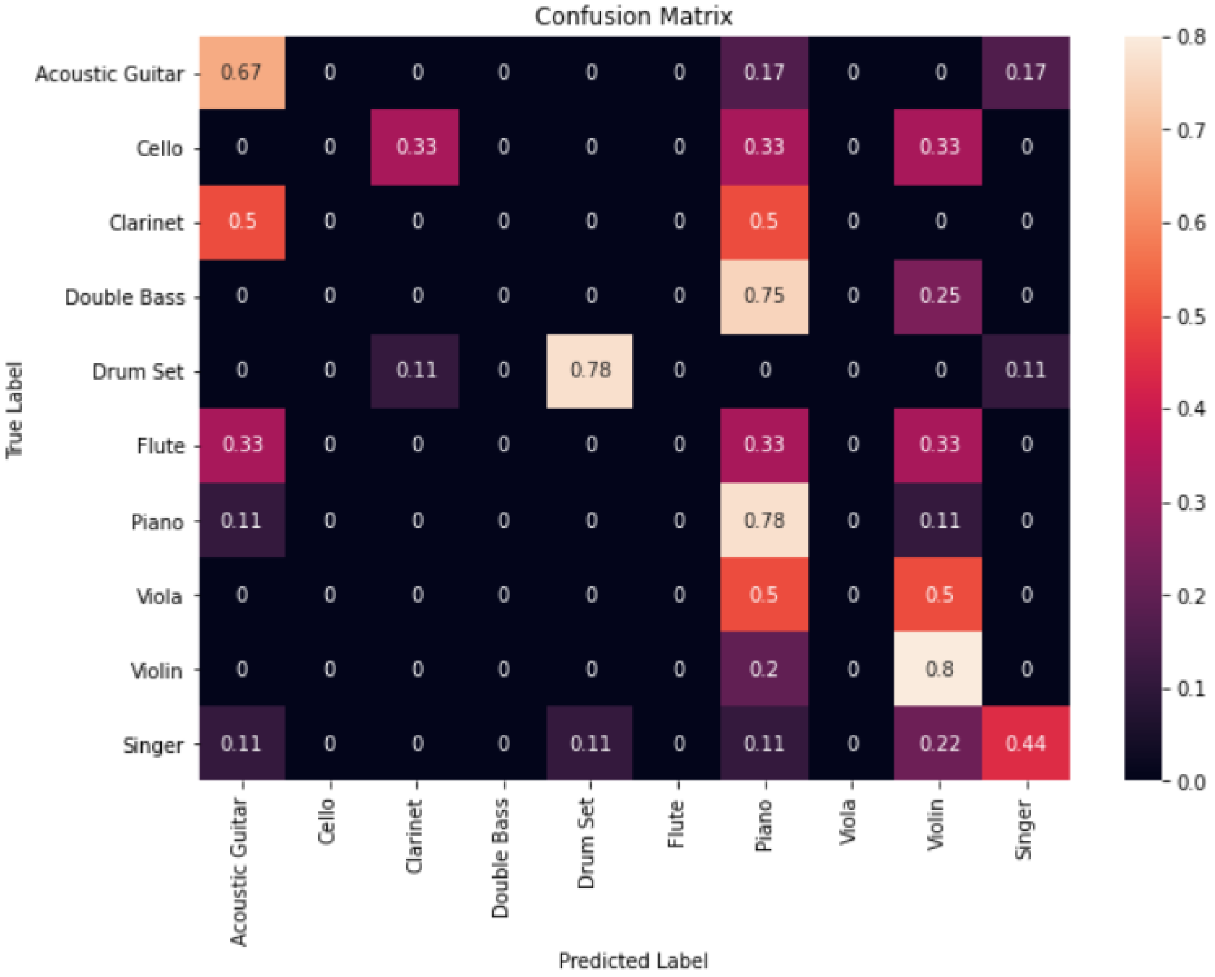
```
True instrument: piano
Predicted instrument: piano

True instrument: singer
Predicted instrument: acoustic guitar
```

Figure 12 shows the first eleven predictions made by our model on the test set. These outputs were chosen at random and not 'cherry picked,' since we wanted to show an accurate estimate of our model's performance. This small snippet shows that drum sets are being predicted quite accurately but our model confuses acoustic guitars and singers. String instruments like violin and cello are also mixed up in the predictions.

These results are neatly summarized in a normalized confusion matrix (Figure 14), providing a better understanding of which instruments were predicted more precisely.

Figure 14: Confusion Matrix for Primary Model

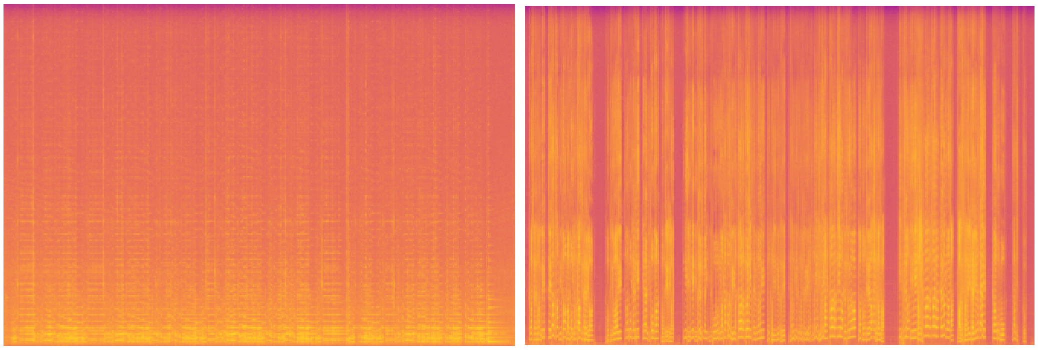


Instruments with the highest accuracy were acoustic guitar, piano, drum set, singer, and violin. Another interesting observation that can be made from the confusion matrix is that many instruments are falsely predicted as either piano or violin. However, most of the true piano and violin tracks were predicted accurately. This property is not exhibited in the drum set category, meaning the model is able to identify drum sets without resulting in a large amount of false positives. These results will be discussed in more detail in section 10.0.

9.0 Evaluation of the Model on New Data

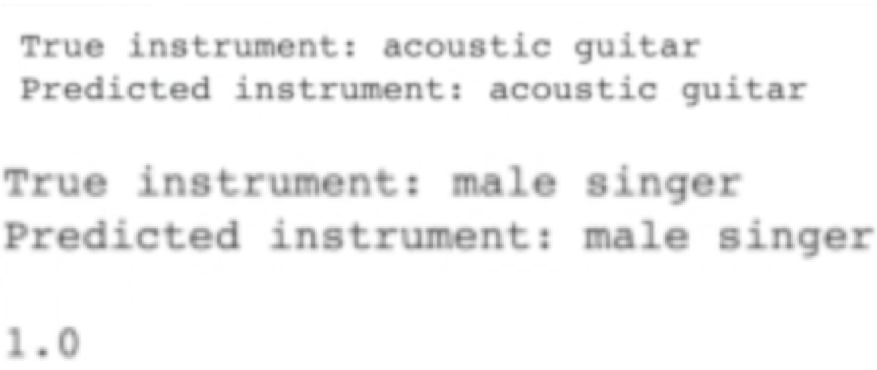
Although the separated test dataset was already ‘new’ data for which we did not tune the hyperparameters, we decided to collect more unseen data to see if our model had a chance at generalizing. To ensure that our model worked for real-world data, Kieran recorded himself singing the classical lieder, “Erlkonig” by Schubert, and playing the acoustic guitar part of “Blackbird” by the Beatles.

Figure 15: Spectrograms for the New Data
(“Erlkonig” - Left, “Blackbird” - Right)



A German song was chosen because we wanted to see if the model was classifying singers well because it was actually detecting the human voice rather than the English language. Additionally, since the tracks were not recorded in a studio, background noise was prevalent, giving a better indication of model generalization. The predictions made by the model for the unseen data are shown in Figure 16, as well as the 100% accuracy it achieved on the set with no hyperparameter tuning.

Figure 16: Predictions for Kieran’s Data



```
True instrument: acoustic guitar
Predicted instrument: acoustic guitar

True instrument: male singer
Predicted instrument: male singer

1.0
```

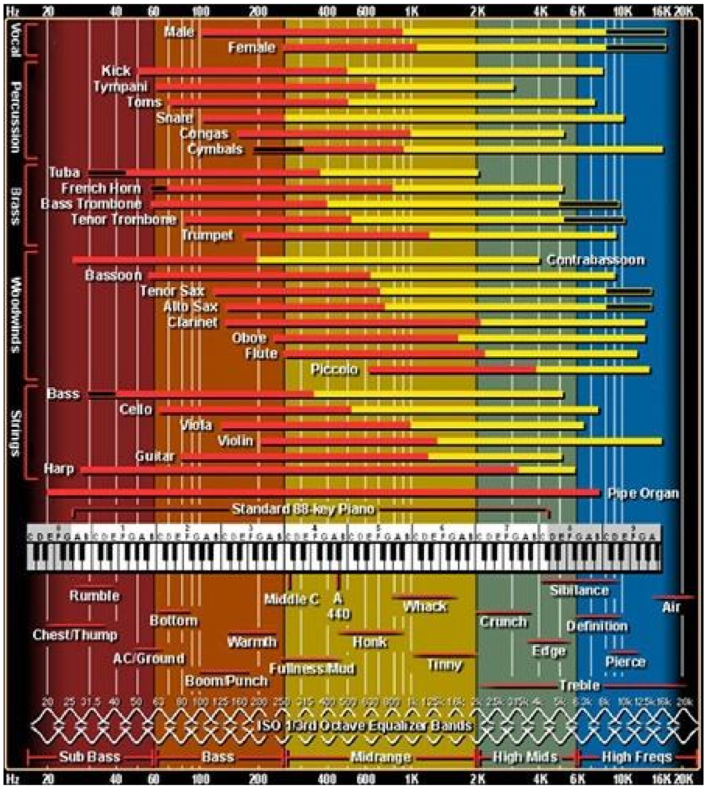
These results were measured before we grouped male and female singers together in a single class. However, our model not only identified the instruments Kieran was using, but his gender as well. Thus, despite the relatively low test-set accuracy compared to what our models have achieved in labs, the CNN we developed exceeded expectations for this complex problem.

10.0 Discussion

Overall, the model performs well in terms of the difficulty of the project scope. A multiclass classification problem with 10 classes has a 10% chance of being correctly classified at random. A 50% accuracy on the test set, while 5 times higher than random chance, is still not reliable for artists to use, especially those who have experience identifying instruments with ease. The model is best described as a proof of concept that machine learning can be used to identify different instruments in musical tracks.

Based on the qualitative results, some instruments like acoustic guitars, pianos, drum sets, violins, and human voices were identified well by our model whereas instruments like clarinets, flutes, and violas had a very low chance of being correctly classified. This is likely because of two fundamental characteristics of music. The first is the difference between percussive instruments and non-percussive instruments. Percussive instruments, like drum sets, are often untuned, only playing one pitch at a time [10]. Additionally, they have a high zero crossing rate, which is the rate at which a sound changes crosses 0 Hz [10]. This is also a quality of vocal music and speech, which often features percussive consonants. The other fundamental characteristic is the idea of Fourier Coefficients and harmonics. Sound propagates in various harmonics, the base of which, the 0th harmonic is called the fundamental frequency [11]. Each instrument can be characterized by the harmonics it resonates most within. However, instruments like the piano cover many harmonics due to its large range (Figure 17), this is why our model was most likely to predict that the instrument was a piano. This is also why most string instruments were predicted to be violins because of its prevalence in the dataset and its large range.

Figure 17: Frequency Ranges of Various Instruments [12]



When compared to the baseline, the model performs significantly better in identifying a wider array of instruments like singer, violin and acoustic guitar whereas the baseline model could only accurately identify piano and drum set. The F1-score and accuracy were much higher for the primary model than the baseline model, however the primary model's F1-score was still 0.45, meaning there is more work to be done in achieving higher precision.

11.0 Ethical Considerations

This project highlights ethical issues pertaining to copyright and music sampling. In terms of the datasets, MedleyDB is meant for strictly educational purposes [13][8]. However, since it contains copyrighted music, the ethical issue of training our model without the consent of the original artists is raised. Additionally, our model can potentially enable other artists and producers to break down the original artists' music and use it for sampling which the original artist may not have consented to [14]. Lastly, the gendered classes in the MedleyDB database of "female singer" and "male singer" could pose an issue for transgender and nonbinary artists. We accounted for this by combining the classes into a single, gender-neutral "singer" class.

12.0 Project Difficulty

This project is quite difficult due to the overlapping nature of instrument ranges making it harder to identify certain instruments over others [12]. Other teams have tried developing models to identify instruments and have shown to be in a similar success range as our model. For example, as mentioned above, Kawwa's model classified 11 instruments using different methods, one of which was a CNN model [5]. This model was trained for 15 hours, achieving an accuracy of 55%, slightly higher than ours [5]. Despite our model being trained with a significantly smaller and more complex dataset, less time, and limited Colab performance, our accuracy was very close to his, illustrating the success of our model relative to the difficulty of our project. It also indicates that the ceiling for a CNN model may be in the sub ~60% range therefore, achieving a higher accuracy requires more advanced models outside the scope of this course.

One reason for the project difficulty is the small amount of data that is available. The fact that we used data with multi-note melodies rather than single tones increased the difficulty of our project. Additionally, many of the spectrograms had interference from other instruments as they weren't perfect isolations of their respective instruments. This issue could not be solved by data augmentation since the differences between instrument spectrograms is incredibly precise and minute. Thus, rotating, cropping, inverting, or pitch-shifting these inputs would have changed the data to where the instrument may have become completely unrecognizable. Additionally, augmenting data manually could take weeks. This activity would be futile since augmenting data would defeat the purpose of our project in the first place since the data would no longer be realistic.

Furthermore, the scale of this project was also larger than anything we had done before. Not only had we not worked with an audio based dataset, we also set out to classify 10 classes which was more than what was asked for in this course's labs.

13.0 References

[1] Défossez, A., Usunier, N., Bottou, L., and Bach, F., “Music Source Separation in the Waveform Domain”, 2019. Available: https://hal.archives-ouvertes.fr/hal-02379796/document. [Accessed: 08-Apr-2021]

[2] F. Vazquez, "Separate Music Tracks with Deep Learning", Medium, 2019. [Online]. Available: https://towardsdatascience.com/separate-music-tracks-with-deep-learning-be4cf4a2c83. [Accessed: 08-Apr-2021].

[3] S. Essid, G. Richard and B. David, "Hierarchical Classification of Musical Instruments on Solo Recordings," *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, Toulouse, France, 2006, pp. V-V, doi: 10.1109/ICASSP.2006.1661401. Available: https://perso.telecom-paristech.fr/essid/papers/SE_ICASSP-06.pdf/. [Accessed: 08-Apr-2021].

[4] K. Martin and B. Vercoe (2000). Sound-source recognition: A theory and computational model.

[5] N. Kawwa, “Can We Guess Musical Instruments With Machine Learning?,” *Medium*, 29-Apr-2019. [Online]. Available: https://medium.com/@nadimkawwa/can-we-guess-musical-instruments-with-machine-learning-afc8790590b8. [Accessed: 09-Apr-2021].

[6] Magenta. 2021. *The NSynth Dataset*. [Online] Available: https://magenta.tensorflow.org/datasets/nsynth. [Accessed: 09-April-2021].

[7] MedleyDB 2.0 Audio | Zenodo, Zenodo, 2021. [Online]. Available: https://zenodo.org/record/1715175#.YCSl2OrQ_VO [Accessed: 07-Apr-2021].

[8] Zenodo.org. 2021. *Zenodo - Research.* [Online] Available: https://zenodo.org/ [Accessed 07-Apr-2021].

[9] A. Jansson, E. Humphrey, N. Montecchio, R. Bittner, A. Kumar and T. Weyde, "Singing Voice Separation With Deep U-Net Convolutional Networks", [Online]. Available: https://ejhumphrey.com/assets/pdf/jansson2017singing.pdf. [Accessed: 07-Apr-2021].

[10] J. Alm and J. Walker, "Time-Frequency Analysis of Musical Instruments", SIAM Review, vol. 44, no. 3, pp. 457-476, 2002. Available: 10.1137/s00361445003822. [Accessed: 08-Apr-2021].

[11] M. Bell, "Fourier analysis in Music - Rhea", Projectrhea.org, 2021. [Online]. Available: https://www.projectrhea.org/rhea/index.php/Fourier_analysis_in_Music. [Accessed: 09-Apr-2021].

[12] LANDR Blog. 2021. *EQ Cheat Sheet: How to Use Instrument Frequency Chart | LANDR Blog*. [Online] Available: https://blog.landr.com/eq-cheat-sheet/?fbclid=IwAR0uyWuRovTxwPE0huXQN3tOmIpRE5ew8r0rHT6Y0FyZOEELj7Ms9HDLDpU [Accessed: 09-Apr- 2021].

[13] DSD100 | SigSep, Sigsep.github.io, 2021. [Online]. Available: https://sigsep.github.io/datasets/dsd100.html. [Accessed: 09-Apr-2021].

[14] www.nolo.com. 2021. *When You Need Permission to Sample Others' Music*. [Online] Available: https://www.nolo.com/legal-encyclopedia/permission-sampled-music-sample-clearance-30165.html [Accessed: 08-Apr-2021].