

GUITAR TABLATURE ESTIMATION WITH A CONVOLUTIONAL NEURAL NETWORK

Andrew Wiggins, Youngmoo Kim

Drexel University, Dept. of Electrical and Computer Engineering

{awiggins, ykim}@drexel.edu

ABSTRACT

Guitar tablature is a popular notation guitarists use to learn and share music. As it stands, most tablatures are created by an experienced guitarist taking the time and effort to annotate a song. As the process is time consuming and requires expertise, we are interested in automating this task. Previous approaches to automatic tablature transcription break the problem into two steps: 1) polyphonic pitch estimation, followed by 2) tablature fingering arrangement. Using a convolutional neural network (CNN) model, we can jointly solve both steps by learning a mapping directly from audio data to tablature. The model can simultaneously leverage physical playability constraints and differences in string timbres implicit in the data to determine the actual fingerings being used by the guitarist. We propose TabCNN, a CNN for estimating guitar tablature from audio of a solo acoustic guitar performance. We train and test our network using microphone recordings from the GuitarSet dataset [24], and TabCNN outperforms a state-of-the-art multipitch estimation algorithm. We also introduce a set of metrics to evaluate guitar tablature estimation.

1. INTRODUCTION

Given the popularity of the guitar as an instrument for both professional musicians and amateur hobbyists, there have been numerous previous works addressing the problem of automatic guitar transcription. Automatic guitar transcription is a task which aims to generate a symbolic transcription instructing a guitarist how to perform, given an audio recording of a guitar performance. In general, the task of polyphonic transcription is challenging, due to the fact that when multiple different pitches sound at once, their overtones may overlap in the frequency domain, making it hard to tell which pitches are sounding. Despite the narrowed focus on a single instrument, transcription of solo guitar audio remains challenging. Since it has six strings, the guitar can produce up to six pitches at a given time. The guitar is also capable of producing a wide variety of timbres, stemming from differences in guitar models, guitar

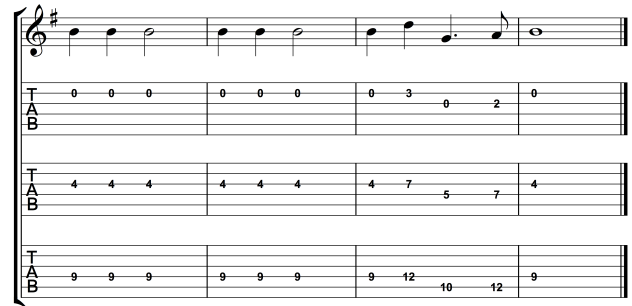



Figure 1. Multiple fingerings (3 bottom staves) can be used to play a given score (top staff) on guitar. These are just 3 of many possible fingerings. In each tablature staff, the horizontal lines represent the 6 guitar strings, and numbers on them indicate the fret on that string to be activated.

strings, audio effects, and strumming, plucking, and picking styles.

Perhaps the greatest challenge in automatic guitar transcription, however, arises from the symbolic representation of guitar music. Rather than using score notation, guitarists commonly use tablature notation to compose, share, and learn music. While music scores display the arrangements of pitches in time, tablature notation additionally indicates which guitar strings and positions along the fretboard were activated to produce the sounding pitches. On guitar, most notes can be played in numerous locations along the fretboard. Figure 1 provides an example of this. These identical pitches played in different locations differ in timbre and can give different characteristics to an overall performance. In order to effectively transcribe a guitar performance from audio, a system needs to estimate both the pitches and fingerings changing over the course of the performance.

Previous works have approached automatic tablature transcription by splitting the problem into two separate steps [5, 6, 25, 26]. First, the systems perform a multipitch estimation on the audio, which determines the set of pitches sounding over the course of the audio. Then, using the estimated pitches, a tablature is arranged, usually by choosing a fingering that maximizes physical ease of play. These methods are designed to output a valid tablature given the detected pitches, but not necessarily the true fingering used by the guitarist during the performance.

In this paper we propose TabCNN¹, a guitar tablature estimation system which outputs the actual fingerings used

 © Andrew Wiggins, Youngmoo Kim. Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). **Attribution:** Andrew Wiggins, Youngmoo Kim. “Guitar Tablature Estimation with a Convolutional Neural Network”, 20th International Society for Music Information Retrieval Conference, Delft, The Netherlands, 2019.

¹ Source code: <https://github.com/andywiggins/tab-cnn>

by the guitarist at the frame-level from audio of a solo performance on a standard 6-string acoustic guitar. The proposed system uses a convolutional neural network (CNN) to learn a direct mapping from audio signal to guitar tablature.

The next section provides background on some related work in automatic guitar transcription and the use of CNNs in music information retrieval. In Section 3 we outline our methodology, including the dataset, preprocessing, and proposed system architecture. In Section 4, we evaluate the proposed system for multipitch estimation and tablature estimation. We further discuss the system performance in Section 5. Finally, in Section 6 we give our conclusions and suggest future work.

2. RELATED WORK

2.1 Automatic Tablature Transcription

Several previous works have addressed the problem of automatic tablature transcription by performing multipitch estimation and then arranging tablature by optimizing the physical ease of play. Burllet and Fujinaga outlined a framework for a guitar transcription web application [5]. Their framework combines a preexisting polyphonic transcription algorithm proposed by Zhou and Reiss [28] with a novel guitar tablature arrangement algorithm that creates a directed acyclic weighted graph of string-fret combinations and finds an optimal path using the A* search algorithm. In [6], Burllet and Hindle utilized the aforementioned tablature arrangement algorithm in conjunction with a novel multipitch estimation algorithm, using deep belief networks to learn framewise pitch estimates from the short-time Fourier transform. Yazawa et al. used latent harmonic allocation (LHA) for multipitch estimation and arranged tablature by filtering the LHA results based on a set of spatial and temporal playability constraints [26]. In [25], the authors applied knowledge of each guitarist’s proficiency to further filter the pitches estimated from LHA and generate a sensible tablature arrangement.

Few previous approaches to automatic tablature transcription that we found seek to learn the true fingering used by the guitarist. In [3], Barbancho et al. used peak-picking from the magnitude spectrum to determine fundamentals and partials for candidate pitches and then analyzed the inharmonicity of the partials to determine the most likely string each pitch was played on. The system’s performance on “free chords” – i.e., not recognition of predetermined chords, or strictly monophonic playing – is respectable, but is limited to a maximum of 4 pitches sounding simultaneously. In [13], Kehling et al. proposed a system that applies the Blind Harmonic Adaptive Decomposition algorithm proposed in [7] for multipitch estimation. After aggregating framewise pitch estimates into note estimates, their system applies Support Vector Machines to classify various performance parameters, including the guitar string each note was played on. The authors show good performance for this system for notewise multipitch estimation and guitar string estimation, but they do not evaluate the

system directly for framewise tablature estimation.

2.2 Related Tasks

There have been a number of works that tackle different problems related to automatic guitar transcription. For guitar chord recognition, Barbancho et al. used a hidden Markov model (HMM) to transcribe guitar chords and fingerings from acoustic features [2]. In [12], Humphrey and Bello approached chord recognition using a convolutional neural network model to output tablature, and this model was trained using a pop music dataset, rather than audio of isolated guitar performances. Hrybyk and Kim used video data in conjunction with audio of solo guitar performances to recognize guitar chords [11].

Regarding the problem of tablature arrangement from symbolic data (not audio), Tuohy and Potter combined a neural network model with a local heuristic-climber to take pieces composed for other instruments and arrange them for guitar [23]. In [18], Mistler used deep neural networks to arrange guitar tablature from sheet music by predicting a fretting cost function and predicting string-fret activations directly.

The problem of arranging solo guitar covers of pop songs was addressed by Ariga et al., who proposed a system which processes pop music audio and generates tablature arrangements that can scale in difficulty [1]. Finally, in the area of automatic music generation, McVicar et al. built an algorithmic composition system to compose tablatres for rhythm and lead guitar parts from an input chord progression [17].

2.3 CNNs for Automatic Music Transcription

While, to our knowledge, the task of guitar tablature estimation has not been approached using convolutional neural networks, CNNs have shown promise for the similar task of automatic piano transcription. In [21], Stigita et al. proposed a hybrid neural network model for piano transcription that combines a CNN for framewise acoustic modelling and a recurrent neural network (RNN)-based music language model. Kelz et al. provided a comparison of various network-based approaches to framewise transcription of piano audio, and the authors argued that the high accuracy and low parameter amount offered by CNNs, compared to other classes of deep neural networks, gives them a “distinct advantage” for piano transcription [14].

The use of CNNs has also been explored for various other tasks within music information retrieval such as musical tempo estimation [20], key classification [15], singing voice detection [19], and instrument classification [9, 10].

3. METHODOLOGY

3.1 Dataset

Since we are interested in learning tablatres for the exact fingerings used by the guitarist, it is important that the dataset contains audio of an actual guitar performance. Previous approaches to guitar transcription utilize audio

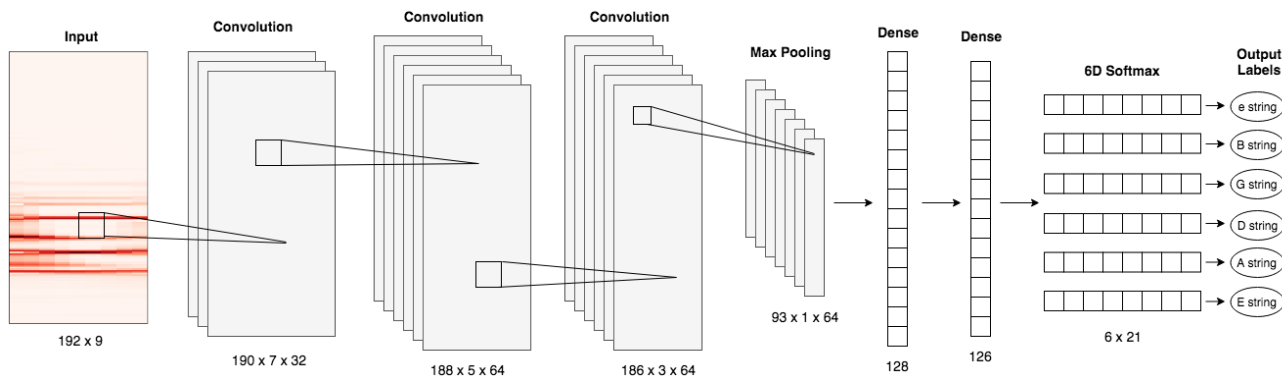


Figure 2. The proposed TabCNN network architecture. The input is a constant-Q spectrogram image of solo acoustic guitar audio. There are a series of 2D convolutional layers followed by a max pooling layer, which learn to extract spatial features relevant for guitar tablature estimation. The dense layers and final softmax function aim to use the learned representation to predict fret-number labels for each of the 6 guitar strings.

data that was automatically generated using MIDI to playback audio samples [5, 6, 25, 26]. As a result, the audio does not represent a performance of specific guitar fingerings. This is problematic for our purposes since there is no ground truth tablature.

We employ the GuitarSet dataset [24], which consists of audio recordings of solo acoustic guitar performances. This dataset was created using a guitar equipped with a hexaphonic pickup. The signals from each individual guitar string were processed separately to produce a frame-level pitch annotation for each of the 6 guitar strings. This dataset enables the ground truth fretting used by the guitarist to be easily accessible. Since the guitar used remained in standard tuning, we can use these pitch estimations to determine the corresponding frets being fingered over the course of the performance. While the hexaphonic signals were used by the GuitarSet authors to create the ground truth annotations, our system uses only the monophonic microphone signal of each song to estimate the tablature.

The GuitarSet contains audio recordings of 360 solo guitar performances, each approximately 30 seconds in length. These performances span every key and cover a variety of styles including bossa nova, rock, and funk. The guitarists were instructed to play two different versions of each song: soloing, which contains mostly single notes, and comping, which means playing chords. Six different guitarists contributed to the dataset, each performing to a provided chord progression, but interpreting it in their own way.

3.2 Audio Preprocessing

During the preprocessing stage, we first downsample the audio from 44100 Hz to 22050 Hz, making the assumption that there is not too much relevant information above 11025 Hz, in order to reduce the dimensionality of the input signals. We normalize each audio clip by its maximum value, to account for any major amplitude differences between clips.

As CNNs are useful in learning spatial features from

images, we are interested in transforming the raw audio data into an image representation. The Short-Time Fourier Transform (STFT), commonly used to represent a signal changing over time and frequency, is not a desirable choice for CNNs because of its high dimensionality. Additionally, since the task at hand involves recognizing musical pitches, it would be advantageous to use a representation with a frequency axis that is linearly spaced with respect to pitch. This allows pitch-invariant features to be learned by the network. For these reasons we employ the constant-Q transform (CQT), which greatly reduces the dimension of the frequency axis by linearly spacing the frequency bins with respect to musical pitch.

Motivated by previous work [12], we use a CQT with 192 bins, spanning 8 octaves. This equates to 24 bins per octave, or 2 bins per semitone. We use a hopsize of 512 samples, which corresponds to a frame rate of about 43 frames per second, a rate sufficient for framewise analysis [10]. Using a sliding context window, we generate input images centered around each frame. In our experiments, we observed the best results using a context window of 9 frames. We pad either side of the CQT with zeros so that the beginning and ending frames of each clip can have a full 9-frame-long context window. Thus, input samples are of size 192×9 at each frame.

3.3 Label Preprocessing

To obtain annotations for each individual frame of audio, we sample the stringwise pitch annotations at the same frame rate of approximately 43 frames per second. For each sampled MIDI pitch value, we round it to the nearest integer, which corresponds to finding the nearest musical pitch. Depending on which string the pitch was played on, we subtract the corresponding string’s open pitch value. The resulting integers correspond to the frets that were activated to produce the sounding pitches. A value of zero indicates that the string was plucked open (while no frets were being pressed). Since there are 19 frets used on the acoustic guitar, and a string may at times be closed (not produce any sound), this results in 21 different fret classes

that each string may be in during any given frame: open, closed, or any one of the 19 frets. We convert each string’s label to a one-hot representation, resulting in label size of 6×21 at each frame.

3.4 Network Architecture

The structure of our neural network model is generally inspired by popular models in computer vision, such as AlexNet [16] and VGGNet [22], which both contain series of convolutions followed by a sub-sampling operation. These architectures terminate in dense layer connections. When using this type of CNN architecture, the expectation is that the early convolutional layers act as a hierarchy of feature extractors, learning spatial filter coefficients that result in feature maps useful for the given classification task. We interpret the final dense layers and output connection to act as a classifier that processes the features to output a final decision or prediction.

The proposed network structure for TabCNN is shown in Figure 2. First, there is a series of three convolutional layers, each with a filter size of 3×3 . The first convolutional layer has 32 filters, and the latter two each have 64. These parameters were determined empirically, based on experiments with the validation data. Each convolution is immediately followed by a Rectified Linear Unit (ReLU) activation. Activation functions, in general, allow for complicated, nonlinear mappings to be learned, and the ReLU activation has been shown to train faster [16] and produce better results [8] than alternatives.

The resulting feature maps are subsampled by a 2×2 max pooling layer, which introduces a slight translation invariance in both time and frequency. The structure is then flattened and followed by a dense layer of dimension 128, this size determined empirically in our experiments. After this dense layer, a ReLU activation is applied. This is connected to a second dense layer of dimension 126, which is reshaped to 6×21 . The output shape comes from the 6 guitar strings and the 21 different fret classes a string can be assigned. Finally, a softmax activation is applied to each of the 6 rows. As a result, similar to the network described in [12], our model learns to output an estimation of six probability mass functions, which represent the probability of each fret class for each string.

3.5 Training Procedure

For training the model, we design a loss function by viewing the problem as 6 simultaneous multiclass classification problems. For multiclass classification, a common practice is to optimize the categorical cross-entropy between the predictions and the targets [10]. For our loss function, we compute the categorical cross-entropy for each string and sum these values. The loss function is computed as in Eqn (1). We use z_{ij} to denote an activation at frame i on string j that belongs to fret class $C_{z_{ij}}$. The term $p[z_{ij} \in C_{z_{ij}}]$ is the probability output by the network of z_{ij} belonging to class $C_{z_{ij}}$. N is the total number of frames in the mini batch.

$$Loss = -\frac{1}{N} \sum_{j=1}^6 \sum_{i=1}^N \log p[z_{ij} \in C_{z_{ij}}] \quad (1)$$

We train using the ADADELTA optimization algorithm [27], which adapts learning rates for parameters based on a window of previous gradient values. We use an initial learning rate of 1.0 and a mini batch size of 128 training samples. We train for 8 epochs, as we noticed overfitting when training for longer. We also employ dropout regularization to combat overfitting, with a dropout rate of 0.25 applied immediately after the max pooling layer, and a second dropout rate of 0.5 applied after the first dense layer.

4. EVALUATION

To evaluate the proposed system, we first review multipitch estimation metrics from the literature [6] and assess the quality of the system as a polyphonic pitch detector. Then, since there are no standards for evaluating the performance of a tablature estimation system, we introduce a set of metrics to measure performance in estimating tablature for the actual fingering used during the performance. In our evaluations we use 6-fold cross validation, holding out one of the 6 guitarists to test on, while training on the remaining 5. We average our numerical results over the 6 folds of data, testing with a total number of 472,560 frames.

4.1 Multipitch Estimation Metrics

In the following equations, we use Y to denote a binary matrix of size $N \times 44$, where N is the total number of testing frames, and the matrix represents the presence or absence of each pitch for each frame of audio. (The guitar can produce a total of 44 distinct pitches.) Y_{gt} contains the ground truth pitch detections for the testing set, while Y_{pred} contains the predicted pitch detections during testing. We use e to denote a vector of all ones, and \odot to denote element-wise multiplication.

4.1.1 Multipitch Precision

We compute multipitch precision using Eqn (2), which calculates the number of correctly identified pitches divided by the total number of predicted pitches. This metric measures how frequently the pitches that are detected are in fact correct.

$$p_{\text{pitch}} = \frac{e^T (Y_{\text{gt}} \odot Y_{\text{pred}}) e}{e^T Y_{\text{pred}} e} \quad (2)$$

4.1.2 Multipitch Recall

We also compute multipitch recall using Eqn (3), which calculates the number of correctly identified pitches divided by the the total number of ground truth pitches. This metric measures how frequently pitches exist in the signal are are detected by the system.

$$r_{\text{pitch}} = \frac{e^T (Y_{\text{gt}} \odot Y_{\text{pred}}) e}{e^T Y_{\text{gt}} e} \quad (3)$$

System	p_{pitch}	r_{pitch}	f_{pitch}
TabCNN	0.900 ± 0.016	0.764 ± 0.043	0.826 ± 0.025
Deep Saliency	0.778 ± 0.092	0.562 ± 0.099	0.646 ± 0.078

Table 1. Multipitch estimation metrics for our system, TabCNN, compared against a baseline, the Deep Saliency f0-estimation algorithm introduced in [4], from experiments carried out in [24]. For all metrics, we report the mean and standard deviation over the entire dataset.

4.1.3 Multipitch F-measure

Finally, we compute multipitch F-measure using Eqn (4), which is the harmonic mean of multipitch precision and recall. This metric summarizes the system’s overall performance for multipitch estimation.

$$f_{\text{pitch}} = \frac{2p_{\text{pitch}}r_{\text{pitch}}}{p_{\text{pitch}} + r_{\text{pitch}}} \quad (4)$$

4.2 Multipitch Estimation Results

The evaluation results of TabCNN for multipitch estimation, alongside a baseline algorithm, are shown in Table 1. For a benchmark, we look to the GuitarSet paper [24], in which the authors employ the Deep Saliency multiple-f0 estimation algorithm in [4] to transcribe frame-wise pitch estimations for the GuitarSet data.² Deep Saliency, although not designed specifically for guitar transcription, achieves a multipitch F-measure of 0.646. The proposed system TabCNN outperforms these baseline results, achieving a multipitch F-measure of 0.826.

Looking at a recent multipitch system designed specifically to operate on guitar signals, in [6] Burlet and Hindle report a multipitch F-measure of 0.71 before frame smoothing with a Hidden Markov Model (HMM), and an F-measure of 0.77 afterward. Our system’s multipitch F-measure of 0.826 without any temporal smoothing is promising. However, these results cannot be compared too closely since the authors were evaluating on a different dataset.

Our results reveal that the proposed system behaves conservatively for multipitch estimation. The multipitch precision being significantly larger than the multipitch recall indicates that the system will more often make an error by missing a detection, rather than reporting a pitch not actually present in the signal. This is a common behavior for multipitch estimation systems, as a pitch can be easily missed if it is able to blend in to the overtones of another pitch present at the same time. We noticed that this often occurs with pitches an octave apart; the higher pitch will often be missed, as it may appear to just be overtones of the lower pitch.

² We acknowledge and thank the authors of [24] who have granted us access to the full numerical results from the experiments conducted in the work.

4.3 Tablature Estimation Metrics

We use Z to denote a binary matrix of size $N \times 120$, N being the total number of testing frames. The dimension of 120 arises from all possible sounding string-fret combinations on guitar (6×20). This matrix represents the presence or absence of each string-fret combination for each frame of audio. Z_{gt} contains the ground truth tablature detections for the testing set, while Z_{pred} contains the predicted tablature detections during testing. Again, we use e to denote a vector of all ones and \odot to denote element-wise multiplication.

4.3.1 Tablature Precision

We define tablature precision, which calculates the number of correctly identified string-fret combinations divided by the total number of predicted string-fret combinations. This metric measures how frequently the tablature detected is in fact correct.

$$p_{\text{tab}} = \frac{e^T(Z_{\text{gt}} \odot Z_{\text{pred}})e}{e^T Z_{\text{pred}} e} \quad (5)$$

4.3.2 Tablature Recall

We also introduce tablature recall, which calculates the number of correctly identified string-fret combinations divided by the total number of ground truth string-fret combinations. This metric measures how frequently tablature existent in the signal is detected by the system.

$$r_{\text{tab}} = \frac{e^T(Z_{\text{gt}} \odot Z_{\text{pred}})e}{e^T Z_{\text{gt}} e} \quad (6)$$

4.3.3 Tablature F-measure

We define multipitch F-measure, which is the harmonic mean of tablature precision and recall. This metric summarizes the system’s overall performance for tablature estimation.

$$f_{\text{tab}} = \frac{2p_{\text{tab}}r_{\text{tab}}}{p_{\text{tab}} + r_{\text{tab}}} \quad (7)$$

4.3.4 Tablature Disambiguation Rate

Finally, we introduce a tablature disambiguation rate (TDR) which is computed by dividing the total number of correctly identified string-fret combinations by the total number of correctly identified pitches. This metric measures how frequently pitches that are correctly identified are assigned the correct tablature.

$$TDR = \frac{e^T(Z_{\text{gt}} \odot Z_{\text{pred}})e}{e^T(Y_{\text{gt}} \odot Y_{\text{pred}})e} \quad (8)$$

4.4 Tablature Estimation Results

The tablature estimation evaluation results for TabCNN are shown in Table 2. While, to our knowledge, there are no prior approaches to directly compare these metrics to, the results are respectable, as each tablature metric is not too far below its multipitch counterpart. As in multipitch estimation, the proposed system achieves a higher tablature

System	p_{tab}	r_{tab}	f_{tab}	TDR
TabCNN	0.809 \pm 0.029	0.696 \pm 0.061	0.748 \pm 0.047	0.899 \pm 0.033

Table 2. Tablature estimation results for the proposed system, TabCNN, using the metrics we introduce to measure performance in fingering prediction. For all metrics, we report the mean and standard deviation over the entire dataset.

precision than tablature recall. This implies that errors made by the system are more often due to missed string-fret combinations than predicting fingerings not present in the signal. The TDR of 0.899 indicates that over 89% of correctly identified pitches are assigned the correct fingering. This value is promising given that the majority of pitches playable on guitar can be played in multiple locations.

5. DISCUSSION

Viewing the predicted tablature alongside the ground truth³, the system appears to output correct string-fret activations a good amount of the time, with the occasional error. To better understand these results, Figure 3 contains examples of 3 common types of errors made by the proposed system.

False alarms occur when a string-fret combination that is predicted is not actually present in the input signal. This type of error negatively impacts the tablature precision metric. In Figure 3 (a), a false alarm occurs when an F3 pitch is mistakenly detected on the 3rd fret of the D-string. This specific error likely happened because the overtones of the F2 pitch, which is present, could be mistaken for the presence of the F3 pitch an octave up. Also, the predicted fingering is a chord shape commonly used by guitarists called a “power chord.”

Missed detections occur when the system fails to detect a string-fret combination that is present in the input signal. These errors hurt the tablature recall score. In Figure 3 (b), the presence of the Ab4 pitch on the 4th fret of the e-string is missed by the system. This type of error could be due to the note having been played quietly or having mostly faded out by the current frame, but still technically being active according to the ground truth. Also, this specific error can be attributed to octave confusion, since the Ab4 could easily be mistaken for overtones of Ab3 note, which was activated on the 6th fret of the D-string.

Finally, miss-frettings occur when a pitch is correctly detected, but it is assigned the incorrect string-fret combination. This is essentially a simultaneous false alarm and missed detection, so both the tablature precision and tablature recall are negatively affected. However, the TDR metric most directly represents how often this third type of error is avoided. In Figure 3 (c), the Ab4 pitch is detected

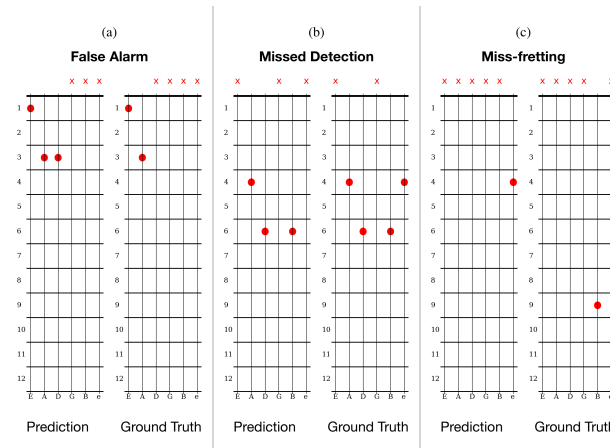


Figure 3. Fretboard diagrams showing examples of 3 common error types made by TabCNN. In each diagram, the vertical lines are guitar strings, and the horizontal lines are the frets. The circles indicate positions of the guitarist’s fingers on the fretboard. An “X” over a string means that string is not sounding.

as an activation on the 4th fret of the e-string. However, the pitch was actually played on the 9th fret of the B-string. In this is specific case, there were no other pitches simultaneously present to cause confusion. Instead, this error is likely due to the system’s failure to predict the correct guitar string based on timbre.

Overall, we observed that errors often occur at the beginning or end of a note. A strategy for improving estimation performance could be the incorporation of a temporal smoothing algorithm.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed TabCNN, a convolutional neural network (CNN) approach to tablature estimation from audio of a solo acoustic guitar performance. Our system performs competitively in guitar multipitch estimation, while providing the advantage of additionally predicting the true fingering used by the guitarist. The guitar community, which frequently shares music in tablature form, can benefit from a system that automates the arduous task of transcribing tablature, while retaining nuance from the original performance, in the form of the exact fretboard positions being used.

Looking ahead to future work, the approach in this paper could be integrated into an end-to-end tablature transcription system. A temporal smoothing method could be added to aggregate these framewise tablature estimations into a note-level transcription. Additionally, we hope that the tablature estimation metrics introduced in this paper can serve to evaluate future guitar transcription approaches that aim to predict the guitarist fingering.

7. REFERENCES

[1] Shunya Ariga, Satoru Fukayama, and Masataka Goto. Song2guitar: A difficulty-aware arrangement system

³ As supplementary materials, video demonstrations showing predicted and ground truth tablature synced with input audio are available at: <https://github.com/andywiggins/tab-cnn>

- for generating guitar solo covers from polyphonic audio of popular music. In *Proceedings of the 18th International Conference on Music Information Retrieval (ISMIR)*, Suzhou, China, 2017.
- [2] Ana M Barbancho, Anssi Klapuri, Lorenzo J Tardón, and Isabel Barbancho. Automatic transcription of guitar chords and fingering from audio. *IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*, 20(3):915–921, 2012.
- [3] Isabel Barbancho, Lorenzo J Tardon, Simone Sammartino, and Ana M Barbancho. Inharmonicity-based method for the automatic generation of guitar tablature. *IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*, 20(6):1857–1868, 2012.
- [4] Rachel M Bittner, Brian McFee, Justin Salamon, Peter Li, and Juan P Bello. Deep salience representations for f0 estimation in polyphonic music. In *Proceedings of the 18th International Conference on Music Information Retrieval (ISMIR)*, Suzhou, China, 2017.
- [5] Gregory Burlet and Ichiro Fujinaga. Robotaba guitar tablature transcription framework. In *Proceedings of the 14th International Conference on Music Information Retrieval (ISMIR)*, Curitiba, Brazil, 2013.
- [6] Gregory Burlet and Abram Hindle. Isolated guitar transcription using a deep belief network. *PeerJ Computer Science*, 3:e109, 2017.
- [7] Benoit Fuentes, Roland Badeau, and Gaël Richard. Blind harmonic adaptive decomposition applied to supervised source separation. In *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, pages 2654–2658. IEEE, 2012.
- [8] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377, 2018.
- [9] Juan S Gómez, Jakob Abeßer, and Estefanía Cano. Jazz solo instrument classification with convolutional neural networks, source separation, and transfer learning. In *Proceedings of the 19th International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2018.
- [10] Yoonchang Han, Jaehun Kim, Kyogu Lee, Yoonchang Han, Jaehun Kim, and Kyogu Lee. Deep convolutional neural networks for predominant instrument recognition in polyphonic music. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 25(1):208–221, 2017.
- [11] Alex Hrybyk and Youngmoo Kim. Combined audio and video analysis for guitar chord identification. *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, Utrecht, Netherlands, 2010.
- [12] Eric J Humphrey and Juan P Bello. From music audio to chord tablature: Teaching deep convolutional networks to play guitar. In *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 6974–6978. IEEE, 2014.
- [13] 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 International Conference on Digital Audio Effects (DAFx)*, pages 219–226, 2014.
- [14] Rainer Kelz, Matthias Dorfer, Filip Korzeniowski, Sebastian Böck, Andreas Arzt, and Gerhard Widmer. On the potential of simple framewise approaches to piano transcription. *arXiv preprint arXiv:1612.05153*, 2016.
- [15] Filip Korzeniowski and Gerhard Widmer. Genre-agnostic key classification with convolutional neural networks. In *Proceedings of the 19th International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2018.
- [16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [17] Matt McVicar, Satoru Fukayama, and Masataka Goto. Autoguitartab: computer-aided composition of rhythm and lead guitar parts in the tablature space. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 23(7):1105–1117, 2015.
- [18] Elias Mistler. Generating guitar tablatures with neural networks. *Master of Science Dissertation, The University of Edinburgh*, 2017.
- [19] Jan Schlüter and Bernhard Lehner. Zero-mean convolutions for level-invariant singing voice detection. In *Proceedings of the 19th International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2018.
- [20] Hendrik Schreiber and Meinard Müller. A single-step approach to musical tempo estimation using a convolutional neural network. In *Proceedings of the 19th International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2018.
- [21] Siddharth Sigtia, Emmanouil Benetos, and Simon Dixon. An end-to-end neural network for polyphonic piano music transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 24(5):927–939, 2016.
- [22] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

- [23] Daniel R Tuohy, Walter D Potter, and Artificial Intelligence Center. An evolved neural network/hc hybrid for tablature creation in ga-based guitar arranging. In *International Computer Music Conference Proceedings*, 2006.
- [24] Qingyang Xi, Rachel M Bittner, Johan Pauwels, Xuzhou Ye, and Juan P Bello. Guitarset: A dataset for guitar transcription. In *Proceedings of the 19th International Conference on Music Information Retrieval (ISMIR), Paris, France*, 2018.
- [25] Kazuki Yazawa, Katsutoshi Itoyama, and Hiroshi G Okuno. Automatic transcription of guitar tablature from audio signals in accordance with player’s proficiency. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3122–3126. IEEE, 2014.
- [26] Kazuki Yazawa, Daichi Sakaue, Kohei Nagira, Katsutoshi Itoyama, and Hiroshi G Okuno. Audio-based guitar tablature transcription using multipitch analysis and playability constraints. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 196–200. IEEE, 2013.
- [27] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.
- [28] Ruohua Zhou and Joshua D Reiss. A real-time polyphonic music transcription system. *Proceedings of the 4th Music Information Retrieval Evaluation eXchange (MIREX)*, 2008.