PIANO COVER GENERATION

Chih-Pin Tan
National Taiwan University
tanchihpin0517@gmail.com

ABSTRACT

Cover song generation, which aims to create a piano cover from a pop song, stands out as a popular way of music making in the music-creative community. In this study, we introduce Piano Cover Generation (PiCoGen) 1 & 2.

PiCoGen1 is a two-stage approach for automatic cover song generation that transcribes the melody line and chord progression of a song given its audio recording, and then uses the resulting lead sheet as the condition to generate a piano cover in the symbolic domain. This approach is advantageous in that it does not required paired data of covers and their original songs for training. Compared to an existing approach that demands such paired data, our evaluation shows that PiCoGen1 demonstrates competitive or even superior performance across songs of different musical genres.

While PiCoGen1 shows certainly successes, however, there are some problems from the two-stage approach. For example, the transcription of melody and chord makes losses of other musical information such as rhythm and vibe. Therefore, we plan to build a piano cover generation system by hiring end-to-end approach. In the survey of recent studies, we notice existing approaches mainly employ supervised learning and the training demands strongly-aligned and paired song-to-piano data, which is built by remapping piano notes to song audio. This would, however, result in the loss of piano information and accordingly cause inconsistencies between the original and remapped piano versions. To overcome this limitation, we propose PiCoGen 2, a transfer learning approach that pre-trains our model on piano-only data and fine-tunes it on weakly-aligned paired data constructed without note remapping. During pre-training, to guide the model to learn piano composition concepts instead of merely transcribing audio, we use an existing lead sheet transcription model as the encoder to extract high-level musical information from the piano recordings. The pre-trained model is then fine-tuned on the paired song-piano data to transfer the learned composition knowledge to the pop song domain. As demonstrated in our evaluation, this training strategy enables our model to achieve high-quality results, outperforming baselines on both objective and subjective metrics across five music genres.

1. PICOGEN 1

Cover song generation, recreating or rearranging the musical elements from an existing piece, is popular within the music-creative community. Musicians craft a cover song with careful consideration of musical components such as melody, chords, rhythm, and performance techniques, in a style or musical genre that may be different from the original piece (e.g., Pop to Jazz), and with different sets of instruments (e.g., using only the piano or the guitar).

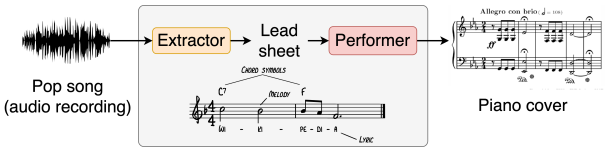

Figure 1. An overview of PiCoGen 1. The model generates a piano cover in two stages: extracts firstly a lead sheet (i.e., melody line and chord progression) from an audio recording of the original song via audio analysis (i.e., transcription), and then turns the extracted lead sheet into a piano performance via conditional symbolic-domain music generation.

Attempts have also been made to create cover songs automatically, relying on a supervised approach trained on paired data of songs and their human-made cover versions. Takamori *et al.* [1] employ a regression model with acoustic features and structural analysis to generate piano covers. Song2Guitar [2] uses a hidden Markov model to generate guitar covers with fundamental frequency (f0), beat, and chord information. More recently, Choi *et al.* present the Pop2Piano model [3], which represents the first attempt that uses deep learning for cover song generation. Specifically, it leverages the MT3 architecture [4], an encoder-decoder Transformer designed to convert an audio signal into a sequence of discrete tokens, to create piano covers of music. In collecting paired data needed for model training, Choi *et al.* find it important to employ heuristics to improve the data quality, involving a matching routine to identify covers of songs from the Internet, and an alignment routine to synchronize the songs and their covers.

To get rid of the reliance on paired data and to supply deep learning models with more training data, we explore instead in this paper a novel approach for piano cover generation that does not need paired data at all. Our approach draws inspiration from musical domain knowledge that a cover and its original song tend to share a common melody line and chord progression, and therefore uses the symbolic-domain *lead sheet* [5, 6] (i.e., a MIDI-like representation of melody+chord) [1] as the intermediate gateway

[1] The lead sheet is strongly linked to the f0 contours and chroma fea-

transferring a musical audio involving arbitrary instrumentation into a piano-only rendition. Specifically, as depicted in Figure 1, we propose a two-stage model that firstly extracts the lead sheet directly from the song audio, and then generates a piano cover given the extracted lead sheet. We refer to the resulting model as **Pi**ano **Co**ver **Gen**eration (PiCoGen).

The previous Pop2Piano model [3] uses a *piano transcription*-like approach and creates piano covers with a *single* stage of audio-to-symbolic conversion with no explainable intermediate representation, trained using paired data of original music and piano covers. In contrast, PiCoGen bypasses the need of such a paired data by decomposing the conversion task into *two* stages. PiCoGen needs other two types of paired data, one for each stage, yet both of which have been available in the research community through efforts on audio-to-symbolic lead sheet transcription [10–12] and symbolic-to-symbolic conditional piano generation [6, 13].

As its name implies, Pop2Piano is evaluated on Pop music only [3]. However, we set forth to investigate how well PiCoGen compares to Pop2Piano on generating the piano covers for up to ten different musical genres, including cases where lead sheet transcription can be difficult. This is to study to which extent the domain knowledge (or assumptions) incorporated by PiCoGen is adequate, and to evaluate the generalizability of both Pop2Piano and PiCoGen.

We invite readers to our project website [2] for audio examples of the generated piano covers and the source code of this project.

2. BACKGROUND

Piano cover generation is a special class of cover song generation where the target output is a piano-only rearrangement of a given music piece. It is expected that the generated cover and the original piece can be identified as different versions of the same music composition, and that the cover itself is pleasant to listen to.

Due to the availability of high-quality piano synthesizers, piano cover generation can be approached via generating piano music in a symbolic representation such as Musical Instrument Digital Interface (MIDI), instead of generating audio signals directly. This is the approach taken by Pop2Piano and similarly this work. As such, piano cover generation is related to the following three tasks.

Automatic Music Transcription (AMT) involves converting music signals into symbolic representations such as piano rolls. *Automatic piano transcription*, for example, aims to generate a musical score-like transcription of all the notes involved in an audio recording of piano performance [4, 14–18]. *Lead sheet transcription*, as another instance of AMT, transcribes the notes corresponding to the melody line and recognizes the chord names involved in the harmonic progression [10–12]. Pop2Piano adopts MT3 [4], the state-of-the-art of piano transcription, as their model backbone as both piano transcription and piano cover generation convert audio into piano scores. A main difference here is that MT3 assumes its input to be audio recordings of piano-only music, so Pop2Piano needs paired data of {original songs, piano covers} for supervised training to cope with input with arbitrary instrumentation.

Symbolic-domain music generation aims to generate novel pieces of music in a symbolic format, usually by representing a music piece as a sequence of discrete tokens [6, 13, 19–23]. The generation process can be either *unconditional* (i.e., from-scratch generation) or *conditional*. A key observation of our work is that piano cover generation can be viewed as a conditional generation task—generating token sequences of piano performances conditioned on pre-given lead sheets. Such a symbolic-to-symbolic conditional piano generation problem has been tackled before [6, 13].

Music style transfer involves the process of altering the style of a music piece to match the style of a provided example, influenced by various attributes such as orchestration, chord progression, and tonality [24–26]. Cover song generation can be considered as a specific type of music style transfer in general.

3. METHODOLOGY

Given an audio segment A of a song, cover song generation establishes a model $f : A \rightarrow S$ that creates an alternative version of the input. For piano cover generation, the output S is pure piano music, which can be represented in the so-called symbolic domain with discrete tokens bearing explicit musical meaning [27]. While many token representations for symbolic music have been proposed in the literature [28], Pop2Piano adopts the *MIDI-like* representation [19], which uses tokens that indicate the pitch, onset time, offset time, and velocity (which is related to perceptual loudness) of each note involved in a piano playing. Pop2Piano can therefore be viewed as a cross-domain sequence-to-sequence model converting a segment of continuous audio waveform to a sequence of discrete tokens.

PiCoGen is different from Pop2Piano mainly in two aspects: the model architecture (two-stage vs. single-stage), and the adopted token representation for symbolic music. We provide details below.

3.1 Proposed Two-stage Model

PiCoGen decomposes f into two steps, $f = e \circ g$. The first step $e : A \rightarrow L$ converts the audio input into an intermediate representation L, while the second step $g : L \rightarrow S$ generates a cover based only on L. In other words, this two-stage approach assumes A and S are conditional independent given L. As shown in Figure 1, a fundamental assumption of PiCoGen is that the lead sheet can serve as such an intermediate representation. When A and S share the same underling lead sheet, we assume that they would sound like the same song. From a style transfer viewpoint,

tures of music, both of which play significant roles in cover song identification [7–9].

[2] https://tanchihpin0517.github.io/PiCoGen

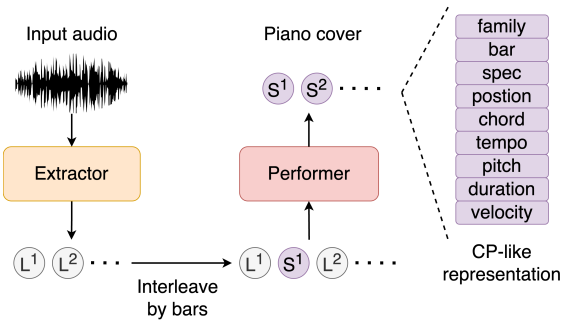

Figure 2. For each bar (musical measure) k of the input, the Extractor transcribes from the input its lead sheet L^k (a token sequence), and the Performer generates autoregressively the piano performance S^k (also a token sequence) for the same bar given the current and preceding sequences of lead sheet $[L^1, L^2, \dots L^k]$ and the preceding piano performances $[S^1, S^2, \dots S^{k-1}]$ organized in an interleaving fashion.

lead sheet is regarded as the "content" to be reserved. Unlike style transfer in general, the content here is a human-readable lead sheet, facilitating interpretable and controllable music generation.

As shown in Figure 1, PiCoGen accordingly contains two sub-models, the *Extractor* that performs e and outputs the corresponding lead sheet sequence L of the input audio A, and the *Performer* that performs g and generates piano token sequence S based on the transcribed lead sheet. Here e and g can be viewed as two sequence-to-sequence problems. In both cases, the objectives are to maximize the probability of the target sequence given the source sequence.

To better inform the Performer of the temporal correspondence between the lead sheet L and the piano S, we follow the approach of Wu & Yang [6] and implement the Performer as a decoder-only Transformer that deals with an interleaved sequence composed of a "bar-wise mix" of L and S, as shown in Figure 2. Specifically, we segment L and S respectively into multiple sub-sequences $L = [L^1, \dots, L^B]$ and $S = [S^1, \dots, S^B]$, where L^k and S^k are the lead sheet sub-sequence and piano sub-sequence for the same, k-th bar (i.e., musical measure) of the input music, and B denotes the number of bars. We use the bar-wise mix $[L^1, S^1, \dots, L^B, S^B]$ to train the Performer, such that the generation for the k-th bar of piano S^k would depend on (i.e., can attend to) the current and preceding sub-sequences of lead sheet $[L^1, L^2, \dots L^k]$ and the preceding piano sub-sequences $[S^1, S^2, \dots S^{k-1}]$.

3.2 Proposed Token Representation

A known weakness of the *MIDI-like* tokens [19] used by Pop2Piano to represent piano music S is the overly large number of tokens needed to represent a musical bar, making it hard to learn long-term dependency [13]. We instead adopt a modified version of compound-word (CP) token representation [13] to make the token sequence compact. The main idea of CP is to group related tokens into a "super token." As shown in Figure 2, tokens belonging to the same super token are combined into a single embedding vector before feeding to the Transformer, which in turn predicts the tokens belonging to the next super token collectively at the same time using different heads. We refer to [13] for details.

Specifically, CP uses the idea of "token classes" to organize the tokens [13]. While the original CP considers only two token classes, metric and non-metric, we consider four token classes so as to have a unified CP-like representation for not only the piano S but also the lead sheet L. They are Spec, Bar, Metric and Note. Spec class includes special tokens, e.g., `[BOS]` and `[EOS]` (i.e., beginning or ending of a sequence). Bar class contains `[bar_src]` and `[bar_tgt]`, which are used to distinguish tokens belongs to lead sheet or piano cover. Metric class is composed with three sub-classes: Position, Tempo, and Chord. Note class is composed with three sub-classes describing a musical note: Pitch, Duration and Velocity. [3]

3.3 Implementation Details

To implement the Extractor of PiCoGen, we employ SheetSage [12], the state-of-the-art for lead sheet transcription. Authors of SheetSage have kindly released a model checkpoint [4] well-trained on the Hook Theory dataset [12], which consists of approximately 40,000 paired data of audio clips and lead sheets. We simply leverage this checkpoint as our Extractor without fine-tuning.

As for the Performer, we adopt the decoder-only architecture of the CP Transformer [13] and train it from scratch. The model has 8 self-attention layers, 8 heads for multi-head attention, 512 hidden dimensions, a sequence length of 1,024 super tokens, and GeLU as the activation function. At inference time, the lead sheet L obtained from the Extractor is converted to the proposed CP-like tokens and then fed to the Performer to generate the piano cover S.

4. EXPERIMENTAL SETUP

4.1 Datasets

Training the Performer requires paired data of lead sheets $\mathcal{D}_l$ and piano performances $\mathcal{D}_p$. For $\mathcal{D}_p$, we adopt the Pop1k7 dataset [13], which encompasses around 1,700 piano covers of various Japanese anime, Korean popular and Western popular songs. To construct $\mathcal{D}_l$, we follow the approach of Wu & Yang [6] and employ symbolic-domain music analysis techniques [29] to recognize the lead sheet from each piano performance.

For model evaluation, we use the GTZAN dataset [30], [5] a well-known public-domain dataset containing

[3] Position indicates the timing shift from the beginning of each bar, in the resolution of 16th note. Tempo indicate the beat-per-minute (BPM), ranging from 32 to 244. Chord is used to represent the chord condition, composed by {chord root, chord quality} pair. Pitch ranges from A0 to C8. Duration indicates the length of each notes, with the same resolution of Position. We limit the maximum duration to the length of 8 quarter notes in this work. Velocity takes MIDI value from 0 to 127.

[4] `https://github.com/chrisdonahue/sheetsage`

[5] `http://marsyas.info/download/data\_sets`

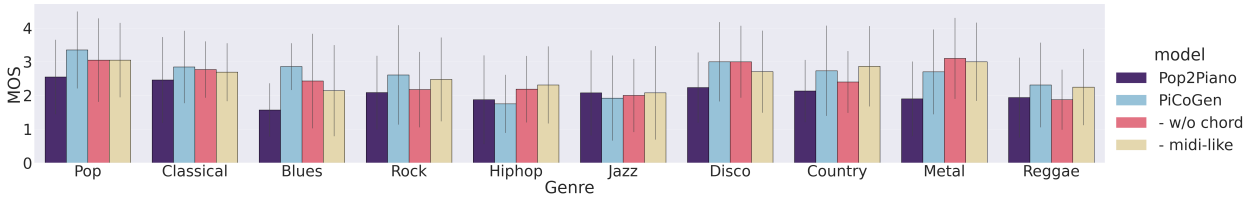

Figure 3. Mean opinion score in the metric OVL in the subjective evaluation, for each of the ten considered genres.

100 30-second song clips for each of 10 selected genres, including Pop, Jazz, Classical, Rock, etc (see Figure 3 for the full list), enabling evaluation beyond Pop music. Note that SheetSage assumes constant tempo and would report errors for songs that violate this assumption. We therefore discard the 117 songs of GTZAN that has this problem, using the remaining 883 songs in our evaluation.

4.2 Baseline and Ablations

We consider Pop2Piano [3] as the baseline, as it represents the state-of-the-art for piano cover generation. Specifically, we use the model checkpoint released by the original authors for testing. [6] Moreover, we train and evaluate two ablated versions of PiCoGen. The first version, denoted as "w/o chord", uses only the melody as the intermediate representation L, namely dropping chord information entirely. This is to study the effect of chords. The second version, denoted as "midi-like," uses the MIDI-like tokens [19] to represent the piano music, instead of the more advanced CP-like representation. This is to use the same token representation as Pop2Piano [3] so as to examine whether any possible performance difference between Pop2Piano and PiCoGen is due to the two-stage architecture or due to token representations. We note that these ablations involve changes only to the intermediate data, and that both ablations adhere to the proposed two-stage strategy.

4.3 Evaluation Metrics

For objective evaluation, we follow Pop2Piano and compute melody chroma accuracy (MCA) [3]. The idea is to compare the top line of the piano MIDI generated by a model with the melody contour of the input audio. According to Pop2Piano [3], the melody contour is computed by separating the vocal of the input via a source separation model [31], and then using pYIN [32] for f0 estimation from the separated vocal. MCA is $\in [0, 1]$ and the higher the better.

For subjective evaluation, we conduct an online user study involving 30 volunteers consisting of 10 amateurs, 17 pro-ams [33], and 3 pros. We generate 31 distinct testing sets, distributing them to the human subjects according to their birth dates. In each testing set, we randomly select 5 genres from the GTZAN dataset and designate one song per genre as the testing target. Consequently, each testing set comprises 5 songs, and we generate 4 results for each song, using Pop2Piano, PiCoGen and the two ablation models, presented to the subject in random order. Subjects are instructed to score the results with a Likert scale from 1 to 5 (the higher the better) in the following five aspects:

- **Similarity (SI)**: How much the piano cover sounds similar to its original song in general?
- **Melody similarity ($\mathbf{SI_m}$)**: The perceived similarity between the melody lines of the piano cover and the original song.
- **Chord similarity ($\mathbf{SI_c}$)**: The similarity in chord progression.
- **Music fluency (FL)**: Does the piano cover sound fluent?
- **Overall (OVL)**: How much do you like the piano cover?

5. RESULTS AND DISCUSSION

Table 1 displays the average MCA and mean opinion scores (MOS) from the user study for Pop only, the genre targeted by Pop2Piano. We see that the proposed PiCoGen and its variants show a lower MCA than Pop2Piano, but outperform it greatly in every subjective metric. Although statistical test does not reveal significant performance difference (due to large stanadard deviation), we view this as an encouraging result since PiCoGen does not use paired data of music and their piano covers at all. Moreover, although the two ablated variants receive moderately lower MOS than PiCoGen, the result is in general comparable, providing further evidences the effectiveness of the two-stage approach. We also note that the MOS in OVL reduces from 3.35 to 3.05 when dropping chords from the intermediate representation, suggesting the superiority of using lead sheet instead of melody only in PiCoGen.

Table 1 also suggests a conflicting result between MCA and its subjective counterpart $\mathrm{SI_m}$. PiCoGen underperforms Pop2Piano in MCA, but its MOS in $\mathrm{SI_m}$ is higher. We conjecture that this might due to the fact that MCA only considers vocal melody, neglecting the melody of the leading instrument, as discussed in [34]. We consider the development of better objective metrics as a future work and focus on the result of the user study here.

Figure 3 shows the MOS in OVL for all the ten genres in GTZAN, extending our evaluation beyond Pop music. Figure 3 shows that PiCoGen outperforms Pop2Piano

[6] https://github.com/sweetcocoa/pop2piano

Model	*objective*	*subjective evaluation* ↑				
	MCA ↑	SI	SI_m	SI_c	FL	OVL
Pop2Piano [3]	**0.25**	2.65	2.85	2.65	2.60	2.55
PiCoGen	0.17	**3.20**	**3.35**	**3.05**	**3.20**	**3.35**
–w/o chord	0.14	3.10	3.25	2.95	3.10	3.05
–midi-like	0.12	2.95	3.10	2.95	3.10	3.05

Table 1. Evaluation result on piano cover generation for only Pop music. Best result per metric highlighted in bold.

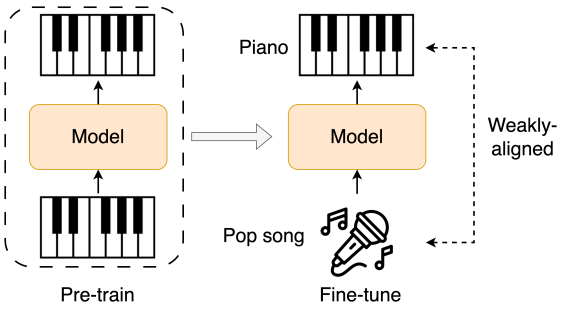

Figure 4. The proposed model is trained with two stages: firstly pre-trained on piano-only data and then fine-tuned on the weakly-aligned song-to-piano pairs.

not only for Pop but also for genres such as Blues, Disco, and Metal, demonstrating the generalizability of PiCoGen to some extent. However, there are also many genres for which Pop2Piano and PiCoGen have similar MOS. In particular, both Pop2Piano and PiCoGen perform poorly for Hip-hop music. Upon scrutinizing the attributes of the testing songs, we identify that the quality of the lead sheets extracted by SheetSage is not satisfactory for some of the genres, especially for Hip-hop. Moreover, the MOS of PiCoGen in Table 1 and Figure 3 ranges from 2.00 to 3.50 in general, leaving considerable room for future improvement given that the scores are from 1 to 5.

There are many interesting avenues to be investigated in future work. First, both Pop2Piano and PiCoGen can be potentially improved by collecting a larger training dataset encompassing more genres. However, we note that Pop2Piano demands both the piano covers and their original songs in diverse genres as the paired data for training. In contrast, it is easier to expand the training data of PiCoGen as we only need the piano covers (not the audio recordings of the original songs). Second, besides data, better result for PiCoGen might be obtained along with advances in lead sheet transcription, and by incorporating intermediate representations of music other than the lead sheet (e.g., rhythmic elements) for musical genres with no dominant melody line. Finally, it might also be interesting to develop a model that combines the strengths of the music-informed design of PiCoGen and the data-driven design of Pop2Piano, leveraging both unpaired and paired data for training.

6. PICOGEN 2

Piano cover generation, which involves recreating or arranging an existing music piece as a new piano version, is popular within music-creative communities and the music production industry. On media sharing sites like YouTube, piano cover creators often have millions of subscribers. Additionally, many music producers create and distribute piano arrangements on music streaming platforms.

In the field of music information retrieval (MIR), attempts have been made to automatically generate piano covers from existing musical pieces. Takamori *et al.* [1] proposed a regression method to generate piano reductions, which can be considered simplified versions of piano covers, utilizing acoustic features and structural analysis of the input music. With the recent surge in deep learning, Choi *et al.* [3] introduced a model named Pop2Piano that tackles piano cover generation by leveraging the concept of piano transcription and employing the MT3 architecture [4], originally designed for transcription, as their model backbone. They collected pop songs and the corresponding piano covers from the Internet and built a song-piano synchronized dataset by "remapping" the piano notes to the song audio with a warping algorithm (thereby modifies, or warps, the piano cover). The algorithm entails evaluating the similarity between the pitch contour of the vocal signal extracted from the song audio with the top line of the piano MIDI. They then trained the model with the synchronized data, guiding the model to learn the pitch and onset/offset timing of each note in the generated piano cover.

However, by scrutinizing the statistics in the ratio of audio length difference and tempo difference (see Table 2) of the song-to-piano paired data they collected, [7] we identify a gap between transcription and cover generation, since the difference ratio should be equal to 1.00 if the piano cover and its original song are temporally aligned to each other. Forcing the piano cover to be synchronized with its original song is unreasonable even they are made to express the same high-level musical ideas [35]. After digging deeply into the process of building synchronized song-to-piano dataset of Pop2Piano, we notice that the note-remapping process—i.e., adjusting piano note timing according to the time mapping function obtained by synchronizing piano notes to the song audio—breaks the relation of original piano notes and thereby incurs loss of piano information. Moreover, from a musical perspective, the way human cre-

[7] https://github.com/sweetcocoa/pop2piano/blob/main/train_dataset.csv

duration deviation	tempo deviation	IOI deviation
1.10 ± 0.12	1.16 ± 0.25	1.14 ± 0.17

Table 2. Some statistics probing the song-to-piano paired data of Pop2Piano [3]. The first two contrast the original songs with their piano covers, evaluating the length of the **duration** (in seconds) of the longer one divided by that of the shorter one, and similarly the deviation ratio in **BPM**. The last statistic is similarly the deviation ratio in terms of the average inter-onset intervals (**IOIs**; in seconds), but between the original and adjusted piano performances.

ates piano covers is inherently different from the way human transcribes music. For cover generation, musicians may firstly analyze the original song in aspects such as melody, chord progression and rhythm section, then decide how to interpret the original song with their composition knowledge, and finally make the piano cover based on the piano performance techniques.

Inspired by the process of human composition for piano cover songs, we propose in this paper a novel approach for piano cover generation by involving the concept of transfer learning [36]. Instead of relying on the *strongly-aligned* pairs [37] that involves note-remapping, we use *weakly-aligned* data with the correspondence in "beat" level between song-piano pairs. This approach introduces no modification to the piano covers, retaining their musical quality. Besides, to mitigate the inaccuracy of data alignment, the model is pre-trained on piano-only data to learn the concept of piano performance first and then fine-tuned on the weakly-aligned paired data to learn the conversion of song to piano. We also employ a prior model pre-trained for lead sheet transcription [12] as the encoder component of our model to help the model learn high-level musical concepts during pre-training.

We compare our model with other baselines in the evaluation to validate the effectiveness of the weak-alignment for pairing and the two-steps training strategy. The result shows that our model outperforms not only in objective metrics but also in user study.

7. BACKGROUND

Piano arrangement, i.e., the process of reconstructing and reconceptualizing a piece, can refer to various conditional music generation tasks, including lead sheet [8] -conditioned accompaniment generation, transcription and reorchestration, and piano reduction [38–40]. Beyond arrangement, piano cover generation involves creating new musical elements and modifying the original elements via improvisation, tempo changes, stylistic shifts, etc. Therefore, piano cover generation is related to the following topics.

Symbolic-domain music generation refers to generating a music piece represented in a symbolic form such as pianorolls and discrete MIDI (Musical Instrument Digital Interface) or REMI-like token [20], rather than audio signals. Looking more closely, this task can be classified as either unconditional generation (i.e., from-scratch generation) or conditional generation. While the objective of piano cover generation is to generate piano audio given a song audio input, we can regard it as a conditional symbolic-domain music generation task because we can generate piano in the MIDI domain first and then use off-the-shelf high-quality piano synthesizers to convert it into audio.

Automatic music transcription (AMT), which aims to precisely transcribe music content from audio signals into a symbolic representation over time, has a close relation to piano cover generation. AMT tasks can be categorized based on the completeness of information captured from the input audio. One category of AMT tasks aims to capture all music content presenting in the audio, such as automatic piano transcription [4, 14–18]. These methods transcribe the complete polyphonic piano performance from the audio signal. Another category focuses on transcribing a reduced representation of the input, like melody transcription [41, 42] and lead sheet transcription [10–12]. These tasks extract only the lead melody line and chord progressions, representing a sparse subset of the full musical content. Piano cover generation also requires the exploration of music content reduction and additionally relies on generative modeling conditioned on the reduced representation. For instance, Pop2Piano uses MT3 [4] as its backbone to convert audio features into a symbolic piano performance representation. Following the paradigm of transcription approaches, Pop2Piano requires paired data consisting of pop songs and their corresponding temporally-synchronized piano cover.

Transfer learning is generally consider as the concept of adopting the model to the target domain by re-using parameters that are trained on a source domain, thereby transferring the knowledge between the domains [43]. There have been several works on transfer learning in the field of MIR, e.g., music classification [44–46] and music recommendation [47, 48]. However, to our best knowledge, there is no previous attempt that applies transfer learning to the task of cover song generation.

8. METHODOLOGY

Piano cover generation can be defined as a sequence-to-sequence (seq2seq) problem, since we view it as a form of conditional symbolic-domain music generation. The objective is to generate a sequence of symbolic tokens Y representing the piano performance, conditioned on the input audio X of the original song.

8.1 Weakly-Aligned Data

In Pop2Piano, Choi *et al.* [3] propose a data preprocessing algorithm to synchronize the piano MIDI to the song audio. They utilize SyncToolBox [49] to analyze the chroma features of two audio segments to obtain a warping path of mapping the time from the piano performance to the song audio. Based on the analysis, they adjust the tim-

[8] A music notation consisting of lead melody and chord progression.

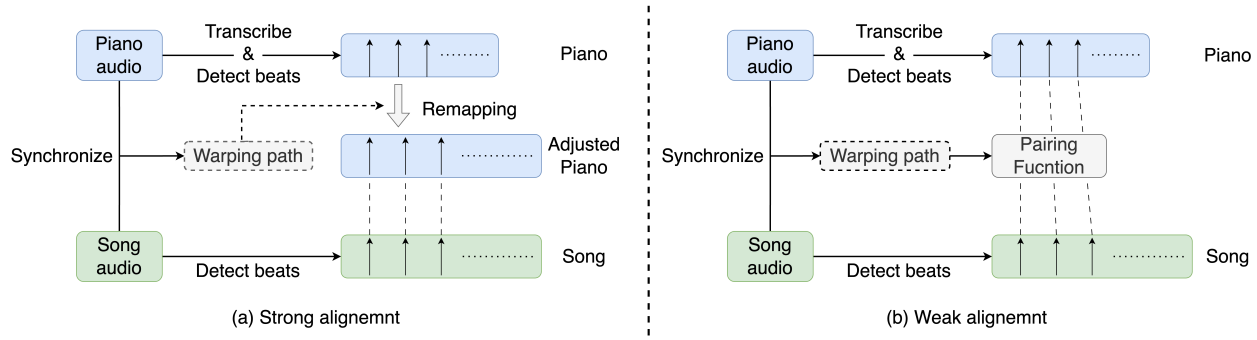

Figure 5. (a) Pop2Piano constructs strongly-aligned data by remapping piano notes to the song audio, altering the piano content. (b) We use weakly-aligned song-to-piano pairs without note-remapping, leavning the piano content intact.

ing of notes transcribed from the piano performance by using a linear mapping function calculated from the temporal warping information. These remapped notes is then quantized to align with the beat locations of the song audio. However, the rhythmic distortion caused by note-remapping is practically unavoidable, even disregarding the inaccuracy of the synchronization process. The chroma feature only reflects a rough overall alignment between the piano performance and song audio which cannot precisely describe the nuanced amount of timing shift for each individual note. This is evident when examining the changes in the inter-onset intervals (IOIs) between the original piano notes and the remapped version, shown in Table 2.

To avoid the rhythmic distortion of note-remapping, we propose a weak-alignment approach without any changes to the timing of piano notes. The idea is to let the alignment rely on only the *beats* of each song-to-piano pair. As shown in Figure 5, we construct the time mapping function F_{map} by computing the warping path for the audio pair like the way done in Pop2Piano. Given a time of piano performance t_p, the function outputs the corresponding time of song audio $t_s = F_{map}(t_p)$ according to the temporal warping information. Specifically, we detect the beat locations with Beat Transformer [50] to get the beat times $Q^p = [q_1^p, \ldots, q_{l_p}^p]$ of the piano performance and $Q^s = [q_1^s, \ldots, q_{l_s}^s]$ of the song audio, where l_p and l_s denote the number of beats of each of them. Then we define an aligning function F_{align} as:

$$F_{align}(i) = \arg\min_j (F_{map}(q_i^p) - q_j^s). \quad (1)$$

For any beat index $i \in [1, ..., l_p]$ of the piano performance, the aligning function outputs the beat index $j \in [1, ..., l_s]$ of the song audio, where q_j^s is the nearest beat time of $F_{map}(q_i^p)$. We consider a song-piano pair to be weakly-aligned if the correspondence between them is determined by the proposed aligning function F_{align}.

8.2 Model

An aerial view of our model is depicted in Figure 6. We employ a decoder-only Transformer to accept an input sequence bundling condition X (song audio) and target Y (piano performance) together, and generates the output tokens for Y autoregressively. This approach of providing

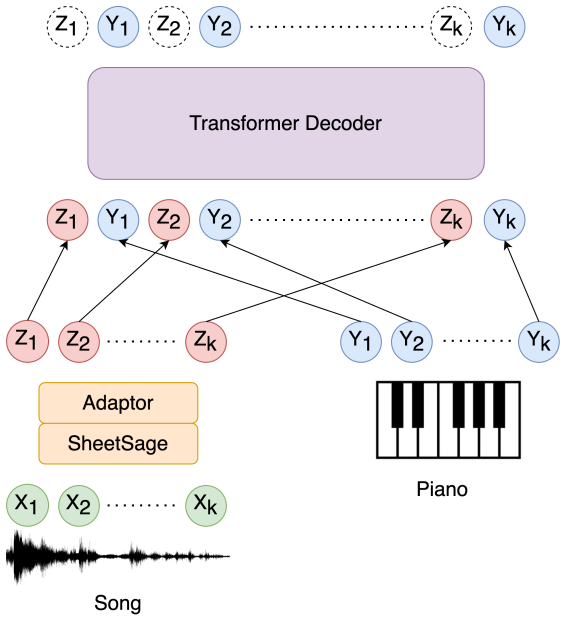

Figure 6. A model overview. The song audio X is encoded to a latent embedding Z, and the decoder accepts the bar-mixed sequence of Z and Y for next-token prediction.

both the condition and target as a bundled input sequence to the Transformer has been applied in previous studies [6,13] and has shown success in better informing the model of the temporal correspondence between the condition and desired output. We divide Y into *bars* with the detected beat information and get $Y = [Y^1, \ldots, Y^{B_p}]$, where B_p is the number of bars in the piano cover, and there exists an song audio sequence $X = [X^1, \ldots, X^{B_p}]$ for Y, where each sub-sequence X^k is weakly aligned to Y^k. We then rearrange them with an interleaving form and train the decoder with the bar-wise mix $S = [X^1, Y^1 \ldots, X^{B_p}, Y^{B_p}]$. The decoder model would learn to generate k-th bar of piano performance Y^k depending on (i.e., can attend to) the current and preceding sub-sequences of song audio $[X^1, \ldots, X^k]$ and the preceding sub-sequences of piano performance $[Y^1, \ldots, Y^{k-1}]$.

To reduce the sequence length of X and extract better musical information, we employ a prior audio encoder to transform X into an intermediate representation Z. Differ-

ent from those works which use Mel-spectrograms [4] or audio codecs [51] for Z, we use SheetSage [12], which is trained for the task of lead sheet transcription, headed with an neural adapter as the prior audio encoder. We consider the output embedding of SheetSage more suitable for piano cover generation, since the lead sheet is often regarded as an important musical semantics bridging a piano cover and its original song. With the prior encoder, the song audio $[X^1, \ldots, X^{B_p}]$ is transformed into the latent embedding $[Z^1, \ldots, Z^{B_p}]$ before passed to the decoder, resulting in the input sequence $[Z^1, Y^1 \ldots, Z^{B_p}, Y^{B_p}]$ of the decoder, as illustrated in Figure 6.

8.3 Transfer Learning

While the weak-alignment approach eliminates inner temporal distortions for piano performance, there can still be alignment errors between the piano segments and their corresponding song segments. This is because a piano cover is not guaranteed, in the beat level, to have a strict one-to-one mapping with the original song.

To prevent the model from being affected by the alignment errors, we propose a training strategy that applies the concept of transfer learning to the training process. We divide the training into two steps: pre-training and fine-tuning. In the *pre-training* stage, we train the model with an input sequence $\bar{S} = [\bar{Y}^1, Y^1, \ldots, \bar{Y}^{B_p}, Y^{B_p}]$ where $\bar{Y}$ is the original piano audio recording of the symbolic piano tokens Y. The same as the song audio, the original recording $\bar{Y}$ is encoded to an inter-representation $\bar{Z}$ by the prior encoder. We expect the model to learn to generate piano performances Y with high-level musical features extracted by SheetSage from the piano audio $\bar{Y}$, rather than merely detecting note onsets/offsets like in a piano transcription task. Importantly, there will be no alignment errors between Y and $\bar{Y}$, ensuring that the model can firstly learn the complete concept of piano composition and generation in the pre-training stage, without being impeded by cross-domain alignment issues.

In the *fine-tuning* stage, we train the model with the mixture of $\bar{S}$ and S. Following [26, 51, 52], we train the model with the objective of minimizing the cross entropy loss on the tokens of piano performance Y. Let L_1 and L_2 stand for the cross entropy losses for $\bar{S}$ and S, respectively. The loss L_p in the pre-training stage and the loss L_f in the fine-tuning stage can be writeen as:

$$\begin{aligned} L_p &= L_1\,, \\ L_f &= \alpha \cdot L_1 + (1-\alpha) \cdot L_2\,, \end{aligned} \tag{2}$$

where α is the weighting factor determining the proportion of losses contributed from $\bar{S}$ and S during fine-tuning. We expect that α helps the model retain the knowledge about piano performance learned from the pre-training stage.

8.4 Data Representation

We adopt a modified version of REMI token representation [20], which is designed to generate high-quality pop piano performances, for the piano performance sequences Y. Our representation consists of 7 token classes. `Spec` contains special tokens such as `[bos]` (beginning-of-sentence) and `[ss]` (song-start) for controlling the model behavior. `Bar` indicates the property of each bars. `Position`, `Chord` and `Tempo` are metric-related tokens for 16th-note offsets within bars, chord changes (11 roots $\times$ 12 qualities), and tempo changes (64 levels). `Pitch`, `Duration` and `Velocity` are note-related tokens for note pitches (A0 to C8), durations (1 to 32 16th-notes), and note velocities (32 levels). There are in total 428 tokens in the vocabulary. In our implementation, `[Bar_start]` and `[Bar_end]` always occur at the start and end of each bar in the input sequence S and $\bar{S}$.

9. EVALUATION

9.1 Dataset

We follow the instructions provided in the Pop2Piano source code to rebuild the training dataset, collecting 5,844 pairs of pop songs and their corresponding piano covers from the Internet. We filter out song pairs with a melody chroma accuracy (MCA) [53] lower than 0.05 or an audio length difference exceeding 15%, leaving 5,503 remaining pairs. In the pre-training stage, all the piano performances from these remaining pairs are used for training. In the fine-tuning stage, we remove invalid bars from the piano performances where the first and last beats of a bar were mapped to the same beat of the original song by the mapping function F_{map}. Around 50% of the bars are removed from the piano performances accordingly.

For objective and subjective evaluations, we collect additional 95 song-to-piano pairs from the Internet, comprising 19 Chinese Pop (**Cpop**), 20 Korean Pop (**Kpop**), 16 Japanese Pop (**Jpop**), 20 Anime Song (**Anime**), and 20 Western Pop (**Western**) pairs. To avoid collisions between the training and testing sets, we make sure that all testing songs were published after November 2022, the released month of the Pop2Piano dataset.

9.2 Experiment Setup

We implement our model using GPT-NeoX [54] as the piano token decoder and SheetSage [12] headed with an adapter network as the song audio encoder. The decoder consists of 8 layers, each with 8 attention heads. The adapter is a 4-layer Transformer encoder with 8 attention heads per layer. Our full model has approximately 39M learnable parameters. [9]

There are 2 ablations compared in the experiment, both of them sharing the same architecture as our full model, but one ablation (Ablation 1) is trained on song-to-piano data without pre-training, and the other ablation (Ablation 2) is trained on piano-only data. For baselines, in addition to Pop2Piano, we also include the piano transcription model by Kong *et al.* [18] to validate the effectiveness of the encoder component in our model.

We train the models with Adam optimizer, learning rate 1e$-$4, batch size 4 and segment length 1,024. The

[9] We use the pre-trained model of SheetSage and freeze its parameters.

Model	*objective evaluation*			*subjective evaluation* ($\in [1, 5]$)		
	$MCA\uparrow$	$GS\uparrow$	$H_4\downarrow$	OVL$\uparrow$	SI$\uparrow$	FL$\uparrow$
Pop2Piano [3]	**0.42** $\pm$ 0.07	0.86 $\pm$ 0.09	2.46 $\pm$ 0.18	2.71 $\pm$ 0.98	2.63 $\pm$ 1.01	2.72 $\pm$ 1.1
Ours	0.17 $\pm$ 0.06	0.84 $\pm$ 0.06	2.46 $\pm$ 0.22	**3.48** $\pm$ 0.93	**3.55** $\pm$ 1.06	**3.66** $\pm$ 1.02
Ablation 1 - w/o pre-train	0.16 $\pm$ 0.05	**0.87** $\pm$ 0.06	**2.45** $\pm$ 0.23	3.09 $\pm$ 1.03	2.96 $\pm$ 1.02	3.22 $\pm$ 1.09
Ablation 2 - w/o fine-tune	0.15 $\pm$ 0.05	0.81 $\pm$ 0.06	2.57 $\pm$ 0.19	3.09 $\pm$ 1.02	3.30 $\pm$ 1.07	3.08 $\pm$ 1.16
Transcription [18]	0.19 $\pm$ 0.06	0.67 $\pm$ 0.09	2.78 $\pm$ 0.30	1.48 $\pm$ 0.74	1.69 $\pm$ 0.88	1.45 $\pm$ 0.71
Human	0.16 $\pm$ 0.06	0.81 $\pm$ 0.06	2.59 $\pm$ 0.18	4.30 $\pm$ 0.87	4.23 $\pm$ 0.95	4.33 $\pm$ 0.9

Table 3. The results of objective evaluations and the MOS of the user study ($\uparrow$ / $\downarrow$: the higher / lower the better).

full model is pre-trained for 100K steps on the piano-only data, and then fine-tuned for an additional 70K steps on the song-to-piano paired data. Ablation 1 is trained from scratch for 100K steps directly on the paired data. Ablation 2 is trained for 50K steps only on the piano-only data, without any exposure to the song-to-piano pairs. During the fine-tuning stage for the full model, we tune the weighting factor α that controls the balance between the piano-only loss and song-to-piano loss, and find that the model achieved the best performance when α is set to 0.25.

In the experiment, all models are conducted to generate piano covers of the 95 testing songs (cf. Section 9.1) for the objective and subjective evaluations. To eliminate the bias caused by the varying quality of piano recordings, the ground truth human piano performances are first transcribed into MIDI note sequences. These MIDI sequences are then synthesized back into audio using the same MIDI synthesizer employed for the model outputs.

9.3 Objective Metrics

We adopt a set of existing metrics to assess the quality of the generated piano covers from different aspects, including similarity to the original song and coherence of the piano performance itself. Specifically, the metrics include:

- **Melody Chroma Accuracy** (MCA) [53] evaluates the similarity between two monophonic melody sequences. The melody line plays a crucial role in deciding whether a song cover accurately captures the contour of the original song. We compute the MCA between the vocals from the test song audio and the top melodic line extracted from the generated piano cover MIDI.
- **Pitch Class histogram Entropy** (H_4) [52] evaluates the harmonic diversity of a musical segment by computing the entropy of the distribution of note pitch class counts. A lower entropy value indicates lower harmonic diversity, but implies a more stable and consistent tonality across the segment. The subscript ("4") indicates the number of bars over which the entropy is calculated.
- **Next-Bar Grooving Pattern Similarity** (GS) is modified from the Grooving Pattern Similarity proposed in [52]. It originally measures the global rhythmic stability across an entire song. Instead of calculating over all pairs in the target, we adapt the metric to focus on local rhythmic stability within a song, evaluating the rhythmic coherence between each bar and its succeeding bar.

9.4 User Study

For subjective evaluation, we conduct an online listening test involving 52 volunteers: 5 professional music producers, 13 amateurs, and 34 pro-amateurs with more than 3-year music training. The volunteers are randomly assigned to distinct test sets, with each set containing 3 songs randomly selected from different genres, and for each song, there are 6 piano performances presented anonymously in random order. These piano performances are a human piano performance, results from our full model and the two ablated versions, and results from the Pop2Piano and piano transcription model baselines. All of them are truncated to 40-second audio clips from the beginning. Subjects are asked to listen to these audio clips and provide ratings on a 5-point Likert scale for the following aspects:

- **Similarity (SI):** The degree of similarity between the piano performances and the original song.
- **Music Fluency (FL):** The degree of perceived fluency in the music, representing the smoothness and coherence of the piano performances.
- **Overall (OVL):** How much do the participants like the piano cover in the personal overall listening experience?

9.5 Results

Table 3 displays the results of the objective evaluation metrics and mean opinion scores (MOS) from the user study. In the objective evaluation, Pop2Piano shows a leading MCA score compared to other models and the human piano performances, which indicates it excels at matching the original song's melodic contour. Except for the transcription baseline model, there is no significant difference in GS and H_4 across models, suggesting comparable local rhythmic coherence and harmonic variety.

Next, we pay attention to the result of user study. Much to our delight, the full model leads with the best scores across all aspects in the user study with statistical significance ($p < 0.05$), but there remains a gap compared to the human reference performances. Ablation 1 achieves higher scores than Pop2Piano in all aspects of the user study, both of which are trained on the paired data. This suggests that utilizing the weakly-aligned paired data, which avoids distorting the original piano performances, helps increase the overall listening experience quality of the model outputs

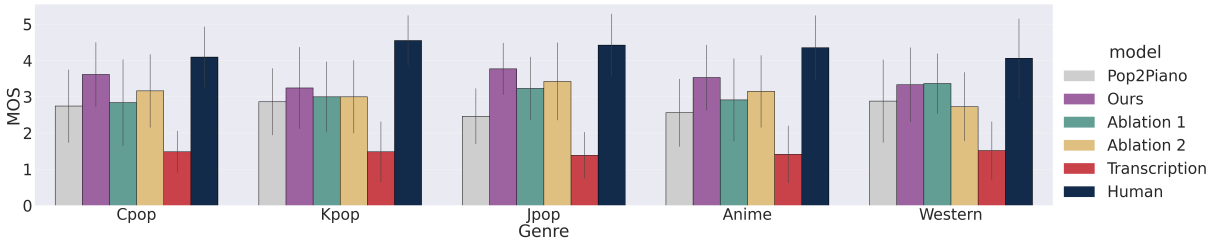

Figure 7. The MOS in overall scores (**OVL**) of the user study in different genres.

for human raters. While both Ablation 2 and the transcription baseline are trained on piano-only data, however, Ablation 2 performs significantly better than the baseline in both objective and subjective evaluations. This can be seen as evidence that SheetSage, as the encoder, extracts more relevant features beneficial for the piano cover generation task compared to the baseline transcription model.

10. DISCUSSION

In the experiment, we notice that while Pop2Piano exhibits a significantly higher MCA score than other models and even human performances, it does not achieve comparably high SI ratings in the user study. We suggest this conflict arises from the assumption in MCA that two melodies must temporally correspond to each other on a fixed "time grid." That is, the corresponding chroma features must be located at precisely the same time instants. For human listening experiences, two similar melodies only need to be coordinated on beats rather than a rigid time grid. Specifically, human perception of melodic similarity allows for the tempo or duration to be slightly changed in the same ratio, as long as their notes are located on the same underlying musical beat positions. As mentioned in Sections 6 & 7, different from transcription or arrangement, a cover song is not usually temporally aligned to the original song, i.e., the musical elements such as tempo, melody, rhythmic changed in the composition process of the piano cover. This temporal flexibility results in lower MCA scores for human-performed piano covers compared.

We also find that the two ablated models have the same **OVL** scores in the subjective evaluation, even though Ablation 2 has never seen any pop song data during training. To investigate the reason behind this, we first examine the piano covers generated by Ablation 2. Figure 8 shows a snippet of a cover generated by this model. We notice it tends to generate repeated short notes, resulting in an unnatural-sounding performance. However, Figure 7 demonstrates the **OVL** scores across different music genres. Interestingly, we see that Ablation 2 outperforms Ablation 1 for the *Cpop*, *Jpop*, and *Anime* genres. Additionally, as shown in Table 3, the former ablation also achieves higher **SI** and lower **FL** scores than the latter. From this observation, we suggest that (i) for short audio clips (less than 40 seconds), human raters may place more emphasis on initial melodic accuracy when judging the overall perceived quality, even if Ablation 1 tends to generate more

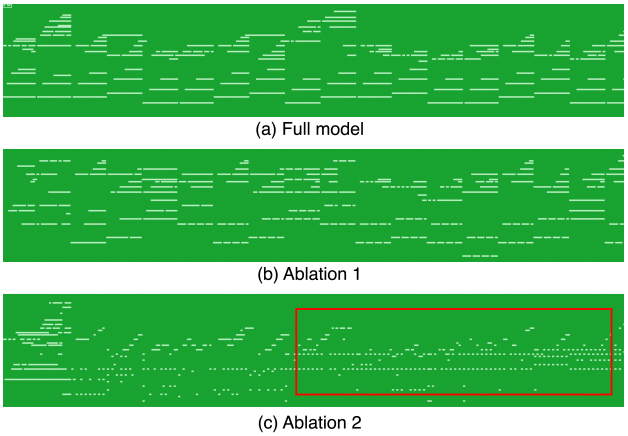

Figure 8. The pianoroll representation of a snippet from an example generated by the models. We observe that Ablation 2 which trained on piano-only data tends to generate repeated short notes.

coherent and natural-sounding results; (ii) Ablation 1 does not effectively learn to precisely capture the melodic contour from the reference song condition due to the inherent alignment errors present in the weakly-aligned song-to-piano paired data it was trained on.

Towards the end of this study, we seek to address the essential question: *what are the key factors that decide how humans judge the quality of a piano cover?* To gain insights, we interview the 5 professional music producers after they complete the user study. Their feedback highlights the following critical aspects: (i) performance skills dominate the judgment of piano cover quality. The ability to produce an expressive, musically coherent piano performance is the primary determining factor; (ii) the accuracy in capturing the original melody and chord progressions also plays an important role. However, *only about 80% fidelity is required*. A piano cover need not be entirely identical to the original song — some creative reinterpretation is allowed and even desirable to distinguish a high-quality arrangement from a mere literal transcription; and (iii) more advanced compositional details such as voice leading, orchestration, and adapting the overall vibe or feeling of the original piece to the piano expression are also crucial for creating a compelling cover.

11. CONCLUSION

In the first part of this study, we have presented PiCoGen 1, a generative model capable of creating a piano cover of an input audio with two distinct steps: extracting a lead sheet from the audio and generating a piano performance based on this lead sheet. Treating the lead sheet as the common ground between the input music and the target piano MIDI, PiCoGen 1 bypasses the need of curating paired data of covers and their original songs for training. Compared to the existing single-stage model Pop2Piano, PiCoGen 1 exhibits comparable or superior performance for Pop music and some other genres.

In the second part, we have presented PiCoGen 2, an approach that applies the concept of transfer learning to the task of piano cover generation. We proposed a training strategy that involves two stages: pre-training on piano-only data to learn fundamental piano performance skills, followed by fine-tuning on weakly-aligned song-to-piano paired examples for the cross-domain translation. A comprehensive set of experiments validated the effectiveness of the proposed transfer learning approach and the use of weakly-aligned data.

As we still require weakly-aligned data in our approach, one direction for future research involves tackling the challenge of piano cover generation without relying on data alignment. Additionally, there remains a need for systematic analysis to evaluate the quality of piano covers and identify the key factors influencing piano cover generation.

12. REFERENCES

[1] H. Takamori, T. Nakatsuka, S. Fukayama, M. Goto, and S. Morishima, "Audio-based automatic generation of a piano reduction score by considering the musical structure," in *Proc. MultiMedia Modeling Conference (MMM)*, 2019.

[2] S. Ariga, S. Fukayama, and M. Goto, "Song2guitar: A difficulty-aware arrangement system for generating guitar solo covers from polyphonic audio of popular music," 2017.

[3] J. Choi and K. Lee, "Pop2piano: Pop audio-based piano cover generation," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[4] J. Gardner, I. Simon, E. Manilow, C. Hawthorne, and J. Engel, "MT3: Multi-task multitrack music transcription," in *Proc. International Conference on Learning Representations (ICLR)*, 2022.

[5] H.-M. Liu and Y.-H. Yang, "Lead sheet generation and arrangement by conditional generative adversarial network," in *Proc. IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2018, pp. 722–727.

[6] S.-L. Wu and Y.-H. Yang, "Compose & Embellish: Well-structured piano performance generation via a two-stage approach," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[7] J. Serra, E. Gómez, and P. Herrera, "Audio cover song identification and similarity: background, approaches, evaluation, and beyond," *In Proc. Advances in music information retrieval*, 2010.

[8] F. Yesiler, C. Tralie, A. A. Correya, D. F. Silva, P. Tovstogan, E. Gómez Gutiérrez, and X. Serra, "Da-tacos: A dataset for cover song identification and understanding," 2019.

[9] D. F. Silva, F. Falcao, and N. Andrade, "Summarizing and comparing music data and its application on cover song identification." 2018.

[10] M. P. Ryynänen and A. P. Klapuri, "Automatic transcription of melody, bass line, and chords in polyphonic music," *Computer Music Journal*, 2008.

[11] J. Weil, T. Sikora, J.-L. Durrieu, and G. Richard, "Automatic generation of lead sheets from polyphonic music signals." in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2009.

[12] C. Donahue, J. Thickstun, and P. Liang, "Melody transcription via generative pre-training," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2022.

[13] W.-Y. Hsiao, J.-Y. Liu, Y.-C. Yeh, and Y.-H. Yang, "Compound Word Transformer: Learning to compose full-song music over dynamic directed hypergraphs," in *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[14] E. Benetos, S. Dixon, Z. Duan, and S. Ewert, "Automatic music transcription: An overview," *In Proc. IEEE Signal Processing Magazine*, 2019.

[15] K. Toyama, T. Akama, Y. Ikemiya, Y. Takida, W.-H. Liao, and Y. Mitsufuji, "Automatic piano transcription with hierarchical frequency-time Transformer," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2023.

[16] C. Hawthorne, E. Elsen, J. Song, A. Roberts, I. Simon, C. Raffel, J. Engel, S. Oore, and D. Eck, "Onsets and Frames: Dual-objective piano transcription," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2018.

[17] C. Hawthorne, I. Simon, R. Swavely, E. Manilow, and J. Engel, "Sequence-to-sequence piano transcription with Transformers," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2021.

[18] Q. Kong, B. Li, X. Song, Y. Wan, and Y. Wang, "High-resolution piano transcription with pedals by regressing onset and offset times," *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 2021.

[19] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, I. Simon, C. Hawthorne, N. Shazeer, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, "Music transformer," 2019.

[20] Y.-S. Huang and Y.-H. Yang, "Pop music Transformer: Beat-based modeling and generation of expressive pop piano compositions," in *Proc. ACM Multimedia (ACM MM)*, 2020.

[21] D. von Rütte, L. Biggio, Y. Kilcher, and T. Hofmann, "Figaro: Generating symbolic music with fine-grained artistic control," 2023.

[22] H.-W. Dong, K. Chen, S. Dubnov, J. McAuley, and T. Berg-Kirkpatrick, "Multitrack music transformer," 2023.

[23] P. Lu, X. Xu, C. Kang, B. Yu, C. Xing, X. Tan, and J. Bian, "Musecoco: Generating symbolic music from text," 2023.

[24] A. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck, "A hierarchical latent vector model for learning long-term structure in music," 2018.

[25] O. Cífka, U. Şimşekli, and G. Richard, "Groove2groove: One-shot music style transfer with supervision from synthetic data," *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 2020.

[26] S.-L. Wu and Y.-H. Yang, "MuseMorphose: Full-song and fine-grained piano music style transfer with one Transformer VAE," *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 2021.

[27] S. Oore, I. Simon, S. Dieleman, D. Eck, and K. Simonyan, "This time with feeling: Learning expressive musical performance," *In Proc. Neural Computing and Applications*, 2018.

[28] N. Fradet, J.-P. Briot, F. Chhel, A. El Fallah Seghrouchni, and N. Gutowski, "MidiTok: A python package for MIDI file tokenization," 2021.

[29] A. L. Uitdenbogerd and J. Zobel, "Manipulation of music for melody matching," 1998.

[30] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *In Proc. IEEE Transactions on Speech and Audio Processing*, 2002.

[31] R. Hennequin, A. Khlif, F. Voituret, and M. Moussallam, "Spleeter: A fast and efficient music source separation tool with pre-trained models," *In Proc. Journal of Open Source Software*, 2020.

[32] M. Mauch and S. Dixon, "pyin: A fundamental frequency estimator using probabilistic threshold distributions," 2014.

[33] S. Sai Vanka, M. Safi, J.-B. Rolland, and G. Fazekas, "Adoption of AI technology in music mixing workflow: an investigation," 2023.

[34] Z. Wang, K. Chen, J. Jiang, Y. Zhang, M. Xu, S. Dai, G. Bin, and G. Xia, "Pop909: A pop-song dataset for music arrangement generation," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2020.

[35] A. Ramaseshan, "Application of multiway methods for dimensinality reduction to music," 2013, master thesis, Aalto University.

[36] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, 2020.

[37] F. Zalkow and M. Müller, "Using weakly aligned score-audio pairs to train deep chroma models for cross-modal music retrieval." in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2020.

[38] A. Elowsson and A. Friberg, "Algorithmic composition of popular music," in *International Conference on Music Perception and Cognition (ICMPC)*, 2012.

[39] E. Nakamura and S. Sagayama, "Automatic piano reduction from ensemble scores based on merged-output hidden markov model," in *International Conference on Mathematics and Computing (ICMC)*, 2015.

[40] Z. Wang, K. Chen, J. Jiang, Y. Zhang, M. Xu, S. Dai, G. Bin, and G. Xia, "POP909: A pop-song dataset for music arrangement generation," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2020.

[41] R. P. Paiva, T. Mendes, and A. Cardoso, "An auditory model based approach for melody detection in polyphonic musical recordings," in *Proc. Computer Music Modeling and Retrieval (CMMR)*, 2004.

[42] ——, "On the detection of melody notes in polyphonic audio," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2005.

[43] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, 2016.

[44] P. Hamel, M. Davies, K. Yoshii, and M. Goto, "Transfer learning in MIR: Sharing learned latent representations for music audio classification and similarity," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2013.

[45] A. Van Den Oord, S. Dieleman, and B. Schrauwen, "Transfer learning by supervised pre-training for audio-based music classification," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2014.

[46] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "Transfer learning for music classification and regression tasks," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2017.

[47] K. Choi, G. Fazekas, and M. Sandler, "Towards playlist generation algorithms using RNNs trained on within-track transitions," in *Proc. Workshop on Surprise, Opposition, and Obstruction in Adaptive and Personalized Systems (SOAP)*, 2016.

[48] D. Liang, M. Zhan, and D. P. Ellis, "Content-aware collaborative music recommendation using pre-trained neural networks." in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2015.

[49] M. Müller, Y. Özer, M. Krause, T. Prätzlich, and J. Driedger, "Sync toolbox: A python package for efficient, robust, and accurate music synchronization," *Journal of Open Source Software*, 2021.

[50] J. Zhao, G. Xia, and Y. Wang, "Beat Transformer: Demixed beat and downbeat tracking with dilated self-attention," *arXiv preprint arXiv:2209.07140*, 2022.

[51] J. Copet, F. Kreuk, I. Gat, T. Remez, D. Kant, G. Synnaeve, Y. Adi, and A. Défossez, "Simple and controllable music generation," in *Proc. Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[52] S.-L. Wu and Y.-H. Yang, "The Jazz Transformer on the front line: Exploring the shortcomings of AI-composed music through quantitative measures," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2020.

[53] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, D. P. Ellis, and C. C. Raffel, "mir_eval: A transparent implementation of common MIR metrics," in *Proc. International Society for Music Information Retrieval (ISMIR)*, 2014.

[54] A. Andonian, Q. Anthony, S. Biderman, S. Black, P. Gali, L. Gao, E. Hallahan, J. Levy-Kramer, C. Leahy, L. Nestler, K. Parker, M. Pieler, J. Phang, S. Purohit, H. Schoelkopf, D. Stander, T. Songz, C. Tigges, B. Thérien, P. Wang, and S. Weinbach, "GPT-NeoX: Large scale autoregressive language modeling in PyTorch," 2023. [Online]. Available: https://www.github.com/eleutherai/gpt-neox