AutoTab: An Interactive System for Automatic Guitar Tablature Transcription

Athos Matos, Daniel Teodolino Barbosa Torres, Adriana Takahashi, Raul Benites Paradeda

[1]Robotics Learning Laboratory, Computer Science Department, State University of Rio Grande do Norte, Natal, Brazil-RN

athosladies,danielteodolino@gmail.com,

adrianatakahashi,raulparadeda@uern.br

Abstract. *This work introduces AutoTab, an interactive system designed to simplify the process of transcribing music into guitar tablature. By combining sound event detection, Convolutional Neural Networks (CNNs), and graph-based optimization algorithms, AutoTab offers an intuitive and accessible solution for musicians of all skill levels. The system employs precise note and chord identification through CNNs, a WebSocket-based server for real-time audio analysis, and specialized algorithms such as the Audio Window Analyser for improved sound event detection and TabGen for optimized tablature generation. Initial results demonstrate robust performance, with the Notes Model achieving 87.8% accuracy in note identification and the Chords Model achieving a 78% Macro F1-score in chord recognition. AutoTab aims to democratize access to high-quality transcription technology, enhancing musical practice and learning for guitarists worldwide. Future work will focus on addressing the challenges in polyphonic transcription and expanding the system's functionality.*

Keywords: Audio processing, Convolutional Neural Networks for music, Analysis of musical elements, Tablature generation, Interactive web application, Note and chord recognition.

1. Introduction

Music is deeply rooted in human culture and provides various forms of enjoyment, particularly for musicians. However, identifying musical elements such as notes and chords remains a challenge, especially for beginners, leading to frustration and, in many cases, the abandonment of musical practice [4, 23]. Guitar tablatures have become a valuable resource for musicians, but manual transcription remains a complex and time-consuming task, even for experienced performers [15, 13].

This challenge is further amplified for guitarists, as a single note can be played in multiple fretboard positions, complicating transcription compared to instruments like the piano, where each note corresponds to a unique piano key [27]. While Automatic Music Transcription (AMT) technologies aim to address these difficulties, they still face significant challenges due to the complexity and harmonic overlap in musical signals. Even human experts often struggle with precise transcription, underscoring the difficulty of designing reliable automated systems [4, 23].

Although commercial solutions like Melody Scanner[1] and Guitar2Tabs[2] exist, their accessibility is often limited by financial constraints. This highlights the relevance of open-source alternatives, fostering greater accessibility and collaborative innovation. In response to this, AutoTab is proposed as an open-source system designed to automatically transcribe audio recordings into guitar tablatures, democratizing access to high-quality transcription technology for musicians of all skill levels.

AutoTab integrates convolutional neural networks (CNNs) for precise note and chord identification, a WebSocket-based server for real-time audio analysis, and specialized algorithms such as the Audio Window Analyser (AWA) for improved sound event detection and TabGen for optimized tablature generation. Extensive testing and validation were conducted to ensure accuracy and practicality, effectively addressing the identified transcription challenges.

This paper is structured as follows: Section 2 presents the background; Section 3 reviews related works; Section 4 describes the methodology; Section 5 shows the results; and Sections 6 and 7 provide the discussion and conclusion, respectively.

2. Background

AutoTab integrates music theory, signal processing, machine learning, and tablature generation for automated transcription. It employs spectrograms, such as the Constant Q Transform (CQT), and the Spectral Flux method to accurately detect sound events. Convolutional Neural Networks (CNNs) are used for pattern recognition, and graph-based algorithms optimise tablature generation by enhancing playability and efficiency.

2.1. Fundamentals of Musical Structure

Music consists of melody, rhythm, and harmony, essential for musical expression and communication [8]. Musical theory supports cognitive, psychomotor, and social development, while musical literacy aids in interpretation and composition [25]. Notes and chords define musical structure, with notes representing specific frequencies and chords combining multiple notes to form harmonic foundations [8].

Musical notation documents and transmits music, primarily through sheet music and tablatures (Figure 1). Sheet music details melody, harmony, and rhythm, akin to written language, and follows specific rules to allow comprehensive musical interpretation [21]. In contrast, tablature simplifies notation, using numbers and symbols to focus on execution, particularly benefiting guitarists and facilitating learning for beginners [21]. These notations complement each other, enhancing musical understanding [19].

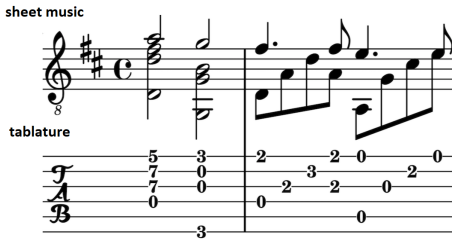
2.2. Signal Processing and Spectral Analysis

Sound is composed of pressure waves that the brain interprets as auditory signals. These waves range from 20 Hz to 20 kHz, with inaudible frequencies beyond this spectrum [9]. In digital form, sound is represented as numerical sequences fluctuating over time, mimicking physical waves. Audio signal processing analyses and manipulates

[1] https://melodyscanner.com/

[2] https://klang.io/pt-br/guitar2tabs/

Figure 1. Representation of the same musical sequence in sheet music and tablature



Source: [18]

these signals for enhancement, compression, and spectrogram generation, playing a key role in capturing features for AutoTab [6]. Spectrograms visually represent sound by mapping frequency on the vertical axis, time on the horizontal axis, and intensity through colour variations [19]. Among the various types, the Constant Q Transform (CQT) is particularly effective for musical analysis, segmenting the frequency spectrum into logarithmic bands to improve note and chord identification [14].

The equation 1 defines the CQT, where $x[n]$ is the input signal, N is the analysis window size, f represents the target frequency, f_{ref} is the reference frequency (typically 440 Hz), and Q is the quality factor influencing spectral resolution:

$$CQT(f) = \sum_{n=0}^{N-1} x[n] \cdot \frac{1}{\sqrt{N}} \cdot e^{-j2\pi qn/N}, \quad q = \frac{f}{f_{ref}} \cdot Q \tag{1}$$

CQT is essential for AutoTab, ensuring precise frequency resolution necessary for accurate note and chord detection, even in complex musical passages.

2.3. Sound Event Detection

Sound event detection, or *onset detection*, identifies the precise timing of sound events, with applications in music segmentation, speech recognition, and seismic monitoring [32]. Several techniques exist, including Zero Crossing Rate (ZCR) for monitoring signal polarity changes [28], Energy-Based Detection for amplitude variations [26], and Onset Detection Function (ODF) for frequency and energy shifts [3]. More advanced methods, such as convolutional networks on spectrograms, enhance precision [24].

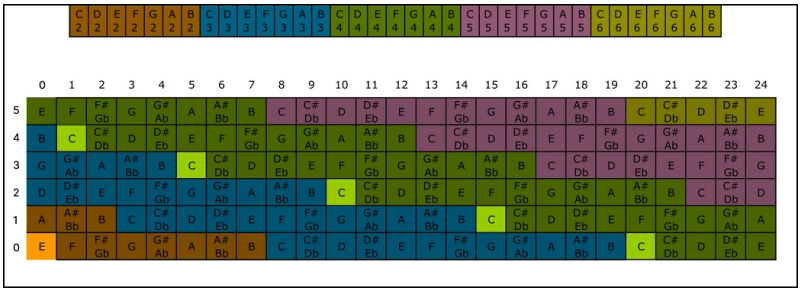
In this study, the *Spectral Flux* method was selected for its ability to detect spectral variations with high accuracy while minimising false positives. This method aligns with AutoTab's requirements, ensuring reliable identification of musical note events [2].

2.4. Neural Networks and Tablature Generation

Neural networks enable adaptive learning for complex classification tasks [11, 10]. AutoTab employs CNNs for efficient analysis of audio spectrograms, ensuring accurate identification of notes and chords. The architecture integrates multi-layer perceptrons, backpropagation, and activation functions like ReLU, enhancing pattern recognition in spectrograms and improving transcription reliability [24]. Tablature generation converts detected notes into a playable format for string instruments [23], addressing polyphony

challenges where a note can be played in multiple fretboard positions [7]. Figure 2 illustrates this complexity.

Figure 2. Positions of C4 note on the guitar fretboard



Source: Prepared by the author

Graph theory plays a crucial role in optimizing note placement for tablature generation. By modeling the fretboard as a graph, where nodes represent possible note positions and edges represent transitions between them, algorithms can determine the most efficient and playable sequence of notes. This approach minimizes finger movement and ensures ergonomic playability, particularly in polyphonic contexts where multiple valid positions exist for the same note. The graph-based method leverages shortest-path algorithms to identify optimal transitions, balancing musical accuracy with practical execution [5].

Recent advances, such as *TimbreTron* [12] and *ProgGP* [17], demonstrate the growing role of machine learning in refining automatic transcription. These methods integrate graph-based optimization with neural networks, further enhancing the accuracy and usability of generated tablatures.

3. Related Works

Recent studies have addressed the challenges of guitar transcription, each contributing unique point of views and methodologies. In this context, the study by [29] introduced a rubric-based metric system to evaluate the playability of guitar chords, assessing factors such as finger positioning, hand movement, and chord complexity. This approach provides a quantitative framework for understanding the ergonomic challenges of guitar playing, which aligns with AutoTab's goal of optimizing tablature generation for playability. However, the study focuses solely on chord playability and does not address the transcription process itself.

In one of these studies, by [14] employed CNNs to transcribe guitar chords from audio waveforms, leveraging transfer learning to improve accuracy. The study demonstrates the effectiveness of CNNs in chord recognition, particularly when trained on diverse datasets. AutoTab builds on this approach by integrating CNNs for both note and chord detection, while also addressing the polyphonic challenges unique to guitar transcription. In a complementary effort, [20] introduced a simplified CNN architecture using 1-max pooling for robust audio event recognition, achieving high accuracy in identifying musical events, even in noisy environments. AutoTab incorporates similar CNN techniques for sound event detection, ensuring reliable transcription of notes and chords in real-world audio recordings.

Among recent contributions, the study by [22] focused on enhancing onset detection using CNNs, achieving high precision in identifying the start of musical notes. The work highlights the importance of accurate onset detection for reliable transcription, a challenge that AutoTab addresses through its Audio Window Analyser (AWA) module, which combines spectral analysis and CNNs for precise note identification. Additionally, [7] introduced a novel approach to polyphonic transcription by modeling continuous pitch contours, addressing the challenge of multiple valid fretboard positions for the same note. This work is particularly relevant to AutoTab's tablature generation module, which leverages graph-based optimization to select the most playable note sequences.

AutoTab builds upon these studies by integrating their successful aspects into a comprehensive, user-friendly system. For example, it combines the CNN-based approaches proposed by [14] and [20] for note and chord detection, the onset detection techniques described in [22] for precise note identification, and the polyphonic transcription methods presented in [7] for optimized tablature generation. Additionally, AutoTab extends these works by introducing a graph-based optimization algorithm to ensure ergonomic playability and by providing an interactive web interface for real-time transcription.

4. Methodology

The methodology employed in this study integrates audio signal processing, sound event detection, machine learning models, and automatic tablature generation into an interactive web-based system. This research follows an experimental design, employing quantitative evaluation metrics such as accuracy and F1-score, combined with usability assessments involving amateur and professional musicians. The system was evaluated across multiple components, including note and chord detection accuracy, processing speed, and user interaction efficiency.

4.1. Dataset Preparation

Three datasets were utilized for training the classifiers: *OwnSet*, *IDMT-SMT-GUITAR_V2* [15], and *GuitarSet* [30]. The *IDMT-SMT-GUITAR_V2* dataset includes authentic recordings of various guitars and musical styles, featuring techniques such as bends, slides, and chords, making it invaluable for training the neural network. The *GuitarSet* focuses on acoustic guitar sounds, providing diverse and complex audio samples with advanced harmonies and chords. The *OwnSet* was developed to address specific gaps, such as missing harmonics and background noise, and includes classical chords, power chords, and noise sounds.

The datasets were organized into a folder named *Raw*, maintaining their original structure. Python scripts were developed to extract and organize audio fragments of individual notes and chords into separate folders. These scripts read annotations to identify the start and end points of each note, classifying segments as chords or isolated notes based on their intervals. The extracted audio fragments were stored in a directory named *Custom*, with subfolders named after the identified notes.

To enhance dataset diversity, an audio augmentation routine was implemented to modify the pitch of the original recordings, generating new audio versions with higher or

lower tones. Each augmented file was stored in a dedicated folder, following the same organisational structure as the original dataset. This process significantly expanded the dataset, increasing the number of individual note files from 22,799 to 84,243 and chord files from 1,757 to 20,998.

Finally, a preprocessing algorithm was developed to prepare the data for neural network training. This algorithm locates, loads, and processes audio files from the *Custom* folder, associating each audio file with its corresponding notes. The processed data is then exported in the Numpy (`.npy`) format, ensuring efficient loading and portability. The dataset is divided into three subsets: one for individual notes, one for chords, and a mixed dataset combining both.

4.2. Classifier Architecture

The classifier architecture for this project was developed using the `TensorFlow` framework [1], chosen for its flexibility, efficiency, and extensive community support. The architecture is based on CNNs, which excel at identifying complex visual patterns, making them ideal for analyzing spectrograms of audio signals. The base architecture, inspired by [20], consists of three main layers: input, convolutional, and output.

Three distinct CNN models were developed, each tailored for specific tasks:

- **`model_notes`**: Specialized in detecting individual notes.
- **`model_chords`**: Focused on recognizing harmonies and chords.
- **`model_mix`**: A generalist model combining the capabilities of the other two, designed to identify both notes and chords.

4.2.1. Input Layer

The input layer accepts a three-dimensional matrix of dimensions 84x173x1, representing the Constant-Q Transform (CQT) spectrogram of an audio signal. This matrix serves as the foundation for identifying musical notes in the analyzed audio.

4.2.2. Convolutional Layers

The model includes two parallel convolutional layers, each equipped with 64 filters. The first layer uses a kernel size of 3, while the second uses a kernel size of 5, both with *valid* padding. Following each convolutional layer, a `GlobalMaxPooling` layer is applied to emulate the efficiency of `max-pooling-1d`, as suggested by [20]. The outputs of these layers are concatenated to optimize the model's ability to identify relevant patterns in the audio data.

4.2.3. Output Layer

The output layer consists of two dense layers: the first with 128 units and the second with 49 units, corresponding to the detectable notes in the dataset. The activation function of the final layer varies depending on the model's purpose:

- For `model_notes`, the `softmax` activation function ensures that only one output unit is activated at a time, indicating a specific note.
- For `model_chords` and `model_mix`, the `sigmoid` activation function allows multiple output units to be activated simultaneously, enabling the identification of chords and polyphonic notes.

The choice of 49 output units aligns with the dataset's structure, facilitating the subsequent tablature generation process. This approach distinguishes the model from others, such as [14], which use six output units to represent guitar strings, highlighting the adaptability of the proposed architecture for diverse musical contexts.

4.3. Automatic Tablature Generation

The TabGen algorithm automates the generation of up to k distinct tablatures from a specific sequence of notes, extracted directly from the *Audio Window Analyser* (AWA) module. The AWA module is responsible for detecting musical notes within audio windows. It processes audio files by converting them into spectrograms using the Short-Time Fourier Transform (STFT) and identifies precise note onsets using methods such as `onset_strength` and `onset_detect`. The module then predicts the notes present in each window by analyzing the Constant-Q Transform (CQT) spectrogram, leveraging a trained neural network to assign confidence scores to the detected notes. This process ensures accurate and efficient note identification, which is essential for subsequent tablature generation.

Based on graph theory and the *k-shortest paths* algorithm [31], TabGen identifies the most playable tablature sequences, optimizing transitions between notes and chords for ease of execution.

4.3.1. Mapping Notes on the Guitar Fretboard

The first step involves creating a map of all possible note positions on a 24-fret guitar, considering standard tuning (E2-A2-D3-G3-B3-E4). This map, generated by a function, associates each note with its exact position on the fretboard, represented by coordinates (x, y). This mapping is essential for calculating the weight of each note, which reflects its playability based on criteria such as hand movement and string proximity.

4.3.2. Graph Construction

TabGen employs graph theory to model the relationships between note positions. For each chord, a preliminary graph is constructed, mapping all possible combinations of note positions. Each combination is assigned a weight based on playability criteria, including:

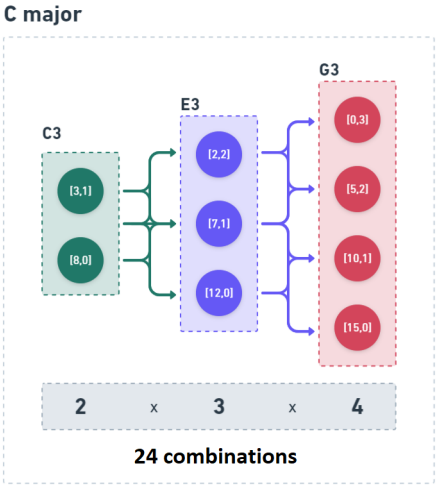
- **Fretboard Proximity**: Evaluates the dispersion of notes along the fretboard, assigning higher weights to combinations requiring greater hand movement.
- **String Proximity**: Assesses the distance between strings, penalizing stretches that increase difficulty.
- **Comfort Zone**: Adjusts weights based on proximity to the guitarist's preferred fretboard position.

For example, a C major chord (C3, E3, G3) can be played in 24 combinations, as illustrated in Figure 3. These combinations are represented as nodes in the graph, connected by edges weighted according to playability.

4.3.3. K-Shortest Paths Algorithm

To generate the most playable tablatures, TabGen applies Yen's algorithm [31], which extends Dijkstra's algorithm to find the k shortest paths in a graph. This approach

Figure 3. Graph representation for tablature generation.



Source: Prepared by the author.

ensures that the algorithm identifies multiple viable tablature sequences, optimizing transitions between chords and minimizing the guitarist's effort.

4.3.4. Output

The final output of TabGen is a vector containing up to k tablature sequences, each represented as a sub-vector of note positions and assigned a weight reflecting its playability. An example of the output structure is shown in Figure 4. This flexibility allows musicians to choose the most suitable tablature based on their preferences and skill level.

Figure 4. Example of tablatures generated by `TabGen`

```
Final Result:
[
   {Tablature: [[(3,1)], [(2,2)], [(0,3),(3,1)(2,2)]], Weight: 100},
   {Tablature: [...], Weight: ...},
   ...
   {Tablature: [(k-th combination)],
   Weight: (weight of the k-th tablature)}
]
```

Source: Prepared by the author.

5. Results

In this section, we present the results obtained by the critical modules of the AutoTab system, focusing on sound event detection and the identification of notes and chords. The analysis was conducted using two datasets: an external one (IDMT_GUITAR_V2) for evaluating sound event detection and an internal one (Custom) for testing the accuracy of the note and chord identification models.

5.1. Sound Event Detection

The sound event detection module, based on the *Spectral Flux* method, was evaluated using 19,341 events. The results showed a hit rate of 87.5% (18,512), with

errors in 3.9% (829) of the samples and a false positive rate of 8.6% (1,823). Although the method demonstrated effectiveness in identifying sound events, the presence of false positives suggests the need for improvements, especially in scenarios with rapid chord sequences or dense sound environments.

5.2. Note and Chord Identification

The note and chord identification models were evaluated using the internal dataset (Custom), which contains a wide variety of note and chord sounds. The results for each model are detailed below.

5.2.1. Chord Model (`model_chords`)

The `model_chords` was tested with 51,748 files, achieving a hit rate of 60% (31,027). However, errors and incorrect detections were significant, with 24.5% (12,700) and 15.5% (8,021), respectively.

5.2.2. Note Model (`model_notes`)

The `model_notes` showed higher accuracy, with a hit rate of 87.8% (77,910) in 88,704 tested files. Errors and incorrect detections were relatively low, with 7.1% (6,333) and 5.0% (4,461), respectively.

5.2.3. Generalist Model (`model_mix`)

The `model_mix`, designed to identify both notes and chords, was tested with 142,826 files, achieving a hit rate of 71.1% (101,581). Errors and incorrect detections were 18.6% (26,521) and 10.3% (14,724), respectively.

5.3. Comparative Analysis

Compared to established methods in the literature, the AutoTab system demonstrated competence but with room for improvement. The *Spectral Flux* method used for sound event detection achieved a hit rate of 87.5%, while CNN-based approaches, such as the one proposed by [24], achieved an F-score of up to 90.3%. For note identification, the `model_notes` achieved an accuracy of 87%, in contrast to the *CREPE Pitch Tracker* [16], which reaches 96%. For chords, the `model_chords` achieved 60% accuracy, while the work of [14] achieved 88.7%.

These results highlight the effectiveness of the AutoTab system but also point to the need for improvements, especially in detecting subtle sound events and identifying chords in polyphonic contexts.

6. Discussion

The AutoTab system has demonstrated robust performance in automatic guitar tablature transcription, leveraging advanced signal processing techniques, CNNs, and graph-based optimization algorithms. The system's ability to accurately detect sound events and identify notes and chords has been validated through extensive testing, achieving an 87.8% accuracy in note identification and a 78% Macro F1-score in chord recognition. These results highlight the effectiveness of the proposed approach, particularly in monophonic contexts where the system excels.

However, challenges remain in polyphonic scenarios, where the complexity of overlapping harmonics and multiple valid fretboard positions for the same note complicates transcription. While the Spectral Flux method used for sound event detection achieved a hit rate of 87.5%, the presence of false positives (8.6%) indicates room for improvement, especially in dense or noisy audio environments. Similarly, the Chord Model (`model_chords`) achieved a lower accuracy of 60%, suggesting that further refinement is needed for reliable chord recognition in polyphonic contexts.

The integration of CNNs for spectrogram analysis has proven effective, with the Note Model (`model_notes`) achieving high accuracy. However, the Generalist Model (`model_mix`), designed to handle both notes and chords, showed a balanced but lower performance (71.1% accuracy), indicating that specialized models may still be necessary for optimal results. These findings align with previous studies, such as [14], which emphasize the challenges of polyphonic transcription and the need for advanced neural network architectures.

The TabGen algorithm, based on graph theory and the k-shortest paths approach, successfully generates playable tablatures by optimizing note transitions for ergonomic execution. This feature enhances the usability of the system, allowing musicians to choose from multiple tablature options based on their preferences and skill levels. However, the algorithm's reliance on predefined playability criteria may limit its adaptability to individual playing styles, suggesting a potential area for future enhancement.

7. Conclusion

The AutoTab system represents a step forward in automatic guitar tablature transcription, offering an accessible and user-friendly solution for musicians of all skill levels. By integrating signal processing, CNNs, and graph-based optimization, the system addresses key challenges in musical transcription, such as sound event detection, note and chord identification, and tablature generation. The interactive web interface further enhances usability, enabling real-time transcription and providing musicians with a practical tool for learning and practice.

Despite its strengths, the system has limitations, particularly in handling polyphonic music and low-quality audio recordings. Future work should focus on improving the accuracy of chord recognition and sound event detection, potentially through the use of more advanced neural network architectures or hybrid approaches combining CNNs with recurrent networks. Additionally, expanding the system's functionality to include features such as tablature regeneration, virtual synthesizers, and database integration could further enhance its utility and appeal.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker,

Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] M. Alonso, G. Richard, and B. David. Extracting note onsets from musical recordings. *2005 IEEE International Conference on Multimedia and Expo*, pages 4 pp.–, 2005.

[3] J.P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M.B. Sandler. A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13(5):1035–1047, 2005.

[4] Emmanouil Benetos, S. Dixon, Z. Duan, and S. Ewert. Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, 36:20–30, 2019.

[5] J. Bondy and U. Murty. Graph theory. pages 1–582, 2008.

[6] V. Chiesa, P. Coughlan, and C. Voss. Development of a technical innovation audit. *Journal of Product Innovation Management*, 13:105–136, 1996.

[7] Frank Cwitkowitz, T. Hirvonen, and A. Klapuri. Fretnet: Continuous-valued pitch contour streaming for polyphonic guitar tablature transcription. *ArXiv*, abs/2212.03023, 2022.

[8] Maria Luisa de Mattos Priolli. Princípios básicos da música para juventude, 1999.

[9] L. Frenzel. Chapter 10 – audio electronics: Digital voice and music dominate. pages 229–246, 2010.

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

[11] Simon S. Haykin. *Neural networks and learning machines*. Pearson Education, Upper Saddle River, NJ, third edition, 2009.

[12] Sicong Huang, Qiyang Li, Cem Anil, Xuchan Bao, Sageev Oore, and Roger B. Grosse. Timbretron: A wavenet(cyclegan(cqt(audio))) pipeline for musical timbre transfer, 2023.

[13] Yogesh Jadhav, Ashish Patel, Rutvij H. Jhaveri, and Roshani Raut. Transfer learning for audio waveform to guitar chord spectrograms using the convolution neural network. *Mobile Information Systems*, 2022:1–11, 2022.

[14] Yogesh Jadhav, Ashish Patel, Rutvij H. Jhaveri, and Roshani Raut. Transfer learning for audio waveform to guitar chord spectrograms using the convolution neural network. *Mobile Information Systems*, 2022:11, 2022.

[15] Christian Kehling, Jakob Abeßer, Christian Dittmar, and Gerald Schuller. Automatic tablature transcription of electric guitar recordings by estimation of score- and instrument-related parameters. In *Proceedings of the 17th International Conference on Digital Audio Effects (DAFx)*, Erlangen, Germany, 2014.

[16] Jong Wook Kim, Justin Salamon, Peter Li, and Juan Pablo Bello. Crepe: A convolutional representation for pitch estimation, 2018.

[17] Jackson Loth, Pedro Sarmento, CJ Carr, Zack Zukowski, and Mathieu Barthet. Proggp: From guitarpro tablature neural generation to progressive metal production, 2023.

[18] Rock Out Loud. Tablature vs sheet music explained, 2025. Accessed: 2025-03-10.

[19] Meinard Müller. *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Springer Cham, 1 edition, 2015.

[20] Huy Phan, Lars Hertel, Marco Maass, and Alfred Mertins. Robust audio event recognition with 1-max pooling convolutional neural networks. *arXiv preprint arXiv:1604.06338*, 2016.

[21] Polifono. Como aprender a ler partitura e por qual motivo, 2020. Acessado em: 18 de Janeiro de 2024.

[22] Hendrik Purwins, Bo Li, Tuomas Virtanen, Jan Schluter, Shuo-Yiin Chang, and Tara Sainath. Deep learning for audio signal processing. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):206–219, may 2019.

[23] Miguel A. Román, Antonio Pertusa, and Jorge Calvo-Zaragoza. A holistic approach to polyphonic music transcription with neural networks, 2019.

[24] Jan Schlüter and Sebastian Böck. Improved musical onset detection with convolutional neural networks. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6979–6983, 2014.

[25] A. C. P. R. Silva. A música no processo de ensino e aprendizagem. Trabalho de Conclusão de Curso (Especialização em Educação: Métodos e Técnicas de Ensino) - Universidade Tecnológica Federal do Paraná, 2021.

[26] Julius O. Smith. *Spectral Audio Signal Processing*. `http`:http://ccrma.stanford.edu/ jos/sasp/`//ccrma.stanford.edu/~jos/sasp/`, accessed 2024. online book, 2011 edition.

[27] D.R. Tuohy and W.D. Potter. Ga-based music arranging for guitar. In *2006 IEEE International Conference on Evolutionary Computation*, pages 1065–1070, 2006.

[28] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

[29] Marcel A. Vélez Vásquez, Mariëlle Baelemans, Jonathan Driedger, Willem Zuidema, and John Ashley Burgoyne. Quantifying the ease of playing song chords on the guitar. In *Proceedings of the 24th International Society for Music Information Retrieval Conference*, pages 725–732, Milan, Italy, 2023. ISMIR. © M.A. Vélez Vásquez, M.C.E. Baelemans, J. Driedger, W.H. Zuidema, and J.A. Burgoyne. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0).

[30] Q. Xi, R. Bittner, J. Pauwels, X. Ye, and J. P. Bello. Guitarset: A dataset for guitar transcription. In *19th International Society for Music Information Retrieval Conference*, Paris, France, Sept 2018.

[31] Jin Y Yen. An algorithm for finding shortest routes from all source nodes to a given destination in general networks. *Quarterly of Applied Mathematics*, 27(4):526–530, 1970.

[32] R. Zhou and J. Reiss. Music onset detection. pages 297–316, 2011.