Transcription of Audio to MIDI Using Deep Learning

Patrick J. Donnelly
Department of Computer Science
Oregon State University
Corvallis OR, United States
patrick.donnelly@oregonstate.edu

Victoria Ebert
Department of Computer Science
Oregon State University
Corvallis OR, United States
ebertv@oregonstate.edu

***Abstract*—We investigate automatic transcription of polyphonic piano music with recurrent neural networks directly from raw audio. Previous approaches to this task rely on spectrograms or other features extracted for training, which requires timely pre-processing. In our approach, we train directly on sequences of 64 milliseconds of samples from the waveform. We evaluate our models on the MAESTRO dataset and achieve promising levels of pitch and onset accuracy but difficulty detecting the correct offset of the notes. Our approach also enables learning the note pitches and velocities at the same time. Together these results demonstrate the potential of relatively simple deep learning architectures to learn music transcription directly from raw audio.**

Keywords—automatic music transcription deep learning, long short-term memory, music information retrieval, raw audio

I. Introduction

Music transcription is the act of listening to a piece of music and writing it down as sheet music. The act of learning to transcribe by ear in an important skill taught in most music degree programs. In these dictation exercises, students practice monophonic dictation, consisting of one voice playing at a time, and polyphonic dictations, in which two or more voices sound simultaneously. As the number of notes playing simultaneously increases, this becomes an increasingly difficult task, even for highly trained musicians.

The task of Automatic Music Transcription (AMT) is an active area of research within the field of music information retrieval. AMT seeks to autonomously convert audio recordings of music to symbolic notation that can be read by musicians. An AMT system that could perform quick and automatic music transcription would many useful applications in musicology, music pedagogy, and music technology.

Musical Instrument Digital Interface (MIDI) files. MIDI is a symbolic representation of music, and was designed in the 1980s to be a simple, cheap way for manufacturers of digital devices to control their synthesizers independently of the device or vendor [1]. Most research in AMT has focused on the piano recording primarily because of availability of large and precise datasets of paired audio and MIDI recordings. Although these datasets feature just a single instrument, they are examples of polyphonic music and therefore the more difficult task of identifying multiple notes sounding at the same time.

AMT is a sequence to sequence learning task in which sequential windows of audio information are fed to a machine learning system and the model is tasked to predict an output sequence representing the notes played at that moment in time. In this work, we present a novel approach for music transcription training on raw audio, without pre-processing, spectral analysis, or feature extraction. Specifically, we train deep learning model directly on the samples of the digital waveform to learn a time-series representation of a MIDI file.

II. Related Work

Automatic music transcription remains an important area of focus in the field of information retrieval and researchers have tried a variety of algorithmic, signal processing, and machine learning approaches. In 2004, one of the earliest attempts at music transcription combined 76 different neural networks, each trained to recognize a piano different note [2]. A similar approach in 2011 combined 87 independent support vector machine classifiers, using a deep belief network in a frame-level note classification system with spectrograms as model input [3].

Other approaches at music transcription have drawn from speech to text methods. One such attempt uses an end-to-end approach to jointly train both language and music models via convolutional neural networks and constant-Q transforms [4]. These same authors also attempted another method using an arbitrary frame-level classifier, where the output was the probability that a given pitch would be in a frame [5]. In related but different approach, the authors attempt to reverse the AMT process by taking a score and attempting to generate the audio. PerformanceNet converts from a score to an audio file using a deep encoder and decoder [6], in an approach based on the text-to-speech network WaveNet [7].

Other researchers have chosen to focus on the onset of notes to aid in transcription. Some approaches use the note onset information in both training and evaluation [8]–[11], while others use the onsets only in their evaluation [3], [4], [12]–[15]. One method of assessing onset information comes from outputting a tuple of the form: `{start, end, note, volume}`[13]. One of the more successful attempts at automatic transcription makes predictions primarily from the onset of each note, and from there, calculates the other data, including the velocity of the note [8]. The authors later extended their work to incorporate [7] to both transcribe and generate music [9]. This study also introduced the MAESTRO dataset, which we use in this work and describe in Section III.B.

Another approach attempted sparse dictionary learning, using a dictionary of multiple atoms of different lengths for

978-1-6654-8157-1/22/$31.00 ©2022 IEEE

each pitch to jointly estimate pitch, onset, and duration in a convolutional sparse coding framework [11]. The authors first learned a convolutional sparse encoding of the dictionary by dictionary by recording audio of each piano note and using the waveform to estimate the activations [15].

Other approaches using spectrograms have examined deep belief networks [16], hidden Markov models [13], recurrent neural networks [16], and transformers [9], [17]. Transformers are relatively new deep learning architecture with mechanisms for self-attention that have found recent success in natural language processing. Recently, Hawthorne *et al.* applied transformer models to segmented spectrograms to perform AMT, significantly outperforming all previous attempts [17].

In this work we explore AMT with recurrent neural networks. Previous attempts using recurrent neural networks are trained on spectrograms [4], [5], [14]. Our approach differs from these other approaches in that we train recurrent neural networks on raw audio from the waveform. Our trained models can predict a piano roll-like representation that can be easily converted to a MIDI file. This is perhaps similar to the output format of [14], but our approach also attempts to learn the velocity of the note.

III. Dataset

Over the years many types of MIDI instruments have been developed, including synthesizers, drum kits, guitars, and wind and brass controllers. Most studies attempting AMT focus on piano recordings, because of the availability high quality data consisting of pairs of audio recordings and MIDI files. Because modern MIDI-enabled pianos can so easily automatically produce high quality synchronized recordings, the majority of research in AMT focuses on these available datasets containing polyphonic piano music.

A. Yamaha Disklavier

A Disklavier piano records precise MIDI information as it is played. Each time the pianist presses a key or pedal, the piano records which key/pedal, timing, and precise velocity the key was pressed. The Disklavier also can play back MIDI files, all while still functioning and sounding like a classic grand piano.

Until recently, researchers investigating AMT relied on the MIDI aligned Piano Sounds (MAPS) dataset [18] which contains 60 Disklavier and 210 synthesizer recordings, and the Saarland dataset [19], containing 50 Disklavier recordings.

B. MAESTRO

In 2018 the **MI**DI and **A**udio **E**dited for **S**ynchronous **TR**acks and **O**rganization (MAESTRO) dataset [9] was introduced. This dataset is an order of magnitude larger than any previous dataset. The data was compiled from over 10 years of recordings at the International Piano E-competition[1] in which competitors perform each piece on a Yamaha Disklavier.

During this competition, each performance was recorded in a concert hall. The dataset contains synchronized pairs of MIDI and audio recordings aligned to within three milliseconds of accuracy. These MIDI recordings allow the judges to re-preform the exact performance played on the piano again in the same physical space as the other competition contestants.

The MAESTRO dataset contains 1276 songs, is 198.7 hours long, and contains 7.04 million individual notes [20]. The authors of the data suggest a partition of the dataset into train, validation, and test splits, as shown in Table I. This partition ensures that same composition, if repeated, occurs within the same split. This split allocates 75% for training, 11% for validation, and 14% for testing. We use this dataset in this work.

TABLE I. Recommended Split for MAESTRO Dataset

Split	Performances (count)	Duration (hours)	Size (GB)	Notes (millions)
Train	962	159.2	96.3	5.66
Validation	137	19.4	11.8	0.64
Test	177	20.0	12.1	0.74
Total	1275	198.7	120.2	7.04

C. Audio Pre-processing

The MAESTRO dataset contains CD quality 44.1 kHz or better 16-bit PCM stereo audio. We first down-sampled the recordings in the MAESTRO dataset to 16 kHz mono audio files. We note that C8, the highest note of the piano, corresponds to a frequency of 4186 Hz an equal temperament. Considering the Nyquist theorem, we preserve frequencies up to one half the sampling rate, or 8 kHz. Although we will lose many of the harmonics for notes of the upper range of the piano, we sufficiently capture all the instrument's fundamental frequencies. Down-sampling to 16 kHz single channel audio allows us to reduce the input size to the neural network significantly, while still retaining sufficient auditory fidelity.

D. MIDI Pre-processing

Although MIDI files are in format easily read by audio software, they are not readily in a format that we can readily align with our temporal audio input. Instead MIDI files represent a series of events containing relative timing information. We compare an audio waveform and MIDI representation in Fig. 1.

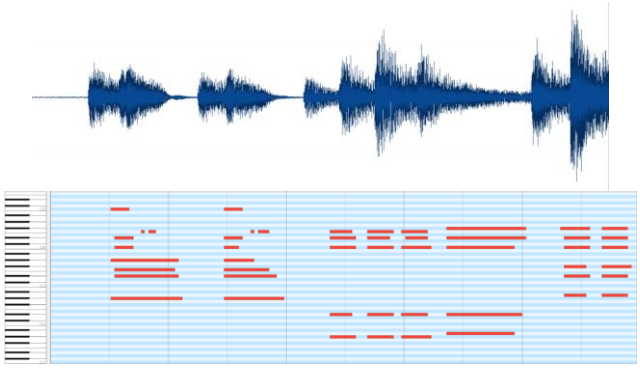


Fig. 1. A visual comparison of an audio waveform (top) and the corresponding MIDI shown in piano-roll notation (bottom).

[1] https://www.piano-e-competition.com/

To align the MIDI files with our audio, we first need to convert the information to a time series. We accomplish this in two steps. From these MIDI files, we need to (1) extract timing information and (2) convert these MIDI files into a time-series that aligns with our audio. First, we converted the MIDI files with the `py-midicsv` library[2] to a CSV file representation from which it was easier to extract note and timing information. Because we only required information about notes, velocities, and timings, we discard meta-information and other data. Second, we converted the CSV representation to a format that synchronizes with the time scale of our audio input. In this work, we segment the audio with a window size 1024 samples of 16 kHz audio, which represents 64 milliseconds. We chose this window size because 1024 is a common window size for audio manipulations, including a fast Fourier transform. We converted the CSV to a time series representation in which each time step is a snapshot of all notes playing in that 64 ms window.

For each 64 ms time step, we store 131 velocity values representing the 128 piano keys plus the three pedals. A MIDI velocity value represents the strength to the performer pressed the piano key ranging $[0..127]$. We normalize the values to [0..1] to use with our neural network. We refer to this time-series representation as MIDI_TS for the remainder of this paper.

E. Audio and MIDI Alignment

The length of songs in the MAESTRO dataset range from 45 seconds to 45 minutes. For processing by the neural network, we split the songs into approximate ten second non-overlapping chunks. More precisely, and given that the MIDI_TS files have time-steps of 64 ms that cannot be divided, our chunks were each 9.984 seconds. The final chunk in each song was zero-padded with silence to reach a duration of ten seconds. We chose this approach in order to be able to process songs of all lengths while minimizing the amount of zero-padding necessary to standardize the song lengths. This results in a pair of 10 second chucks of audio and MIDI_TS, which serve as the input and expected output used to train our models.

IV. Experimental Design

In this section we describe the model architecture, experimental design, post-processing, and evaluation metrics.

A. Model

The Long Short Term Memory (LSTM) architecture is a type of recurrent neural network architecture that uses a series of gates to be able to retain some memory of data that came previously. The Bidirectional LSTM (BLSTM) retains memory in both the forward and backwards directions when training.

In this work we use a BLSTM with two hidden layers. Our input layer corresponds of 1024 audio samples. Our output layer consists of 131 piano notes or pedals. In between the input and output layers, our two hidden layers each step down the dimensionality by a reduction of two, with sizes 512 and 256, respectively. We included a drop of 0.2 for each hidden layer to compensate for any overfitting of the model that may occur. We calculate model loss with mean squared error.

B. Eliminating Quiet Noise

Unsurprising, the neural network learned very small values of noise for all the piano notes that were not playing. In post-processing, we set any predicted velocity values < 0.1 to 0.0. We designed our model to predict across all 127 MIDI notes. However, the piano ranges from MIDI note 21 to 108, representing the 88 piano keys. We chose to predict across all 127 MIDI notes to allow our model the flexibility of future generalizations to other instruments that have different ranges than the piano. As expected, after this post-processing the 40 unused MIDI notes on a piano were predicted to be 0 in all cases.

C. Converting back to MIDI

Our model produces output in the MIDI_TS representation described in Section III.D. To enable listening to this output, we reverse the conversion process back to a MIDI file so they can be played by standard software. In the predicted songs returned by our model, each note had a different velocity at each of its 64 ms timesteps. This is because we spliced the sequences, learned them sequentially, and then reconstructed them back to a song. Given our process, this has the effect of producing a new MIDI note every 64 ms, with an ever so slightly different velocity than the note before it. This does not present a problem when evaluating our output mathematically, as the velocity is treated as binary. However, this is not desirable when listening to the predicted MIDI. To combat this, we implemented a smoothing function in which each note was replaced with the average velocity of the predicted timestamps throughout its duration.

D. Evaluational Metrics

We evaluated our results with the `mir_eval` toolkit library [21]. This library is an implementation of many common music information retrieval evaluation metrics. From this toolkit, we use the `mir_eval.transcription` metric, which counts a predicted notes as correct if it matches the onset of the expected note within ± 50 ms. The `mir_eval` currently does not support evaluating the piano pedals, so we exclude them from evaluation here to be addressed as future work. The `mir_eval.transcription` metric produces a set of 14 values, grouped in four categories: (1) Pitch, Onset, Offset, (2) Pitch, Onset, (3) Onset, and (4) Offset. Each of these include precision, recall, and F-measure metrics. The first two categories that measure pitch also includes average overlap ratio.

V. Results

We trained our model on the training set of 962 pieces and tested on the held-out test set of 177 pieces.

A. Evaluating the Test Set

We computed the `mir_eval`metrics for each individual song before averaging these results to produce a summary measure of the entire test set, shown in Table II.

Our model does fairly well in its ability to detect note onsets ($F_1 = 0.71$). When we consider pitch and onset together

[2] https://pypi.org/project/py-midicsv/

(F_1 = 0.63), we observe a reduction in our score, indicating that we did not identify pitch quite as well as onset alone. However, we when we consider pitch, onset, and offset together (F_1 = 0.26), we note our models were wholly unsuccessful at correctly determining the offset (end) of the notes. This is because the notes continue to resonate, especially when a pedal is applied, after the release of the MIDI note. Our model learns these longer note duration from the audio, but this is penalized by evaluation metrics including offsets.

TABLE II. EVALUATION OF THE TEST SET (N=177)

	Pitch, onset, & offset	*Pitch & onset*	*Onset only*	*Offset only*
Precision	.26	.65	.73	.55
Recall	.25	.62	.70	.52
F_1-Measure	.26	.63	.71	.53
Avg. Overlap Ratio	.97	.68	-	-

We find it encouraging that we observe relatively similar scores for precision and recall across of all of our results. This indicates that our model's errors are balanced between false negatives and false positives and not skewed to one or the other.

B. Comparision with Other Work

Next, we compare our results to other recent studies in the literature, as shown in Table III. The results published before 2019 used the MAPS dataset [18] which contains 65 hours of Disklavier piano recordings. Studies published in 2019 or later use the MAESTRO dataset that we use in this work.

We find it encouraging that we outperform two relatively recent results that are referenced today. These approaches learn from spectrograms with constant-Q transforms [5], [22]. In their 2016 publication the authors note that "time-frequency representations in the form of spectrograms still seem to have a distinct advantage over raw audio input" [22]. In this work, we use raw audio input to achieve a 46% improvement when considering pitch and offset and a 29% improvement when considering pitch, onset, and offset.

However, the three most recent state of the art results [8], [13], [17] all outperform our current approach. These approaches also use spectrograms for input. One gains improvement by jointly training separate models for onset and offset detection [8] and another uses a probabilistic model to better align note predictions from the neural network model [13]. In the most recent result in the literature, researchers at Google applied a state of the art encoder-decoder Transformer architecture [17].

TABLE III. COMPARISON WITH OTHER RECENT RESULTS

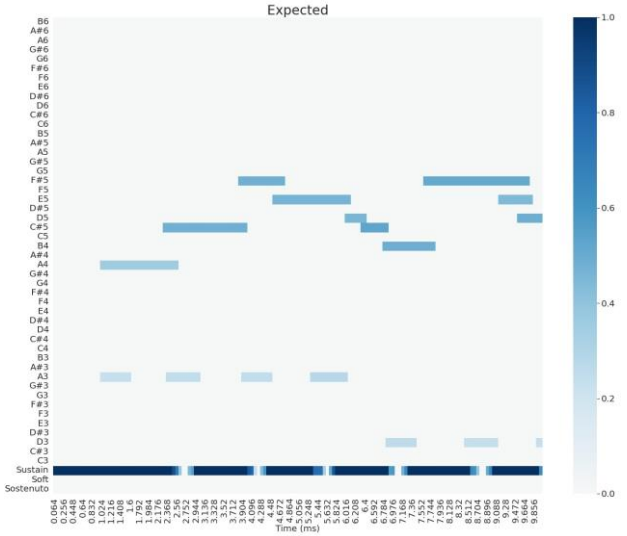
	Pitch, onset, & offset			*Pitch & onset*		
Study	***P***	***R***	***F_1***	***P***	***R***	***F_1***
Hawthorne, 2021 [17]	-	-	83.9	-	-	96.0
Hawthorne, 2018 [8]	51.3	49.3	50.2	84.2	80.8	82.3
Kelz, 2019 [13]	61.9	51.7	56.1	90.2	74.8	81.4
This Work, 2022	26.0	25.3	25.5	64.6	62.8	63.0
Kelz, 2016 [22]	20.1	27.8	23.1	44.3	91.3	50.9
Sigtia, 2016 [4]	17.6	19.7	18.4	45.0	49.6	46.6

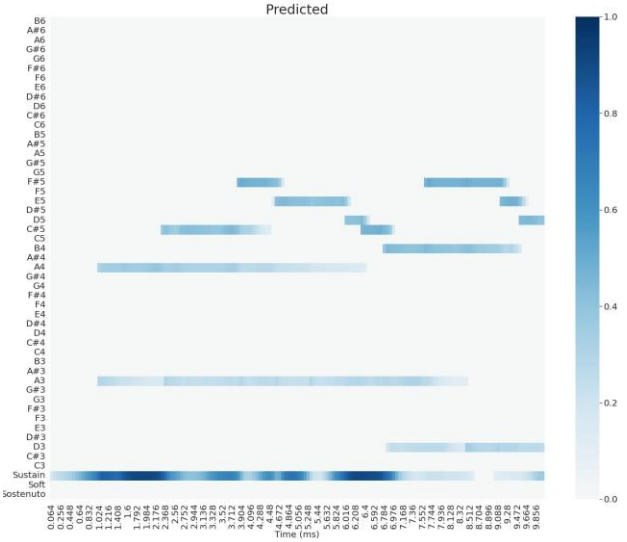
C. Evaluation: Sonata in A Major, K.208

To better understand the efficacy our model's transcriptions, we examine the prediction for short excerpt of a piano sonata, Sonata in A Major, K. 208 by Domenico Scarlatti. In TABLE IV. we show the evaluation of the first 10 seconds of this piece.

TABLE IV. EVALUATION OF THE FIRST 10 SECONDS OF K.208

	Pitch, onset, & offset	*Pitch & onset*	*Onset only*	*Offset only*
Precision	.347	.669	.823	.734
Recall	.267	.516	.634	.556
F_1-Measure	.302	.582	.716	.639
Avg. Overlap Ratio	.993	.760	-	-





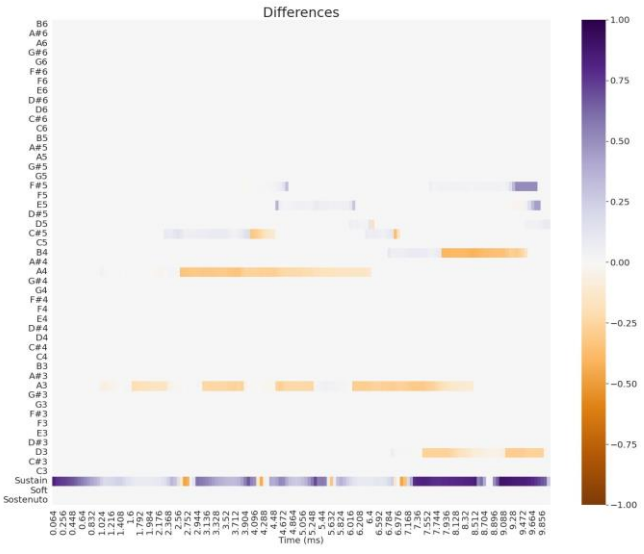


Fig. 2. Visualization of the notes, times, and velocities for the first ten seconds of Scarlatti's Sonata in A Major, K.208, showing the true MIDI output (top), our predicted output (middle), and the difference which highlights our model's errors (bottom).

This excerpt illustrates the use of the sustain pedal and our difficult in predicting note offsets. After the pedal releases, we capture the resonance of the notes as they decay in the concert hall. In Fig. 2 we compare the true MIDI notes and our predicted notes, and we highlight the specific errors made by our model. We can see that our model does not separate well the repeated notes when the pedal is applied. Otherwise, our predictions were generally accurate with respect to pitch and note onset.

VI. Conclusion

One of the primary limitation of our current approach is our treatment of the three piano pedals. The sustain pedal removes the dampers from the strings which permits strings reverberate for longer. The reverberation after the note offset is captured in the audio recordings but not in the MIDI files. In this attempt, we treated the pedals like additional piano keys. However, our model did not predict the pedals with fidelity. As future work we will follow the approach taken by [8], [9] which preprocesses the MIDI to adjust note offsets based on the pedal actions. By integrating the pedal information *a priori*, we expect to predict more accurate offsets, as well as improve onset detection in cases where notes are reiterated when a pedal is pressed.

Additionally, we evaluated the performance of our models with the `mir_eval` library in order to compare our results with other relevant studies. However this evaluation toolkit does not currently measure the predicted velocity of notes, evaluating each note in binary as pressed or note pressed. We will extend these metrics to include evaluation of the velocities values.

We have demonstrated the use of a BLSTM for the automatic transcription of solo piano performance directly from a raw audio file. The majority of approaches for AMT must pre-process the audio to extract spectral or acoustic features. Our approach learns directly from the sequence of samples of a waveform without timely preprocessing. Approaches that can perform quick inference will be necessary for future technologies that seek to understand music in near or delayed real-time. Additionally our approach is able to learn the velocities of the MIDI notes directly at the same time it learns their pitch, onset, and offset. Although our results do not yet match the state-of-the-art, we demonstrate that intentional onset detection is perhaps not necessary to explicitly determine the start of notes when transcribing with deep learning.

References

[1] G. Loy, "Musicians Make a Standard: The MIDI Phenomenon," *Computer Music Journal*, vol. 9, no. 4, pp. 8–26, 1985, doi: 10.2307/3679619.

[2] M. Marolt, "A Connectionist Approach to Automatic Transcription of Polyphonic Piano Music," *IEEE Trans. Multimedia*, vol. 6, no. 3, pp. 439–449, Jun. 2004, doi: 10.1109/TMM.2004.827507.

[3] J. Nam, J. Ngiam, H. Lee, and M. Slaney, "A Classification-Based Polyphonic Piano Transcription Approach Using Learned Feature Representations," p. 7, 2011.

[4] S. Sigtia, E. Benetos, and S. Dixon, "An End-to-End Neural Network for Polyphonic Music Transcription," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 5, pp. 927--939, 2016.

[5] S. Sigtia, E. Benetos, N. Boulanger-Lewandowski, T. Weyde, A. S. d'Avila Garcez, and S. Dixon, "A Hybrid Recurrent Neural Network For Music Transcription," *arXiv:1411.1623 [cs]*, Nov. 2014, Accessed: Mar. 30, 2020. [Online]. Available: http://arxiv.org/abs/1411.1623

[6] B. Wang and Y.-H. Yang, "PerformanceNet: Score-to-Audio Music Generation with Multi-Band Convolutional Residual Network," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, Art. no. 01, Jul. 2019, doi: 10.1609/aaai.v33i01.33011174.

[7] A. van den Oord *et al.*, *WaveNet: A Generative Model for Raw Audio*. 2016.

[8] C. Hawthorne *et al.*, "Onsets and Frames: Dual-Objective Piano Transcription," *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR 2018*, Jun. 2018, Accessed: Sep. 14, 2020. [Online]. Available: http://arxiv.org/abs/1710.11153

[9] C. Hawthorne *et al.*, "Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset," *7th International Conference on Learning Representations, ICLR 2019*, Jan. 2019, Accessed: Jun. 28, 2020. [Online]. Available: http://arxiv.org/abs/1810.12247

[10] J. J. Valero-Mas, E. Benetos, and J. M. Iñesta, "Assessing the Relevance of Onset Information for Note Tracking in Piano Music Transcription," p. 8, 2017.

[11] A. Cogliati, Z. Duan, and B. Wohlberg, "Piano Transcription With Convolutional Sparse Lateral Inhibition," *IEEE Signal Processing Letters*, vol. 24, no. 4, pp. 392–396, Apr. 2017, doi: 10.1109/LSP.2017.2666183.

[12] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," *CoRR*, vol. abs/1409.3215, 2014, [Online]. Available: http://arxiv.org/abs/1409.3215

[13] R. Kelz, S. Böck, and G. Widmer, "Deep Polyphonic ADSR Piano Note Transcription," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 246–250, May 2019, doi: 10.1109/ICASSP.2019.8683582.

[14] S. Bock and M. Schedl, "Polyphonic piano note transcription with recurrent neural networks," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Kyoto, Japan, Mar. 2012, pp. 121–124. doi: 10.1109/ICASSP.2012.6287832.

[15] A. Cogliati, Z. Duan, and B. Wohlberg, "Context-Dependent Piano Music Transcription With Convolutional Sparse Coding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, pp. 1–1, Dec. 2016, doi: 10.1109/TASLP.2016.2598305.

[16] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Audio Chord Recognition with Recurrent Neural Networks.," in *ISMIR*, 2013, pp. 335–340.

[17] C. Hawthorne, I. Simon, R. Swavely, E. Manilow, and J. Engel, “Sequence-To-Sequence Piano Transcription With Transformers,” p. 8, 2021.

[18] V. Emiya, N. Bertin, B. David, and R. Badeau, *MAPS - A piano database for multipitch estimation and automatic transcription of music*. 2010.

[19] M. Müller, V. Konz, W. Bogler, and V. Arifi-Müller, “Saarland Music Data (SMD),” *Late-Breaking and Demo Session of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[20] “The MAESTRO Dataset,” *Magenta*. https://magenta.tensorflow.org/datasets/maestro (accessed Jul. 07, 2020).

[21] C. Raffel *et al.*, “mir_eval: A Transparent Implementation of Common MIR Metrics,” Oct. 2014.

[22] R. Kelz, M. Dorfer, F. Korzeniowski, S. Böck, A. Arzt, and G. Widmer, “On the Potential of Simple Framewise Approaches to Piano Transcription,” *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR 2016)*, Dec. 2016.