MUSIC TRANSCRIPTION WITH CONVOLUTIONAL SEQUENCE-TO-SEQUENCE MODELS

Karen Ullrich
Univerity of Amsterdam
k.ullrich@uva.nl

Eelco van der Wel
University of Amsterdam
author1@gmail.com

ABSTRACT

Automatic Music Transcription (AMT) is a fundamental problem in Music Information Retrieval (MIR). The challenge is to translate an audio sequence to a symbolic representation of music. Recently, convolutional neural networks (CNNs) have been successfully applied to the task by translating frames of audio [44, 46]. However, those models can by their nature not model temporal relations and long time dependencies. Furthermore, it is extremely labor intense to get annotations for supervised learning in this setting. We propose a model that overcomes all these problems. The convolutional sequence to sequence (Cseq2seq) model applies a CNN to learn a low dimensional representation of audio frames and a sequential model to translate these learned features to a symbolic representation directly. Our approach has three advantages over other methods: (i) extracting audio frame representations and learning the sequential model is jointly trained end-to-end, (ii) the recurrent model can capture temporal features in musical pieces in order to improve transcription, and (iii) our model learns from entire sequences as opposed to temporally accurately annotated onsets and offsets for each note thus making it possible to train on large already existing corpora of music. For the purpose of testing our method we created our own dataset of 17K monophonic songs and respective MusicXML files. Initial experiments proof the validity of our approach.

1. INTRODUCTION

Automatic music transcription (AMT) is a challenging problem for humans and machines. The task at hand is to find a mapping $f : x \to y$ that translates an audio sequence x to a symbolic representation of that sequence y. The difficulty is no surprise because in the most general case, polyphonic AMT, separating the sources of sound alone, e.g. one key stroke on a piano from another, is already a highly under determined problem. Thus, any sufficient model needs to learn strong priors over the audio sequences it receives as input in order to perform well. Even if a model does learn these priors sufficiently it can not be guaranteed that the task at hand is well defined. For example, the harmonics of two distinct notes of possibly different instruments can have complex interactions. Furthermore, noise or recording technique may limit the prior assumptions that can be made. The space of expected events is huge as well: Musical pieces come in a great range of styles, forms, instrumentations and even playing techniques. However, the fact that machine performance lags behind human performance [30] is a strong indicator for the room of improvement for these models. Thus it is reasonable to believe that a good model needs to have the capacity to learn priors over musical sequences for example the (probabilistic) rules western music is following with respect to tempo, harmony or timbre. It has been the subject of several studies to work in this prior knowledge without restricting the flexibility of a model too much. One of the key limitations for state-of-the-art models is the lack of annotated data of sufficient size and diversity.

Notice that ATM falls in the regime of perceptional problems. Within this field, deep learning has been contributing remarkable improvements on several tasks, initially mainly in computer vision (CV) later also in several other domains such as natural language processing. There is reason to believe that Music Information Retrial (MIR) tasks are more challenging than CV tasks for example due to the ambiguity of annotation even to human perceivers. However, several pioneering studies in deep learning have shown significant improvement in various MIR challenges such as onset and structural boundary detection [43, 49], piano transcription [44], genre classification [18, 50] or sound generation [5] to just name a few. This gives reason to believe in the power of such techniques.

Within the deep learning domain there are two popular models: the *Convolutional Neural Network* (CNN) and the *Recurrent Neural Network* (RNN). CNNs had enormous success in classification tasks such as image recognition. They seem to break the curse of dimensionality by learning locally low dimensional representations of their input. By stacking many of these modules in a hierarchical manner, a global understanding of the input as a whole can be achieved (for illustration see [55]). The other popular model, RNNs, is applied to sequence modeling. These models can be understood as a generalized version of hidden Markov models. They are used for language modeling such as text generation or language translation. For the latter example *sequence to sequence* (seq2seq) mod-

Attribution: Karen Ullrich, Eelco van der Wel. “Music Transcription with Convolutional Sequence-to-Sequence models”, 18th International Society for Music Information Retrieval Conference, Suzhou, China, 2017.

els, a subclass of RNNs, are well known. Here a sequence of, for example, English is feed into a neural network to output a hidden state that contains all the information of the sequence. This hidden state is then fed into another model that generates the sequence with the same meaning but in a different language. This model is superior to others because it does not translate ”word by word” thus can for example deal with different grammatical structure from source and target language such as word order.

In music translation tasks such as optical music recognition or music transcription, we are often faced with the same problems. Dependencies need to be ”kept in mind” and later be remembered at a different place in the sequence, for example when translating sheet music to piano roll representation one needs the model to have the capacity to remember the key signature. This is why we propose to apply the seq2seq model to music translation tasks such as ATM. However, since audio streams are very high dimensional we propose to preprocess the data by first computing a spectral representation of the audio input and consequently applying a CNN for dimensionality reduction before its fed to the seq2seq model. The CNN and seq2seq model can be trained jointly and end-to-end and thus benefit one another. Similar to the original proposed seq2seq models that train on entire sentences of source and target language rather than words by word translations the annotation effort to train these models is minimal since large corpora of suitable training data already exist.

We relate to work of others in the next section. In section 3, we will outline how we create a simple dataset to test out method. In section 4, we will describe the proposed method in detail followed by inital experiments and and extensive discussion of model criticism and future work in section 5 and 6.

2. RELATED WORK

AMT systems are usually complex pipelines that perform the following subtasks: pitch detection, onset/offset detection, instrument identification, rhythm parsing, identification of dynamics and expressions and typesetting. Depending on the context, an AMT system for western music does either percussive instrument transcription or multi-pitch analysis. The latter one knows two main approaches: analysis on the frame and on the note level. Note level analysis identifies notes by onset and offset detection. The identified notes are consequently classified [13, 22, 31, 37, 38]. However, a bottleneck of these methods is the accuracy of the onset detection method. Another unsupervised method is clustering harmonic temporal structures [28]. Alternatively, the audio signal can be modeled as a hidden Markov model that transitions between notes [42]. The same approach can also be used to model the signal as a mixture of note spectra [14,24].

In contrast to note level predictions, frame level approaches subdivide the audio stream into temporally equivalent frames. Multi-pitch prediction is performed on each frame independently. The predictions are usually made in the time or frequency domain. More specifically for time domain models, there are biologically inspired models [31,34,48] and probabilistic models [12,16,52] . Most recent algorithms perform in the frequency domain. Here for each frame a spectral representation such as the ERB filterbank, STFT or CQT spectrum, is computed.

The central idea of frequency domain approaches is that the given spectrum is a linear superposition of several pitches' spectra. [29] and [2] subtract detected pitches from the signal spectrum and iteratively proceed until the spectral frame is explained sufficiently. A range of methods focused on the most dominant peaks in the spectrum [21, 23, 39, 40, 53]. The most sophisticated methods in this area model the full spectrum either as a mixture model [26, 28, 45, 54], compute the eigen-spectra [1, 3, 6–9, 17, 25, 27, 37] or perform classification on the frames [10,31,36,41,44].

To our knowledge, our method is the first proposed that does not a ”word-by-word”/ ”frame-by-frame” translation but rather gathers the information of a sequence and translates it as a whole to a symbolic representation. The advantage of that model is that it can learn relevant priors on the signal since it does not consider frames independently. These priors could learn concepts from data that map our understanding of musicology and are thus expected to be superior to other methods. Furthermore, while still be considered supervised models seq2seq models have little labeling work.

3. DATASET

In the context of this project, we collaborated with the MuseScore sheet music archive [35], a public database of user-generated scores. The archive hosts scores from various genres, clefs, key and time signatures. The data is originally stored in MusicXML format. It serves as basis for generating audio input files and corresponding ground truth. From the entire data base we extracted 17K monophonic scores and if available BPM rates. These were randomly assigned to training (60%), validation (25%) and test (25%) set. A list of the specific files in use can be found online [1] .

More precisely, we generated data points in the following fashion, a MusicXML score is split such that one fragment contains maximally 4 bars. We do not generate the entire sequence but splits to guarantee approximately equal length of sound files. This is not a general limitation of this method but does allow us to train the model faster. For the audio creation we chose the BPM rate as provided by the particular MusicXML file. If no rate is available we uniform randomly sample a rate in $[80, 180]$. We generated audio sequences with the timidity synthesizer [2] using the fluid general midi sound font for piano and stored them as mp3. Labels are threefold: they provide information about the pitch and the duration of a note in quarter notes and seconds. Pitch values are represented as categorical data in their Western notation. Note that one pitch class is the rest,

[1] https://github.com/anonymous

[2] https://github.com/m13253/timidity

notated as r. Alternatively, they can be represented as continuous labels according to their frequency. In this setting, rests can be a problem since there is no frequency connect to silence. We choose a high negative value to represent a rest. Duration values are given in quarter length or seconds. The easiest encoding is to choose continuous labels. However, in the case of quarter length labels we can also rely on categorical data. Note that we can use the given data as categorical or continuous labels. The stop token is chosen to be $(r, 0)$. Finally, we restrict the maximum number of events in one sequence to 48. Note that for training we need one pair representing pitch and duration.

4. METHOD

We introduce the *Convolutional sequence-to-sequence* (Cseq2seq) model. We represent an audio stream in the frequency domain. Consequently, the stream is fragmented into a series of overlapping spectrogram excerpts. Each fragment is fed into a CNN for dimensionality reduction. The reduced representation serves as input to an RNN model that encodes the information in the sequence. Another RNN model serves as decoder to generate the output sequence.

4.1 Preprocessing

We generate a spectral representation of the input sequence. For each audio sequence, we compute a magnitude spectrogram with a window size of 46.6 ms (2048 samples at 44.1 kHz) and 50% overlap. We apply an equivalent rectangular filterbank of 200 triangular filters from 27.5 Hz to 16 kHz. The entire preprocessing pipeline was realized with Essentia [11]. Alternatively, we provide constant Q transformed sequences. With 16 bins per octave and 7 octaves resulting in 112 bins. This feature was computed with librosa [33].

4.2 Convolutional Sequence-to-Sequence model

The spectral representation of a musical piece with index i is split into a series of spectrogram excerpts $\mathbf{X}^{(i)} = \{\mathbf{x}_t^{(i)}\}_{t=1}^{T}$ of T frames with 25% overlap. We propose to couple a CNN with a seq2seq model and train the combination jointly. The CNN represents the automated feature extractor aka for each except $\mathbf{x}_t^{(i)}$ it extracts meaningful information from the spectral representation and compresses it. This low dimensional representation $\tilde{\mathbf{x}}_t^{(i)}$ is than the input to the recurrent model that decodes the sequence $\tilde{\mathbf{X}}^{(i)} = \{\tilde{\mathbf{x}}_t^{(i)}\}_{t=1}^{T}$ to a hidden state $\mathbf{H}$ that ideally contains all information of the sequence much like a sufficient statistics. Consequently the information is being ”translated” to the symbolic space with another RNN, the decoder, to the output sequence $\mathbf{Y}^{(i)} = \{\mathbf{y}_t^{(i)}\}_{t=1}^{T}$. More specifically, we choose LSTM models as our RNNs due to their ability to learn long term dependencies better . The whole model is illustrated in Figure 1.

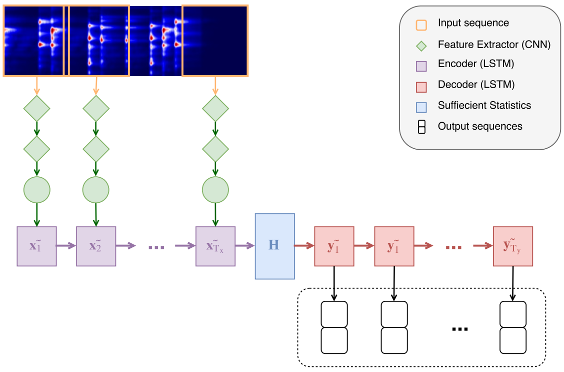

Figure 1. *Convolutional sequence to sequence model*: A spectral audio representation of N frames is fed into a CNN (green). The sequence of consequent representations is than submitted to an encoder LSTM (purple) that puts out a hidden state representing the input sequence. This hidden state is finally used to generate the output sequence via the decoder LSTM (red). Note that equal color represents units within the system, each unit shares parameter. However, the entire system is trained jointly.

4.3 Objective

The model may be trained with categorical data, i.e., pitch classes and duration in quarter notes or with continuous labels with frequencies and durations in seconds . The former method would naturally be trained with the categorical cross entropy loss, whereas the latter would be trained with mean squared error.

4.4 Training

The input to the Cseq2seq model are batches of series of spectrogram excerpts of T frames. Note that the spectrograms are padded with zeros so that all sequences in a batch are equally long. Each frame is passed through the CNN. The representation is than passed on to the LSTM-encoder, which computes a hidden state. Based on this hidden state, an LSTM-decoder generates an output sequence to match the labels given as (pitch, duration) which is padded as well with stop tokens.

We train the system with sequence mini batches of size 64. The objective is the categorical cross entropy or mean squared error depending on the labeling we choose (see section 3). We use the Adam optimizer with a notability small learning rate of 8×10^{-4}. We apply 15% dropout to the inputs and 25% in the convolutional network. We train for 50 epochs. Training a single Cseq2seq on an Nvidia GTX Titan X graphics card took 30h to 60h. Note that the method is almost trainable end-to-end, however, the spectral representation can be seen as hand engineered feature.

5. EXPERIMENTS

We present initial experiments with the Cseq2seq model on the Musescore dataset. While the novelty of our approach does not allow us to compare with current methods directly, we determine the best modeling choices and examine how sensitive the model is to augmentation.

5.1 Evaluation

Because our approach is so different from other methods most of the common evaluation measures can not be applied directly. Our method can neither be specified as frame nor as note based system. Since it is translating an audio stream directly into symbol representation. We will instead report a pitch and a duration accuracy for categorical data. If the system puts out a correct pitch and duration this will be a successful note detection, which will also be reported. In the case of continuous output, following the authors of [20] and [28], duration is counted as correct if it is within $\pm 50ms$ of the ground truth. The pitch will be rounded to its next class. Note that our system can by definition not produce any false positives or negatives, all output is regarded as a prediction.

5.2 Initial experiment

We perform initial experiments to determine successful models. First, we test one of the most important modeling choices: weather to predict categorical or continuous outputs. Categorical durations will be presented in quarter notes, continuous ones in seconds. Obviously, durations and pitches are (almost) linearly related in the proposed representations thus we expect continuous output to perform well. On the other hand, neural networks are known to perform best on categorical data. To our surprise, categorical prediction networks outperformed continuous ones strongly even though they had to guess the note duration with different BPM rates. Thus all future experiments will be carried out on categorical output networks.

We furthermore tested the effect of log-scaling and normalizing the spectral representations. For the CQT representations, we find those measures to not perform better than the raw input. ERB bands on the other hand benefit from normalization.

5.3 Feature extractor

Computing an optimal representation for the sequential model is an important part in the translation process. Our method consists of ”hand engineered” features, the spectral representations, and learned features, the CNN part of the model. We experiment with different choices for either of the two components. We vary the spectral representation between ERB bands and CQT features and experiment with 3 different network architectures. We call them A, B and C. The motivation for these choices is the following conflict. Introducing convolutional layers and sub-sampling operations introduces translation equivariance and invariance, respectively, a feature that we might not desire in the frequency domain. Thus we test a fully connected architecture in model A, an architecture with strided convolutions only in the time domain and finally a model with both. The precise specification can be found in table 1. For all experiments we set the following hyperparameters, for the activation functions we choose relu units and the LSTM has 256 units. Furthermore, we use dropout with a probability of 25% and a window size of 3.8s with 50% overlap.

	A	B	C
layer 1	1024	16 × [5,3], [1,2]	16× [5,3], [2,2]
layer 2	512	32× [5,3], [1,2]	32 × [5,3], [2,2]
layer 3	256	32× [5,3], [1,2]	32 × [5,3], [2,2]
layer 4		256	256

Table 1. Network architectures for feature extractor models. Fully connected layers are simply identified by number of units, convolutional layers are specified # of filters × kernel size, striding

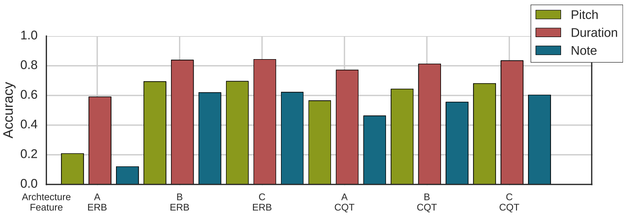

Figure 2. We test our model on various feature extractors. We report pitch, duration and note accuracy.

The results of this experiment are visualized in figure 2. We see that there is barely a difference between the two spectral representations. However, the choice of model does seem to matter. Somewhat counterintuitive model C works best. We suspect that to be related to the importance of dimensionality reduction.

5.4 RNN capacity

After having established good choices to extract features from incoming frames, we turn to an optimal recurrent model. There are two quantities that need to be chosen carefully. One is the information that needs to be encoded by the feature extractor and one is the amount of information to be encoded by the recurrent model. These properties correspond to the window size and the amount of hidden units in encoder and decoder, respectively. Ideally, there is a balance between the work the feature extractor and the recurrent model need to accomplish. Too small sizes window sizes might be a problem for the RNN because it can not resolve long time dependencies. Too large sizes might be a problem because the CNN needs to store too much information in the features. In this experiment, we vary window sizes from 1.8s and 3.7s to 5.5s. Furthermore, we vary the number of hidden units in both LSTMs between 256 and 512. We fix the feature extractor to ERB bands and a CNN model with architecture C. We continue training with the same dropout rates as in the previous experiment. Again we train the model for 50 epochs with Adam. We present results on our validation set in figure 3.

We find the best model performance with large recurrent model capacity and a small window size. This finding

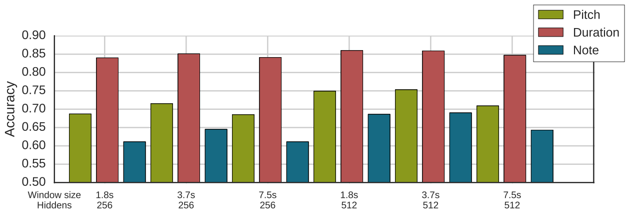

Figure 3. We optimize the proposed model's recurrent part on the validation set. We experiment with different variations of window size and recurrent model capacity. We report the pitch, duration and note accuracy.

Noise level	0.1 dB	1 dB
Test set without augmentation		
Pitch accuracy	0.732	0.418
Duration accuracy	0.853	0.600
Note accuracy	0.666	0.284
Test set with augmentation		
Pitch accuracy	0.732	0.725
Duration accuracy	0.852	0.844
Note accuracy	0.666	0.663

Table 2. Final results: We trained the best model architecture as determined earlier on a training set with augmentations with varying levels of noise. We tested the resulting model on the validation set and the tested with additional augmentations relating to the training augmentation.

is not surprising. It is expected that if we segment a sequence in many small pieces the RNNs need to have to resolve longer time dependencies. We clearly see that the performance drops significantly when we restrict the RNN capacity to 256 hidden units. In contrast to those results, the results for the larger context vary only little since RNN and CNN ”share the work” of encoding more evenly.

5.5 Data augmentation

In a final experiment, we determine if data augmentation does benefit the training. Data augmentation is a popular way to enrich artificial data such that it extrapolates to real wold data, for example, in scenarios where there is only artificial training data available. We apply pink noise on the audio sequences and report the accuracy of the validation data with and without this noise. We present results with small, moderate and large induced noise levels in table 5.5

We find low levels of pink noise to neither benefit nor detriment the performance of the network. Moderate noise does benefit the overall accuracy, whereas too much noise obfuscates the information in the data.

In our final experiment, we train the network with varying levels of pink noise by uniform randomly sampling its dB rate per training example in $[0, 1]$. We evaluate the performance of this experiment on the test set. For the non-augmented test set we achieve scores of 0.723, 0.847 and 0.654 for pitch, duration and total accuracy, respectively and 0.721, 0.845 and 0.650 for the augmented test dataset. Hence, we can train a single model that is robust to a wide range of noise present in the signal.

6. DISCUSSION AND FUTURE WORK

In this study we present a novel approach to ATM. Our solution is an important step towards an end-to-end trainable system. We combine the benefits of differentiable feature extractors such as CNNs with recurrent models that can pick up long time dependencies in data. We need both of these properties to tackle ATM successfully. More precisely, we propose the convolutional sequence-to-sequence model. We pass spectrogram excerpts through a CNN, the consequent representation is fed into a sequence-to-sequence model. Ideally, the model distributes the difficulty of this task to its components. The problem of relevant feature extraction is carried out by the CNN while the seq2seq model learns long time dependencies and data priors such as derived by musicology automatically from the data. Our model is preferable not just because the model can capture the complexity of the data well, but it is to our knowledge, the first method that does not rely on note level annotations but rather on sequence annotation, i.e., audio recordings and respective scores. Thus, we do not only propose a very flexible model but also one that can be trained with data that exists en mass already. There is no need of on- and offset annotations which is often considered as a bottleneck of ATM methods.

In experiments we determine the best modeling choices and we can show that the model is robust to synthetic recording noise. We achieve convincing results on monophonic scores. We are sure we could improve these results by additional information such as the BPM rate. In future efforts, we will extend this method to polyphonic scores. This however does require us to change the labeling scheme to a version that is closely related to the MIDI or piano roll format. We elaborate on the form of such a format in appendix 8.1.

However, the format is not the only challenge in order to extrapolate to multi-pitch prediction. Given the proposed multi-pitch labeling scheme, target sequences will be substantially longer thus our recurrent models will need more capacity, and further enroll longer sequences over time through which we need to back-propagate. This poses substantial computational challenges. To address the latter one its is recommendable to use a dynamic deep learning framework such as torch [15] or chainer [47]. To address the problem of longer time dependencies, we refer to work by [4], that address this problem with a so called attention mechanism.

Our main focus and challenge for future work, however, will be to replace the spectral representation and CNN by a fully differentiable feature extractor. Recently, there were promising results such as [19, 46, 51] but also biologically inspired models [32] that show that this goal is in reach. The former authors achieve astonishing results by interpret the CNN as a feature extractor and a recurrent model.

Finally, we want to test our approach on multi-pitch prediction and real world recordings in a competitive setting. For that we need to make approximations between the accuracy measures in use today and the method that we proposed.

We would like to point out that the proposed method is a very general approach to address music translation tasks. Another example that we can easily generalize our method to would be optical music recognition. But it would also be applicable over the limits of MIR to tasks such as handwriting recognition or video tagging.

7. REFERENCES

[1] S. A. Abdallah and M. D. Plumbley. Unsupervised analysis of polyphonic music by sparse coding. *IEEE Transactions on Neural Networks*, 17(1):179–196, January 2006.

[2] F. Argenti, P. Nesi, and G. Pantaleo. Automatic transcription of polyphonic music based on the constant-Q bispectral analysis. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(6):1610–1630, 2011.

[3] I. Ari, U. Simsekli, A.T. Cemgil, and L. Akarun. Large scale polyphonic music transcription using randomized matrix decompositions. In *Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, pages 2020–2024, 2012.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[5] Bhavik R Bakshi and George Stephanopoulos. Wavenet: A multiresolution, hierarchical neural network with localized learning. *AIChE Journal*, 39(1):57–81, 1993.

[6] M. Bay, A. F. Ehmann, J. W. beauchamp, P. Smaragdis, and J. S. Downie. Second fiddle is important too: pitch tracking individual voices in polyphonic music. In *International Symposium on Music Information Retrieval Conference*, pages 319–324, October 2012.

[7] E. Benetos, R. Badeau, T. Weyde, and G. Richard. Template adaptation for improving automatic music transcription. In *15th International Society for Music Information Retrieval Conference (ISMIR)*, pages 175–180, October 2014.

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

[9] N. Bertin, R. Badeau, and E. Vincent. Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):538–549, March 2010.

[10] S. Böck and M. Schedl. Polyphonic piano note transcription with recurrent neural networks. In *IEEE International Conference on Audio, Speech, and Signal Processing*, pages 121–124, March 2012.

[11] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, Perfecto Herrera, Oscar Mayor, Gerard Roma, Justin Salamon, José R Zapata, and Xavier Serra. Essentia: An audio analysis library for music information retrieval. In *ISMIR*, pages 493–498. Citeseer, 2013.

[12] A. T. Cemgil, H. J. Kappen, and D. Barber. A generative model for music transcription. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(2):679–694, March 2006.

[13] A. Cogliati and Z. Duan. Piano music transcription with fast convolutional sparse coding. In *Proc. IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015.

[14] Andrea Cogliati and Zhiyao Duan. Piano music transcription modeling note temporal evolution. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 429–433. IEEE, 2015.

[15] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011.

[16] Manuel Davy and SJ Godsill. Bayesian harmonic models for musical signal analysis. *Bayesian Statistics*, 7:105–124, 2003.

[17] A. Dessein, A. Cont, and G. Lemaitre. Real-time polyphonic music transcription with non-negative matrix factorization and beta-divergence. In *International Symposium on Music Information Retrieval Conference*, pages 489–494, August 2010.

[18] Sander Dieleman, Philémon Brakel, and Benjamin Schrauwen. Audio-based music classification with a pretrained convolutional network. In *12th International Society for Music Information Retrieval Conference (ISMIR-2011)*, pages 669–674. University of Miami, 2011.

[19] Sander Dieleman and Benjamin Schrauwen. End-to-end learning for music audio. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 6964–6968. IEEE, 2014.

[20] S. Dixon. On the computer recognition of solo piano music. In *2000 Australasian Computer Music Conference*, pages 31–37, July 2000.

[21] Z. Duan, B. Pardo, and C. Zhang. Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(8):2121–2133, November 2010.

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

[23] Valentin Emiya, Roland Badeau, and Bertrand David. Automatic transcription of piano music based on hmm tracking of jointly-estimated pitches. In *Signal Processing Conference, 2008 16th European*, pages 1–5. IEEE, 2008.

[24] S. Ewert, M.D. Plumbley, and M. Sandler. A dynamic programming variant of non-negative matrix deconvolution for the transcription of struck string instruments. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 569–573, 2015.

[25] B. Fuentes, R. Badeau, and G. Richard. Harmonic adaptive latent component analysis of audio and application to music transcription. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(9):1854–1866, September 2013.

[26] M. Goto. A real-time music-scene-description system: predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication*, 43:311–329, 2004.

[27] G. Grindlay and D. Ellis. Transcribing multi-instrument polyphonic music with hierarchical eigeninstruments. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1159–1169, October 2011.

[28] H. Kameoka, T. Nishimoto, and S. Sagayama. A multi-pitch analyzer based on harmonic temporal structured clustering. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(3):982–994, March 2007.

[29] A. Klapuri. Multiple fundamental frequency estimation based on harmonicity and spectral smoothness. *IEEE Transactions on Speech and Audio Processing*, 11(6):804–816, November 2003.

[30] Anssi Klapuri and Manuel Davy. *Signal processing methods for music transcription*. Springer Science & Business Media, 2007.

[31] M. Marolt. A connectionist approach to automatic transcription of polyphonic piano music. *IEEE Transactions on Multimedia*, 6(3):439–449, June 2004.

[32] J H McDermott and E P Simoncelli. Sound texture perception via statistics of peripheral auditory representations. In *34th midWinter Meeting, Assoc. for Research in Otolaryngology*, Baltimore, MD, Feb 29-23 2011.

[33] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, 2015.

[34] R. Meddis and M. J. Hewitt. Virtual pitch and phase sensitivity of a computer model of the auditory periphery. i: pitch identification. *Journal of the Acoustical Society of America*, 89:2866–2882, 1991.

[35] MuseScore.

[36] J. Nam, , J. Ngiam, H. Lee, and M. Slaney. A classification-based polyphonic piano transcription approach using learned feature representations. In *International Symposium on Music Information Retrieval Conference*, pages 175–180, October 2011.

[37] K. O'Hanlon, H. Nagano, and Mark Plumbley. Structured sparsity for automatic music transcription. In *IEEE International Conference on Audio, Speech, and Signal Processing*, pages 441–444, March 2012.

[38] Peter P. Grosche, B. Schuller, M. Müller, and G. Rigoll. Automatic transcription of recorded music. *Acta Acustica united with Acustica*, 98(2):199–215, March 2012.

[39] P.H. Peeling and S.J. Godsill. Multiple pitch estimation using non-homogeneous poisson processes. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1133–1143, October 2011.

[40] A. Pertusa and J. M. Iñesta. Multiple fundamental frequency estimation using Gaussian smoothness. In *IEEE International Conference on Audio, Speech, and Signal Processing*, pages 105–108, April 2008.

[41] G. Poliner and D. Ellis. A discriminative model for polyphonic piano transcription. *EURASIP Journal on Advances in Signal Processing*, (8):154–162, January 2007.

[42] M. Ryynänen and A. Klapuri. Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music Journal*, 32(3):72–86, fall 2008.

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

[44] 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.

[45] 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, New Paltz, USA, August 2003.

[46] John Thickstun, Zaid Harchaoui, and Sham Kakade. Learning features of music from scratch. *ICLR*, 2017.

[47] Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton. Chainer: a next-generation open source framework for deep learning. In *Proceedings of workshop on machine learning systems (LearningSys) in the twenty-ninth annual conference on neural information processing systems (NIPS)*, 2015.

[48] T. Tolonen and M. Karjalainen. A computationally efficient multipitch analysis model. *IEEE Transactions on Speech and Audio Processing*, 8(6):708–716, November 2000.

[49] Karen Ullrich, Jan Schlüter, and Thomas Grill. Boundary detection in music structure analysis using convolutional neural networks. In *ISMIR*, pages 417–422, 2014.

[50] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *Advances in neural information processing systems*, pages 2643–2651, 2013.

[51] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *CoRR abs/1609.03499*, 2016.

[52] Paul J Walmsley, Simon J Godsill, and Peter JW Rayner. Polyphonic pitch tracking using joint bayesian estimation of multiple frame parameters. In *Applications of Signal Processing to Audio and Acoustics, 1999 IEEE Workshop on*, pages 119–122. IEEE, 1999.

[53] C. Yeh, A. Röbel, and X. Rodet. Multiple fundamental frequency estimation and polyphony inference of polyphonic music signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1116–1126, August 2010.

[54] K. Yoshii and M. Goto. A nonparametric Bayesian multipitch analyzer based on infinite latent harmonic allocation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(3):717–730, March 2012.

[55] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

8. APPENDIX

8.1 Extension to Multi-Pitch prediction

The labeling format as presented in the paper can not be applied to multi-pitch prediction. An extension is however straight forward. The proposed labeling format relates to the well known piano roll or midi format. More precisely, there is a minimum time resolution that defines the length of an event also known as ticks. Each tick can contain one, several or no events. There needs to be an indicator for the start s and the end e of a tick and a stop token S. For example, we might find the sequence [s C4 C5 e s C4 e s C4 e s e s e s e s C5 e S]. The notes C4 and C5 are hold for three ticks and one tick, respectively. Followed by a three tick rest and another one tick long C5. Note, that there is no need for a duration indication in this setting anymore. An advantage of this notation is that temporal errors can not accumulate over time. A disadvantage is the categorical nature of the pitch description. Naturally, we would want a continuous output since this is to be expected to benefit the model. However, we believe with good modeling choices it is possible to work that prior knowledge back into the model.