End-to-End Real-World Polyphonic Piano Audio-to-Score Transcription with Hierarchical Decoding

Wei Zeng[1] , **Xian He**[2] and **Ye Wang**[1,2]
[1]Integrative Sciences and Engineering Programme, NUS Graduate School
[2]School of Computing, National University of Singapore
{w.zeng, xian.he}@u.nus.edu, wangye@comp.nus.edu.sg

Abstract

Piano audio-to-score transcription (A2S) is an important yet underexplored task with extensive applications for music composition, practice, and analysis. However, existing end-to-end piano A2S systems faced difficulties in retrieving bar-level information such as key and time signatures, and have been trained and evaluated with only synthetic data. To address these limitations, we propose a sequence-to-sequence (Seq2Seq) model with a hierarchical decoder that aligns with the hierarchical structure of musical scores, enabling the transcription of score information at both the bar and note levels by multi-task learning. To bridge the gap between synthetic data and recordings of human performance, we propose a two-stage training scheme, which involves pre-training the model using an expressive performance rendering (EPR) system on synthetic audio, followed by fine-tuning the model using recordings of human performance. To preserve the voicing structure for score reconstruction, we propose a pre-processing method for **Kern scores in scenarios with an unconstrained number of voices. Experimental results support the effectiveness of our proposed approaches, in terms of both transcription performance on synthetic audio data in comparison to the current state-of-the-art, and the first experiment on human recordings.

1 Introduction

Audio-to-score transcription (A2S) is a notation-level automatic music transcription (AMT) task that aims to transcribe audio into human- or machine-readable musical scores. By providing detailed notations of musical compositions from audio recordings, A2S functions as a useful tool for music performance and music content analysis [Shibata *et al.*, 2021]. The piano, inherently polyphonic and covering a wide range of pitch, serves as an ideal model, offering potential extensions to other polyphonic music forms. A2S usually involves several interrelated subtasks, such as multi-pitch detection, rhythm quantization, voice separation, and key/time signature estimation.

Early A2S work [Cogliati *et al.*, 2016; Nakamura *et al.*, 2018; Shibata *et al.*, 2021] majorly decomposes A2S into subtasks and addresses them individually. From [Carvalho and Smaragdis, 2017], there has been a notable trend towards using end-to-end models for A2S, such as general Seq2Seq models [Liu *et al.*, 2021] and connectionist temporal classification (CTC) models [Román *et al.*, 2018; Román *et al.*, 2019; Arroyo *et al.*, 2022]. Compared to early decomposition-based methods, these end-to-end models tackle A2S more comprehensively, mitigating the potential problem of error accumulation from different sub-tasks.

However, the current state of end-to-end A2S models is still unfledged and thus cannot well address the following challenges.

Complexity of musical structures. Musical scores include various elements at different hierarchical levels [Koelsch *et al.*, 2013], such as notes, keys, and time signatures. This structural complexity is difficult to be well modelled by previous end-to-end A2S models, as existing studies [Liu *et al.*, 2021; Román *et al.*, 2019; Arroyo *et al.*, 2022] primarily focus on note transcription, neglecting key and time signature transcription. Additionally, the polyphonic nature of piano music adds to the challenge, particularly in scenarios with an unconstrained number of voices [Arroyo *et al.*, 2022].

Real-world evaluation gap. The availability of datasets for A2S transcription is limited, especially when compared to frame-level transcription datasets [Hawthorne *et al.*, 2019]. Existing end-to-end A2S systems primarily rely on synthetic audio directly from the quantized score MIDI for training and evaluation, which fails to capture human expressions in tempo, dynamics, and articulations. This reliance ignores the discrepancies between synthetic data and real-world recordings of human performance, limiting the real-world application of these models. To date, there has been no empirical evaluation of these end-to-end A2S systems using real-world audio recordings, underscoring a critical gap in the field.

In this paper, we introduce a novel Seq2Seq A2S model, leveraging a hierarchical decoder to align with the hierarchical structure of piano music. As depicted in Figure 1, the decoder first generates bar-level representations from the encoded audio representation, where key and time signatures are predicted. Subsequently, it further decodes these bar-level representations into note-level representations for note

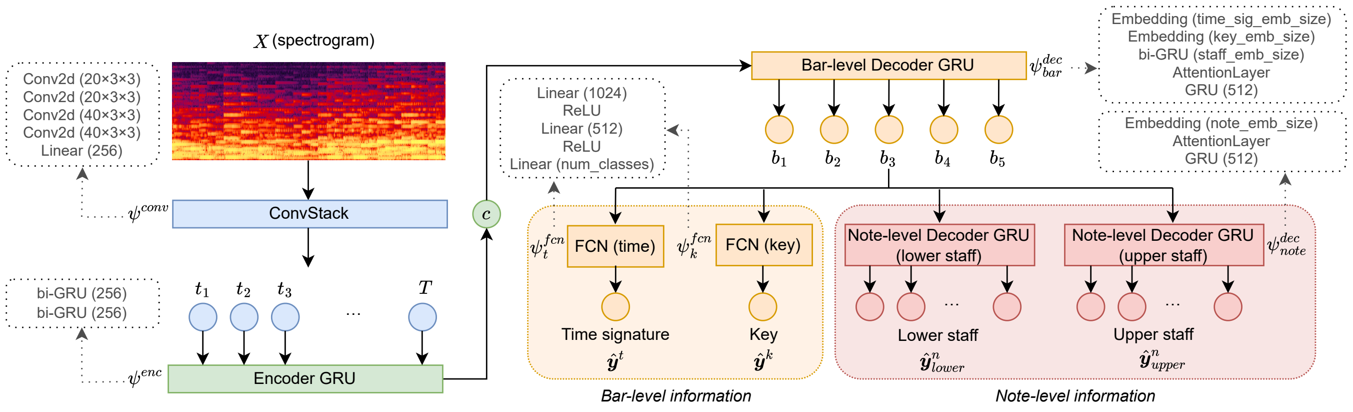


Figure 1: The proposed piano A2S model with a hierarchical decoder to transcribe the audio into both bar-level and note-level information. The model first encodes a Variable-Q Transform spectrogram X into audio context representation c. Subsequently, a bar-level Decoder decodes c into bar-level representation b_i for the i-th bar. Time signatures ($\hat{\boldsymbol{y}}^t$) and keys ($\hat{\boldsymbol{y}}^k$) are transcribed at this level given bar-level representation $\boldsymbol{b}$, while two note-level Decoders further decode $\boldsymbol{b}$ into note sequences of the lower staff ($\hat{\boldsymbol{y}}^n_{lower}$) and upper staff ($\hat{\boldsymbol{y}}^n_{upper}$), respectively.

transcriptions. To bridge the gap between synthetic sound and real-world recordings of human performance, we propose a two-stage training scheme, which pre-trains the model on a large synthetic dataset generated from the state-of-the-art expressive performance rendering (EPR) system [Jeong *et al.*, 2019a], and then fine-tunes the model on real-world piano recordings from human performance. Additionally, we extend previous work on score representation [Arroyo *et al.*, 2022] by proposing a pre-processing method to serialize **Kern scores into token sequences while preserving the voicing structure. This representation strategy allows for the reconstruction of the final musical scores from predicted sequences. Our model is evaluated on both synthetic data and real piano recordings selected from the ASAP dataset [Foscarin *et al.*, 2020]. These experiments demonstrate improvements over prior end-to-end systems and underscore the efficacy of our proposed training scheme.

To summarize, our main contributions are[1]:

- **A hierarchical A2S model by multi-task learning**: We introduce a novel end-to-end A2S model with a hierarchical decoder to transcribe the audio into both bar-level information, including key and time signatures, and note-level information, i.e., the note sequences.
- **A two-stage training scheme for A2S:** To bridge the gap between synthetic sound and real-world recordings, we first pre-train the model on a synthetic dataset generated from an EPR system and subsequently fine-tune it using real-world piano recordings from human performance. Experiments on human recordings show the effectiveness of our proposed training scheme.
- **A score representation method for unconstrained voices:** We introduce a pre-processing approach to serialize **Kern piano scores into tokens while preserving their inherent voicing structures, facilitating the reconstruction of scores with unconstrained number of voices.

[1]Our code, pre-trained models, and demos can be found at https://github.com/wei-zeng98/piano-a2s.

2 Related Work

2.1 A2S and Score Representation

Existing end-to-end methods transcribe audio directly into score representations like LilyPond and **Kern. [Liu *et al.*, 2021], for example, used a general Seq2Seq model to predict target sequences for both left and right hands in the LilyPond format. Meanwhile, [Román *et al.*, 2019] and [Arroyo *et al.*, 2022] applied CTC loss to predict the target score sequence in the **Kern format. Among these studies, [Liu *et al.*, 2021] and [Arroyo *et al.*, 2022] emphasized polyphonic representation in their work. However, [Liu *et al.*, 2021] is constrained to scenarios with only one voice per staff, thereby restricting its applicability to scores containing multiple voices occurring simultaneously for one hand. While [Arroyo *et al.*, 2022] first attempted to tackle the unconstrained voices problem, their approach compromised the voicing structure during the serialization of the score, making it difficult to reconstruct the transcribed musical scores from the predictions.

2.2 Expressive Performance Rendering Systems

EPR systems are used for generating human-like performance MIDI given score files by delivering emotions and messages through subtle controls of tempo, dynamics, articulations and other expressive elements. Recent EPR systems have achieved promising results using neural network-based approaches [Jeong *et al.*, 2019a; Jeong *et al.*, 2019b; Renault *et al.*, 2023]. Among these systems, [Jeong *et al.*, 2019a] combined recurrent neural network (RNN) with hierarchical attention network and conditional variational autoencoder (CVAE), allowing users to specify a composer as an input for the output performance style.

3 Methodology

3.1 Problem Formulation

The primary objective of the A2S task is to transcribe provided piano recordings into sequences of musical notes containing pitch and note values, along with time and key signatures. Initially, the audio recordings are segmented into 5-bar clips, with segmentation based on the downbeat of each bar. These audio clips are then transformed into spectrograms, denoted as $X \in \mathbb{R}^{T \times F}$, where T is the number of time frames and F is the number of frequency bins.

Given the spectrograms as input, the A2S model predicts the note sequences in **Kern format for both the lower ($\boldsymbol{y}^n_{lower}$) and upper ($\boldsymbol{y}^n_{upper}$) staffs, as well as the sequences of time ($\boldsymbol{y}^t$) and key ($\boldsymbol{y}^k$) signatures of each bar respectively.

Specifically, the note sequence is denoted as $\boldsymbol{y}^n = (y^n_1, y^n_2, \cdots, y^n_N)$, where each $y^n_i \in \Sigma^{n'}$ and N is the length of the sequence. Here, $\Sigma^{n'}$ comprises the **Kern score representation vocabulary Σ^n and special symbols $\{\langle\text{SOS}\rangle, \langle\text{EOS}\rangle, \langle\text{PAD}\rangle\}$. The time and key signatures are represented in sequences with a length of 5 corresponding to each bar: $\boldsymbol{y}^t = (y^t_1, y^t_2, \cdots, y^t_5)$, and $\boldsymbol{y}^k = (y^k_1, y^k_2, \cdots, y^k_5)$, where $y^t_i \in \Sigma^t$ and $y^k_i \in \Sigma^k$. Σ^t and Σ^k are vocabularies for time and key signatures, respectively. We keep 7 common time signatures for Σ^t as summarized in Table 1, and 14 keys for Σ^k. The representation of Σ^n is detailed in Section 3.2.

3.2 Data Representation

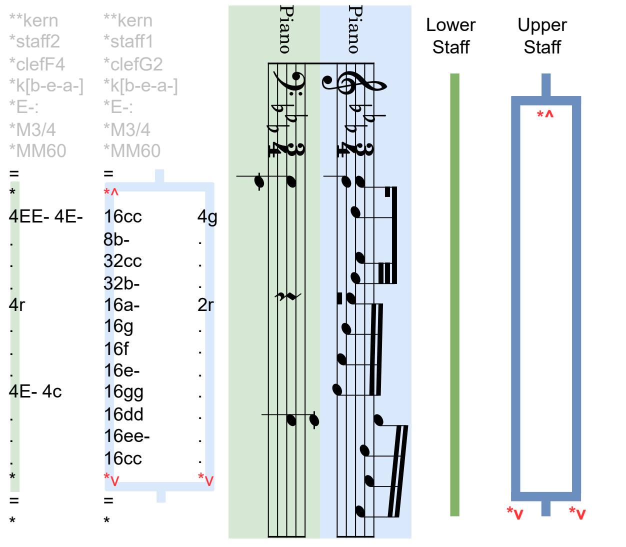
Introduction to **Kern Representation

We adopted **Kern as our score representation due to its clear and well-organized structure that accommodates multiple voices, as well as the abundance of data in the Humdrum database, from which we collect our HumSyn dataset as detailed in Section 4.1. Figure 2 shows an example of a **Kern representation for a one-bar piano score together with its original musical score and simplified voicing structure. In this example, the lower staff consists of a single voice, while the upper staff consists of two voices. Consequently, the **Kern score is encoded with three *spines*, each corresponding to a distinct voice. It is important to distinguish between the concept of polyphony and multiple voices: in the lower staff of the excerpt, there are two notes occurring simultaneously, indicating its polyphonic characteristic, but these simultaneous notes belong to a single voice. In a scenario with unconstrained voices, multiple voices may appear within the same staff, as shown in the upper staff of the excerpt. To manage this, **Kern representation allows for the division and merging of spines using specific identifiers, namely '*^' and '*v', which aids in maintaining and tracking the voicing structure. In **Kern representation, concurrent notes within a voice are separated by blank spaces, while notes belonging to different voices are separated by tabs.

The voicing structure within **Kern representation heavily relies on accurate predictions of the identifiers. An incorrect prediction of these identifiers can result in the loss of the voicing structure, posing significant challenges during the reconstruction process. For instance, as illustrated in the voicing structure of Figure 2, if the model fails to make a correct prediction regarding the '*^' token, the middle voice is ambiguous to either the lower staff or the upper staff, misleading the reconstruction process.

Proposed Pre-processing Method

To gain more insights on the voicing structure in piano scores for better pre-processing, we analyzed 235 scores from the



```
**kern       **kern
*staff2      *staff1
*clefF4      *clefG2
*k[b-e-a-]   *k[b-e-a-]
*E-:         *E-:
*M3/4        *M3/4
*MM60        *MM60
=            =
*            *^
4EE- 4E-     16cc      4g
.            8b-       .
.            32cc      .
.            32b-      .
4r           16a-      2r
.            16g       .
.            16f       .
.            16e-      .
4E- 4c       16gg      .
.            16dd      .
.            16ee-     .
.            16cc      .
*            *v        *v
=            =
*            *
```

Figure 2: A sample excerpt of multiple voicing: On the left, **Kern representation. On the center, the original score. On the right, the voicing structure of the excerpt. After pre-processing, the lower staff is serialized as: $\{4, EE-, \langle b\rangle, 4, E-, \backslash n, 4, r...\}$, and the upper staff is serialized as :$\{16, cc, \backslash t, 4, g, \backslash n, 8, b-, ...\}$, where $\langle b\rangle$ is the token for a blank space. The identifiers are manually added back as post-processing.

ASAP dataset, and counted the number of voices present in each staff at the bar level. The results revealed that bars containing no more than two voices per staff comprised 95.2% of the total in the lower staff and 97.6% in the upper staff. Taking into account the possibility of notation errors in the score files, this observation leads us to a robust conclusion: in the majority of scenarios, a single staff typically contains no more than two voices. Building upon this premise, we propose to separate the **Kern representation into lower and upper staffs and predict them individually within the model. This approach allows us to eliminate the identifiers within each staff without introducing ambiguity to the voicing structure. To reconstruct the score from the prediction, we manually reintroduce the identifiers during the post-processing stage, specifically before each transition between one voice and two voices.

Additionally, **Kern does not constrain the order of notes within chords or between different voices. It also allows for the inclusion of empty voices within scores. To ensure consistency in our data representation, we employ the following steps as pre-processing:

- Separate the lower staff and upper staff individually, and remove the identifiers '*^' and '*v' from the **Kern representation for each staff.
- Eliminate empty voices, and merge two parallel voices if their notes have identical durations.
- Arrange concurrent notes within the same voice in ascending order of pitch.
- Organize the voices so that the higher voice appears on the left stem, while the lower voice appears on the right stem.

3.3 Model Architecture

The A2S model is comprised of a ConvStack (ψ^{conv}), an Encoder (ψ^{enc}), a bar-level Decoder (ψ^{dec}_{bar}), note-level Decoders (ψ^{dec}_{note}) and Fully Connected Network (FCN) layers (ψ^{fcn}_{t}, ψ^{fcn}_{k}), as shown in Figure 1. The input spectrogram firstly passes through ψ^{conv} for feature extraction. We use padding to fix the time dimension, such that

$$\psi^{conv}(X) \in \mathbb{R}^{T\times d}, \tag{1}$$

where d is the feature dimension of the output of ConvStack.

Next, the Encoder encodes the features into a single representation:

$$outs, c = \psi^{enc}(\psi^{conv}(X)), \tag{2}$$

where $outs \in \mathbb{R}^{T\times d}$ is the outputs of the last layer, and $c \in \mathbb{R}^{d}$ is the audio content representation taken from the last hidden state.

For the Decoder, we developed a hierarchical structure capable of handling both bar-level and note-level information. Starting from the Encoder outputs $outs$ and the encoded audio representation c, we employ the bar-level Decoder ψ^{dec}_{bar} to decode it into bar representations, where each b_i denotes the feature representation for the i-th bar. At the bar level, FCN layers ψ^{fcn}_{t} and ψ^{fcn}_{k} are employed to make predictions for the time signature $\hat{y}^t_i$ and key signature $\hat{y}^k_i$ given b_i. Subsequently, using these bar-level representations as inputs, two note-level Decoders further decode the music notes into **Kern sequences for the lower and the upper staff respectively, resulting in the predicted note sequences $\hat{\boldsymbol{y}}^n_{lower}$ and $\hat{\boldsymbol{y}}^n_{upper}$. This hierarchical decoding process is summarized in Algorithm 1 in more details.

For the training, we employ multi-task learning [Zhang and Yang, 2021]. We use the negative log-likelihood loss (NLLL) for each subtask, including key, time signature, and note sequences for both the lower and upper staffs:

$$l_{\text{subtask}} = \text{NLLL}(\hat{\boldsymbol{y}}^{\text{subtask}}, \boldsymbol{y}^{\text{subtask}}) \tag{3}$$

The total loss is the summation of these individual losses:

$$l = l_{\text{key}} + l_{\text{time}} + l_{\text{lower}} + l_{\text{upper}}. \tag{4}$$

3.4 Training Scheme

Synthetic data has been extensively used in A2S tasks due to its ease of access and potential for augmentation. In contrast, obtaining real piano recordings from human performance is challenging and less accessible. In order to bridge the gap between synthetic sound and real human performance while leveraging the easy accessibility of synthetic sound, we propose a two-stage training scheme, as illustrated in Figure 3. The pre-training leverages data synthesized from the state-of-the-art EPR model, VirtuosoNet [Jeong *et al.*, 2019a]. Compared to data directly generated from score MIDI, these synthetic data are more similar to human performance as they capture some subtle details in piano performance such as deviations in note onsets, durations, velocities, and pedal usage. The model is pre-trained on synthetic audio from the EPR system. In the subsequent fine-tuning stage, we handpick a small set of real-world recordings of human performance from the ASAP dataset [Foscarin *et al.*, 2020] to fine-tune the pre-trained model by transfer learning.

Algorithm 1 The Hierarchical Decoder

Input: Encoded audio content representation c, and Encoder outputs of the last layer $outs$

Parameter: Bar-level hidden state $\boldsymbol{h}^b$, bar-level token $\widetilde{\boldsymbol{b}}$, bar-level feature $\boldsymbol{b}$, bar-level attention $\boldsymbol{a}$, note-level hidden state $\boldsymbol{h}^n$, note-level token $\widetilde{\boldsymbol{n}}$, note-level feature $\boldsymbol{n}$, attention layer Attn($\cdot$), and embedding layer Emb($\cdot$)

Output: Predicted time signature $\hat{\boldsymbol{y}}^t$, key signature $\hat{\boldsymbol{y}}^k$, and note sequences $\hat{\boldsymbol{y}}^n$ ($\hat{\boldsymbol{y}}^n_{lower}$, $\hat{\boldsymbol{y}}^n_{upper}$)

```
1:  Let $h^b_0 = c$, $\widetilde{b}_0 = \langle\text{SOS}\rangle$.
2:  for i = 1, 2, ..., 5 do
3:      $a_i = \text{Attn}(h^b_{i-1}, outs)$
4:      $b_i, h^b_i = \psi^{dec}_{bar}(\text{Cat}(\widetilde{b}_{i-1}, a_i), h^b_{i-1})$
5:      $\hat{y}^t_i = \text{log\_softmax}(\psi^{fcn}_t(\text{Cat}(b_i, a_i)))$
6:      $\hat{y}^k_i = \text{log\_softmax}(\psi^{fcn}_k(\text{Cat}(b_i, a_i)))$
7:      Let $h^n_{i,0} = b_i$, $\widetilde{n}_{i,0} = \langle\text{SOS}\rangle$
8:      for t = 1, 2, ..., max_steps do
9:          $n_{i,t}, h^n_{i,t} = \psi^{dec}_{note}(\text{Cat}(\widetilde{n}_{i,t-1}, \text{Attn}(h^n_{i,t-1}, outs)),$
                $h^n_{i,t-1})$
10:         $\hat{y}^n_{i,t} = \text{log\_softmax}(n_{i,t})$
11:         $\widetilde{n}_{i,t} = \text{Emb}(\text{argmax}(\hat{y}^n_{i,t}))$
12:         if $\widetilde{n}_{i,t} == \langle\text{EOS}\rangle$ then
13:             break
14:         end if
15:     end for
16:     staff_emb = GRU ($\hat{y}^n_{i,1:\text{max\_steps}}$)
17:     $\widetilde{b}_i = \text{Cat}(\text{staff\_emb}, \text{Emb}(\text{argmax}(\hat{y}^t_i)),$
            $\text{Emb}(\text{argmax}(\hat{y}^k_i)))$
18: end for
19: return $\hat{\boldsymbol{y}}^t$, $\hat{\boldsymbol{y}}^k$, $\hat{\boldsymbol{y}}^n$
```

4 Experiments

4.1 Data Preparing

Corpora

For the pre-training stage, we consider two score corpora: MuseScore website[2] and Humdrum repository[3]. From MuseScore, we choose the same MuseSyn dataset used in [Liu *et al.*, 2021], primarily comprising pop music. From the Humdrum repository, we select piano sonatas by Beethoven[4], Haydn[5], Mozart[6], Scarlatti[7], as well as Joplin's compositions[8] and Chopin's works[9], covering a range of styles from classical to ragtime. This combined corpus from Humdrum database is referred to as the HumSyn dataset.

For pre-training, we randomly select 80% of the pieces from the HumSyn dataset and combine them with the training

[2]https://musescore.com/hub/piano

[3]http://kern.ccarh.org/

[4]https://github.com/craigsapp/beethoven-piano-sonatas.git

[5]https://github.com/craigsapp/haydn-piano-sonatas.git

[6]https://github.com/craigsapp/mozart-piano-sonatas.git

[7]https://github.com/craigsapp/scarlatti-keyboard-sonatas.git

[8]https://github.com/craigsapp/joplin.git

[9]https://github.com/pl-wnifc/humdrum-chopin-first-editions.git

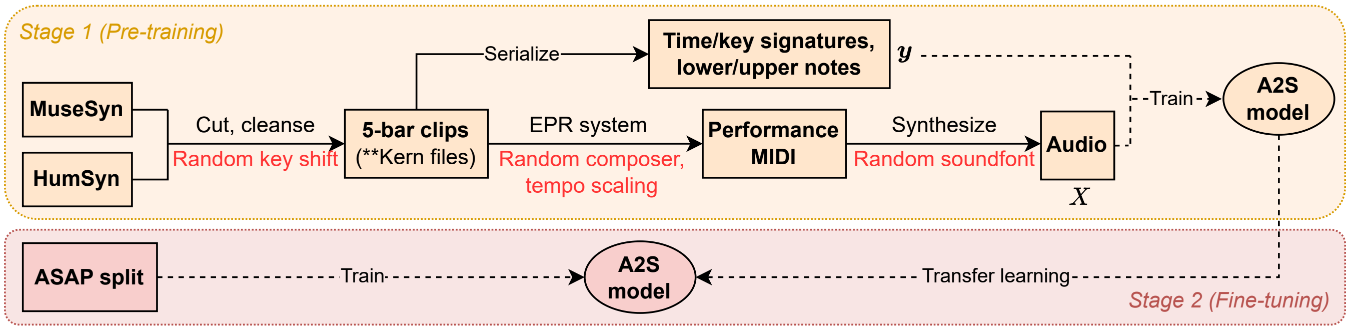


Figure 3: The two-stage training scheme including pre-training on synthetic data from an EPR system, and fine-tuning on human recordings.

Dataset	Split	Num	Time Signatures
MuseSyn	train	169	4/4,3/4,2/4,6/8,2/2,12/8
(Pop)	valid	20	4/4,3/4,2/2
	test	21	4/4,3/4,6/8
HumSyn	train	382	4/4,3/4,2/4,6/8,2/2,12/8,3/8
(Classical,	valid	46	4/4,3/4,2/4,6/8,2/2,3/8
Ragtime)	test	45	4/4,3/4,2/4,6/8,2/2,12/8,3/8
ASAP	train	14	4/4,3/4,2/4,2/2
(Classical)	test	25	4/4,3/4,2/4,6/8,2/2,3/8

Table 1: Statistics about number of musical scores and time signatures in different splits used in the experiments.

split of the MuseSyn dataset together as the training set. Similarly, we create validation and test sets by randomly selecting 10% of the pieces from the HumSyn dataset and combining them with the corresponding splits of the MuseSyn dataset. This is to ensure a consistency on the MuseSyn test split with [Liu *et al.*, 2021] for a fair comparison.

For the fine-tuning, we use the ASAP dataset [Foscarin *et al.*, 2020] with annotated alignments between scores and human performances. From ASAP, we select 14 songs, each featuring recordings by one or more different performers, with a total of 58 recordings in our chosen subset. Additionally, we choose 25 different compositions for the test set, including a total of 80 recordings performed by various artists. We confirmed that these splits are disjoint and cannot lead to data leakages. The statistics about MuseSyn, HumSyn, and ASAP split are summarized in Table 1.

Data Synthesis Pipeline

The data synthesis pipeline for the pre-training stage is illustrated in Figure 3. Firstly, we segment the scores into 5-bar clips and cleanse the scores to eliminate ornamental elements such as editorial marks and grace notes. We also exclude double sharps and flats for a reasonable vocabulary size.

For the training process, we employ four sequential steps for data augmentation on each of the segmented clip, including a random key shift within a range of four semitones, a random selection from 15 composers within the EPR system [Jeong *et al.*, 2019a], a random tempo scaling between 0.85 and 1.15 for the performance MIDI, and a random selection from four piano soundfonts during the synthesizing process. These data augmentation methods expand the final training set to 10× of its original size, with key signatures spanning a wide range covering number of sharps from -6 to 7.

For the validation and testing processes, we keep the original key and tempo, while generating performance MIDI using four EPR composers for every score clip: *Score* (generating MIDI directly from the score, *id*), *Bach (id)*, *Mozart (ood)*, and *Chopin (ood)*. The term *id*, or in-distribution, indicates the composer is used in training, while *ood*, or out-of-distribution, is not. Additionally, we synthesize audio using three distinct piano soundfonts for every MIDI clip: *UprightPianoKW (Upright, id)*, *SalamanderGrandPiano (Salamander, id)* and *YDP-GrandPiano (YDP, ood)*.

4.2 Metrics

We evaluate the score transcription using two established metrics, in accordance with prior end-to-end A2S research [Liu *et al.*, 2021; Arroyo *et al.*, 2022; Román *et al.*, 2019]: word error rate (WER) and the MV2H metric [McLeod and Steedman, 2018; McLeod, 2019]. Additionally, to evaluate the performance of key and time signature transcription, we employ the F1-score for bar-level transcriptions. Below are the details of these metrics:

- **WER**: We calculate the WER by comparing the transcription score with the original in **Kern sequences, treating each individual token as a word. We report both the overall WER and the individual WER for the two staffs, denoted as WER_l and WER_u for the lower and upper staff, respectively.
- **MV2H**: MV2H evaluation includes four sub-metrics: multi-pitch detection accuracy (F_p), voice separation accuracy (F_{voi}), note value detection accuracy (F_{val}), and harmonic detection (F_{harm}). Since we do not constrain metrical prediction at the note level, we have excluded the sub-metric for metrical alignment. Instead, we predict the time signature at the bar level and assess its performance using the F1-score. The overall metric (F_{MV2H}) is computed as the average of the four sub-metrics. We employ the non-aligned version of MV2H [McLeod, 2019], which is suitable for our non-aligned scenario by using automatic alignment.
- **F1-score**: Due to the imbalanced distribution of key and time signature labels, we employ the macro F1-score for key (f_k) and time signature (f_t) respectively.

Model	F_p	F_{voi}	F_{val}	F_{harm}	F_{MV2H}
baseline (*Joint*)	71.1	**90.8**	94.4	-	85.4
Ours (score)	**81.2**	90.4	**95.3**	82.1	**87.2**
Ours (EPR)	80.0	87.7	92.4	84.0	86.0

Table 2: Comparison of our models with the state-of-the-art baseline model on MuseSyn test split synthesized from score MIDI.

4.3 Experimental Setup

We build our hierarchical piano A2S model using the PyTorch library and SpeechBrain toolkit [Ravanelli *et al.*, 2021]. The duration of the synthesized audio clips ranges from 4 to 12 seconds. We transform the audio into variable-Q transform (VQT) spectrograms [Schörkhuber *et al.*, 2014], following [Liu *et al.*, 2021]. The audio sample rate and hop length are 16kHz and 160. We employ 60 bins per octave across a total of 8 octaves, and the gamma value for the VQT is 20. During the training process of the pre-training stage, we employ teacher forcing [Lamb *et al.*, 2016] to expedite the model convergence. Initially, the teacher forcing ratio is set to 0.7, with a decay rate of 0.99 for each epoch. In the training process of the fine-tuning stage, the teacher forcing ratio remains fixed at 0.6, given the limited size of the dataset and knowledge acquired during the pre-training stage.

4.4 Comparison with Baseline Model

We select the Joint model with Reshaped representation (*Joint*) from [Liu *et al.*, 2021] as our baseline model. This choice is motivated by several factors: it specializes in polyphonic piano transcription, adopts an end-to-end approach, provides a publicly available test split, and uses a comparable MV2H metric. In contrast, some other models, like [Arroyo *et al.*, 2022], randomize test split and are not directly comparable to our work due to the use of different tokenizers and reliance solely on the WER metric.

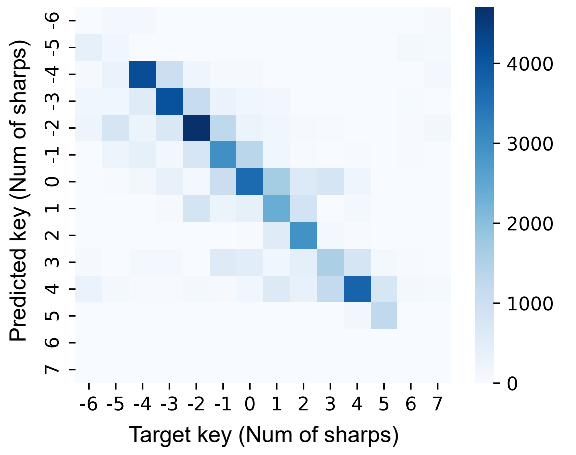
We synthesize the same training split in two different ways: 1) synthesizing directly from the score MIDI, and 2) using the EPR system for synthesizing. Models pre-trained on these two training splits are denoted as *Ours (score)* and *Ours (EPR)* respectively. To enable a fair comparison, we synthesize the MuseSyn test split directly from score MIDI, following the methodology employed in [Liu *et al.*, 2021].

We compare our models with the baseline *Joint* model on MuseSyn test split in terms of MV2H metrics. From Table 2, it is evident that our model exhibits improvements over the baseline *Joint* model, particularly in multi-pitch detection and note value detection. Notably, on multi-pitch detection, *Ours (score)* shows an increase of 10.1%. Furthermore, we observe that *Ours (score)* exhibits higher accuracy compared to *Ours (EPR)* on the test split of MuseSyn synthesized from the score MIDI, but *Ours (score)* and *Ours (EPR)* perform the opposite on human recordings, as elaborated in the following sections.

4.5 Results from the Pre-training Stage

Generalization Capability

To further investigate the impact of EPR composers and piano soundfonts on *Ours (EPR)*'s performance, as well as its ability to generalize to previously unseen performance styles and



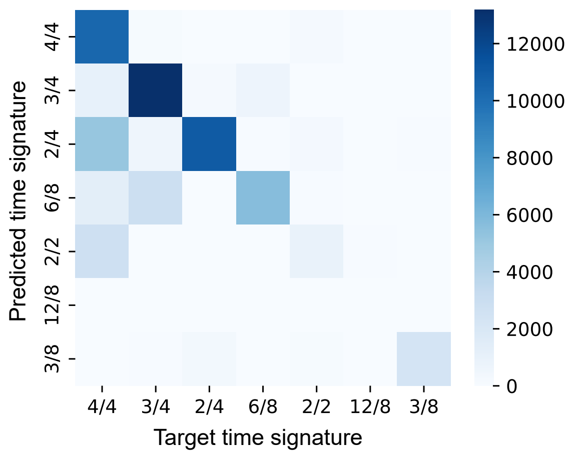


Figure 4: The confusion matrices of key (upper) and time signature (lower) of *Ours (EPR)* in the pre-training stage.

soundfonts in the pre-training stage, we summarize the performance of *Ours (EPR)* on MV2H with respect to different composers and soundfonts in Table 4. The results reveal that the model demonstrates strong generalization capabilities to unseen performance styles. However, it is worth noting that *ood* soundfonts can have a more significant influence on performance, with an acceptable degradation over *id* soundfonts. This can be attributed to two main factors: firstly, the diversity in performance styles arising not only from EPR composers but also from various compositions themselves, enhancing the model's generalizability to unseen performance styles; and secondly, the model's exposure to a limited set of soundfonts during pre-training, resulting in a similarity of learned sound characteristics.

Bar-level Information Transcription

To gain further insights into the model's key and time signature prediction, we visualize the confusion matrices for the test split of *Ours (EPR)*, as depicted in Figure 4. The matrices illustrate the number of bars along with their corresponding targets and predictions. In music theory, key distance is used to describe the varying degrees of relevance between musical keys. It is defined as the difference in the number of sharps or flats in the key signatures [Krumhansl *et al.*, 1982]. There-

Model	Fine-tune	F_p	F_{voi}	F_{val}	F_{harm}	F_{MV2H}	WER_l	WER_u	WER	f_k	f_t
Ours (score)	No	39.8	75.8	80.1	46.9	60.6	84.0	91.7	87.8	**37.6**	38.7
	Yes	60.4	87.8	**90.9**	50.0	72.3	61.5	56.8	59.2	36.3	62.6
Ours (EPR)	No	53.5	83.2	89.8	51.4	69.4	67.0	65.7	66.4	36.6	56.4
	Yes	**63.3**	**88.4**	90.7	**54.5**	**74.2**	**57.0**	**53.2**	**55.1**	36.5	**71.2**

Table 3: The MV2H, WER and F1-score results of *Ours (score)* and *Ours (EPR)* on ASAP test split before and after fine-tuning.

Test Split	F_p	F_{voi}	F_{val}	F_{harm}	F_{MV2H}
Score (id)	79.8	87.3	**92.8**	65.9	81.4
Bach (id)	**82.8**	89.0	92.5	67.1	82.8
Mozart (ood)	82.0	**89.2**	**92.8**	**67.5**	**82.9**
Chopin (ood)	80.2	88.8	92.4	67.0	82.1
Upright (id)	**83.5**	89.5	**92.9**	67.0	83.2
Salamander (id)	83.2	**90.0**	92.8	**67.1**	**83.3**
YDP (ood)	76.9	86.3	92.1	66.5	80.4
Overall	81.2	88.6	92.6	66.9	82.3

Table 4: MV2H metric of *Ours (EPR)* on test splits synthesized by different EPR composers and piano soundfonts.

fore, in Figure 4, adjacent keys have the closest key distance. From Figure 4, we observe a pronounced trend in the key confusion matrix: the model tends to make mistakes in keys with close distances. Similarly, in time signature detection, the model shows a tendency to confuse similar time signatures, such as 4/4 and 2/2, 3/4 and 6/8, etc.

4.6 Results from the Fine-tuning Stage

Effectiveness of Using EPR Models for Pre-training

To investigate the impact of the EPR system used in the pre-training stage, we compare the performance of *Ours (score)* and *Ours (EPR)* on the ASAP test split before and after fine-tuning. The MV2H, WER and F1-score are summarized in Table 3. From the table, it is evident that *Ours (EPR)* outperforms *Ours (score)* on almost all metrics, both before and after fine-tuning. Even with only a small number of songs selected for fine-tuning, both models exhibit improvements across most metrics. Taking these points above, we conclude that audio synthesized from EPR systems capture more similarities from human performances, which is a more efficient way for pre-training. These results underscore the effectiveness of utilizing EPR systems for pre-training.

Case Study

We present a sample result from the test ASAP split transcribed by *Ours (EPR)* after fine-tuning on the ASAP training set, as shown in Figure 5. Notably, it is selected from real-world human recordings, with the model transcribing all score information. Upon closer examination, several interesting observations can be made. While the entire composition is based on the key of *D* minor, as indicated by one flat in the upper part of Figure 5, there is a temporary shift to *A* minor key within the audio clip. Remarkably, our model correctly identifies this potential key change, even though the key signature of the original piece remains the same. This demonstrates the model's capability to detect such key variations.

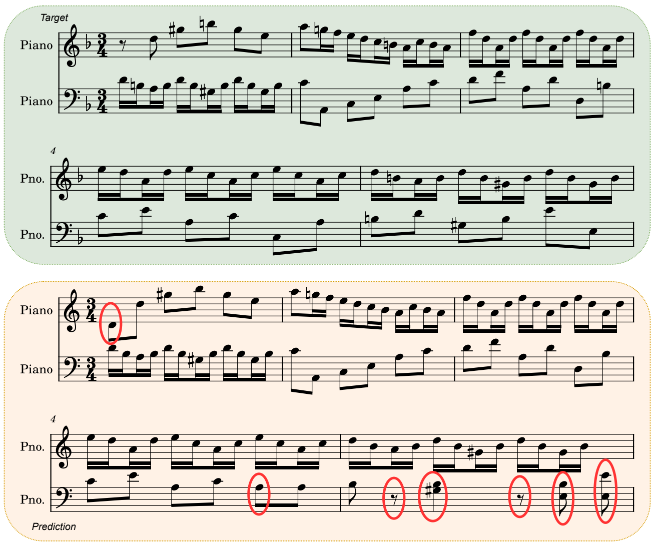


Figure 5: A sample score (upper) and its transcription result (lower) from the fine-tuned *Ours (EPR)* model. The excerpt is selected from *Prelude and Fugue in D minor, BWV 875 (Bach, Johann Sebastian)*, performed by HONG04M.

5 Conclusion

In this paper, we introduced a novel end-to-end piano A2S model targeting at two main challenges in existing work: difficulty in modelling hierarchical musical structures, and the discrepancies between synthetic data and real-world recordings from human performance. We proposed a Seq2Seq model with a hierarchical decoder that transcribes both the bar-level and note-level information. To bridge the gap between synthetic data and real-world recordings, we proposed a two-stage training scheme to pre-train the model on synthetic data from an EPR system, and fine-tune the model on human performance. Furthermore, to preserve the voicing structure in the musical score, we proposed a pre-processing method for **Kern representation for score reconstruction. We conducted experiments on the pre-trained model to evaluate its generalization capabilities in both in-distribution and out-of-distribution scenarios. We also conducted the first experiment for end-to-end A2S systems on piano recordings of human performance on ASAP dataset and validated the effectiveness of our proposed training scheme.

Acknowledgments

We would like to thank anonymous reviewers for their valuable suggestions. We also appreciated Joon Siong Yap's work on audio synthesis and Qihao Liang's writing advice.

References

[Arroyo *et al.*, 2022] Víctor Arroyo, Jose J Valero-Mas, Jorge Calvo-Zaragoza, and Antonio Pertusa. Neural audio-to-score music transcription for unconstrained polyphony using compact output representations. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4603–4607. IEEE, 2022.

[Carvalho and Smaragdis, 2017] Ralf Gunter Correa Carvalho and Paris Smaragdis. Towards end-to-end polyphonic music transcription: Transforming music audio directly to a score. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 151–155. IEEE, 2017.

[Cogliati *et al.*, 2016] Andrea Cogliati, David Temperley, and Zhiyao Duan. Transcribing human piano performances into music notation. In *ISMIR*, pages 758–764, 2016.

[Foscarin *et al.*, 2020] Francesco Foscarin, Andrew Mcleod, Philippe Rigaux, Florent Jacquemard, and Masahiko Sakai. Asap: a dataset of aligned scores and performances for piano transcription. In *International Society for Music Information Retrieval Conference*, pages 534–541. ISMIR, November 2020.

[Hawthorne *et al.*, 2019] 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. In *International Conference on Learning Representations*, 2019.

[Jeong *et al.*, 2019a] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, Kyogu Lee, and Juhan Nam. Virtuosonet: A hierarchical rnn-based system for modeling expressive piano performance. In *ISMIR*, pages 908–915, 2019.

[Jeong *et al.*, 2019b] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, and Juhan Nam. Graph neural network for music score data and modeling expressive piano performance. In *International conference on machine learning*, pages 3060–3070. PMLR, 2019.

[Koelsch *et al.*, 2013] Stefan Koelsch, Martin Rohrmeier, Renzo Torrecuso, and Sebastian Jentschke. Processing of hierarchical syntactic structure in music. *Proceedings of the National Academy of Sciences*, 110(38):15443–15448, 2013.

[Krumhansl *et al.*, 1982] Carol L Krumhansl, Jamshed Bharucha, and Mary A Castellano. Key distance effects on perceived harmonic structure in music. *Perception & Psychophysics*, 32(2):96–108, 1982.

[Lamb *et al.*, 2016] Alex M Lamb, Anirudh Goyal Alias Parth Goyal, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. *Advances in neural information processing systems*, 29, 2016.

[Liu *et al.*, 2021] Lele Liu, Veronica Morfi, and Emmanouil Benetos. Joint multi-pitch detection and score transcription for polyphonic piano music. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 281–285. IEEE, 2021.

[McLeod and Steedman, 2018] Andrew McLeod and Mark Steedman. Evaluating automatic polyphonic music transcription. In *ISMIR*, pages 42–49, 2018.

[McLeod, 2019] Andrew McLeod. Evaluating non-aligned musical score transcriptions with mv2h. *arXiv preprint arXiv:1906.00566*, 2019.

[Nakamura *et al.*, 2018] Eita Nakamura, Emmanouil Benetos, Kazuyoshi Yoshii, and Simon Dixon. Towards complete polyphonic music transcription: Integrating multi-pitch detection and rhythm quantization. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 101–105. IEEE, 2018.

[Ravanelli *et al.*, 2021] Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, et al. Speechbrain: A general-purpose speech toolkit. *arXiv preprint arXiv:2106.04624*, 2021.

[Renault *et al.*, 2023] Lenny Renault, Rémi Mignot, and Axel Roebel. Expressive piano performance rendering from unpaired data. In *International Conference on Digital Audio Effects (DAFx23)*, 2023.

[Román *et al.*, 2018] Miguel A Román, Antonio Pertusa, and Jorge Calvo-Zaragoza. An end-to-end framework for audio-to-score music transcription on monophonic excerpts. In *ISMIR*, pages 34–41, 2018.

[Román *et al.*, 2019] Miguel A Román, Antonio Pertusa, and Jorge Calvo-Zaragoza. A holistic approach to polyphonic music transcription with neural networks. *arXiv preprint arXiv:1910.12086*, 2019.

[Schörkhuber *et al.*, 2014] Christian Schörkhuber, Anssi Klapuri, Nicki Holighaus, and Monika Dörfler. A matlab toolbox for efficient perfect reconstruction time-frequency transforms with log-frequency resolution. In *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*. Audio Engineering Society, 2014.

[Shibata *et al.*, 2021] Kentaro Shibata, Eita Nakamura, and Kazuyoshi Yoshii. Non-local musical statistics as guides for audio-to-score piano transcription. *Information Sciences*, 566:262–280, 2021.

[Zhang and Yang, 2021] Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609, 2021.