Guitar Tuning Detection

McCormack Chew
University of Rochester
mchew2@ur.rochester.edu

Declan Parker
University of Rochester
dpark31@u.rochester.edu

Abstract

This paper presents a method for classifying seven different guitar tunings based solely on audio input. Audio clips are sorted into groups of likely tunings, and feature embeddings are extracted using an L3 model. The method then utilizes support vector machines to classify these feature embeddings. Our method is the first of its kind to be trained on a diverse dataset, encompassing various genres and tonalities. It shows potential, but demonstrates the challenge, and potential impracticalities, of differentiating some tuning groups from each other.

1. Introduction

The guitar is a very versatile instrument. From folk to country to classic rock to metal, the instrument takes on many different playing styles and sounds. One of the main enablers of the varying sounds throughout different genres is alternate tunings, which is when a guitar's strings are tuned in a manner that is not standard. This standard tuning is called E Standard. Alternate tunings include low drop tunings, like Drop D and Drop C, common in metal and rock songs due to the ease of playing heavy and impactful sounding power chords, and open tunings, like Open D and Open G, common in folk and indie due to the ease of playing open chords, chords that include one or more unmuted non-fretted strings. Both drop and open tunings allow guitarists to more easily reach certain tone colors and keys, and most tunings are typically named after their lowest string. For instance, power chords are chords that only feature the 1st and 5th scale degrees and are easier to play in drop tunings because a single finger can be used to play them. Meanwhile, open tunings make major chords easier to play as the strings are tuned to a major triad.

For a guitarist trying to learn a song, determining the tuning is typically the first thing they need to do. If they are reading off tablature notation, a notation system that marks which string and corresponding fret should be played at certain times, the tuning is typically written at the top of the music sheet so that the following notation produces the correct notes heard in the song. While many songs are playable in different tunings, the fingerings needed to translate a song originally played in one tuning to another could differ vastly and be more difficult to play. Of course, many times songs must be played in a specific tuning, especially songs where the guitarist tunes down, like to Drop C.

This project is motivated by efforts to formulate a model that can determine the tuning of songs with little to no documentation of the tuning used. We use the assumption that the same note played on two differently tuned guitar strings will have different spectral qualities. We confirmed this phenomenon quantitatively and visually by plotting spectrograms of single notes, as well as chords and calculating the cosine similarity between the spectra. Figure 1 and figure 2 show the spectrograms for the note E2 when played in E Standard tuning and Drop C tuning respectively. The note shows a noticeable beating effect in Drop C tuning which is much less pronounced in E Standard. Additionally, certain frequencies show different decay times between the two tunings. In particular, frequencies between 1000 and 2000 Hz generally retain their energy for longer in E Standard, which gives the note an audibly darker tone. The calculated cosine similarity of these two notes is 0.80%.

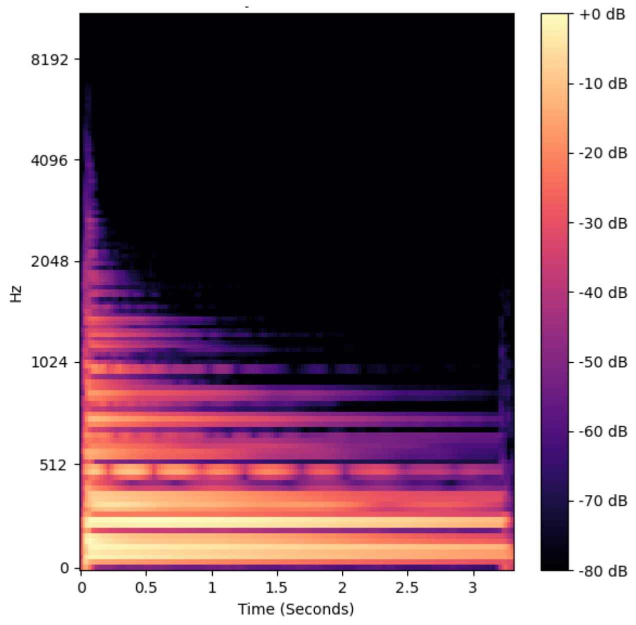

Figure 1. Mel Spectrogram of E2 played in E Standard tuning

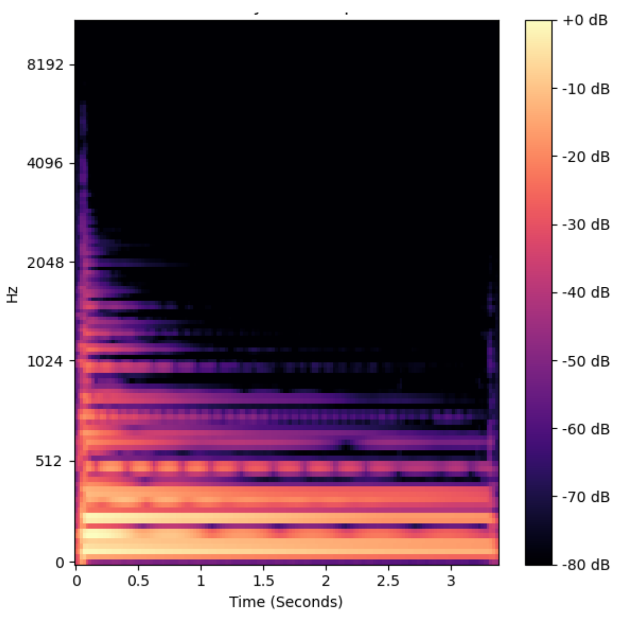

Figure 2. Mel Spectrogram of E2 played in Drop C tuning

We theorize that a model could be trained to learn the nuances of multiple tunings and identify which tuning is used when given an audio example. Such a model could identify points within recordings that feature uniquely tuned strings, recognizing alternate tunings and classifying them appropriately. While this paper outlines a proof of concept, this idea could be extended to many tunings and trained on specific tunings and genres that a guitarist favors or is trying to learn. The names of the tunings that our model is trained to distinguish between and the notes that each open string is tuned to in each case can be found in Table 1.

The rest of this paper is structured as follows: Section 2 details our labeled training data collection, Section 3 outlines the methods and formulation of our model, Section 4 discusses ongoing results, and in Section 5 we explore potential future exploration of the topic.

String	6	5	4	3	2	1
E Standard	**E**	**A**	**D**	**G**	**B**	**E**
Drop D	**D**	**A**	**D**	**G**	**B**	**E**
Drop C	**C**	**G**	**C**	**F**	**A**	**D**
D Standard	**D**	**G**	**C**	**F**	**A**	**D**
Open D	**D**	**A**	**D**	**F#**	**A**	**D**
Open G	**D**	**G**	**D**	**G**	**B**	**D**
F Maj 9	**F**	**A**	**C**	**G**	**C**	**E**

Table 1. Open string notes for each tuning used. String 6 corresponds to the lowest pitched string.

2. Training data

2.1 Data Challenges and Previous Research

A significant hurdle we had to overcome in order to train a guitar tuning classifier was the lack of appropriate training data. Previous research combining machine learning and guitar tunings is scarce, with the only other example online being a Cardiff University student's project [1]. This project found moderate success employing support vector machines alongside a convolutional neural network to classify guitar tunings, but the training data was limited to exclusively Joni Mitchell songs. This training set introduces some potential limitations to the applications of the classifier, as the songs were all acoustic, and fell under similar genres.

We aimed to develop a classifier which was more capable of handling different genres and guitar tones. Unfortunately, no pre-existing dataset of diverse guitar songs labeled by tuning existed. Even among unlabeled sets, it is difficult to know if any contain enough instances of nonstandard tunings to make them identifiable to a classifier. Knowing this, we curated our own dataset to best suit our purposes.

2.2 Goal and Initial Data Collection

We began by identifying ten songs of each tuning, utilizing the websites Ultimate Guitar [2], and Guitar Tuning Database [3], which each compile lists of popular guitar songs sorted by tuning. We recognized that some biases in the classifier might be likely if we picked the songs arbitrarily. For example, tunings like Drop C are most popular in genres like metal, featuring heavily distorted guitars. If every example of Drop C was heavily distorted, and none of the E standard examples were distorted, it is likely that the identifier would base its classifications more on guitar tone than spectral nuances. In order to counteract this effect, we selected instances of distorted and clean toned guitars for each tuning when possible. After acquiring high quality versions of all of the selected songs, an issue with this data became apparent. While the Joni Mitchell dataset had some limitations, it did have the benefit of featuring consistently minimal instrumentation. We knew that if we trained our model on songs featuring a full mixture of instruments, we ran the risk of our model differentiating between tunings using features of non-guitar elements of the songs. In order to mitigate this, we ran all of our tracks through a stem separator in order to isolate the guitar and minimize the impact of any other instruments on the classifier.

2.3 Tests, Refinement, and Dataset Overview

Initial tests, which used the entirety of the separated stems as training inputs, produced poor results, with accuracy under 50%. Our leading theory for the cause of these results is the sparseness of the input tracks. A majority of the stems contained stretches of silence during parts of the songs that did not feature guitar, which could greatly skew the model's understanding of each tuning. Additionally, some songs simply did not contain enough data to be representative of the tuning the guitar was in. For instance, a song in Drop D tuning that only plays the low D note a couple times in a song is widely indistinguishable from a song in E standard, due to how similar the tunings otherwise are. At this point, it was clear that we would need to be a lot more selective in the data we trained our model on. Thus, we manually sliced up our audio into ten second clips which only contained guitar, and made a concerted effort to choose clips which featured the notes most characteristic of each tuning. The importance of clip selection for certain tuning also has certain implications for selecting clips to be identified, which will be explored below.

One unmentioned drawback inherent in stem-separated tracks is that audio artifacts are unavoidable, and we acknowledge that much of our stem-separated audio contains remnants of non-guitar instruments, and also is missing some harmonic content from the original guitar. To reduce the impact of these artifacts, we expanded our dataset by including solo guitar audio sourced from YouTube videos. These supplementary clips, unaffected by artifacts are intended to make our model robust to a greater variety of inputs.

Our complete dataset includes 260 ten second audio clips, 164 of which are sourced from stem separated tracks, and 96 of which are from YouTube guitarists.

3. Method

We employ a three-step design. First, we detect the lowest pitch within the sample. Next, we extract features from the sample using OpenL3. Finally, we classify the feature vectors based on the tuning label and the lowest detected pitch, utilizing support vector machines

3.1 Pitch Detection

Pitch detection is employed at the user level, primarily to detect the presence of notes lower in pitch than E2, the lowest pitch found in E Standard tuning. By determining the lowest note in any given audio clip, we are able to rule out the possibility of any tunings in which the note is out of range. This preliminarily sorts audio clips in order to improve classification results.

To perform pitch detection, we use the CREPE Pitch Tracker [4]. CREPE utilizes a deep convolutional neural network which was trained on over twenty one hours of audio in order to accurately classify pitch. Despite being intended for use on monophonic audio, we found that it still performed well for our purposes. The model takes in an audio file, and generates an estimate for the most likely note every ten milliseconds, as well as a confidence rating for that estimate. If an audio input is found to contain a frequency under that of E2 with a confidence level of over 50%, the clip is flagged, and can no longer be classified as any tuning which does not contain the low note. Given its current parameters, the pitch detection module has yet to falsely detect a low note in a high tuning such as E Standard or FMaj9. It does however, sometimes mislabel low notes as being an octave higher, meaning the audio is never flagged. This is likely a consequence of inputting polyphonic music into CREPE, which was trained on monophonic data. Despite the occasional false negative, audio which is not flagged is still able to be classified as a low tuning later on in the process, meaning the pitch detection step should only ever improve classification accuracy.

3.2 Embedding Model

To generate feature vectors, we use OpenL3 [5], which contains pretrained L3-Net audio embedding models. This is a Convolution neural network, and the particular model used here is trained on about 2 million ten second Youtube videos of music performances from AudioSet [6]. The input to the model is 256 band Mel Frequency spectrograms with window size of 1 second and hop size of 0.5 second, and the output dimensionality is 6144. The labeled tuning audio data is cut to 10 seconds, so the output of the model is 20 frames per sample.

3.2 Classification

Based on the lowest note detected, each frame and label are inputted into a corresponding support vector machine for classification. If the lowest note in a clip is D2 or lower, we train a support vector machine with the corresponding feature vector frames without E Standard and FMaj9 tuning classes. Since the lowest note possible in those tunings is E and F, respectively, we can eliminate them as potential tunings, which is especially helpful in detecting Drop D tuning as it only differs from E Standard in its lowest string. If the lowest note detected is above an E, all tunings must be considered. The support vector machines performed best with a polynomial kernel.

The labeled dataset is split 80% training and 20% test, with the training data being shuffled and test data kept in order. Each embedding frame's label is predicted by the support vector machine, so the test data must be kept in order since each ten second sample features 20 frames. The prediction with the most occurrences within the 20 frames is used as the clip prediction.

4. Results

4.1 First Model and Audio Selection

The first model we trained performed well on the test set. The tunings we included in this model were Drop D, Drop C, E Standard, and FMaj9, and it achieved 83.6% accuracy on the test set. As we trained another model on three additional tunings, D Standard, Open D, and Open G, and only saw a one percentage point reduction in accuracy, we realized something with the dataset was incorrect. Much of our dataset featured files longer than ten seconds, and in an effort to augment the size of the training data we cut the clip into multiple ten second clips. However, we couldn't subsequently keep the ten second clips from the same longer file together in either the training or test set, and since many of the longer clips were repeated riffs or chords, the model seemed to be memorizing parts of the training data on the test set.

4.2 Current Model

Once we cut each file to about ten seconds long and made sure none were repeated from other areas of a song, we saw a more accurate representation of the performance of the model. Trained on the full dataset of every tuning the model achieved 71.7% accuracy, and when training a separate classifier to detect tunings not including E Standard and FMaj9, since we can eliminate those when pitches below E2 are detected, the model was 77.5% accurate. Averaged together while taking into account the sizes of the datasets with and without E Standard and FMaj9 files, the overall accuracy of the combined models is 74.2%.

Model	Accuracy
1	83.6%
2	82.8%
3	71.7%
4	77.5%
5	74.2%

Table 2. Model 1 contains tunings Drop C, Drop D, E Standard, and FMaj9. Model 2 contains the same tunings as Model 1 as well as Open G, Open D, and D Standard. Model 3 contains the same tunings as Model 2 but without dataset leakage. Model 4 is the same as Model 3 but with E Standard and FMaj9 tunings taken out of the training and test sets. Model 5 is the average test accuracy between Model 3 and Model 4.

4.3 Limitations

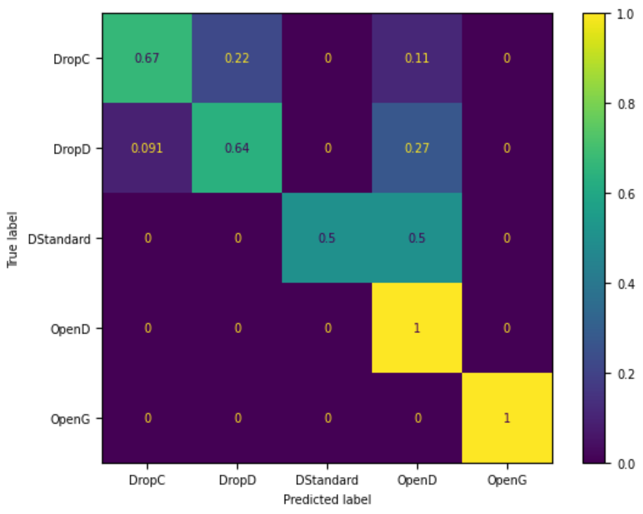

Figure 3. Confusion matrix of Model 4 from Table 2

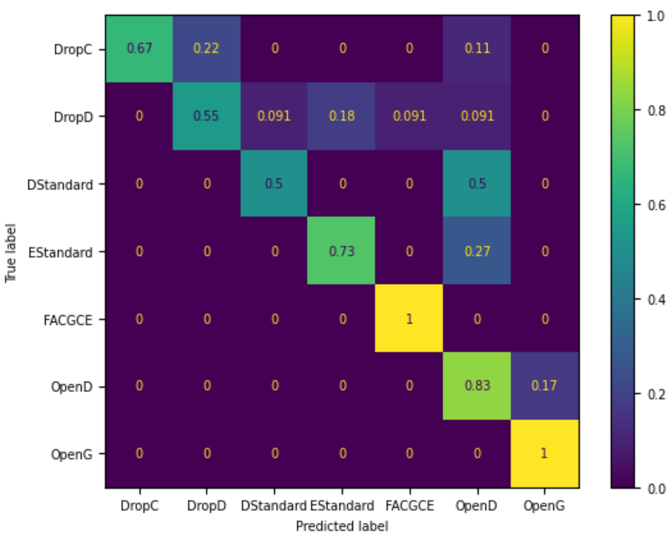

Figure 4. Confusion matrix of Model 5 from Table 2

As Figures 3 and 4 demonstrate, the model struggles heavily with tunings that feature D2 on the lowest string. Drop D, Open D, and D Standard are generally the worst performing tunings, which makes intuitive sense. Drop D and Open D share the same 3 lowest strings, which are used more often than the higher 3 strings in most of the dataset and popular music in general, so differentiating between them should be difficult. D Standard also shares 3 strings with Open D, so they were also difficult to classify separately.

It was surprising that drop tunings were rarely confused with their most similar tunings, which for Drop C is D Standard and for Drop D is E Standard, as they share all but the lowest string. This points to the lowest string having the highest weight within the model.

Other issues become known when inputting single files for classification. The model tends to classify very distorted guitars as Drop C, even though we have made an effort to diversify the dataset on the tuning level. When tuned to Drop C, a guitarist is technically able to play any song in the database since open notes in other tunings just become fretted notes with Drop C, which points to the distortion of the guitar masking timbral differences between songs actually played in Drop C and songs played in higher tunings. Another consideration that must be made when inputting files for classification is the importance of using segments which contain the specific notes characteristic of each tuning. It is best practice to locate the lowest note played in a song by ear, and make sure that it is included in the input clip.

5. Conclusions and Further Work

Our results suggest that there is potential for classification of guitar tunings from audio through machine learning.

Our initial assumption was that the dominant differentiating factor between tunings would be the timbral qualities of the same notes on differently tuned strings. However, this was made difficult due to the lack of consistency in guitar tone, especially considering the prevalence of guitar with distortion effects masking the natural timbral intricacies of the guitar, and single notes between similar tunings being identical on certain strings. Despite this, our model was still able to differentiate between tunings with a moderate level of success, indicating that timbral differences between tunings do exist and are detectable with a model involving embeddings.

This method could be enhanced by incorporating genre and more accurate pitch or chord detection. As guitarists, we know that open tunings are almost never used in certain genres, so eliminating those from being predicted for a metal song, for example, would generally give the model better performance. A more comprehensive improvement would be to lean into the differences between the typical chords played in alternate tunings. Going through a full song and picking out chords, then detecting those chords as either major or fifth chords (power chord) and subsequently inputting them into OpenL3 for embeddings, could help the current model latch on to the major differences between open and drop tunings with the same lower string pitch.

We're also curious if our method could be improved by using a dataset without any artifacts due to stem separation. This was not possible in the timespan of this project, but further work in sampling alternate-tuned guitars or synthetic guitars could make artifact-free data possible in the future.

6. REFERENCES

[1] D. Roy, "Deep Learning Guitar Tunings," in *One Semester Individual Project.*, Cardiff, Wales. May, 2019.

[2] https://www.ultimate-guitar.com/ (accessed Dec. 1, 2023).

[3] https://gtdb.org/ (accessed Dec. 1, 2023).

[4] Kim, J. W., Salamon, J., Li, P., and Bello, J. P., "CREPE: A Convolutional Representation for Pitch Estimation", <i>arXiv e-prints</i>, 2018. doi:10.48550/arXiv.1802.06182.

[5] Aurora Cramer, Ho-Hsiang Wu, Justin Salamon, and Juan Pablo Bello. "Look, Listen and Learn More: Design Choices for Deep Audio Embeddings", IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP), pages 3852–3856, Brighton, UK, May 2019.

[6] F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audioset: An ontology and human-labeled dataset for audio events," in IEEE ICASSP, 2017, pp. 776–780.