Leveraging Computer Vision Networks for Guitar Tablature Transcription

Charbel El Achkar[1,2], Raphaël Couturier[2], Abdallah Makhoul[2], and Talar Atéchian[1]

[1] TICKET Lab, Antonine University (UA), Baabda, Lebanon
{charbel.elachkar,talar.atechian}@ua.edu.lb
[2] Université de Franche-Comté, CNRS, institut FEMTO-ST, F-90000 Belfort, France
{charbel.el_achkar,raphael.couturier,abdallah.makhoul}@univ-fcomte.fr

Abstract. Generating music-related notations offers assistance for musicians in the path of replicating the music using a specific instrument. In this paper, we evaluate the state-of-the-art guitar tablature transcription network named TabCNN against state-of-the-art computer vision networks. The evaluation is performed using the same dataset as well as the same evaluation metrics of TabCNN. Furthermore, we propose a new CNN-based network named TabInception to transcribe guitar-related notations, also called guitar tablatures. The network relies on a custom inception block converged by dense layers. The TabInception network outperforms the TabCNN in terms of multi-pitch precision (MP), tablature precision (TP), and tablature F-measure (TF). Moreover, the Swin Transformer achieves the best score in terms of multi-pitch recall (MR) and tablature recall (TR), while the Vision Transformer achieves the best score in terms of multi-pitch F-measure (MF). Motivated by the previous insights, we train the networks with more epochs and propose another network named Inception Transformer (InT) to surpass all the estimation metrics of TabCNN using a single network. The InT network relies on an inception block converged by a Transformer Encoder. The TabInception and the InT network outperformed all estimation metrics of TabCNN except the tablature disambiguation rate (TDR) when trained using a bigger epoch size.

Keywords: Guitar Tablature Transcription · Computer Vision · Automatic Music Transcription.

1 Introduction

Over the last decade, researchers have been exploring the benefits of their innovations in music-related fields while producing tools that can facilitate musicians' daily tasks. One of the latter fields is automatic music transcription (AMT). The AMT is the task of generating a symbolic notation, and instructing a musician how to play a song using a specific instrument. Several studies have been conducted in the AMT field, but only a few of them dealt with the guitar instrument [3, 8, 2]. As for automatic guitar transcription, the guitarist generally

relies on both the music score and the tablature notation to play the song in question. The music score represents the distribution of pitches in time, and the tablature notation defines the guitar strings and the position of the fingers along the fretboard to produce those pitches.
This paper explores several computer vision techniques for automatic guitar transcription. Inspired by the TabCNN model published in [23], Constant-Q spectrograms are generated from each audio track and computed through Computer Vision approaches as visual representations of the audio data. Furthermore, we propose a new CNN-based network named TabInception that relies on Inception and Dense Blocks for automatic guitar transcription. Moreover, we propose another network named Inception Transformer (InT) to attempt to improve the results of TabInception and other featured networks. The Int network relies on an Inception Block converged by the Transformer Encoder Block proposed in [21]. Thus, the leading purpose of this study is to evaluate the TabCNN network against state-of-the-art computer vision networks while proposing new networks that might be capable of outperforming the latter network in the field of guitar tablature transcription. All the aforementioned networks, in addition to the TabInception and the InT network, are evaluated using the GuitarSet dataset published in [25], by the fact that TabCNN was assessed earlier against this dataset [23]. At a broader level, the aim is to explore which of the shallow networks like TabCNN or the deeper networks, such as the proposed ones, can perform better on music transcription use cases. The remainder of this paper is organized as follows: In Section 2, recent automatic guitar transcription studies are discussed. Section 3 presents the selected dataset in addition to the adopted preprocessing procedure. Section 4 interprets the proposed networks for automatic guitar transcription, while Section 5 compares the proposed networks with state-of-the-art CNNs and Transformer-based networks in terms of multi-pitch and tablature estimation metrics. Section 6 concludes the work and gives some directions for future work.

2 Related Work

The technological advances and innovations in the field of human-computer interaction were reflected in various practices. A novel method for gesture recognition was proposed in [26], emphasising the need for improved data generalisation. Another initiative extended innovation by discussing multimodal learning and its relevance in computer vision [4]. Advances in technology such as EMG signal analysis helped in the development of highly accurate recognition methods [13] after demonstrating the continuous progress in this area. Another area of recognition is automatic tablature transcription, which we are developing in this study.
Many studies are proposed for automatic tablature transcription, but only a few seek to detect the real fretting of the guitarist. One of the first approaches leverages the fundamentals and partials for candidate pitches to determine the most used string per pitch [3]. This approach is limited to detecting no more

than four pitches sounding simultaneous. Two years later, a system for applying the Blind Harmonic Adaptive Decomposition Algorithm was developed to classify several performance parameters, including the detection of the note's guitar string [8]. This system is not evaluated for framewise tablature estimation. Nevertheless, it is considered an insightful approach for multi-pitch estimation and guitar tablature estimation. Additionally, several studies focused on the guitar in their pursuit of automatic transcription. For instance, A. M. Barbancho et al. [2] transcribed guitar chords and fingering using a hidden Markov Model, while Humphrey and Bello [11] took the benefit of a convolutional neural network (CNN) model to achieve chord recognition.
The results of the latter approach encouraged the researchers to take advantage of CNN for similar music-related tasks. A combination of a CNN for framewise acoustic modelling and a recurrent neural network (RNN) model is proposed for piano transcription in [17].
The use of neural networks for music-related tasks helped in providing solutions for tablature arrangement problems [20]). It tackled various music information retrieval tasks such as instrument classification [9, 10], music genre classification [7], and singing voice detection [16]. It also helped in achieving the first guitar tablature estimation model using CNNs. The model was trained using solo acoustic guitar performances of the GuitarSet dataset presented in Xi et al. (2019) [25], while outperforming state-of-the-art multi-pitch estimation algorithms. This paper also introduced a set of metrics found to be specific for evaluating guitar tablature estimation models, as described in Wiggins and Kim (2019) [23]. Several attempts took place to improve the TabCNN's results presented in [23]. One of those attempts was the thesis report in Maaiveld et al. (2021) [15]. It yielded insights into the CNNs' functioning for automatic music transcription. The proposition relied on several adaptations such as data augmentation, Oracle method adaptation, and increasing the amount of training data. The latter study was not able to outperform the results of the TabCNN [23]) but presented intuitive conclusions, such as the fact that Dense layers play a major role in tablature estimation CNNs and that the size of the dataset is a key factor in the model's performance.
The fast growth of neural networks encouraged researchers to test the latest approaches in the music industry. An unsupervised pitch estimation model was reported by Wiggins and Kim (2020) [24] to analyse audio clips by estimating their pitches and amplitudes. The model was not tested through experiments but gave thoughtful ideas for further unsupervised acoustic guitar transcription attempts. Also, a method for generating note-level transcription for guitar transcription is proposed to demonstrate successful transcription using notes rather than frames [12]. This work outperformed the conventional frame-level CNN methods. Nevertheless, it did not outperform all TabCNN's estimation metrics results [23].

Last but not least, a unified model and methodology for estimating pitch contours took place to transcribe guitar tablatures [5]. It produced pitch estimates with a higher resolution than modern models. However, and to the best

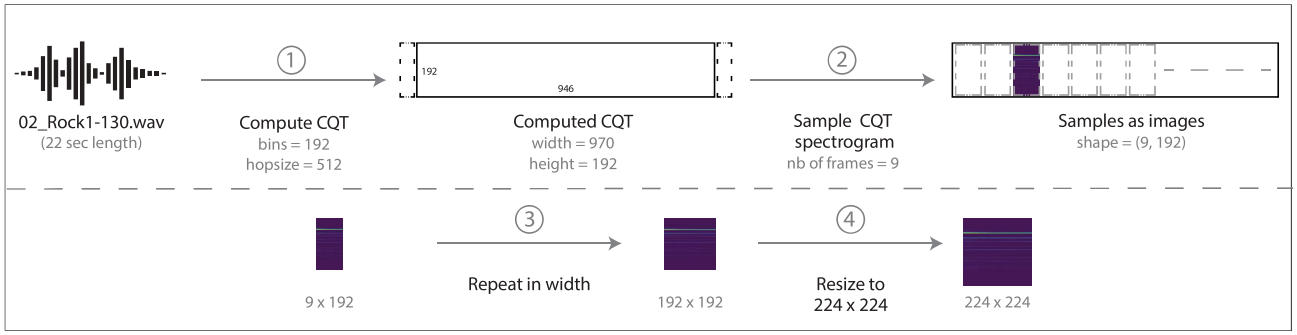

Fig. 1. Audio to Image transformation through Constant-Q Transform computation

of our knowledge, neither the approaches listed in this section nor any other associated work can outperform all TabCNN's [23] estimation metrics for guitar tablature transcription.

3 Data Selection And Preparation

The TabCNN model proposed in [23] holds the state-of-the-art record for guitar tablature transcription using CNNs. In this study, the same dataset chosen in TabCNN is used, in addition to the preprocessing procedure for computing audio features to images.

Similar to the TabCNN approach [23], the audio recordings were downsampled from 44100 to 22050 Hz to reduce the input signals' dimension. The input signals were normalized to obtain an identical range of amplitudes among all the recordings. This normalization is essential to achieve the next step: computing the convenient audio signal feature out of each recording. Inspired by previous experiences in guitar tablature transcription, the Constant-Q Transform (CQT) is adopted as the feature to compute. For this reason, and to directly compare all studied networks with TabCNN, similar CQT parameters are adopted. As shown in Fig. 1, and using the *Librosa* Python library, the CQT is computed over the audio recording in the first place. A value of 192 is selected for the bins and 512 for the hopsize parameter. The bins parameter consists of the intervals between samples in the frequency domain. It is estimated by dividing the sampling rate by the Fast Fourier Transform (FFT) size. On the other hand, the hopsize is the number of samples between each successive FFT window. It is processed by dividing the FFT size by an integer defining the overlap factor of FFT windows. As for this parameter selection validation, we plotted the chroma_cqt features using the selected bins and hopsize parameters. The chroma features captures harmonic and melodic characteristics of music while being robust to changes in timbre and instrumentation. In this case, the chroma_cqt analyses these musical features following the CQT parameter already computed. While visualizing the

plots, it was found that the produced chroma features were slightly noisy and unclear. Thus, the number of bins per octave parameter was scaled from its default value (12) to 24 to clarify the computed CQT by increasing its resolution. The CQT is then computed using the new parameter values: hopsize = 512, number of bins = 192, and number of bins per octave = 24. At this stage, the computed CQT can be obtained as a visual representation of size 970x192, while adding zero padding on both sides of the CQT to achieve the sampling step (the initial size is 946x192 since the audio used in this example has a 22 seconds length and the hopsize used corresponds to 43 frames per seconds approximately). In addition, the sampling step (second in Fig. 1) is where the sliding context window of 9 frames takes place to generate multiple images of size 9x192 out of the initial computed CQT. The entire process results in multiple CQT images out of the same audio recording. Each image concerns nine successive frames of the initially computed CQT. It was essential to resize the sampled CQT images into square-shaped images to compare the proposed and the existing approaches with state-of-the-art computer vision networks. The majority of the latter networks are trained and evaluated using squared images. Thus, the need to resize the images to the smallest recurrent size, 224x224. Consequently, using the function of the *numpy* Python library, we repeated the same pixels of the sampled image in width to achieve a size of 192x192. Then, the images were resized from 192x192 to 224x224. It is important to note that this is the most convenient resizing technique, since resizing from 9x192 directly to 224x224 may distort the image content. Also, both versions were kept, the 9x192 sampled images and the 224x224 resized images for further network comparisons. Concerning the annotations, the same approach in [23] is adapted to sample the stringwise pitch features stored in the JAMS files. These features are transformed into binary matrices. Each matrix represents a frame belonging to a computed audio recording.

```
[[1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
```

Fig. 2. Label associated to the 512th frame of the 02_Rock_1_130.wav recording

Fig. 2 represents a matrix associated with one of the frames in a guitar recording. The matrix is of shape 6x21, equal to the six strings of the guitar having 21 different fret classes. Since the GuitarSet is recorded using an acoustic guitar of 19 frets, the remaining two frets correspond to two descriptive states of a guitar string. The first fret associated with the first column (from left to right) of the matrix indicates if the string is in an open state (no frets are pressed),

while the second fret (second column) indicates if the latter is in a closed state. The remaining 19 frets correspond to the remaining 19 columns of the matrix to define the pressed fret at a given frame.

4 Proposed Networks

4.1 The TabInception Network

Inspired by the insightful conclusion in [15], especially the point mentioning the essential role of Dense layers in guitar tablature transcription, a custom CNN-based network named TabInception is proposed. As shown in Fig. 3, the TabInception starts with an input layer taking images of shape (192, 9, 1). Thus, it involves swapping the axes of the computed images in the preprocessing steps to provide a proper data fitting. Consequently, we propose adding a two-dimensional convolutional layer of 32 filters adjacent to a Batch Normalization, a Relu activation, and a Max Pooling layer with a pool size equal to (4,1). The output of the latter bundle is fed into an Inception block that can be described as follows:
The proposed Inception block uses a similar architecture to the Inception v4 architecture implemented in [18], while adding Batch Normalization and Relu activation layers among adjacent Conv2D layers.
Fig. 3 shows a high-level visualization of the Inception block, where several base convolutional blocks (base_conv_block) are interconnected together and are concatenated at the end with a MaxPooling2D layer. Each base_conv_block consists of a Batch Normalization, a Relu activation, and a Conv2D layer with 32 filters. This technique ensures that the adopted inception approach will be less likely to over-fit. Also, the Batch Normalization improves memory optimization to back-propagation while reducing the intensive computations caused by convolutional layers. After concatenating the Inception block's calculations, the output is fed to the Transition Block. As presented in Fig. 3, the Transition block is the same as a base_conv_block with the addition of an AveragePooling2D layer after the Conv2D one. This approach is essential to downsample the huge spatial dimensions caused by the Inception block, and to converge the network into its decisive and final layers. Since TabInception concerns guitar tablature transcription, the network should be able to compute multidimensional calculations. Hence, the use of the Flatten layer to convert the sixth channelled output to a single channelled one for Dense layer calculations. Each of the sixth channels consists of a guitar string having 21 frets. The Dense calculations are dropped out with a value of 0.5 while re-iterating the Dense computations using a number of units equal to the multiplication of the number of strings and frets ($6 \times 21 = 126$ units). The output of the concluding layer is reshaped back to (6, 21) to compute the activation of each guitar string separately. Finally, the softmax_by_string activation function proposed in [23] is used to concatenate the separately computed six softmax calculations and to unify the output.

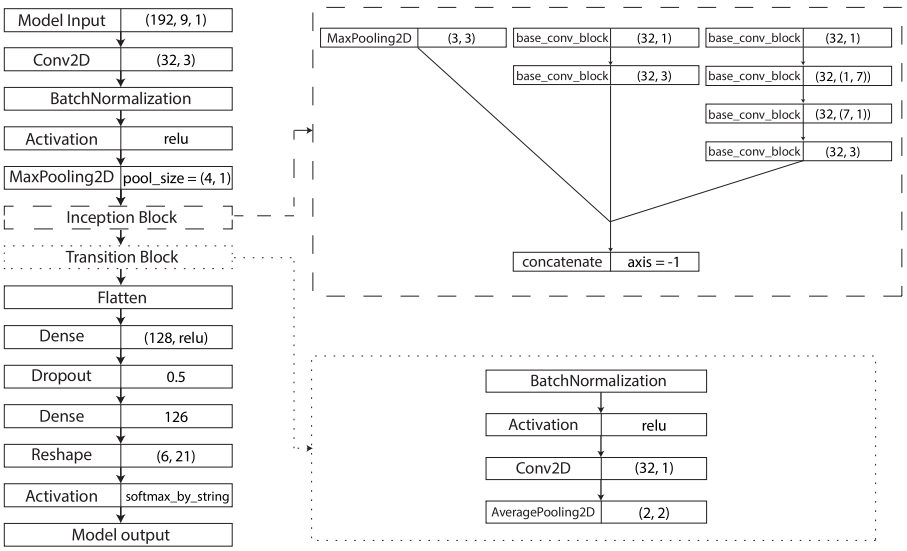

Fig. 3. Architecture of the TabInception network

4.2 The Inception Transformer Network

Inspired by the precision of the TabInception network and the recall and the F-measure of the Transformer-based models, the InT proposes a fusion between the Inception block of the TabInception network and the Transformer Encoder proposed in [21] and adopted in the Vision Transformer (ViT) model [6].
Similar to TabInception, The InT network relies on similar Input layers except using a number of filters equal to 64 instead of 32, as well as adding 4 strides to the initial Conv2D layer. The increased number of filters is adapted into the Inception Block of the InT network. The latter Block is identical to the one used in TabInception except for the number of filters. Furthermore, the computed calculations are concatenated and reshaped to (96, 64) to match the input shape needed for the Transformer Encoder. The Transformer Encoder adopted in [6] expects a sequence of embeddings vector that serves as input. These vectors consist of positional embeddings in addition to those of previously generated patches. As for the Transformer Encoder of the InT network, it expects a sequence of positional embedding along with the reshaped tensors produced out of the previously mentioned Inception block. Thus, the idea of generating patches and feeding them to the encoder is replaced by loading the encoder with convolutional-based tensors. The Transformer Encoder is responsible for alternating mutlihead self-attention blocks with MLP blocks. A LayerNormalization layer is applied before every block in addition to a residual connection after every block inside the encoder [22, 1]. The Transformer Encoder used in the proposed

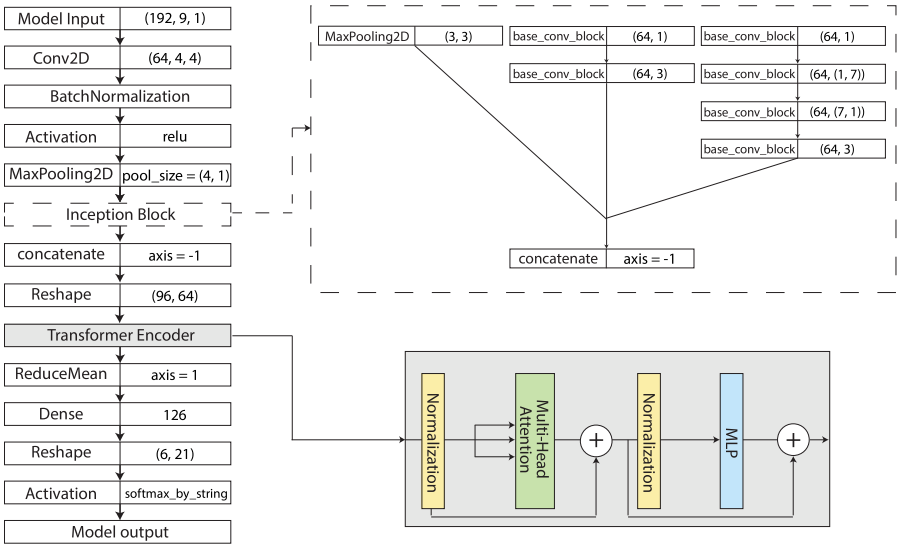

Fig. 4. Architecture of the Inception Transformer (InT) network

InT network relies on six transformer layers, an MLP dimension of 128, and a patch size equal to 4. The output of the latter encoder is fed to a ReduceMean layer to reduce the dimension of the tensor for the succeeding Dense layer. The Dense layer of shape 126 is then fed to the same concluding layers as TabInception to compute the activation of each guitar string separately. The same activation function and optimizer are adopted for both, the TabInception and the Inception Transformer networks.

5 Experiments

In this section, the TabCNN network is compared to the TabInception and InT networks, in addition to state-of-the-art computer vision (CV) networks such as [14, 19, 6]. The CV networks were modified slightly by reshaping their decisive layers to provide a unified output across all networks (shape of (6, 21) as a matrix of 6 strings with 21 frets each). The implementation took place using the official version of the CV networks or its equivalent in *Keras*. The same parameters and hyperparameters are used across all the networks for better comparison with TabCNN. A batch size of 128 and a 6-fold cross-validation training method were selected while relying on the preprocessed CQT as input images to the network. The images were divided using an 85% training and 15% testing ratio for all networks.

Image Size	Epochs	Network	MP	MR	MF	TP	TR	TF	TDR
192x9	8	TabCNN	*0.9 ±0.016*	*0.764 ±0.043*	*0.826 ±0.025*	*0.809 ±0.029*	*0.696 ±0.061*	*0.748 ±0.047*	*0.899 ±0.033*
192x9	16	TabCNN	0.927 ±0.008	0.7805 ±0.003	0.8474 ±0.0176	0.8101 ±0.0115	0.711 ±0.0268	0.757 ±0.0151	0.873 ±0.006
192x9	300 with ES	TabCNN	0.8549 ±0.013	0.722 ±0.021	0.782 ±0.0172	0.815 ±0.0185	0.697 ±0.0346	0.751 ±0.0237	**0.953** ±0.0136
192x9	8	TabInception	0.941 ±0.008	0.7189 ±0.031	0.815 ±0.0176	0.7973 ±0.0115	0.6455 ±0.0268	0.7134 ±0.0151	0.8473 ±0.006
192x9	16	TabInception	**0.9688** ±0.0192	0.7454 ±0.0518	0.8425 ±0.0317	**0.8519** ±0.0199	0.6911 ±0.0608	**0.7631** ±0.0443	0.8793 ±0.0244
192x9	300 with ES	TabInception	**0.9533** ±0.0147	0.7834 ±0.0339	0.86 ±0.0187	**0.8639** ±0.0158	**0.739** ±0.0356	**0.7965** ±0.0221	**0.906** ±0.0104
192x9	8	ViT	0.908 ±0.0165	0.8209 ±0.0373	0.8622 ±0.0204	0.7291 ±0.0329	0.7144 ±0.0444	0.7216 ±0.0349	0.802 ±0.0313
192x9	16	ViT	0.882 ±0.0066	0.8 ±0.0155	0.839 ±0.0056	0.7043 ±0.0074	0.6901 ±0.0183	0.6971 ±0.0099	0.798 ±0.0093
192x9	300 with ES	ViT	0.937 ±0.0115	**0.8524** ±0.0264	**0.8927** ±0.013	0.7586 ±0.0201	**0.7441** ±0.0313	0.7512 ±0.0224	0.8096 ±0.0203
192x9	8	InT	0.8785 ±0.0083	0.8213 ±0.0092	0.8489 ±0.0056	0.7202 ±0.0128	0.7206 ±0.0132	0.7203 ±0.0117	0.8198 ±0.0089
192x9	16	InT	0.891 ±0.0031	0.82 ±0.0062	0.854 ±0.0031	0.7134 ±0.0356	0.7019 ±0.0738	0.7076 ±0.0473	0.8 ±0.0034
192x9	300 with ES	InT	**0.9481** ±0.0057	**0.914** ±0.0077	**0.9307** ±0.0043	**0.8551** ±0.0242	**0.8041** ±0.0435	**0.828** ±0.0295	**0.901** ±0.00615
224x224	8	SwinTF	0.8875 ±0.0146	0.8034 ±0.0374	0.843 ±0.0146	0.709 ±0.0226	0.693 ±0.0212	0.7 ±0.041	0.798 ±0.031
224x224	16	SwinTF	0.9035 ±0.0075	0.8421 ±0.0034	0.8717 ±0.0024	0.7331 ±0.0019	0.726 ±0.0069	0.729 ±0.002	0.8114 ±0.007
224x224	300 with ES	SwinTF	0.9259 ±0.011	**0.8531** ±0.0204	**0.888** ±0.0085	0.7307 ±0.0122	0.7191 ±0.014	0.7248 ±0.0215	0.789 ±0.019
224x224	8	EfficientNetB0	0.839 ±0.0176	0.7691 ±0.0071	0.8025 ±0.016	0.6739 ±0.026	0.6406 ±0.048	0.656 ±0.034	0.803 ±0.031
224x224	16	EfficientNetB0	0.861 ±0.006	0.691 ±0.067	0.766 ±0.0386	0.733 ±0.0359	0.615 ±0.0695	0.668 ±0.0475	0.851 ±0.0401
224x224	300 with ES	EfficientNetB0	0.8947 ±0.0118	0.7747 ±0.037	0.83 ±0.0273	0.748 ±0.0309	0.6723 ±0.0587	0.708 ±0.0407	0.836 ±0.0355

Table 1. Comparative table for guitar tablature transcription using computer vision networks. The best score per metric is highlighted in **black**, the second best in **green**, and the third best in **red**.

The Swin Transformer (SwinTF) [14] and the EfficientNetB0 [19] [1] networks perform their best when trained using squared images since they rely on patch-based architectural structures. Therefore, it was favourable to experiment with both networks using a squared image format instead of performing architecture changes for fitting non-squared images. Hence, the resized 224x224 CQT images are used for these approaches. In contrast, the 192x9 CQT images are adopted to train the TabCNN, the TabInception, The InT, and the Vision Transformer (VIT) networks [6]) by the fact that they are not image size dependent. Thus, the proposed networks can be directly compared with the TabCNN network [23], while presenting other approaches where an image resizing may impact the training results.
It is important to mention that the VIT network relies on a patch-based structure. Nevertheless, it can be fed with non-squared images by its ability to transform each image into patches of equal width and height size. Thus, the input images are transformed into patches before being fed to the VIT encoder. In this experiment, and for the VIT network exclusively, we adopt a patch size of 4 and a hidden size of 64 after performing several empirical tests where both parameters were varied to maximize the evaluation results. Consequently, each of the 192x9 input images is transformed into 64 patches, having a size of 64x64 for each patch. The latter is conducted using the patch generation function proposed in the original code of the VIT [6] for rescaling and transforming the input images into patches. Table 1 presents all the networks that we compare with TabCNN. The first training of TabCNN is written in *italic* to indicate that its results are shown as they appear in the official contribution. Contrarily, the remaining training is performed in our test environment. The table header presents seven different multi-pitch and tablature estimation metrics. Each metric manifests an essential role already proposed in [23]. The metrics referenced in Table 1 are the following: Multi-pitch Precision (MP), Multi-pitch Recall (MR), Multi-pitch F-measure (MF), Tablature Precision (TP), Tablature Recall (TR), Tablature F-measure (TF), and Tablature Disambiguation Rate (TDR). As shown in Table 1, the networks were trained using two different epoch sizes. An epoch size of 8 is used to compare the results with the official TabCNN results. Furthermore, an epoch size of 16 is adopted to identify the behaviour of each network using longer iterations. At an epoch size of 8, the TabInception outperformed the TabCNN by 4.1 percentage points (*pp* in terms of multi-pitch precision (MP). On the other hand, the InT network and the transformer-based networks (ViT and SwinTF) can either outperform or obtain the same results as the TabCNN in terms of multi-pitch recall (MR) and F-measure (MF). The VIT exceeded the TabCNN by 3.62 *pp* in terms of MF and 5.69 *pp* in terms of MR. Also, the SwinTF exceeded TabCNN's MF by 1.7 *pp*, and TabCNN's MR by 3.94 *pp*. As for the InT network, it exceeded TabCNN's MF by 2.29 *pp* and TabCNN's MR by 5.73 *pp*. These results show that the TabInception network is a good solu-

[1] The B0 base model of EfficientNet is the only selected model for this experiment since it is the only compatible model for computing 224x224-sized images. As for the SwinTF, we use the base architecture for this experiment, also known as swin_b.

tion for better pitch detection, while both, the proposed InT network and the transformer-based networks are better options when the comparison concerns the MR and MF metrics. The TabInception network outperformed the TabCNN network in terms of MP, TP, and TF metrics when increasing the epoch size to 16. It achieved the greatest results concerning the multi-pitch precision metric. It outperformed TabCNN's MP by 4.18 *pp*, the TP by 4.17 *pp*, and the TF by 0.61 *pp*. As for the proposed InT network, it surpassed the TabCNN network in terms of MR and MF. Moreover, among the remaining networks, the SwinTF improved its results using an increased epoch size. Contrarily, the EfficientNetB0 could not exceed TabCNN's results in either epochs variations.
Motivated by the increase in metrics when raising the epochs size, we configured an epochs size of 300 while establishing the early stopping mechanism with a patience value equal to 5. Thus, the models will keep training until they reach a safe point to stop without overfitting. All the tests produced using the latter configuration are highlighted in a dashed outline in Table 1 to discriminate the latter from the legacy configuration (8–16 epochs without an early stopping mechanism). **green** Also, we highlight the best score per metric with a **black** bold color, the second best with a **green** bold color, and the third best with a **red** bold color. The results show that the TabInception network achieved the best result in terms of TP, and the InT network achieved the best results for MR, MF, TR, and TF. Both proposed networks were able to surpass all of TabCNN's results except the TDR metric. The TabCNN preserved the best result in terms of TDR in that case. The significant TDR value of TabCNN is due to the closer MP and TP values compared to the remaining networks. As for the SwinTF, the ViT, and the EfficientNetB0, some of their metrics' results increased but could not considerably surpass TabCNN's values at all times.

6 Conclusion and Future Work

In this paper, two networks were proposed for guitar tablature transcription. The first network, TabInception, relies on a custom inception block converged by dense layers. The second network, Inception Transformer (InT), relies on a similar inception block of TabInception converged by a Transformer Encoder. Both networks were compared against the state-of-the-art guitar tablature transcription network named TabCNN and other recent computer vision networks. The experiment results showed that the proposed networks can outperform the TabCNN in terms of multi-pitch precision (MP), multi-pitch recall (MR), multi-pitch F-measure (MF), tablature precision (TP), tablature recall (TR), and tablature F-measure (TF). Our future work should focus on exploring the performance and the usability of both proposed networks for transcribing tablatures of other string instruments such as the violin, cello, and harp. Furthermore, it would be essential to test the proposed networks on computer vision use cases beyond the tablature transcription or even the music field to better evaluate and explore the importance of such contribution. The source code for imple-

menting the discussed networks is publicly available on the following repository: `https://github.com/elachkarcharbel/Guitar-Tablature-Transcription`

Acknowledgments

This work was performed using HPC resources from GENCI–IDRIS (Grant 2021-AD011013289). It is funded by the "Agence Universitaire de la Francophonie" (AUF) and supported by the EIPHI Graduate School (contract ANR-17-EURE-0002).

References

1. Baevski, A., Auli, M.: Adaptive input representations for neural language modeling. arXiv preprint arXiv:1809.10853 (2018)
2. Barbancho, A.M., Klapuri, A., Tardon, L.J., Barbancho, I.: Automatic transcription of guitar chords and fingering from audio. IEEE Transactions on Audio, Speech, and Language Processing **20**(3), 915–921 (2012). https://doi.org/10.1109/TASL.2011.2174227
3. Barbancho, I., Tardon, L.J., Sammartino, S., Barbancho, A.M.: Inharmonicity-based method for the automatic generation of guitar tablature. IEEE Transactions on Audio, Speech, and Language Processing **20**(6), 1857–1868 (2012). https://doi.org/10.1109/TASL.2012.2191281
4. Bayoudh, K., Knani, R., Hamdaoui, F., Mtibaa, A.: A survey on deep multimodal learning for computer vision: advances, trends, applications, and datasets. The Visual Computer pp. 1–32 (2021)
5. Cwitkowitz, F., Hirvonen, T., Klapuri, A.: Fretnet: Continuous-valued pitch contour streaming for polyphonic guitar tablature transcription. arXiv preprint arXiv:2212.03023 (2022)
6. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
7. El Achkar, C., Couturier, R., Atéchian, T., Makhoul, A.: Combining reduction and dense blocks for music genre classification. In: Mantoro, T., Lee, M., Ayu, M.A., Wong, K.W., Hidayanto, A.N. (eds.) Neural Information Processing. pp. 752–760. Springer International Publishing, Cham (2021)
8. Fuentes, B., Badeau, R., Richard, G.: Blind harmonic adaptive decomposition applied to supervised source separation. In: 2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO). pp. 2654–2658 (2012)
9. Gómez, J.S., Abeßer, J., Cano, E.: Jazz Solo Instrument Classification with Convolutional Neural Networks, Source Separation, and Transfer Learning. In: Proceedings of the 19th International Society for Music Information Retrieval Conference. pp. 577–584. ISMIR, Paris, France (Sep 2018). https://doi.org/10.5281/zenodo.1492481, `https://doi.org/10.5281/zenodo.1492481`
10. Han, Y., Kim, J., Lee, K.: Deep convolutional neural networks for predominant instrument recognition in polyphonic music. IEEE/ACM Transactions on Audio, Speech, and Language Processing **25**(1), 208–221 (2017). https://doi.org/10.1109/TASLP.2016.2632307

11. Humphrey, E.J., Bello, J.P.: From music audio to chord tablature: Teaching deep convolutional networks toplay guitar. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 6974–6978 (2014). https://doi.org/10.1109/ICASSP.2014.6854952
12. Kim, S., Hayashi, T., Toda, T.: Note-level automatic guitar transcription using attention mechanism. In: 2022 30th European Signal Processing Conference (EUSIPCO). pp. 229–233. IEEE (2022)
13. Li, J., Wei, L., Wen, Y., Liu, X., Wang, H.: An approach to continuous hand movement recognition using semg based on features fusion. The Visual Computer **39**(5), 2065–2079 (2023)
14. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 10012–10022 (2021)
15. Maaiveld, T., Driedger, J., Yela, D., Meroño-Peñuela, A.: Automatic tablature estimation with convolutional neural networks: Approaches and limitations (04 2021). https://doi.org/10.13140/RG.2.2.13906.48320
16. Schlüter, J., Lehner, B.: Zero-mean convolutions for level-invariant singing voice detection (09 2018)
17. Sigtia, S., Benetos, E., Dixon, S.: An end-to-end neural network for polyphonic piano music transcription. IEEE/ACM Transactions on Audio, Speech, and Language Processing **24**(5), 927–939 (2016). https://doi.org/10.1109/TASLP.2016.2533858
18. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: Proceedings of the AAAI conference on artificial intelligence. vol. 31 (2017)
19. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
20. Tuohy, D.R., Potter, W.D.: An evolved neural network/hc hybrid for tablature creation in ga-based guitar arranging. In: International Conference on Mathematics and Computing (2006)
21. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
22. Wang, Q., Li, B., Xiao, T., Zhu, J., Li, C., Wong, D.F., Chao, L.S.: Learning deep transformer models for machine translation. arXiv preprint arXiv:1906.01787 (2019)
23. Wiggins, A., Kim, Y.: Guitar Tablature Estimation with a Convolutional Neural Network. In: Proceedings of the 20th International Society for Music Information Retrieval Conference. pp. 284–291. ISMIR, Delft, The Netherlands (Nov 2019). https://doi.org/10.5281/zenodo.3527800
24. Wiggins, A., Kim, Y.: Towards unsupervised acoustic guitar transcription. Journal, vol **7**(2), 43–55 (2020)
25. Xi, Q., Bittner, R.M., Pauwels, J., Ye, X., Bello, J.P.: Guitarset (Aug 2019). https://doi.org/10.5281/zenodo.3371780, `https://doi.org/10.5281/zenodo.3371780`
26. Zeghoud, S., Ali, S.G., Ertugrul, E., Kamel, A., Sheng, B., Li, P., Chi, X., Kim, J., Mao, L.: Real-time spatial normalization for dynamic gesture classification. The Visual Computer pp. 1–13 (2022)