Acoustic Classification of Guitar Tunings with Deep Learning

Edward Hulme
hulmeed@cardiff.ac.uk
Cardiff University
Cardiff, UK

David Marshall
marshallad@cardiff.ac.uk
Cardiff University
Cardiff, UK

Kirill Sidorov
sidorovk@cardiff.ac.uk
Cardiff University
Cardiff, UK

Andrew Jones
jonesac@cardiff.ac.uk
Cardiff University
Cardiff, UK

ABSTRACT

A guitar tuning is the allocation of pitches to the open strings of the guitar. A wide variety of guitar tunings are featured in genres such as blues, classical, folk, and rock. Standard tuning provides a convenient placing of intervals and a manageable selection of fingerings. However, numerous other tunings are frequently used as they offer different harmonic possibilities and playing methods.

A robust method for the acoustic classification of guitar tunings would provide the following benefits for digital libraries for musicology: (i) guitar tuning tags could be assigned to music recordings; these tags could be used to better organise, retrieve, and analyse music in digital libraries, (ii) tuning classification could be integrated into an automatic music transcription system, thus facilitating the production of more accurate and fine-grained symbolic representations of guitar recordings, (iii) insights acquired through guitar tunings research, would be helpful when designing systems for indexing, analysing, and transcribing other string instruments.

Neural networks offer a promising approach for the automated identification of guitar tunings as they can learn useful features for complex discriminative tasks. Furthermore, they can learn directly from unstructured data, thereby reducing the need for elaborate feature extraction techniques.

Thus, we evaluate the potential of neural networks for the acoustic classification of guitar tunings. A dataset of authentic song recordings, which featured polyphonic acoustic guitar performances in various tunings, was compiled and annotated. Additionally, a dataset of synthetic polyphonic guitar audio in 5 different tunings was generated with sample-based audio software and tablatures. Using audio converted into log mel spectrograms and chromagrams as input, convolutional neural networks were trained to classify guitar tunings. The resulting models were tested using unseen data from disparate recording conditions. The best performing systems attained a classification accuracy of 97.5% (2 tuning classes) and 73.9% (5 tuning classes).

This research provides evidence that neural networks can classify guitar tunings from music audio recordings; produces novel annotated datasets that contain authentic and synthetic guitar audio, which can serve as a benchmark for future guitar tuning research; proposes new methods for the collection, annotation, processing, and synthetic generation of guitar data.

DLfM 2024, June 27, 2024, Stellenbosch, South Africa
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-1720-8/24/06
https://doi.org/10.1145/3660570.3660574

CCS CONCEPTS

• **Applied computing → Sound and music computing**; • **Information systems → Music retrieval**;

KEYWORDS

guitar tunings, transcription, audio datasets, music indexing, neural networks, metadata

ACM Reference Format:
Edward Hulme, David Marshall, Kirill Sidorov, and Andrew Jones. 2024. Acoustic Classification of Guitar Tunings with Deep Learning. In *11th International Conference on Digital Libraries for Musicology (DLfM 2024), June 27, 2024, Stellenbosch, South Africa.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3660570.3660574

1 INTRODUCTION

Standard tuning[1] for the six-string guitar was firmly established by the 1800s and today it is the most frequently used tuning [18]. However, altered tunings are also used frequently—they have a bearing on which notes can be produced using open strings,[2] and hence on the timbre and harmony of the guitar. They can provide convenient fingerings and *open chords* that facilitate special guitar techniques, enable distinctive sonorities that are integral to certain guitar styles, inspire new compositions, and force one out of traditional performance habits. In this work, when the interval pattern between the open strings of a guitar deviates from standard tuning, the guitar is considered to be in an *altered tuning* [18].

Altered guitar tunings feature in genres such as blues, classical, folk, and rock. For instance, in the maskanda music of South Africa, the tuning used by guitarists "varies from standard tuning in that the high string is tuned to d' instead of e' ", and other tunings exist, "some pertaining to specific styles and others 'invented' by musicians to suit their individual characteristic styles" [10, p. 122]. Furthermore, altered tunings are present in the music of many of the world's most inventive guitarists such as Ali Farka Touré, Elizabeth Cotten, João Pernambuco, Joni Mitchell, and Robert Fripp.

Many styles of guitar music are rooted in oral/aural traditions, so reliable performance information—such as the guitar tuning and capo position—and accurate transcriptions are not available for most guitar recordings. Consequently, knowledge about the tunings that are associated with certain guitar styles could be lost if methods for identifying guitar tunings are not developed. Additionally, if the tuning used on a guitar recording is unknown, transcriptions are likely to be inaccurate.

It is also important to note that many guitarists learn music by listening carefully to recordings; this approach allows them to extract nuances from the music that notation fails to encapsulate.

[1] In *standard tuning*, the guitar is tuned to the following notes from low to high (6th to 1st string): E_2, A_2, D_3, G_3, B_3, E_4.
[2] *Open string* refers to the unobstructed full string, located between the bridge and nut.

Thus, when a musician or musicologist is learning or analysing a guitar recording, an accurate and succinct piece of information about the tuning, may reveal more than a detailed transcription (that wrongly presumes the guitar is in standard tuning).

Research related to the automatic identification of guitar tunings is underdeveloped. To the best knowledge of the authors, no publication has dealt with the classification of guitar tunings from guitar audio recordings. Nevertheless, there is a wide body of research devoted to tasks that are closely related to guitar tuning classification, such as guitar string detection, chord estimation, and automatic music transcription (AMT). Traditionally, approaches to these tasks have involved the extraction of various acoustic features (e.g., f_0, inharmonicity coefficient) [2] to describe the tonal content of the signal, classical machine learning models (e.g., HMMs, SVMs) [24] and/or constraint-based algorithms (e.g., heuristic cost functions, plausibility filters) [3] to estimate string/chord/note candidates [11, 13, 21]. More recently, approaches to these tasks have utilised neural networks (e.g., CNNs, RNNs, transformers) [22, 35] with time-frequency representations of audio data (e.g., spectrograms, constant-Q transforms) [50] to learn a mapping directly from audio to symbolic music [7, 9, 40]. It should be noted that the research discussed above only considers guitars in standard tuning. Nevertheless, the proposed methods could be adapted for altered guitar tunings.

The identification of guitar tunings from symbolic music has been investigated by Khatri & Dillingham [25]. The authors proposed deep learning (DL) and rule-based methods to predict the tuning of a guitar piece from its MIDI transcription. The first method employed supervised learning with a recurrent neural network (RNN). The RNN model could successfully identify standard and open C tunings, but had difficulty identifying open D and open G tunings. The second method employed a dynamic programming algorithm to determine the optimal note locations a song could have in a given tuning. The dynamic programming algorithm performed well on songs in open C and open D, but struggled with songs in standard and open G. Khatri & Dillingham's research provides some evidence that both DL and rule-based methods can be used to classify guitar tunings from symbolic music.

The primary aim of the work reported in the present paper is to investigate the appropriateness of neural networks (NNs) for the acoustic classification of guitar tunings. We define *guitar tuning classification* (GTC) as the identification of a particular guitar tuning from a recording that contains a guitar performance.

The work is driven by recent developments in deep learning. NNs can learn useful features for discriminative tasks, when trained on large amounts of data. They can learn directly from unstructured data, thus reducing the need for more elaborate feature extraction techniques. An overview of NNs is beyond the scope of this work—for more information, see [15, 39]. DL methods are now frequently used for music information retrieval tasks [4] and NNs are an integral part of current state-of-the-art AMT systems [19]. As NNs have a proven track record for music information retrieval tasks, we hypothesise that *a neural network has the capacity to learn discriminative musical features (e.g., harmony, key, pitch) and fine-grained features (e.g., harmonic spectrum, inharmonicities) relevant to GTC from labelled time-frequency representations of music recordings; these features pertain to the tuning of the guitar, enabling the network to distinguish between different guitar tunings.*

The acoustic classification of guitar tunings is a nascent topic, but research in this area is essential for the development of robust systems for music audio tagging and AMT. Further information regarding this work is available online.[3]

2 BACKGROUND

2.1 The Classification Task

The aim is to create a system that, when given a guitar audio clip, returns a decision regarding the type of guitar tuning that is present in the recording. A *supervised learning* approach is employed. A convolutional neural network (CNN) is trained on a corpus of labelled guitar audio samples; the audio is converted into a log mel spectrogram or chromagram representation and inputted into the CNN. The samples are correctly marked with guitar tuning labels. After training, the model is given new samples, and it predicts which categories the samples belong to. Authentic and synthetic guitar tuning datasets have been created specifically for this task.

This work derives methods and adapts CNNs from the fields of bird audio detection [47] and speech recognition [51]. In these tasks an NN must learn to recognise and classify many different sound event characteristics [26]—this is also true of GTC, so there are clear parallels between the tasks. Furthermore, CNNs were chosen for the task as evidence indicates they perform well on guitar-related MIR tasks, and an established baseline model for guitar tablature transcription (GTT) is CNN-based [50].

There are two methods for representing the tuning of a guitar: (i) specifying the absolute pitch of each string, (ii) specifying the intervals between strings. We investigate both methods for determining the tuning. The first method identifies the tuning by absolute pitch; this provides exactitude, but without certain constraints (e.g., the exclusion of recordings in which a *capo* is used on the guitar) the number of tuning classes yielded could quickly become unmanageable. The second method identifies the tuning by interval profile; *interval profile* refers to the distance of the open strings from each other as measured in semitones [43]. With this method, the strings of two guitars could be tuned to different absolute pitches, but if they shared the same interval profile, they would also share the same tuning class. An advantage of this method is that it provides flexibility and a manageable number of tuning classes, when guitars are transposed via downtuning or the use of a capo.

2.2 Notation

A robust GTC system could provide fine-grained and accurate transcriptions for various types of notation (see Fig. 1), particularly for tablatures. Tablature is a notational system that places numbers on horizontal lines—the numbers represent the frets and the lines represent the strings of the instrument. Tablature is highly intuitive for guitarists as it resolves "the ambiguity between note pitch and fretboard position" [37, page 26]. However, when the tuning of a guitar is altered, the relationship between note pitch and fretboard

[3] https://github.com/edhulme/guitar-tuning-classification

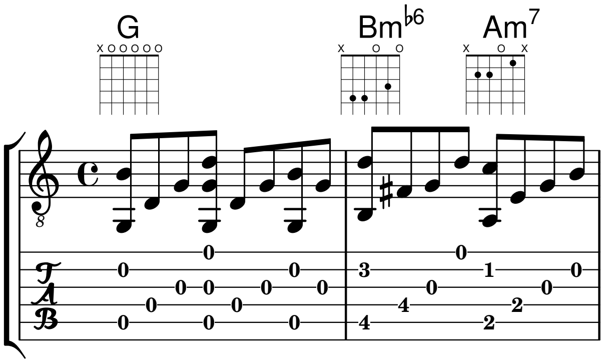

Figure 1: Staff, tab, and chord chart notations combined. Guitar tuned to *open G.*

position changes. Therefore, it is necessary to know the tuning of the guitar, if the tablature transcription is to be encoded correctly.

Regarding performance information, if someone wishes to learn or analyse a guitar recording 'by ear', then information about the tuning and/or capo position would facilitate these activities.

2.3 Guitar Characteristics

Inharmonicity. A simple analysis of the relationship between the harmonic partials and the fundamental frequency of a vibrating string—where f_n is the frequency of the nth harmonic, f_1 is the frequency of the 1st harmonic, and n is the harmonic number—can be written as $f_n = nf_1; n \geq 1$. However, the harmonics in an actual string are higher in frequency than predicted here. This phenomenon is known as *inharmonicity.* The occurrence of inharmonicity in strings, due to their internal stiffness, was first recognised by Lord Rayleigh [42].

The inharmonicity coefficient B is expressed from the radius of the string r, the string's tension T, its lengh L, and Young's modulus Y as: $B = \frac{\pi^3 Y r^4}{4TL^2}$. An adjustment to $f_n = nf_1$ produces the following equation $f_n = nf_0\sqrt{1+Bn^2}$, where f_0 is the fundamental frequency in the absence of inharmonicity i.e., when $B = 0$. In reality $B > 0$ as there is always some internal stiffness in physical strings. Thus, the new harmonic partials will be higher in frequency compared to the corresponding partials in a string with no stiffness [38].

We can define the inharmonicity of a string as "the deviation of the partials from integer multiples of the string's fundamental vibration frequency" [38]. Inharmonicities are inherent in vibrating strings, and the inharmonicity of a guitar string is determined by its radius, tension, length, and Young's modulus. A hypothesis can be derived from this information. The string tension and string length required to produce a particular note on a guitar string change depending on the open note the string is tuned to. The inharmonicity of a string also changes depending on its length and tension. Thus, *a neural network may be able to learn distinct inharmonicity features exhibited by different guitar tunings; the network could then use these features to differentiate between guitar tunings.*

Chord Voicing. "A chord voicing refers to the placement of notes in a chord structure" [27, page 1]. When the order that the notes of a chord appear in changes, or when the octaves that the notes appear in change, different voicings of the chord tones are produced. The voicings used in guitar music often vary as a result of the guitar tuning. To illustrate this, we can compare an open D chord[4] played in standard and open D tuning (see Table 1). Although both voicings contain the same three notes the voicings are clearly different: 6 notes are played in open D, whereas only 4 are played in standard; the order the notes appear in varies; the octaves the notes appear in are sometimes distinct (e.g., F#); the strings notes are played on differ (except D_3). This example highlights how altering the tuning significantly changes the voicing of chords, even when the chords are very simple. Guitar music generally features multiple chords, and these chords can be much more complex than the D chord presented above—this creates the potential for many distinct voicings. Thus, *we hypothesise that each guitar tuning has a collection of signature voicings associated with it; a neural network can learn these voicings as features and use them to identify guitar tunings from audio.*

Table 1: Open D chord in standard tuning and open D tuning.

Tuning	String Number (x = *no note*)					
	6th	5th	4th	3rd	2nd	1st
Standard	x	x	D_3	A_3	D_4	$F\#_4$
Open D	D_2	A_2	D_3	$F\#_3$	A_3	D_4

Pitch Range. The lowest pitch that is playable on the guitar can vary depending on the tuning.[5] For instance, in open D the lowest pitch is a tone lower than in standard tuning. In this type of scenario it should be relatively easy to train an NN—or configure a pitch estimation algorithm—to exclude certain tunings when pitches below a given threshold occur. However, in real-world scenarios certain attributes of the guitar make the task more complicated: (i) a capo raises the lowest pitch that is playable, (ii) guitar tunings do not always conform with the A440 pitch standard, and (iii) a tuning maintains its interval profile when every string is tuned up or down by an equal number of semitones. Abundant evidence of guitarists utilising these attributes was found when collecting acoustic guitar recordings and transcriptions for this study. Dataset 1 (see Section 3.1.2)—which is comprised of song recordings by Joni Mitchell—provides evidence regarding each of the respective attributes: (i) a capo is used on 25 of the 49 songs, (ii) the tuning deviates from A440 on various live recordings, (iii) certain common tunings are downtuned e.g., on the 'The Gallery' from the album *Clouds*, the guitar has the interval profile of open G, but the strings are downtuned by a semitone. Moreover, similar evidence regarding these attributes was found when we analysed official GuitarPro tablatures [41] by various artists, and amateur acoustic guitar performances from the AudioSet dataset [14]. Thus, since these attributes are frequently used by guitarists, it is important to consider them if robust guitar transcription systems are to be developed.

Harmonic Spectrum. A guitar produces notes when the strings vibrate between the bridge and nut or bridge and frets. A recent study [23] indicates that roughly 20% of the notes used by guitarists,

[4] An *open chord* is a chord that contains strings that are not fretted.

[5] The tuning can also vary the highest playable pitch, but guitarists use the upper register of the guitar less frequently [23].

playing in standard tuning without the use of a capo, are played on the open strings (i.e., 20% of vibrations occur between the bridge and nut). The harmonic spectrum differs between open and fretted notes. These differences are perceived by humans as subtle variations in timbre; experienced musicians can differentiate between open and fretted notes by listening attentively. Thus, *we hypothesise that variations in the harmonic spectrum between open and fretted notes can be learned as features by a neural network, and used to determine the pitch of open strings*. However, it is important to note that if a capo is used on a guitar these, differences in the harmonic spectrum would not occur as the strings could only vibrate between the bridge and frets—although the capo might produce other distinct variations in the harmonic spectrum in this scenario.

Pitch Classes, Scales, and Keys. Evidence suggests that there are tuning-specific preferred pitch classes, scales, and keys. In [23], a diverse corpus of 1022 professionally transcribed guitar tablatures was analysed to determine the most common string, fret, and hand positions used by guitarists playing in standard tuning, without the use of a capo. The pitch classes that occurred most frequently in the corpus were E, G, A, B, C, and D. These notes are used in the keys of C major, G major, A minor, and E minor, and also feature in their corresponding heptatonic and pentatonic scales; this indicates that these keys and scales are likely to occur frequently in standard tuning. When an open tuning is used, the chord produced by the open strings may be an indicator of the key—an analysis of 40 songs in 5 different tunings supports this theory,[3] but a larger sample needs to be analysed to provide more conclusive evidence. *We hypothesise that a neural network can learn tuning-specific preferred pitch classes, scales, and keys as features for guitar tuning classification.* Key changes, consonance/dissonance, and chromaticism may also be more common in certain tunings.

3 METHODS

3.1 Data

3.1.1 Labels. A label, indicating the guitar tuning used on a song, was assigned to every audio file. The main references used to determine the guitar tuning featured on a particular recording were *The Joni Mitchell Complete—Guitar Songbook Edition* [5] and official GuitarPro transcriptions [41]—these were reliable sources as the transcriptions were made by professional musicians (if multiple tunings were featured the file was discarded). In many of the recordings/transcriptions, a capo is used on the guitar. Therefore, we devised a flexible labelling system that enables the tuning classes to be determined by absolute pitch or interval profile. The dataset labelling method is described below:

Example: `x75435_EBEG#BE_C2_Cactus_Tree_StaS_12`

- `x75435`: Denotes the interval profile. The `x` represents the 6th string, and the numbers represent the intervals between strings in semitones e.g., `x7` indicates that the 5th string is 7 semitones above the 6th string in pitch, `75` indicates that the 4th string is 5 semitones above the 5th string etc.
- `EBEG#BE`: Shows the note each string is tuned to (6th to 1st string)
- `C2`: This indicates the capo position. The number after `C` denotes the fret position of the capo. *No capo* = `C0`
- `Cactus_Tree`: The name of the song (sometimes abbreviated in all caps)
- `StaS`: Indicates the album (when abbreviated uppercase and lowercase letters are used)
- `12`: Clip number (only relevant to files in the training set)

3.1.2 Dataset 1: Joni Mitchell Song Recordings. Dataset 1 consists of 49 WAV files (44.1 kHz, 16 bit, stereo). The files contain audio of variable length (≈ 3 min). The guitars in the audio are in various different tunings and are labelled accordingly.[3] The audio content of the dataset was derived from Joni Mitchell song recordings. Mitchell's songs feature steel string acoustic guitar performances in altered tunings, making them suitable for GTC tasks. Moreover, using Mitchell's songs seems apt as her music marks a compelling moment in the history of guitar tunings. Altered tunings were an integral part of her sound. Mitchell's popularity in the sixties and seventies introduced listeners to an array of unfamiliar chord voicings, and her guitar playing inspired guitarists to experiment with altered tunings [43]. The songs were recorded between 1968 and 1972. They were taken from 5 studio albums [30–34] and 2 live albums [28, 29]. The guitar performances on the recordings consist predominantly of chord progressions that are fingerpicked or strummed; 14 different tunings are used by Mitchell. All the recordings featured vocals, and sometimes instruments other than the guitar were also present. Thus, the source separation algorithm Spleeter 2.4.0 [20] was applied to isolate the guitar signal. In preliminary tests, models were trained independently on unprocessed audio and source separated audio. When these models were evaluated, the results indicated that source separation was beneficial to classification performance (see Section 4.1). Therefore, source separation was applied in all the subsequent tests. Spleeter is a powerful tool, but it does produce artifacts (e.g., extraneous filtering, distortion). Furthermore, the Spleeter model had no explicit 'guitar' stem, so the 'other' stem from the model output was considered to contain the guitar parts. Thus, while the model removed vocals, speech, piano, bass, and drums from the audio, any other instrumentation remained in the audio along with the guitar. However, the use of other instruments was relatively infrequent, so the detrimental effect this could have on model performance is thought to be low; when the instrumentation of a song was considered to be problematic for the Spleeter model, it was discarded from the dataset. On some of the studio recordings, two guitars in different tunings were present simultaneously. These recordings could not be labelled accurately, so they were removed from the dataset. To the best knowledge of the authors, an algorithm that can disentangle multiple guitar parts from recordings is not available at the time of writing. Mitchell's live recordings contained sections in which she was tuning her guitar. The audio could not be correctly labelled for these sections, so they were removed with an audio editor. The live recordings also contained sections in which Mitchell was talking to the audience. However, after the guitar had been isolated, these sections contained silence; these sections were also removed.

3.1.3 Dataset 2: Multi-artist Song Recordings. Dataset 2 consists of 54 WAV files (44.1 kHz, 16 bit, stereo). The files contain audio of variable length (≈ 3 min). There are 5 tuning classes in the dataset: standard, drop D, DADGAD, open D, open G. The recordings are mainly by singer-songwriters. All recordings feature steel string acoustic guitar. When labelling the audio, official GuitarPro tablatures were used to determine the tuning and capo position [41]. Most songs are studio recordings, but live recordings are also present. Dataset 2 was created after Dataset 1, when a more advanced source separation algorithm—Demucs 4.1.0a1 [44]—was available. Demucs featured an experimental model with an explicit 'guitar' stem. Two expert listeners compared the Demucs 'guitar' and Spleeter 'other' outputs. The Demucs output was judged to provide superior audio quality. It also removed a wider variety of instruments from recordings (although it could be temperamental in this respect); this made Demucs more flexible and helped to streamline the audio editing process. Additionally, Dataset 2 was created so trained models could be tested using data from disparate conditions. Thus, applying a different source separation algorithm had the desirable effect of increasing the disparity between datasets.

3.1.4 Dataset 3: Multi-artist Synthetic Audio. Dataset 3 consists of 245 WAV files (44.1 kHz, 16 bit, mono). The files contain high-quality sample-based guitar audio renderings of song tablature transcriptions. The audio is of variable length ($\approx$ 3 min). There are five tuning classes—standard (90 songs), drop D (86), open D (39), DADGAD (18), open G (12). The audio in Dataset 3 was created by rendering tablatures as audio with Ample Sound *AGT* [45], a sample-based acoustic guitar Virtual Studio Technology (VST) instrument. AGT includes a 'tuner', which allows the user to downtune each string by 1 or 2 semitones. This functionality makes it possible to tune the guitar to various common tunings—analysis of the downtuned audio produced by the VST indicated that the guitar samples were recorded at the pitches they corresponded to (i.e., resampling or sample rate pitch shifting was not applied). The tablatures used were professionally transcribed, so the tuning information and note/string combinations in the transcriptions were accurate. The synthetic data was generated by loading the tablatures in the VST and exporting them as audio renderings; this process was automated with Dawdreamer (a Python-based audio framework that emulates a DAW) [6]. Regarding the settings in the VST: the data was rendered in mono; a 'neutral' guitar timbre was employed (i.e., no audio effects were applied) using a single VST steel string acoustic guitar model; the playing style was set to 'fingerstyle'. In this work, the experiments conducted using Dataset 3 investigated an NNs capacity to learn useful features related to pitch, scale, and key for the acoustic classifcation of guitar tunings. Thus, a diverse range of timbres was not a priority. However, using the framework developed in this study, a large synthetic audio dataset, with a diverse range of timbres and VST guitar models, will be created—this will be used to study guitar tuning characteristics such as inharmonicity and the harmonic spectrum.

Dataset 3 was used to investigate the ability of a CNN to classify tunings by absolute pitch; to simplify the problem the use of a capo was not permitted. However, many of the transcriptions featured a capo. Therefore, we transposed the tablatures by setting the capo parameter to 0 (i.e., no capo) before rendering. Furthermore some tablatures featured 'downtuning' or 'uptuning'; these transcriptions had to be assigned to an appropriate pitch range. Both of these procedures were automated with PyGuitarPro [1].

While a synthetic guitar dataset already exists [52], to the best knowledge of the authors, Dataset 3 is the first sample-based guitar dataset that features a variety of altered tunings. Furthermore, a different method was used to render the symbolic music as audio. In [52], string level MIDI is rendered individually and the string-level audio signals are then mixed by averaging. Instead, we loaded each tablature via Dawdreamer in the VST and rendered the string-level MIDI data jointly using the VST's specialised TabPlayer functionality. With this highly efficient approach the GuitarPro tablature format preserves the note/string combinations, and the joint string-level rendering produces synthetic guitar audio that sounds cohesive and dynamic.

3.2 Experiments

The use of a capo on the guitar in various recordings is likely to make the classification task considerably more challenging. Therefore, *interval profile data partitions were made that featured a capo (3.2.1–3.2.2), they were used in 2 class experiments (4.1, 4.2, and 4.3).* In these partitions and experiments the terms *open D* and *open G* refer to the *tuning type* (e.g., *open D* = $x75435$) and not the absolute pitch. Additionally, *an absolute pitch data partition was made that featured no capo (3.2.3), it was used in 5 class experiments (4.4)*—here, the class names refer to the absolute pitch (e.g., *open D = DADF#AD*).

3.2.1 Data Partition 1: Open D/Other. This partition only features data from Dataset 1. It was used to (i) evaluate how well *CNN 1* (3.2.6) could differentiate between the *open D type* tuning and various other tunings, (ii) determine a suitable sample length for CNN 1, (iii) evaluate how audio processing techniques affected performance. Regarding the final point, *only in preliminary experiments (4.1) was a subset of Data Partition 1 used that did not have audio processing applied (e.g., source separation, removal of extraneous material). Isolated guitar audio was used in all subsequent experiments.* The labelled and edited song files were divided into a training set ($\approx$ 80%) and a test set ($\approx$ 20%). In each set there were 2 classes: (i) the positive class `x75435` (open D type), and (ii) the negative class `other` (tunings not matching x75435). The song files were randomly selected for each set. The selection process was random, aside from the following conditions: different recordings of the same song could not appear in both the training set and test set; studio recordings and live recordings should appear in both the training set and test set. These conditions were enforced to prevent the model from overfitting on characteristics that were not related to the guitar tuning. Although Dataset 1 is relatively small, and only features one artist, the audio recordings that it is comprised of were recorded with a variety of different tools in various different locations, so a catalogue of varied data is spread across the partitions. This should help to constrain a model, so it disregards extraneous features such as recording conditions, and focuses instead on features related to the guitar tuning. GTC is a novel and complex task, so limiting some early experiments to the work of a single artist and 2 tuning classes, helps to provide some consistency and simplify the problem. The training set contained 15 song files from the `x75435` class, and 14 song files from the `other` class. The test set contained the 4 song files from the `x75435` class, and 4 song files from the `other` class.[3] Songs were sliced into 1s/3s/9s clips prior to input into CNN 1. The clips from the training set were randomly split into 2 subsets—*training* = 80%, *validation* = 20% (this also occurred in the subsequent partitions/experiments).

3.2.2 Data Partition 2: Open D/Open G. This partition features 23 recordings from Dataset 1 and 8 recordings from Dataset 2. It was used to evaluate the ability of a CNN to classify guitar tunings as x75435 (*open D type*) or x57543 (*open G type*). Furthermore, it was used to compare model performance on authentic test data from similar conditions with a Joni Mitchell (*JM*) test set, and disparate conditions with a multi-artist (*MA*) test set. The files in Dataset 1 were divided into a training set (≈ 80%) and a JM test set (≈ 20%). A sample of 8 suitable songs was taken from Dataset 2, for the MA test set. In each set there were two classes: (i) x75435 (*open D type*), and (ii) x57543 (*open G*). The training set contained 11 song files from the x75435 class, and 8 song files from the x57543 class. The JM test set contained 2 song files from each class. The MA test set contained 4 songs from each class.[3] Songs were sliced into 9s clips prior to input into CNN 1.

3.2.3 Data Partition 3: Multiclass. This partition features 245 recordings from Dataset 3 and 46 recordings from Dataset 2. It was used to evaluate how well a CNN could differentiate between 5 tuning classes (3.1.4), when trained on synthetic data and tested on synthetic data from similar conditions and authentic data from disparate conditions. It is the only data partition in which a capo is not present in any of the guitar audio; this was done so that we could more easily investigate the ability of an NN to learn features related to pitch class for tuning classification. The files in Dataset 3 were divided into a training set (≈ 80%) and a test set (≈ 20%). A sample of 46 suitable songs was taken from Dataset 2, so the trained model could be evaluated using authentic data from disparate conditions. Songs were sliced into 30s clips (preliminary tests indicated this was an effective length) prior to input into CNN 2.

3.2.4 Audio Pre-processing. To preserve the high frequency content and the dynamic range, the sample rate was set at 44.1 kHz and the bit depth at 16-bit, with the log mel spectrogram input. These features were deemed to be less vital for the chromagram input, so the audio was downsampled to 22.5 kHz. Normalisation was applied to ensure the amplitude was consistent. To retain the dynamic range of the music, song files were normalised in their entirety. The alignment level was set at -18 dBFS as the EBU recommends this as the maximum alignment level in digital systems [48]. Models may learn different features depending on the sample length; to provide insights into the effect sample length has on model performance, the CNN was trained and tested independently on samples of 1s, 3s, 9s, 30s in length. Song files were sliced into samples of the desired length; zero padding was automatically applied to samples that were too short, and samples that were too long were automatically cropped. If clips were multichannel, the first audio channel was used as input and any additional channels were ignored.

3.2.5 Input Representations. The input representations used in the experiments were spectrograms (4.1), *log mel spectrograms* (4.2–4.3), and *chromagrams* (4.4) (see Fig. 2–3). Perceptually relevant representations of audio data can improve the performance of DL models designed for MIR tasks [4]. The log mel spectrogram was chosen as an input representation as it models human perception of loudness and pitch, and it is "efficient in its size while preserving the most perceptually important information" [8]. Log mel spectrograms are also used effectively in DL frameworks for tasks such as generative audio modelling [12] and AMT [19]. To produce the log mel spectrogram: the audio was transformed to the frequency domain by STFT. The STFT was converted into a mel-frequency STFT with 128 mel-filterbanks. Window size for the FFT was set to 512 samples. The Hann window function was applied to the FFT windows. The hop length between STFT windows was set to 256 samples. The decibel scale was applied to the mel spectrogram tensor with the minimum negative cutoff set to -80 dB.

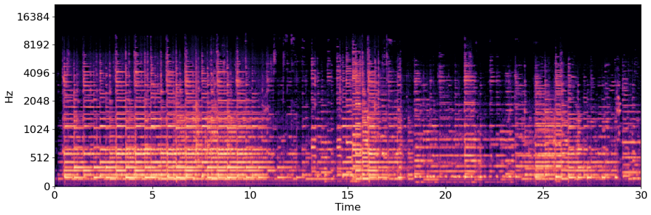

Figure 2: Log mel spectrogram sample from Dataset 1.

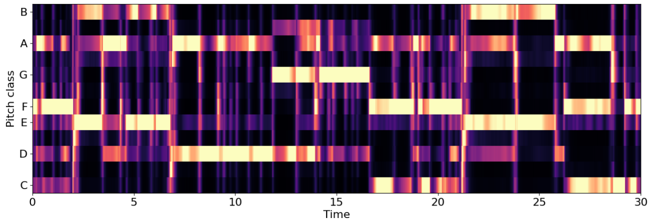

Figure 3: Chromagram sample from Dataset 2.

The second input representation was a chromagram. Pitch can be divided into two elements, *tone height* and *chroma*. "The tone height refers to the octave number and the chroma to the respective pitch spelling attribute contained in the set $\{C, C^{\#}, D, \ldots, B\}$... A *pitch class* is defined as the set of all pitches that share the same chroma" [36, p. 123]. Chroma features aggregate all spectral information that pertain to a given pitch class into one coefficient. A chromagram can be derived from a pitch-based log-frequency spectrogram by summing all pitch coefficients that are part of the same chroma [36]. While a large amount of information is lost in a chroma representation, for certain tasks, "this loss in information is desired since it introduces a high degree of robustness to variations in timbre" [36, p. 124]. Additionally, it is more compact than many other input representations, and it allows us to discard tone height as a feature; this is useful when studying the ability of an NN to use key and scale as features, while ignoring other pertinent features that can be easily detected by less complex algorithms (e.g., a basic pitch estimation algorithm could be used to determine the lowest pitch that is playable in a given tuning).

3.2.6 CNNs. Two relatively small CNNs were chosen to ensure the systems were economical and to reduce overfitting. *CNN 1* was adapted from an architecture designed for bird audio detection [17, 26]; in preliminary tests it performed well when trained on spectrograms, outperforming a much larger CNN, and an NN comprised only of dense layers. It was used in the 2 class experiments with spectrograms (4.1) and log mel spectrograms (4.2–4.3). *CNN 2* was adapted from an architecture designed for keyword spotting [49, 51]. It was used in the 5 class experiment (4.4). In preliminary tests it performed well when trained on synthetic data. The model was originally designed to take MFCCs as input, but various inputs were tried in our preliminary tests—a chromagram was chosen as input, since the performance of models trained on chromagrams was comparable with models trained on more complex and memory intensive representations (e.g., CQTs, spectrograms). Additionally, most information related to the harmonic spectrum, pitch range, and inharmonicity is not present in the chromagram, which enabled us to make inferences about how the CNN may learn features such as pitch classes, scales, and key.

CNN 1[3] was comprised of 4 convolutional layers, followed by 3 dense layers. A sequence of 4 combinations of convolution and pooling condensed the input into 16 feature maps. The condensed features were classified by 3 dense layers with 256, 32 and 1 unit(s). A Leaky ReLU activation function was used for hidden layers and a sigmoid function for the output layer. Batch normalisation was applied after every layer, and dropout was applied after every dense layer [46]. The total number of trainable network parameters was 679,889 with 9s samples as input. Training was carried out over 25–50 epochs; learning curves were used to select a suitable number of training epochs. The batch size was 16. Data was shuffled to ensure the network was unaffected by the order in which samples were presented to it. A binary-crossentropy loss function calculated the difference between the network output and the expected output. The learning rate was set to 0.001 and the Adam optimisation algorithm updated the network weights. Most models could be trained within minutes using an NVIDIA GeForce RTX 3060 Ti GPU, due to the relatively small size of the datasets/networks. Trained models were tested using independent test sets. Inputted test samples returned probability outputs between 0 and 1 in a continuous range. This output data was used to plot ROC curves and determine AUC scores. The probability outputs were also used to make nominal predictions (e.g., "open D", "other") on samples and songs.

CNN 2[3] was comprised of 3 convolutional layers, followed by 2 dense layers. A sequence of 3 combinations of convolution and pooling condensed the input into 64 feature maps. The condensed features were classified by 2 dense layers with 64 and 5 units. A ReLu activation function was used for the hidden layers and a softmax function for the output layer. Batch normalisation was applied to the convolutional layers and dropout was applied after the first dense layer. The total number of trainable network parameters was 353,605 with 30s samples as input. Training and testing was the same as with CNN 1, except the batch size was 32 and the sparse categorical crossentropy loss function was used.

4 RESULTS AND DISCUSSION

Models were trained and tested 5 times with different random seeds (e.g., random weight initialisation, random training/validation split etc.). Receiver operating characteristic (ROC) curve and area under the curve (AUC) were used to evaluate performance in the 2 class studies (see Sections 4.1, 4.2, and 4.3). The ROC curve shows the true positive rate against the false positive rate at all classification thresholds. AUC gives an aggregate measure of the 2D area under the ROC curve; an advantage of AUC is its classification-threshold invariance [16]. F-score was used in the 5 class study (see Section 4.4). The F-scores were calculated for each label and their weighted average was found—this approach was appropriate for the multiclass targets and accounted for label imbalance. Accuracy was also used (Sample Classification = ACC 1, Song Classification = ACC 2). In preliminary tests (see Section 4.1) a spectrogram was used as input; the best performing model in these tests attained an AUC of 0.771. In Sections 4.2 and 4.3 a log mel spectrogram was used as input. This improved performance—the best performing model attained an AUC of 0.893. Section 4.4 was the only experiment that used CNN 2, chromagrams, synthetic data, and 5 tuning classes; the best performing model achieved a classification accuracy of 73.9% on the synthetic test set and 67.4% on the authentic test set.

Table 2: AUC for CNN 1 models with audio processing.

Model	AUC	
	Mean	SD
Source sep + edit	0.771	(0.06)
Source sep	0.749	(0.04)
No processing	0.698	(0.03)

4.1 Open D/Other: Data Processing Study

Table 2 indicates how differently processed versions of Data Partition 1 affected performance. CNN 1 performance increased noticeably when source separation was applied. Model performance also increased as a result of *audio editing* (see Section 3.1.2).

4.2 Open D/Other: Sample Length Study

Figure 4 and Table 3 show that the model trained on 9s samples was most effective, and performance deteriorated with shorter samples. We could infer from this that, for GTC, fine-grained features (e.g., inharmonicity, harmonic spectrum) are less useful than longer term temporal features (e.g., chord voicings, transitions between chords). However, it is also possible that the model was unable to learn these fine-grained features due to other factors such as the small size of the dataset and extensive capo usage. Additionally, the CNN was not originally designed to receive samples of less than 3s, so it is possible that an NN that is specially designed to extract fine-grained features could derive information that is equally useful from shorter samples, and the improved performance with longer samples is simply an attribute of this particular architecture. Table 3 shows that models trained on 9s and 3s clips achieved high AUC and accuracy scores. The 9s models achieved an average song classification accuracy of 97.5%. This result is very promising, especially as the model had to make classification decisions irrespective of the position of the capo on the guitar. The results suggests that CNNs can be used effectively for GTC. However there are a number of caveats that should be considered when assessing this result: (i) there were only 8 songs in the test set, (ii) the task of multiclass tunings classification is likely to be considerably harder, (iii) the data in the train and test sets was derived from a single artist.

4.3 Open D/Open G Study

Table 4 shows that the model performs well on test data from similar conditions, achieving a song classification accuracy of 95.0%.

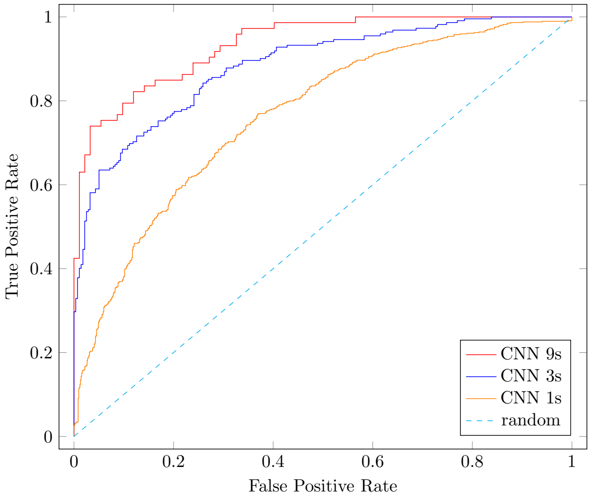

Figure 4: ROC plots (selected) for Open D/Other models.

Table 3: AUC and ACC for CNN 1/Data Partition 1.

Model	AUC		ACC 1		ACC 2	
	Mean	SD	Mean	SD	Mean	SD
9s	0.893	(0.03)	81.8%	(2.30)	97.5%	(5.00)
3s	0.823	(0.06)	74.2%	(5.37)	87.5%	(13.69)
1s	0.627	(0.12)	57.4%	(10.15)	52.5%	(14.58)

Table 4: AUC and ACC scores for CNN1/Data Partition 2.

Model	AUC		ACC 1		ACC 2	
	Mean	SD	Mean	SD	Mean	SD
JM	0.867	(0.05)	80.0%	(4.58)	95.0%	(10.0)
MA	0.577	(0.05)	57.3%	(4.30)	65.0%	(5.0)

This result, which is comparable with the results in Section 4.2, is encouraging; it provides further evidence that NNs can be used effectively for GTC. However, while the model exhibited skill when tested with data from disparate conditions, accuracy decreased markedly (-30%).

4.4 Multiclass Study

Table 5 and Table 6 show that the model performed reasonably well on the synthetic test set, attaining an average F-score of 0.67 across the 5 classes. The model did not perform as well on the authentic test set, with an average F-score of 0.52. Nevertheless, this is a reasonable result if we consider the following factors: the model had to differentiate between five tuning classes; the chromagram input does not include lowest pitch range information which is likely to have made the identification of standard tuning much easier; the model was trained on an imbalanced synthetic dataset; the authentic test set featured noisy real-world data from disparate conditions. Figure 5 shows tuning predictions from a selected model that was tested on real songs from Dataset 2.

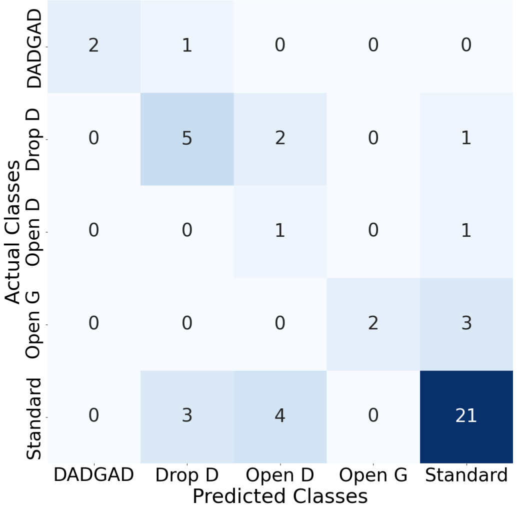

Figure 5: Predictions on an independent test set of real songs.

Table 5: Average F-score for five tuning class models.

Test Set	F_1 Clips		F_1 Songs	
	Mean	SD	Mean	SD
Synthetic	0.649	(0.03)	0.666	(0.05)
Authentic	0.474	(0.06)	0.516	(0.10)

Table 6: Average accuracy for five tuning class models.

Test Set	ACC Clips		ACC Songs	
	Mean	SD	Mean	SD
Synthetic	65.4%	(0.03)	67.8%	(0.05)
Authentic	43.4%	(0.07)	48.3%	(0.11)

5 CONCLUSION AND FUTURE WORK

This research provides evidence that neural networks can classify guitar tunings from audio. Future work will investigate capo and open string detection with spectrograms/CQTs. The modelling and generalisation proficiency of DL models improves significantly with more data. Therefore, a priority in future work will be the creation of a large synthetic GTC dataset with a diverse range of timbres and time alignment between audio and tablature; this can be easily achieved with the approach proposed in Section 3.1.4. An algorithm that outputs a separate tuning prediction for each string would provide greater flexibility for GTC. It could enable the identification of tunings not present in the training data. However, the creation of such an algorithm is likely to be challenging, so the feasibility of this approach requires investigation. Traditional DSP and ML methods may be more appropriate for certain GTC tasks, so they require investigation. Audio segmentation in this work did not account for the onset of notes/chords/bars. Future work, will investigate onset detection techniques for GTC, so individual chords, notes, and bars can be isolated, before input into a model.

REFERENCES

[1] Sviatoslav Abakumov. 2023. PyGuitarPro. Retrieved 2024-02-21 from https://github.com/Perlence/PyGuitarPro?tab=readme-ov-file

[2] Jakob Abeßer. 2013. Automatic String Detection for Bass Guitar and Electric Guitar. In *From Sounds to Music and Emotions*, Mitsuko Aramaki, Mathieu Barthet, Richard Kronland-Martinet, and Sølvi Ystad (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 333–352. https://doi.org/10.1007/978-3-642-41248-6_18

[3] Ana M. Barbancho, Anssi Klapuri, Lorenzo J. Tardon, and Isabel Barbancho. 2012. Automatic Transcription of Guitar Chords and Fingering From Audio. *IEEE Transactions on Audio, Speech, and Language Processing* 20, 3 (2012), 915–921. https://doi.org/10.1109/TASL.2011.2174227

[4] Emmanouil Benetos, Simon Dixon, Zhiyao Duan, and Sebastian Ewert. 2019. Automatic Music Transcription: An Overview. *IEEE Signal Processing Magazine* 36, 1 (2019), 20–30. https://doi.org/10.1109/MSP.2018.2869928

[5] Joel Bernstein and Daniel Libertino. 1996. *Joni Mitchell Complete Guitar Songbook Edition.* Alfred Publishing Co., Inc., Los Angeles, CA, USA.

[6] David Braun. 2021. DawDreamer: Bridging the Gap Between Digital Audio Workstations and Python Interfaces. https://doi.org/10.48550/arXiv.2111.09931 arXiv:2111.09931 [cs.SD]

[7] Gregory Burlet and Abram Hindle. 2017. Isolated guitar transcription using a deep belief network. *PeerJ Computer Science* 3 (2017), e109–e109. https://doi.org/10.7717/peerj-cs.109

[8] Keunwoo Choi, György Fazekas, Kyunghyun Cho, and Mark Sandler. 2017. A Tutorial on Deep Learning for Music Information Retrieval. (2017). https://doi.org/10.48550/arXiv.1709.04396 arXiv:1709.04396

[9] Frank Cwitkowitz, Toni Hirvonen, and Anssi Klapuri. 2023. Fretnet: Continuous-Valued Pitch Contour Streaming For Polyphonic Guitar Tablature Transcription. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* 1–5. https://doi.org/10.1109/ICASSP49357.2023.10094825

[10] Nollene Davies. 1994. The Guitar in Zulu "maskanda" Tradition. *The World of Music* 36, 2 (1994), 118–137. http://www.jstor.org/stable/43561390

[11] Christian Dittmar, Andreas Männchen, and Jakob Abeber. 2013. Real-time guitar string detection for music education software. In *2013 14th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS).* 1–4. https://doi.org/10.1109/WIAMIS.2013.6616120

[12] Jesse Engel, Lamtharn Hantrakul, Chenjie Gu, and Adam Roberts. 2020. DDSP: Differentiable Digital Signal Processing. *arXiv e-prints* (Jan. 2020). https://doi.org/10.48550/arXiv.2001.04643

[13] Xander Fiss and Andres Kwasinski. 2011. Automatic real-time electric guitar audio transcription. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* 373–376. https://doi.org/10.1109/ICASSP.2011.5946418

[14] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. 2017. Audio Set: An ontology and human-labeled dataset for audio events. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2017).* New Orleans, LA.

[15] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning.* MIT Press. http://www.deeplearningbook.org.

[16] GoogleDevlopers. 2022. Classification: ROC Curve and AUC. Retrieved 2023-05-02 from https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

[17] Thomas Grill and Jan Schlüter. 2017. Two convolutional neural networks for bird detection in audio signals. In *2017 25th European Signal Processing Conference (EUSIPCO).* 1764–1768. https://doi.org/10.23919/EUSIPCO.2017.8081512

[18] Mark Hanson. 1995. *The complete book of alternate tunings.* Accent on Music, West Linn, OR, USA.

[19] Curtis Glenn-Macway Hawthorne, Ian Simon, Rigel Jacob Swavely, Ethan Manilow, and Jesse Engel. 2021. Sequence-to-Sequence Piano Transcription with Transformers. https://doi.org/10.48550/arXiv.2107.09142 arXiv:2107.09142

[20] Romain Hennequin, Anis Khlif, Felix Voituret, and Manuel Moussallam. 2020. Spleeter: a fast and efficient music source separation tool with pre-trained models. *The Journal of Open Source Software* 5, 50 (June 2020), 2154. https://doi.org/10.21105/joss.02154

[21] Eric J. Humphrey and Juan P. Bello. 2014. From music audio to chord tablature: Teaching deep convolutional networks toplay guitar. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* 6974–6978. https://doi.org/10.1109/ICASSP.2014.6854952

[22] Yogesh Jadhav, Ashish Patel, Rutvij H. Jhaveri, Roshani Raut, and Saqib Hakak. 2022. Transfer Learning for Audio Waveform to Guitar Chord Spectrograms Using the Convolution Neural Network. *Mob. Inf. Syst.* 2022 (jan 2022), 11 pages. https://doi.org/10.1155/2022/8544765

[23] David Regnier Jules Cournut, Nicolas Martin. 2021. What are the most used guitar positions?. In *8th International Conference on Digital Libraries for Musicology (DLfM2021).* 84–92. https://doi.org/10.1145/3469013.3469024

[24] Christian Kehling, Jakob Abeßer, Christian Dittmar, and Gerald Schuller. 2014. Automatic Tablature Transcription of Electric Guitar Recordings by Estimation of Score-and Instrument-Related Parameters. In *DAFx.* 219–226.

[25] Varun Khatri and Lukas Dillingham. 2020. *Guitar Tuning Identification.* Technical Report. University of Rochester, Department of Electrical and Computer Engineering.

[26] Sidrah Liaqat, Narjes Bozorg, Neenu Jose, Patrick Conrey, Antony Tamasi, and Michael T Johnson. 2018. Domain tuning methods for bird audio detection. DCASE.

[27] MasterClass. 2021. How to Use Chord Voicing in Music. Retrieved 2023-05-11 from https://www.masterclass.com/articles/how-to-use-chord-voicing-in-music

[28] Joni Mitchell. 1968. Club 47. Troubador Records - CD 5 060446 070178.

[29] Joni Mitchell. 1968. Philadelphia Folk Festival. Retrieved June 29, 2023 from https://www.youtube.com/watch?v=9futFx1iEU4

[30] Joni Mitchell. 1969. Clouds. Reprise Records - CD 6341.

[31] Joni Mitchell. 1970. Ladies of the Canyon. Reprise Records - CD 6376.

[32] Joni Mitchell. 1971. Blue. Reprise Records - CD 2038.

[33] Joni Mitchell. 1972. For the Roses. Elektra/Asylum Records - CD 7559-60624-2.

[34] Joni Mitchell and David Crosby. 1968. Song to a Seagull. Reprise Records - CD 6293.

[35] Himadri Mukherjee, Ankita Dhar, Sk. Md. Obaidullah, K. C. Santosh, Santanu Phadikar, and Kaushik Roy. 2019. Segregating Musical Chords for Automatic Music Transcription: A LSTM-RNN Approach. In *Pattern Recognition and Machine Intelligence.* Springer International Publishing, Cham, 427–435. https://doi.org/10.1007/978-3-030-34872-4_47

[36] Meinard Müller. 2015. *Fundamentals of Music Processing.* Springer, Switzerland. https://doi.org/10.1007/978-3-319-21945-5

[37] Meinard Müller and Tech Anssi Klapuri. 2014. *Automatic transcription of bass guitar tracks applied for music genre classification and sound synthesis.* Ph. D. Dissertation. Gustav-Kirchhoff-Straße 1, 98693 Ilmenau, Germany.

[38] Chris J Murray and Scott B Whitfield. 2022. Inharmonicity in plucked guitar strings. *American Journal of Physics* 90, 7 (2022), 487–493. https://doi.org/10.1119/5.0064373

[39] Michael A Nielsen. 2015. *Neural networks and deep learning.* Vol. 25. Determination press San Francisco, CA, USA. http://neuralnetworksanddeeplearning.com/.

[40] Julien Osmalsky, Jean-Jacques Embrechts, Marc Van Droogenbroeck, and Sébastien Pierard. May 2012. Neural networks for musical chords recognition. In *Journées d'informatique musicale* (Mons, Belgium).

[41] Guitar Pro. 2023. Tabs. Retrieved 2024-02-17 from https://www.guitar-pro.com/tabs/artists

[42] John William Strutt Baron Rayleigh. 1878. *The theory of sound.* Vol. 2. Macmillan, London, England.

[43] Rikky Rooksby. 2010. *How to Write Songs in Altered Guitar Tunings.* Backbeat Books, London, England.

[44] Simon Rouard, Francisco Massa, and Alexandre Défossez. 2023. Hybrid Transformers for Music Source Separation. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* 1–5. https://doi.org/10.1109/ICASSP49357.2023.10096956

[45] Ample Sound. 2024. Ample Guitar T. Retrieved 2024-02-21 from https://www.amplesound.net/en/pro-pd.asp?id=6

[46] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15, 56 (2014), 1929–1958. http://jmlr.org/papers/v15/srivastava14a.html

[47] Dan Stowell, Michael D. Wood, Hanna Pamuła, Yannis Stylianou, and Hervé Glotin. 2019. Automatic acoustic detection of birds through deep learning: The first Bird Audio Detection challenge. *Methods in Ecology and Evolution* 10, 3 (2019), 368–380. https://doi.org/10.1111/2041-210X.13103 arXiv:https://besjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.13103

[48] European Broadcasting Union. 2000. Alignment level in digital audio production equipment and recorders. EBU Technical Recommendation R68-2000. https://tech.ebu.ch/docs/r/r068.pdf

[49] Valerio Velardo. 2020. Deep Learning Audio Application from Design to Deployment. Retrieved 2024-02-16 from https://github.com/musikalkemist/Deep-Learning-Audio-Application-From-Design-to-Deployment/tree/master

[50] Andrew Wiggins and Youngmoo E Kim. 2019. Guitar Tablature Estimation with a Convolutional Neural Network.. In *Proceedings of the 20th International Society for Music Information Retrieval Conference.* 284–291. https://doi.org/10.5281/zenodo.3527800

[51] Xiaowei Qin Ximin Li, Xiaodong Wei. 2020. Small-Footprint Keyword Spotting with Multi-Scale Temporal Convolution. In *Interspeech 2020.* 1987–1991. https://doi.org/10.21437/Interspeech.2020-3177

[52] Yongyi Zang, Yi Zhong, Frank Cwitkowitz, and Zhiyao Duan. 2024. SynthTab: Leveraging Synthesized Data for Guitar Tablature Transcription. https://doi.org/10.48550/arXiv.2309.09085 arXiv:2309.09085 [cs.SD]