AN END-TO-END FRAMEWORK FOR AUDIO-TO-SCORE MUSIC TRANSCRIPTION ON MONOPHONIC EXCERPTS

Miguel A. Román
U.I. for Computing Research
University of Alicante
Alicante, Spain
mroman@dlsi.ua.es

Antonio Pertusa
U.I. for Computing Research
University of Alicante
Alicante, Spain
pertusa@ua.es

Jorge Calvo-Zaragoza
PRHLT Research Center
Universitat Politècnica de València
Valencia, Spain
jcalvo@prhlt.upv.es

ABSTRACT

In this work, we present an end-to-end framework for audio-to-score transcription. To the best of our knowledge, this is the first automatic music transcription approach which obtains directly a symbolic score from audio, instead of performing separate stages for piano-roll estimation (pitch detection and note tracking), meter detection or key estimation. The proposed method is based on a Convolutional Recurrent Neural Network architecture directly trained with pairs of spectrograms and their corresponding symbolic scores in Western notation. Unlike standard pitch estimation methods, the proposed architecture does not need the music symbols to be aligned with their audio frames thanks to a Connectionist Temporal Classification loss function. Training and evaluation were performed using a large dataset of short monophonic scores (incipits) from the RISM collection, that were synthesized to get the ground-truth data. Although there is still room for improvement, most musical symbols were correctly detected and the evaluation results validate the proposed approach. We believe that this end-to-end framework opens new avenues for automatic music transcription.

1. INTRODUCTION

Automatic Music Transcription (AMT) is a very relevant field within the Music Information Retrieval (MIR) community. This task can be defined as the automated process of converting an audio recording into any kind of musically-meaningful structured format. The usefulness of this process is very broad, especially for MIR algorithms such as content-based music search, symbolic music similarity, or symbolic musicological analysis.

However, this is a challenging task and state-of-the-art methods currently obtain a performance significantly below a human expert. In order to obtain a complete score from a waveform, it is necessary to perform pitch detection, note onset/offset detection, loudness estimation and quantization, instrument recognition, extraction of rhythmic information, and time quantization [2].

Most music transcription systems focus on two of these stages: pitch detection, where pitches at each time frame of the audio are estimated, and note tracking [32], where the estimations of the previous step are discretized into sequences of 3-tuples (onset, offset, pitch). The output in this case is a piano-roll, that is, a two-dimensional representation of notes across time [2]. Multiple pitch estimation techniques include spectrogram factorization methods [1, 3, 28] and discriminative approaches, which perform frame-by-frame pitch estimation using statistical models [10], signal processing methods [23, 35], or machine learning techniques [4] including deep neural networks [17, 27, 30]. Some works also integrate musical language models into the pitch estimation process to resolve output ambiguities [27, 34].

Supervised learning approaches for piano-roll estimation require the ground truth to be aligned for training. Matching pitches frame by frame with their corresponding waveform samples is a time-consuming task and, although there are some efforts in this direction with datasets such as MAPS [10], RWC [11] or MusicNet [29], currently there are no very large AMT corpora. Beyond the difficulty of performing an accurate annotation, frame-by-frame estimation has some additional issues to be taken into account. For example, when a whole note is played using a plucked string instrument such as a guitar, the quick decay of its harmonic amplitudes produces frames with a very low intensity at the end of the note, causing ambiguities when labeling the offset frames.

In addition, as pointed out in [2], AMT algorithms are usually developed independently to carry out individual tasks such as multiple pitch detection, beat tracking and instrument recognition. Some existing AMT methods, such as the ones proposed in [19–21], also include rhythm estimation and time quantization. Still, the challenge remains to combine the outputs of the individual tasks to perform joint estimation of all parameters, in order to avoid the cascading of errors when algorithms are combined sequentially.

In this work we intend to open a new framework to address the AMT task. Our proposal is to consider end-to-

end machine learning strategies, with which this task can be carried out holistically. In other words, we aim at using a waveform as input, and directly obtaining a music score at the output taking into account all its components (pitches, note durations, time signature, key signature, etc.) jointly.

The task of directly estimating a symbolic score from audio is certainly different from that of estimating a piano-roll. While piano-roll estimation aims to extract what has been played from the audio as exact as possible, in the score transcription task the goal is to obtain a symbolic representation from what the musician read, which includes abstracting away some information such as loudness.

For this, we address score estimation using Deep Neural Networks. We specifically consider the use of a Convolutional Recurrent Neural Network, which is responsible of both processing the input spectrogram to extract meaningful features and predict an output sequence that represents the music contained in a given audio recording. Thanks to the Connectionist Temporal Classification (CTC) loss function, this kind of networks can be trained in terms of pairs (input, output), without needing of dividing the process into smaller stages or providing framewise annotations. The idea is that the prediction is forced to be encoded in terms of actual music-notation elements.

It is important to emphasize that the objective of this work is not to outperform the accuracy of previous approaches, but to propose a framework with which to address the AMT task. In order to demonstrate the feasibility of this formulation, our experiments are restricted to a constrained scenario, using audio recordings from monophonic scores that were synthesized using a piano. We are aware that the main challenge in AMT is to deal with polyphonic real music. In a future work we plan to extend the proposed approach to detect polyphonic scores, although its effectiveness with sound mixtures is yet to be studied.

The evaluation results in this constrained scenario validates the proposed framework and show that the the proposed approach obtains reliable results, correctly detecting most musical symbols.

The rest of the paper is organized as follows: the corpus used for evaluation is described in Section 2; the holistic neural framework proposed for the AMT task is described in Section 3; the series of experiments carried out are detailed in Section 4; and finally, the conclusions of the current work are summarized in Section 5, pointing out some interesting avenues for future work as well.

2. DATASET

In order to get the ground truth for our framework, we used the RISM [1] collection [26], which currently contains more than one million incipits (short monophonic music excerpts). This corpus is very useful for music retrieval tasks because of its size and the fact that it contains real music written by human composers [31]. Spectrograms from synthesized incipits are the inputs to our method, and their corresponding symbolic scores are the outputs. The scheme of the proposed method can be seen in Figure 1.

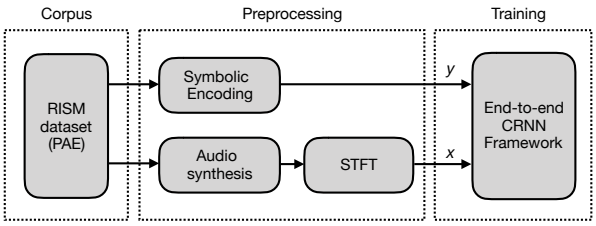


Figure 1: Data acquisition for training. RISM incipits are converted into our music notation format and magnitude spectrograms (Short-Time Fourier Transform, STFT) are also computed from synthesized versions of the incipits. The inputs of the proposed framework (x) are the symbolic data and the outputs are the spectrograms (y). Frame-by-frame alignment is not necessary.

2.1 Preprocessing

RISM incipits are formatted in Plaine & Easie Code (PAE). We randomly selected a subset of 71,400 incipits in Western notation and converted them into the music notation format that can be seen in Table 1, where each symbol is encoded using a single character. This notation is oriented to represent the music as a language, similarly to what a speech recognition system does. Following this analogy, we consider a music note as a *word* (for example, C♯4♩) containing several *characters* from an alphabet set Σ that can be seen in Table 1, and which is separated to other words by blank spaces. Rests are represented in the same way, with a word consisting of the rest symbol and its duration. In addition to notes and rests, the alphabet set includes clefs, key and time signatures, measure bars and note ties. Every musical symbol in Table 1 is encoded for our framework using a single element (one character).

In order to get the audio files, the RISM PAE incipits were converted into Music Encoding Initiative (MEI) format, and then translated again into MIDI using Meico [2], which unlike Verovio [3] takes into account the key signature.

The synthesis from MIDI files was performed using timidity with the piano program of the default soundfont, obtaining monoaural audio files at 16kHz. Then, magnitude spectrograms were calculated using a 64ms (1024 samples) Hamming window with a 16ms hop (256 samples). All incipits were synthesized using random tempo values in the range [96-144] bpm in order to make the network work with different speeds.

3. FRAMEWORK

We describe in this section the neural model that allows us to face the AMT task directly from an audio signal to a sequence of meaningful symbols.

[1] The complete set of RISM incipits can be downloaded from `https://opac.rism.info/index.php?id=8&L=1&id=8`

[2] `https://github.com/cemfi/meico`

[3] `http://www.verovio.org/index.xhtml`

Class	Symbol	Count	Histogram
Global	Blank	1,526,051	
Clef	G2	39,337	
	F4	4,414	
	C1	22,468	
	C3	1,981	
	C4	3,200	
Key	D♭M	112	
	A♭M	1,065	
	E♭M	6,815	
	B♭M	8,950	
	FM	11,599	
	CM	15,488	
	GM	10,309	
	DM	10,861	
	AM	4,933	
	EM	1,216	
	BM	52	
Pitch	A	87,323	
	B	88,004	
	C	95,190	
	D	100,014	
	E	80,780	
	F	75,579	
	G	84,953	
	♭	70,557	
	♯	55,471	
	Rest	89,635	
Octave	2	2,937	
	3	44,170	
	4	274,590	
	5	284,081	
	6	6,065	
Duration	𝅝	8,686	
	𝅗𝅥	52,541	
	♩	172,933	
	♪	226,395	
	𝅘𝅥𝅯	72,124	
	𝅘𝅥𝅰	6,711	
	.	72,453	
	Tie	10,393	
Time	4/4	27,855	
	2/2	13,848	
	3/4	11,595	
	2/4	7,569	
	6/8	4,950	
	3/8	2,916	
	3/2	1,199	
	12/8	592	
	6/4	417	
	4/2	305	
	9/8	154	
	Barline	245,239	

Table 1: Symbols of the alphabet Σ. Notes are encoded using words of three to five symbols (for example, C♯4♩.).

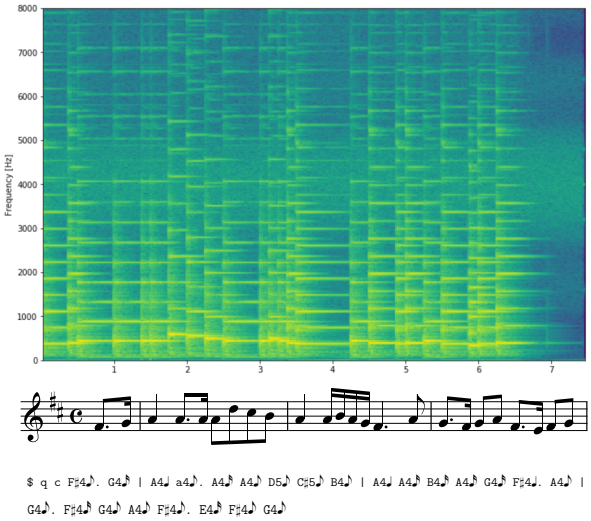


Figure 2: Example of a magnitude spectrogram (x) synthesized from a RISM incipit (center). The symbolic encoding representation used for the CRNN (y) is shown below, where the character '$' is the G2 clef, 'q' is the key signature DM, the symbol 'c' is used to encode 4/4 and ' | ' represents the barline. Similarly to speech recognition, words are separated by blank spaces.

Formally, let $\mathcal{X} = \{(x_1, y_1), (x_2, y_2), ...\}$ be our end-to-end application domain, where x_i is an audio recording represented by its magnitude spectrogram, and y_i denotes its corresponding ground-truth sequence from a fixed alphabet set Σ.

The problem of AMT can be reformulated as retrieving the most likely sequence of symbols $\hat{y}$ given an input spectrogram x. That is:

$$\hat{y} = \arg\max_{y \in \Sigma^*} P(y|x) \quad (1)$$

We formulate this statistical framework by means of Recurrent Neural Networks (RNN), as they allow handling sequences [12]. Ultimately, therefore, the RNN will be responsible of producing the sequence of musical symbols that fulfills Eq. 1. Nevertheless, on top of it, we add a Convolutional Neural Network (CNN), which learns how to process the input signal to represent it in a meaningful way for the task at issue [36]. Since both types of networks consist of feed-forward operations, the training stage can be carried out jointly by simply connecting the output of the last layer of the CNN with the input of the first layer of the RNN, which leads to a Convolutional Recurrent Neural Network (CRNN). A similar topology was previously applied to drum transcription in [33], although not in an end-to-end fashion.

Our work is conducted over a supervised learning scenario. Therefore, it is assumed that we can make use of a set $\mathcal{T} \subset \mathcal{X}$ with which to train the model. Initially, the traditional training mechanism for a CRNN needs to be provided with the expected output for each frame of the input. As introduced above, for each recording the training set only contains its corresponding sequence of expected

symbols, without any kind of explicit information about their location within the input signal. This scenario can be solved by means of the so-called Connectionist Temporal Classification (CTC) loss function [13].

Given an input x, CTC provides a means to optimize the CRNN parameters in order to directly output its correct sequence y. In other works, CTC directly optimizes $P(y|x)$. Since the ground-truth is not aligned at the frame level, that is, it is unknown the alignment between the frames of the recurrent part and the output symbols, CTC integrates over all possible alignments. It only considers monotonic alignments (left-to-right constraint), which is a valid assumption in our task.

Although optimizing the aforementioned probability is computationally expensive, CTC performs a local optimization using an Expectation-Maximization algorithm similar to that used for training Hidden Markov Models [24]. However, given that CTC integrates over all possible alignments, its main limitation is that the cost of the optimization procedure grows rapidly with the length of the sequences.

Note that CTC is used only for training. At the inference stage, the CRNN still predicts a symbol for each frame of the recurrent block. To indicate a separation between symbols, or to handle those frames in which there is no symbol, CTC considers an additional symbol in the alphabet that indicates this situation (*blank* symbol).

3.1 Implementation details

Finding the best instantiation of a CRNN for the case of AMT is out of the scope of this work, but we are inspired by the *Deep Speech 2* [8] topology, which was especially designed for the task of Automatic Speech Recognition (ASR). Although ASR and AMT are different tasks they are related, and so the use of this architecture allows us to obtain valuable results without having to make an exhaustive search of the best neural topology.

Nonetheless, we made small modifications to the original architecture in order to adjust its behavior to AMT. The specification of our neural topology is detailed in Table 2. It consists of 2 convolutional layers and 3 recurrent layers. Convolutional layers are composed of convolutional filters followed by Batch Normalization [16], and the non-linear hard hyperbolic tangent (HardTanh) activation function [14]. Furthermore, bi-directional recurrent layers are configured as Gated Recurrent Units (GRU) [7], with Batch Normalization as well. On top of the last recurrent output, a fully-connected layer is placed with as many neurons as symbols of the vocabulary (plus 1, because of the *blank* symbol). The use of the *softmax* activation allows us to interpret the output of this last layer as a posterior probability over the vocabulary [6].

The training stage is carried out by providing pairs of spectrograms with their corresponding unaligned sequence of musical symbols. The optimization procedure follows stochastic gradient descent (SGD) [5] with Nesterov momentum of 0.9, gradient L2 Norm clipping of 400, and a mini-batch size of 20 samples, which modifies the network weights to minimize the CTC loss function through back-propagation. The learning rate was initially set to 0.0003, but it was annealed by a factor of 1.1 after each epoch to favor convergence. The model was trained during 20 epochs, fixing the weights according to the best result over the validation set.

Input($1024 \times T$)
Convolutional block
Conv($32, 41 \times 11, 2 \times 2$), BatchNorm(), HardTanh()
Conv($32, 21 \times 11, 2 \times 1$), BatchNorm(), HardTanh()
Recurrent block
B-GRU(1024), BatchNorm()
B-GRU(1024), BatchNorm()
B-GRU(1024), BatchNorm()
Dense($\|\Sigma\| + 1$), Softmax()

Table 2: Instantiation of the CRNN used in this work for audio-to-score AMT, consisting of 2 convolutional layers and 3 recurrent layers. Notation: Input($h \times w$) means an input spectrogram of height h and width w; Conv($n, k_h \times k_w, s_h \times s_w$) denotes a convolution operator of n filters, kernel size of $k_h \times k_w$, and stride of $s_h \times s_w$; BatchNorm() denotes a batch normalization procedure; HardTanh() represents the *hard hyperbolic tangent* activation; B-GRU(n) means a bi-directional Gated Recurrent Units of n neurons; $Dense(n)$ denotes a fully-connected layer of n neurons; and $Softmax()$ represents the *softmax* activation function. Σ denotes the character-wise alphabet considered.

Once the CRNN is trained with the previous procedure, it can be used to output a discrete symbol sequence from a given spectrogram. The model yields character-level predictions in each frame. In order to provide an actual symbol sequence, it is necessary to both collapse repeating characters and discarding *blank* characters. Since there could be several frame-level sequences that result in the same sequence of musical symbols, the final decoding is conducted by a beam search procedure [37], with a beam width set to 10.

4. EXPERIMENTS

4.1 Setup

The proposed framework is evaluated using the corpus described in Section 2.1.

Experiments are performed dividing the available data into three independent partitions: $49,980$ samples (118.03 hours) for training, $10,710$ samples (25.34 hours) for validation, and $10,710$ samples (25.36 hours) for the test set, which is used to evaluate the actual performance.

Given the differences with existing AMT approaches, our results are not directly comparable with any previous work. Likewise, there are no standard evaluation metrics with which to evaluate this framework.

Here, we propose a series of metrics especially considered for evaluating the presented approach. In particular,

we are inspired by other tasks, like ASR or Optical Character Recognition (OCR), that are also formulated expecting a sequence of symbols as output. Analogously to these tasks, we also assume that the output consists of individual *characters* (pitches, durations, alterations, ...) that build complete *words* (such as notes). Therefore, the performance can be evaluated in terms of Character Error Rate (CER) and Word Error Rate (WER). These metrics are defined as the number of elementary editing operations (insertion, deletion, or substitution) to convert the hypotheses of the system into the ground-truth sequences, at the character and word level, respectively. They compute this cost in a normalized way according to the length of the ground-truth sequences. Even assuming that these metrics are not optimal for the task of AMT, we hope that they allow us to validate the approach and draw reasonable conclusions from our experimental results.

In order to get some baseline results that can be compared to other works, we also applied the evaluation metric used in [19] for piano-roll alignment tasks. The total number of notes in the ground truth is denoted by N_{GT}, that of estimated notes by N_{est}. The number of notes with pitch errors is denoted by N_p, that of extra notes by N_e, and that of missing notes by N_m. The number of matched notes is defined as $N_{match} = N_{GT} - N_m = N_{est} - N_e$. Then we define the pitch error rate as $E_p = N_p/N_{GT}$, extra note rate as $E_e = N_e/N_{est}$, and missing note rate as $E_m = N_m/N_{GT}$. Onset/offsets errors are also reported in [19]. As we are dealing with note durations instead of onsets/offsets, we include an alternative error metric E_d which is calculated similarly to the pitch error E_p but using note duration errors, denoted by N_d. Thus, we define the duration error rate as $E_d = N_d/N_{GT}$.

4.2 Results

Figure 3 shows the evolution of the errors during the training process. As can be seen, the convergence is fast and the best results on the validation set are obtained at epoch 18, reporting a CER of 5.53 and a WER of 15.98. In the test set, a CER of 5.36 and a WER of 15.67 are obtained. These results are very similar to those from the validation set, thus proving that there is no over-fitting and the model generalizes well.

After an in-depth analysis of the test set transcriptions obtained, we observed that the majority of errors are due to wrong time signatures, barline locations, and clefs. This result was expected in our prior analysis, as even for a human it would be difficult to identify them based on the short audio excerpts we provide to our model (the average number of music measures of the audio excerpts is 4.4). Furthermore, there are some time signatures that contain the same number of notes per measure and therefore they require more musical context to identify them correctly (e.g. 4/4 and 2/2 time signatures), as shown in Figure 4. In other cases, one of these specific errors causes the appearance of many others, as seem to happen with the time signature in the example of Figure 5. In order to address these ambiguities, normalization techniques could be employed (for instance, changing all 2/2 by their equivalent notation in 4/4).

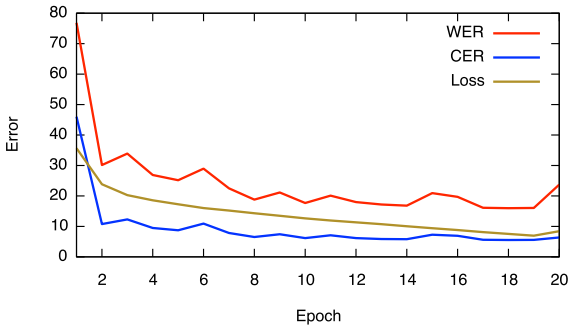
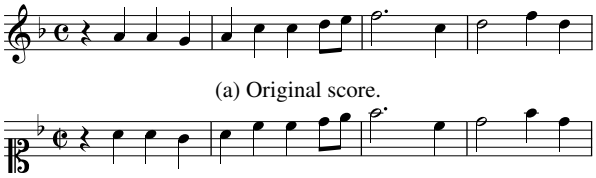


Figure 3: Evolution curves of the CTC loss, CER, and WER over the validation set with respect to the training epoch. The lowest WER (15.98) and CER (5.53) figures are obtained at epoch 18.



(a) Original score.

(b) Transcribed score.

Figure 4: Example of transcription performance. Note that the two mistakes made (clef and time signature) belong to music notation ambiguities.

In spite of all these difficulties, some samples are perfectly recognized, as the one depicted in Figure 6.

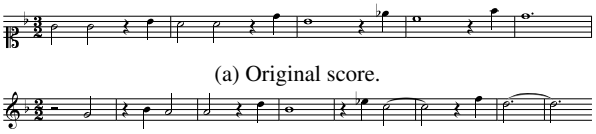
We provide the results of the evaluation metric proposed in [19] for estimated notes, and for estimated notes and rests combined (in this case, E_p does not change). As can be seen in Table 3, the error rates are quite low compared to [19], but this is due to the fact that our audio files are monophonic and synthesized. In addition, most transcription errors are due to wrong estimations of time signatures, subsequently yielding wrong barline locations as previously explained.

5. CONCLUSIONS

In this work, we propose a new formulation of AMT in the form of an audio-to-score task. In summary, the advantages of this formulation over piano-roll estimation are: 1) it is not required to have a frame-by-frame annotation aligned with the audio, therefore potentially more data

Table 3: Note pitch error rate (E_p), missing symbol rate (E_m), extra symbol rate (E_e) and symbol duration error rate (E_d) considering only notes and notes plus rests.

	E_p	E_m	E_e	E_d
Notes	0.99%	2.63%	1.81%	0.71%
Notes+Rests	0.99%	4.94%	2.51%	1.23%



(a) Original score.

(b) Transcribed score.

Figure 5: Example of a transcription with several mistakes. Here, the unusual time signature 3/2 (wrongly detected) propagates the errors to the notes.

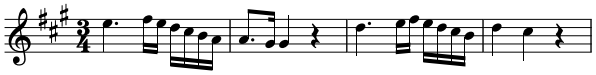


Figure 6: Example of a correctly transcribed score.

could be acquired for training; 2) the obtained outputs are musically meaningful; 3) the frame-by-frame annotation ambiguities are avoided, although on the other hand there are music notation ambiguities to deal with; 4) the task is addressed holistically instead of using a pipeline of individual processes, avoiding the cascading of errors when they are combined sequentially, and 5) musical models are implicitly inferred as it occurs with language models in speech recognition.

We validated the proposed framework using a CRNN with a CTC loss function trained on RISM incipits, correctly predicting around 84% of symbols for monophonic scores synthesized with a piano sound at different tempos. It is important to note that some symbols such as barlines, rests, ties, time signatures or key signatures were no explicitly present in spectrograms but they were correctly inferred from the context.

A qualitative analysis of the performance reported that many errors occurred because of music notation ambiguities. Although they decreased the WER and CER figures, "wrong" outputs are musically correct and equivalent to the ground-truth scores in most cases.

As a future work, we are planning first to extend it for polyphonic sources, and also to perform instrument recognition. In order to deal with polyphony, a chord could be considered as a "word" containing "syllabus" (the individual notes), for example: C4𝅗𝅥E4𝅗𝅥G4𝅗𝅥. An additional symbol could be added to indicate the instrument (for example, PC4♩ could represent a quarter note of pitch C4 played on Piano).

As pointed out in [18], previous experiments on deep neural networks dealing with framewise multiple pitch detection showed that unseen combinations are hard to detect. A partial solution to this problem might involve a modification of the loss function for the network to disentangle individual notes explicitly and learn to decompose a (nonlinear) mixture of signals into its constituent parts. We believe that, unlike what happens in this framewise detection, CTC loss may be able to break the observed glass-ceiling, given that ASR methods using this architecture are capable of generalizing to detect unseen words from its constituent (character) elements. Nonetheless, additional experiments on AMT should confirm this hypothesis.

Synthesized scores were used to perform the experiments, although ideally real data should be evaluated. For this, we are planning to use datasets such as Lakh [25], which contains audio files with their corresponding MIDIs. Given the computational cost of CTC, the proposed framework needs to use short segments. Therefore, it is necessary to have aligned barlines to split both the audio and the corresponding score ground truth into smaller pieces. This could be done using a score following method [9,22]. This preprocessing could introduce some errors due to wrong alignments, but there is a more suitable alternative: to train the CRNN using full scores along with their complete real audio files, which is the ultimate goal of the proposed framework. This is possible and could be done by considering the recently proposed *online* CTC [15] function, which efficiently adapts to any sequence length.

Another obvious future work is to find a more adequate network architecture and evaluate alternative hyperparameters to increase the accuracy. CNN and RNN topologies evaluated in previous AMT works [17, 27] should be investigated for this task.

In conclusion, in this work we show that it is feasible to perform end-to-end transcription from monophonic audio files to scores. We are fully aware that experiments were made in a very controlled and simplified environment and there is still much work to do in order to perform a complete transcription. But we believe that the proposed framework opens a new exciting research area given the huge amount of data that could potentially be used for training, and its practical utility for musicians who could obtain directly a score from audio.

6. ACKNOWLEDGEMENT

This work was supported by the University of Alicante through grant GRE-16-04 and its University Institute for Computing Research (IUII), and by the Spanish Ministerio de Economía, Industria y Competitividad through HispaMus project (TIN2017-86576-R) and Juan de la Cierva - Formación grant (Ref. FJCI-2016-27873).

7. REFERENCES

[1] E. Benetos and S. Dixon. A shift-invariant latent variable model for automatic music transcription. *Computer Music Journal*, 36(4):81–94, 2012.

[2] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri. Automatic Music Transcription: Challenges and Future Directions. *Journal of Intelligent Information Systems*, 41(3):407–434, 2013.

[3] E. Benetos and T. Weyde. An efficient temporally-constrained probabilistic model for multiple-instrument music transcription. In *16th International Conference on Music Information Retrieval*, pages 701–707, 2015.

[4] S. Böck and M. Schedl. Polyphonic piano note transcription with recurrent neural networks. In *IEEE In-*

ternational Conference on Acoustics, Speech and Signal Processing, pages 121–124, 2012.

[5] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[6] H. Bourlard and C. Wellekens. Links Between Markov Models and Multilayer Perceptrons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(11):1167–1178, 1990.

[7] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In *Proceedings of the Eighth Workshop on Syntax, Semantic sand Structure in Statistical Translation*, pages 103–111, 2014.

[8] D. Amodei et al. Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin. In *33nd International Conference on Machine Learning*, pages 173–182, 2016.

[9] R. B. Dannenberg and C. Raphael. Music score alignment and computer accompaniment. *Communications of the ACM*, 49(8):38–43, 2006.

[10] V. Emiya, R. Badeau, and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *IEEE Trans. on Audio, Speech, and Language Processing*, 18(6):1643–1654, 2010.

[11] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC Music Database: Popular, Classical and Jazz Music Databases. In *3rd International Conference on Music Information Retrieval*, pages 287–288, 2002.

[12] A. Graves. *Supervised Sequence Labelling with recurrent neural networks*. PhD thesis, Technical University Munich, 2008.

[13] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *23rd International Conference on Machine Learning*, International Conference on Machine Learning, pages 369–376. ACM, 2006.

[14] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized neural networks. In *Advances in Neural Information Processing Systems 29*, pages 4107–4115, 2016.

[15] K. Hwang and W. Sung. Online Sequence Training of Recurrent Neural Networks with Connectionist Temporal Classification. *CoRR*, abs/1511.06841, 2015.

[16] S. Ioffe and C. Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July*, pages 448–456, 2015.

[17] R. Kelz, M. Dorfer, F. Korzeniowski, S. Böck, A. Arzt, and G. Widmer. On the Potential of Simple Framewise Approaches to Piano Transcription. In *17th International Conference on Music Information Retrieval*, 2016.

[18] R. Kelz and G. Widmer. An experimental analysis of the entanglement problem in neural-network-based music transcription systems. *arXiv preprint arXiv:1702.00025*, 2017.

[19] E. Nakamura, E. Benetos, K. Yoshii, and S. Dixon. Towards complete polyphonic music transcription: Integrating multi-pitch detection and rhythm quantization. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2018.

[20] E. Nakamura, K. Yoshii, and S. Dixon. Note Value Recognition for Piano Transcription Using Markov Random Fields. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25:1542–1554, 2017.

[21] E. Nakamura, K. Yoshii, and S. Sagayama. Rhythm Transcription of Polyphonic Piano Music Based on Merged-Output HMM for Multiple Voices. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25:794–806, 2017.

[22] N. Orio, S. Lemouton, and D. Schwarz. Score following: State of the art and new developments. In *Proc. of the 2003 Conference on New interfaces for Musical Expression*, pages 36–41. National University of Singapore, 2003.

[23] A. Pertusa and J. M. Iñesta. Efficient methods for joint estimation of multiple fundamental frequencies in music signals. *EURASIP Journal on Advances in Signal Processing*, 2012(1):27, Feb 2012.

[24] L. Rabiner and B.-H. Juang. *Fundamentals of speech recognition*. Prentice hall, 1993.

[25] C. Raffel. *Learning-Based Methods for Comparing Sequences, with Applications to Audio-to-MIDI Alignment and Matching*. PhD thesis, Columbia University, 2016.

[26] RISM. Répertoire International des Sources Musicales, 2017.

[27] S. Sigtia, E. Benetos, and S. Dixon. An End-to-end Neural Network for polyphonic piano music transcription. *IEEE Transactions on Audio, Speech and Language Processing*, 24(5):927–939, 2016.

[28] P. Smaragdis and J. C. Brown. Non-negative Matrix Factorization for Polyphonic Music Transcription. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 177–180, 2003.

[29] J. Thickstun, Z. Harchaoui, and S. Kakade. Learning features of music from scratch. In *International Conference on Learning Representations (ICLR)*, 2017.

[30] J. Thickstun, Z. Harchaoui, D. Foster, and S. M. Kakade. Invariances and data augmentation for supervised music transcription. *arXiv preprint arXiv:1711.04845*, 2017.

[31] R. Typke, M. Den Hoed, J. De Nooijer, F. Wiering, and R. C. Veltkamp. A ground truth for half a million musical incipits. *Journal of Digital Information Management*, pages 34–39, 2005.

[32] J. J. Valero-Mas, E. Benetos, and J. M. Iñesta. A supervised classification approach for note tracking in polyphonic piano transcription. *Journal of New Music Research*, pages 1–15, 2018.

[33] R. Vogl, M. Dorfer, G. Widmer, and P. Knees. Drum transcription via joint beat and drum modeling using convolutional recurrent neural networks. In *18th International Conference on Music Information Retrieval*, pages 150–157, 2017.

[34] Q. Wang, R. Zhou, and Y. Yan. Polyphonic Piano Transcription with a Note-Based Music Language Model. *Applied Sciences*, 8(3), 2018.

[35] C. Yeh, A. Robel, and X. Rodet. Multiple fundamental frequency estimation of polyphonic music signals. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3. IEEE, 2005.

[36] M. D. Zeiler and R. Fergus. Visualizing and Understanding Convolutional Networks. In *13th European Conference on Computer Vision*, pages 818–833, September 2014.

[37] T. Zenkel, R. Sanabria, F. Metze, J. Niehues, M. Sperber, S. Stüker, and A. Waibel. Comparison of decoding strategies for CTC acoustic models. In *18th Annual Conference of the International Speech Communication Association*, pages 513–517, 2017.