ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-6327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICASSP49357.2023.10095653

POP2PIANO : POP AUDIO-BASED PIANO COVER GENERATION

*Jongho Choi**

Rebellions Inc.
jongho.choi@rebellions.ai

Kyogu Lee

Department of Intelligence and Information,
Interdisciplinary Program in Artificial Intelligence,
AI Institute, Seoul National University
kglee@snu.ac.kr

ABSTRACT

Piano covers of pop music are enjoyed by many people. However, the task of automatically generating piano covers of pop music is still understudied. This is partly due to the lack of synchronized {Pop, Piano Cover} data pairs, which made it challenging to apply the latest data-intensive deep learning-based methods. To leverage the power of the data-driven approach, we make a large amount of paired and synchronized {Pop, Piano Cover} data using an automated pipeline. In this paper, we present Pop2Piano, a Transformer network that generates piano covers given waveforms of pop music. To the best of our knowledge, this is the first model to generate a piano cover directly from pop audio without using melody and chord extraction modules. We show that Pop2Piano, trained with our dataset, is capable of producing plausible piano covers.

***Index Terms*—** Piano Cover, Music Arrangement, Transformer, Music Synchronization

1. INTRODUCTION

Piano cover refers to a musical piece in which all musical elements of an existing song are recreated or arranged in the form of a piano performance. Piano covers of pop music are one of the widely enjoyed forms of music. For example, people use them for music education purposes, and piano cover creators have millions of subscribers on media such as Youtube.

In order for a human to create a piano cover, it is necessary to recognize all musical elements such as melodies, chords or moods from the original audio and reinterpret them into musically appropriate piano performances. Therefore, making a piano cover is not an easy task even for humans as it is a creative task and requires musical knowledge.

Previous studies have focused on arranging given pop audio to other instruments [1, 2] using multiple external modules to extract explicit musical information, such as melodies and chords. However, creating a piano cover involves various implicit musical characteristics, such as the atmosphere of the music and the arranger's style. Therefore, we believe that research on end-to-end conversion between audio and piano covers can also be valuable.

Meanwhile, Deep learning has demonstrated excellent performance in modeling high-dimensional music audio data. However, to the best of our knowledge, there has been no deep learning study on piano cover modeling using waveforms directly from pop music. This may be due to the lack of large amounts of synchronized paired data for such modeling.

In this study, we introduce a Transformer-based piano cover generation model, called Pop2Piano, which generates a piano performance (MIDI) from the waveform of pop music. Our contributions are summarized as follows:

- We propose a novel approach to piano cover generation. For this, We build 300-hour of synchronized {Pop, Piano Cover} dataset, called **P**iano Cover **S**ynchronized to **P**op Audio (PSP), and introduce the preprocessing system used to make this dataset.
- We design Pop2Piano, a Transformer network that generates piano covers from given audio. It does not use a melody or chord extractor but uses the spectrogram itself as an input. It also controls the covering style using the arranger token.
- We upload a list of data and preprocessing codes to reproduce the PSP dataset, and release an executable Pop2Piano demo on Colab.

2. RELATED WORK

2.1. Automatic Music Transcription

Automatic music transcription (AMT) is the task of estimating note information from musical instrument waveforms. In piano AMT, several studies have used CNN and RNN architectures to extract features and model note onset, offset, and frame [3, 4, 5]. Some studies have employed the Transformer model and spectrograms to generate MIDI [6, 7]. Multi-instrument AMT aims to estimate note information for each instrument from mixed sounds of several instruments. Cerberus [8] and MT3 [7] are two such studies, with RNN and Transformer models, respectively.

These studies have yielded promising results with large synchronized instrument-audio datasets, such as MAESTRO [9] and Slakh2100 [10]. To train a model for piano cover generation, we build a 300-hour synchronized piano cover dataset. While AMT and piano cover generation share similarities in converting audio to musical notes, they differ in that piano cover generation does not have a single correct answer. For example, the monophonic vocal of the original song can be expressed in various ways, such as voicing or doubling, and the harmony may appear as various accompaniment textures depending on the original song's atmosphere or arranger's style.

*The first author performed the work while at Music and Audio Research Group, Seoul National University.

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2022-0-00320, Artificial intelligence research about cross-modal dialogue modeling for one-on-one multi-modal interactions)

https://sweetcocoa.github.io/pop2piano_samples

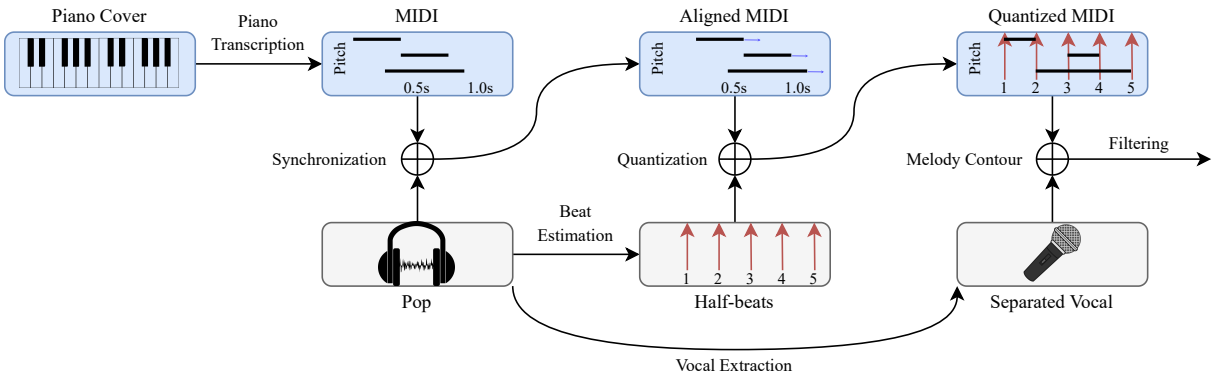

Fig. 1. A preprocessing pipeline for synchronizing and filtering paired {`Pop, Piano Cover`} audio data.

2.2. Music Transformation

Music transformation is an active research area aimed at manipulating music data. One common approach is music style transfer, which is the task of transforming a music piece from one style to another while preserving the original musical characteristics [11, 12, 13].

In many studies, there is no unified definition of music style, so in this study, we use the term style to describe the musical interpretation and composition process that varies from arranger to arranger when creating a piano cover of a given pop song. There are several studies on music style transfer in this sense [12, 14]

Another common approach is music reduction, which is the process of rearranging a piece of music to be performed by a smaller group of instruments or a single performer, while preserving its musical content [15, 16]. Studies on audio-based instrument covers can be considered as music reduction to a single instrument. Previous works have utilized external modules to extract melodies and chords from audio. For example, Song2Guitar [1] generated guitar covers from pop audio by using modules to extract melodies, chords, and beats. Then, a guitar tab score was generated by statistically modeling the fingering probability. Similarly, in *Takamori et al.* [2], melodies, chords, and choruses were extracted using external modules. A piano score was then generated using rule-based methods. In contrast, our study, Pop2Piano, only uses a beat extractor to generate piano notes directly from pop audio.

3. DATASET

The main bottleneck in modeling raw waveform pairs is a long range of dependencies, as they are computationally challenging to learn the high-dimensional semantics of music. An alternative to that is a method of training short-length waveform segments in pairs. However, this method requires paired data to be synchronized; otherwise, there is a risk that the correct label will not be included in the paired segmented data. As the collected data pair was not synchronized, we developed a preprocessing pipeline to obtain synchronized {`Pop, Piano Cover`} data, as shown in Fig 1.

3.1. Preprocessing

3.1.1. Synchronizing Paired Music

First, we convert piano cover audio into MIDI, using the piano transcription model [5]. Second, we roughly align piano MIDI with pop audio. We obtain a warping path using *SynctoolBox* [17] and then use it to modify the note timings of MIDI via linear interpolation. This provides a simple synchronization between the piano cover and pop audio. Third, the note timings are quantized into 8th-note units. After extracting beats from pop audio using *Essentia*, the onset and offset of each note in MIDI are quantized to the nearest 8th-note beat. If the onsets and offsets of the quantized notes are the same, the offset is moved to the next beat. In this way, the entropy of data can be lowered by changing the time unit of a note from continuous-time (seconds) to quantized time (beats).

3.1.2. Filtering Low Quality Samples

There are cases where the data pairs are unsuitable for various reasons, such as the difference in musical progress or having different keys from the original song. To filter out these cases, all data with a melody chroma accuracy (MCA) [18] of 0.15 or less, or an audio length difference of 20% or more are discarded. Melody chroma accuracy is calculated between the pitch contour of the vocal signal extracted from the audio and the top line of the MIDI.

We use *Spleeter* [19] to separate the vocal signal. Then, to get the melody contours of pop music, the f0 sequence of the vocal is calculated using *Librosa* [20] *pYIN* [21]. The sample rate is 44100 and the hop length is 1024.

3.2. Piano Cover Synchronized to Pop Audio (PSP)

We collect 5989 piano covers from 21 arrangers and corresponding pop songs on YouTube. We then synchronized and filter {`Pop, Piano Cover`}. As a result, a total of 4989 tracks (307 hours) are left, which are used as a training set, PSP. Note that the piano cover included in the PSP is unique, but the original song is not. This will help the model train separately the style of the piano cover according to the arranger conditions and the acoustic characteristics of the given audio.

4. MODEL

4.1. Inputs and Outputs

Pop2Piano uses a log-Mel spectrogram of pop audio as an encoder input. The sampling rate is 22050, the window size is 4096, and the hop size is 1024. In addition, the arranger token, which indicates who arranged the target piano cover, is embedded and appended before

https://essentia.upf.edu/

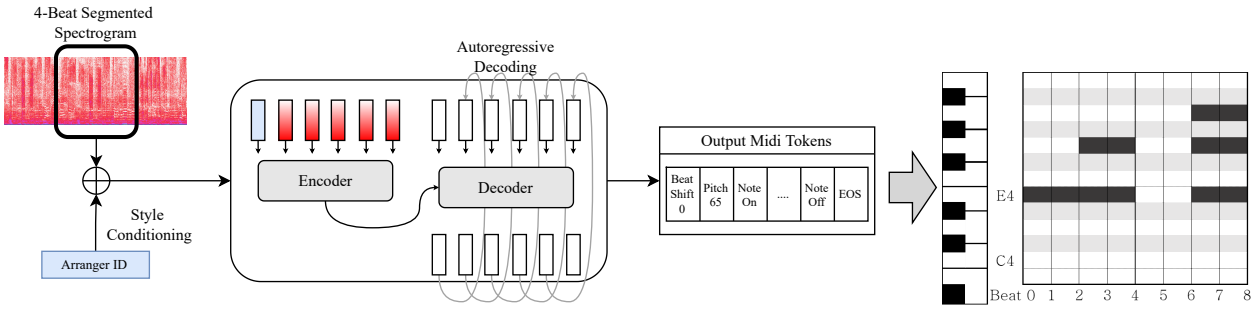

Fig. 2. The architecture of our model is an encoder-decoder Transformer. Each input position for the encoder is one frame of the spectrogram. We concatenated an embedding vector representing a target arranger style to the spectrogram. Output MIDI tokens are autoregressively generated from the decoder.

the first frame of the spectrogram. Each step of the decoder output is chosen from the following types of tokens:

Note Pitch [128 values] Indicates a pitch event for one of the MIDI pitches. But only the 88 pitches corresponding to piano keys are actually used.

Note On/Off [2 values] Determines whether previous Note Pitch events are interpreted as note-on or note-off.

Beat Shift [100 values] Indicates the relative time shift within the segment quantized into 8th-note beats. It will apply to all subsequent note-related events until the next Beat Shift event. We define the vocabulary with Beat Shifts up to 50 beats, but because time resets for each segment, In practice we use only about 10 events of this type.

EOS, PAD [2 values] Indicates the end of the sequence and the padding of the sequence.

The idea for this dictionary was taken from Transformer-based AMT studies [6, 7]. For a detailed example of this token's interpretation, See Fig 3. For each input, the model autoregressively generates outputs until an EOS token is generated in the decoder. To generate a piano cover of pop audio of arbitrary length, in the inference stage, the audio is sequentially cropped by 4 beats and used as input to the model, then generated tokens (except for EOS) are concatenated. After that, the relative beats of the generated tokens are converted into absolute time using the absolute time information of the beat extracted from the original song and then converted into a standard MIDI file.

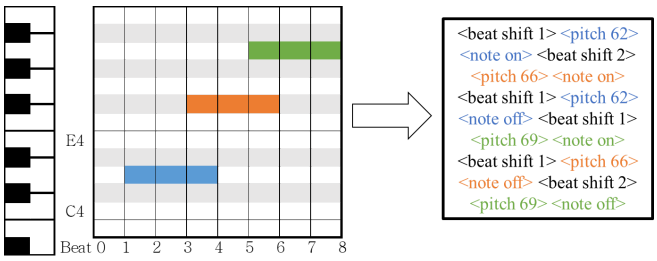

Fig. 3. An example of piano tokenization. the beat shift token means a relative time shift from that point in time.

4.2. Architecture

The Pop2Piano model architecture is T5-small [22] used for MT3 [7]. It is a Transformer network with an encoder-decoder structure. The number of learnable parameters is about 59M. Unlike MT3 [7], the relative positional embedding of the original T5 is used instead of the absolute positional embedding. Additionally, a learnable embedding layer is used for embedding the arranger style. An overview of these processes is shown in Fig 2.

5. EXPERIMENT

In this experiment, we check whether Pop2Piano trained with PSP (Pop2Piano$_{PSP}$) can generate a plausible piano cover and verify that it follows a specific arranger's style.

5.1. Training Setup

We train the model with audio extracted at random 4-beat lengths. We repeat this process 2000 times for the entire PSP dataset. Assuming that the average music has a bpm of 120, we can estimate that about 5500 hours of audio were used for training using the PSP dataset. The network is optimized using the AdaFactor [23] and its learning rate is 0.001.

5.2. Generation Result

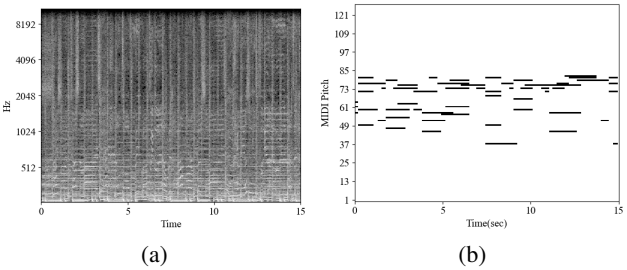

Fig. 4. (a) A Melspectrogram of an input pop audio. (b) A piano roll of output MIDI notes corresponding to input audio.

Fig 4 is an example of a piano cover generated using arbitrary pop music as an input. The generated sample can be listened to on

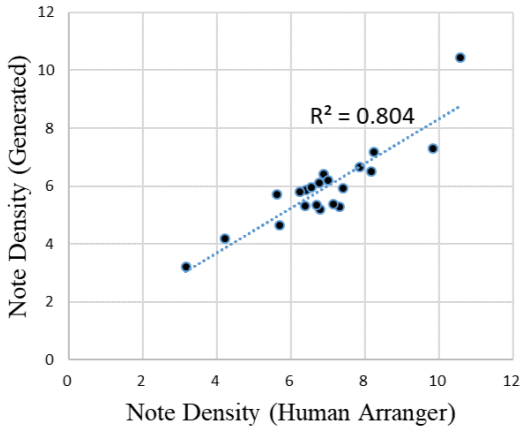

Fig. 5. The dots represent the note density of the piano cover generated by being conditioned with that arranger. The unit is the number of notes per second.

Original Songs	Arranger	Average MCA
POP909$_F$	Human	0.395
PSP	Human	0.3493
POP909$_F$	Pop2Piano$_{PSP}$	**0.402** $\pm$ **0.021**

Table 1. Average Melody Chroma Accuracy (AMCA) of human-made and generated piano covers. The piano covers of Pop2Piano are generated for all 21 arrangers of PSP, and its standard deviation according to arrangers are shown in the table.

the link. The original song is a complex audio signal mixed with various sounds such as vocals, bass, and percussion. Nevertheless, it can be seen that the generated piano cover shows the vocal melody line of the original song as stacked notes, and also includes a plausible accompaniment that follows the harmony of the original song. Also, various piano cover samples generated by Pop2piano can be listened to with the original song on the demo page.

Additionally, we would like to confirm whether the generated piano cover not only plausibly followed the original song but also covered it in the style of the arranger we designated. However, the cover style is very implicit, so there are few ways to evaluate quantitatively. Inspired by the difference in note densities among arrangers, we measure the note densities of the piano cover generated by Pop2Piano$_{PSP}$ to indirectly verify whether it generates a piano cover in the target arranger's style. As a result, Fig 5 shows that the generated piano cover follows the note density of the target arranger with high linearity of $R^2 = 0.804$.

We assess the quality of the generated piano covers by computing the MCA, as described in Section 3.1.2. Interestingly, our results show that the MCA of the generated piano covers is similar to that of the human-created POP909 dataset [24], even when trained without explicit melodies. POP909 is a piano MIDI dataset of 909 Chinese pop songs, with separate tracks for melody, bridge, and accompaniment. However, we merge all the tracks for consistency in our experiments, and like PSP, we synchronize the {`Pop, Piano Cover`} pairs through our preprocessing pipeline. After filtering, 817 tracks remain, which we denote as POP909$_F$ in the rest of this paper. The

https://sweetcocoa.github.io/pop2piano_samples/fig_on_paper.html

(%)	GT$_F$	Pop2Piano$_{PSP}$	Pop2Piano$_{POP909}$
vs GT$_F$	-	**29.6**	22.24
vs Pop2Piano$_{PSP}$	*70.4*	-	36
vs Pop2Piano$_{POP909}$	*77.76*	**64**	-
MOS	*3.771*	**3.216**	2.856

Table 2. The winning rate and Mean Opinion Score(MOS) according to the piano cover model. GT$_F$ denotes synchronized piano covers created by human arrangers.

MCA values are presented in Table 1. Note that a higher MCA does not necessarily indicate better piano covers, and thus, evaluating the quality of the generated piano covers will also be a good follow-up study.

5.3. Subjective Evaluation

We measure the subjective quality of the generated piano covers by evaluating their naturalness. Since there is no prior study publicly available, we use a Pop2Piano network trained with POP909 (Pop2Piano$_{POP909}$) as a baseline model. The comparison with this model indicates how the PSP dataset affected the performance of the Pop2Piano model.

We conduct a user study. There are 25 participants and no professional musicians are included. Participants do not know which model has made each piano cover. We select 10-seconds from the beginning of the chorus of 25 songs that are not included in the training dataset. For each song, a piano cover made by a human arranger is used as GT. However, since GT also needs to be synchronized with the original song, the synchronization pipeline is applied and that data is denoted as GT$_F$. After listening to the original song and piano covers, the participants evaluate how naturally the given piano cover arranged the original song with 1-5 points.

In the listening evaluation, 70% of participants prefer the piano cover generated by Pop2Piano$_{PSP}$ to Pop2Piano$_{POP909}$. In MOS analysis between Pop2Piano$_{PSP}$ and Pop2Piano$_{POP909}$, using paired one-sided Wilcoxon test, we reject H_0 at 99% confidence intervals $(0.222, \texttt{inf})$ with $p = 3.34e{-}05$. See Table 2.

5.4. Limitation

We recognize that some improvements can be made to our model. For instance, Pop2Piano uses only four-beat length audio for the context of input. Therefore, features such as melody contour or texture of accompaniment have less consistency when generating longer than four-beat. Also, time quantization based on eighth note beats prevents the model from generating piano covers with other rhythms such as triplets, 16th notes, and trills.

6. CONCLUSION

We present Pop2Piano, a novel study on generating pop piano covers directly from audio without using melody or chord extraction modules based on a Transformer network. We collect PSP, 300 hours of paired {`Pop, Piano Cover`} datasets, to train the model. And we design a pipeline that synchronizes them in a form suitable for training neural networks. We open-source the list of data and code needed to reproduce the PSP. We show and evaluate that Pop2Piano$_{PSP}$ can generate plausible pop piano covers and also can mimic the style of a specific arranger.

7. REFERENCES

[1] Shunya Ariga, Satoru Fukayama, and Masataka Goto, "Song2guitar: A difficulty-aware arrangement system for generating guitar solo covers from polyphonic audio of popular music.," in *ISMIR*. Suzhou, 2017, pp. 568–574.

[2] Hirofumi Takamori, Takayuki Nakatsuka, Satoru Fukayama, Masataka Goto, and Shigeo Morishima, "Audio-based automatic generation of a piano reduction score by considering the musical structure," in *International Conference on Multimedia Modeling*. Springer, 2019, pp. 169–181.

[3] Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse Engel, Sageev Oore, and Douglas Eck, "Onsets and frames: Dual-objective piano transcription," *arXiv preprint arXiv:1710.11153*, 2017.

[4] Jong Wook Kim and Juan Pablo Bello, "Adversarial learning for improved onsets and frames music transcription," *arXiv preprint arXiv:1906.08512*, 2019.

[5] Qiuqiang Kong, Bochen Li, Xuchen Song, Yuan Wan, and Yuxuan Wang, "High-resolution piano transcription with pedals by regressing onset and offset times," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3707–3717, 2021.

[6] Curtis Hawthorne, Ian Simon, Rigel Swavely, Ethan Manilow, and Jesse Engel, "Sequence-to-sequence piano transcription with transformers," *arXiv preprint arXiv:2107.09142*, 2021.

[7] Josh Gardner, Ian Simon, Ethan Manilow, Curtis Hawthorne, and Jesse Engel, "Mt3: Multi-task multitrack music transcription," *arXiv preprint arXiv:2111.03017*, 2021.

[8] Ethan Manilow, Prem Seetharaman, and Bryan Pardo, "Simultaneous separation and transcription of mixtures with multiple polyphonic and percussive instruments," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 771–775.

[9] Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck, "Enabling factorized piano music modeling and generation with the maestro dataset," *arXiv preprint arXiv:1810.12247*, 2018.

[10] Ethan Manilow, Gordon Wichern, Prem Seetharaman, and Jonathan Le Roux, "Cutting music source separation some slakh: A dataset to study the impact of training data quality and quantity," in *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2019, pp. 45–49.

[11] Ondřej Cífka, Umut Şimşekli, and Gaël Richard, "Groove2groove: one-shot music style transfer with supervision from synthetic data," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2638–2650, 2020.

[12] Kristy Choi, Curtis Hawthorne, Ian Simon, Monica Dinculescu, and Jesse Engel, "Encoding musical style with transformer autoencoders," 2019.

[13] Wei Tsung Lu, Li Su, et al., "Transferring the style of homophonic music using recurrent neural networks and autoregressive model.," in *ISMIR*, 2018, pp. 740–746.

[14] OpenAI, "Musenet," `https://openai.com/blog/musenet`, 2019.

[15] Shih-Chuan Chiu, Man-Kwan Shan, and Jiun-Long Huang, "Automatic system for the arrangement of piano reductions," in *2009 11th IEEE International Symposium on Multimedia*. IEEE, 2009, pp. 459–464.

[16] Sho Onuma and Masatoshi Hamanaka, "Piano arrangement system based on composers' arrangement processes.," in *ICMC*, 2010.

[17] Meinard Müller, Yigitcan Özer, Michael Krause, Thomas Prätzlich, and Jonathan Driedger, "Sync toolbox: A python package for efficient, robust, and accurate music synchronization," *Journal of Open Source Software*, vol. 6, no. 64, pp. 3434, 2021.

[18] Colin Raffel, Brian McFee, Eric J Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, Daniel PW Ellis, and C Colin Raffel, "mir_eval: A transparent implementation of common mir metrics," in *In Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR*. Citeseer, 2014.

[19] Romain Hennequin, Anis Khlif, Felix Voituret, and Manuel Moussallam, "Spleeter: a fast and efficient music source separation tool with pre-trained models," *Journal of Open Source Software*, vol. 5, no. 50, pp. 2154, 2020, Deezer Research.

[20] Brian McFee, Alexandros Metsai, Matt McVicar, Stefan Balke, Carl Thomé, Colin Raffel, Frank Zalkow, Ayoub Malek, Dana, Kyungyun Lee, Oriol Nieto, Dan Ellis, Jack Mason, Eric Battenberg, Scott Seyfarth, Ryuichi Yamamoto, viktorandreevichmorozov, Keunwoo Choi, Josh Moore, Rachel Bittner, Shunsuke Hidaka, Ziyao Wei, nullmightybofo, Adam Weiss, Darío Hereñú, Fabian-Robert Stöter, Pius Friesch, Matt Vollrath, Taewoon Kim, and Thassilo, "librosa/librosa: 0.9.1," Feb. 2022.

[21] Matthias Mauch and Simon Dixon, "Pyin: A fundamental frequency estimator using probabilistic threshold distributions," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 659–663.

[22] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *arXiv preprint arXiv:1910.10683*, 2019.

[23] Noam Shazeer and Mitchell Stern, "Adafactor: Adaptive learning rates with sublinear memory cost," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4596–4604.

[24] Ziyu Wang, Ke Chen, Junyan Jiang, Yiyi Zhang, Maoran Xu, Shuqi Dai, Xianbin Gu, and Gus Xia, "Pop909: A pop-song dataset for music arrangement generation," *arXiv preprint arXiv:2008.07142*, 2020.