Automatic Lyrics Transcription of Polyphonic Music With Lyrics-Chord Multi-Task Learning

Xiaoxue Gao, *Student Member, IEEE*, Chitralekha Gupta, *Member, IEEE*, and Haizhou Li, *Fellow, IEEE*

***Abstract*—Lyrics are the words that make up a song, while chords are harmonic sets of multiple notes in music. Lyrics and chords are generally essential information in music, i.e. unaccompanied singing vocals mixed with instrumental music, representing important components in polyphonic music. In a traditional lyrics transcription task, we first extract the singing vocals from the polyphonic music and then transcribe the resulting singing vocals, where the two steps are optimized independently. In this paper, we propose novel end-to-end network architectures that are designed to disentangle lyrics from chords in polyphonic music for effective lyrics transcription in a single step, where we consider chords as musical words, analogously to lexical words as lyrics intuitively. We start by studying a single-task lyrics transcriber as the reference baseline and the initial model to develop the multi-task lyrics transcription solutions. The main idea is to take advantage of chord transcription available in the training data through multi-task training to improve lyrics transcription. The experiments show that the proposed multitask lyrics transcriber significantly outperforms other competing solutions, with a word error rate (WER) of 31.82% on a standard test dataset.**

***Index Terms*—Automatic lyrics transcription in polyphonic music, multi-task learning, singing voice separation, music information retrieval.**

I. Introduction

UNLIKE automatic speech recognition (ASR) [1], [2], automatic lyrics transcription in polyphonic music has not been well studied. It aims to recognize the lyrics from singing vocals in the presence of instrumental music accompaniment. Lyrics transcription is an enabling technology for many downstream applications, such as the automatic generation of karaoke lyrical content, music video subtitling, and query-by-singing.

Manuscript received 23 September 2021; revised 20 March 2022 and 20 June 2022; accepted 20 June 2022. Date of publication 19 July 2022; date of current version 25 July 2022. The work of Chitralekha Gupta was supported by Academic Research Council, Ministry of Education (ARC, MOE), Singapore, under Grant MOE2018-T2-2-127. The work by Haizhou Li was supported in part by the Guangdong Provincial Key Laboratory of Big Data Computing and in part by the The Chinese University of Hong Kong, Shenzhen, China under Grant B10120210117-KP02. This work was supported in part by the Agency for Science, Technology and Research (A*STAR) through the AME Programmatic Funding Scheme under Project A18A2b0046, and in part by A*STAR through RIE2020 Advanced Manufacturing and Engineering Domain (AME) Programmatic Grant under Grant A1687b0033, Project Title: Spiking Neural Networks. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Hema A Murthy.

Xiaoxue Gao is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117583 (e-mail: xiaoxue.gao@u.nus.edu).

Chitralekha Gupta is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117583 (e-mail: chitralekha@nus.edu.sg).

Haizhou Li is with the The Chinese University of Hong Kong, Shenzhen 518172, China, also with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117583, and also with the Kriston AI Lab, Xiamen, China (e-mail: haizhou.li@u.nus.edu).

Digital Object Identifier 10.1109/TASLP.2022.3190742

Singing manifests a higher degree of variation in pronunciation and prosody than speech [3]; therefore, lyrics transcription of singing vocals is more challenging than automatic speech recognition (ASR). Early studies on lyrics transcription mostly adopted standard ASR techniques, such as hidden Markov models (HMMs) and deep neural networks (DNNs) [4]–[6], which are not specifically designed for singing processing. Furthermore, instrumental music accompaniment typically adds a complex and structured source of musical information to the singing vocals, therefore adversely impacting intelligibility [7]. The solutions to lyrics transcription in polyphonic music can be grouped into two broad categories: 1) *extraction-transcription*, which extracts singing vocals first through a source separation model and then transcribes the extracted vocals [8]–[11], and 2) *direct modeling*, which models polyphonic audio, i.e. singing vocals mixed with instrumental music directly [12]–[14].

The extraction-transcription approach typically follows the standard speech recognition process [15], [16]. In other words, lyrics transcription is performed in the same way as speech recognition. As a result, its performance is adversely affected by imperfect singing voice extraction frontend [8], [13]. Furthermore, the voice extraction and transcription modules are trained separately, which leads to a possible mismatch between training and testing. The extraction task may require clean vocals in addition to the polyphonic music for training [17], which are not always available.

The direct modeling approach does not remove the instrumental music, but rather makes use of it. Stoller *et al.* [12] developed a system based on a Wave-U-Net architecture to predict character probabilities directly from raw audio. This system performed well for the task of lyrics-to-audio alignment; however, it showed a high word error rate (WER) in lyrics transcription. Demirel *et al.* [14] introduced a multistream time-delay neural network based on the Kaldi hybrid architecture. Gupta *et al.* [13] proposed a music-informed acoustic model that incorporated music genre-specific information into acoustic models. This approach outperforms all other systems in the MIREX 2020 lyrics transcription task [18]. The study in [13] suggests that lyrics acoustic models can benefit from musical information. e.g., music genre, available in the instrumental music.

Both chords and lyrics are important elements of polyphonic music. A chord is a combination of notes that sound simultaneously or nearly simultaneously [19]–[23]. Chords play an important role in music [24], as they carry information useful for determining the structure of the music and represent the musical harmonic context and rhythmic properties [20]. Chords in sheet music also provide information so that the piece can be reproduced [20]. Therefore, chord transcription [19], [25]–[32] has attracted great attention for applications such as music segmentation [33], [34] and music structure analysis [26].

Chords and lyrics have some similar properties in most cases. First, chords belong to a finite set, which is similar to the format of the finite vocabulary of lexical words. Second, in polyphonic music audio, a set of chords is used to form chord progressions with distinctive harmonic flow [35], similar to the combination of lexical words producing meaningful sentences. Third, the start of a chord usually synchronizes with the start of a word, and a chord may be sustained over multiple lyrical words [36]. Mauch *et al.* [36] attempted to exploit this correlation between chords and lyrics for the purpose of lyrics-to-audio alignment in polyphonic music, and they achieved boosting accuracy from 38.4% to 88.0% with the aid of chord transcription. They suggested that chords and lyrics complement each other in a manner of representing and bridging musical and lyrical information in polyphonic music.

In this paper, we propose a novel end-to-end framework for lyrics and chord transcription, denoted as *multi-transcriber*, which shares a common encoder but has two separate decoders. We optimize the multi-transcriber with a multi-task learning strategy. At run-time, the multi-transcriber network concurrently transcribes lyrics and chords directly from input polyphonic music.

This work is motivated by several previous findings: a) singing vocals are correlated with music accompaniment; therefore, appropriate joint feature representation between the two could be helpful for lyrics transcription; b) the direct modeling approach that optimizes the system for the overall objective of lyrics transcription outperforms the extraction-transcription approach, which highlights the advantage of jointly modeling lyrics and chords; c) lyrics transcription in polyphonic music resembles the problem of speech recognition in a multi-talker environment [37]–[39], where studies show that it is possible to decode overlapped speech streams concurrently without explicitly separating them.

The contributions of this paper include the following:

- We propose a novel solution to the lyrics transcription problem through a joint lyric-chord decoding strategy.
- We formulate and validate an end-to-end transformer-based framework for lyrics transcription for the first time.
- We achieve state-of-the-art lyrics transcription performance for both monophonic music (singing vocals) and polyphonic music (singing vocals mixed with music accompaniment).

The rest of this paper is organized as follows. In Section II, we present the related work to set the stage for this study. Section III introduces the direct modeling pipelines for lyrics and chord transcription. In Section IV, we formulate the multi-transcriber network in detail. In Section V, we present the experimental setup and a set of reference models for comparison. In Section VI, we discuss the experimental results. Finally, Section VII concludes the study.

II. Related Work

Singing vocals differ from speech in many ways [3], [40]. They exhibit a larger pitch range and higher average pitch than speech [41]. For example, the typical fundamental frequency for female speech is between 165 and 200 Hz, while that for singing vocal can reach more than 1 kHz [42]. Other differences are reflected in the duration, pronunciation and vibrato characteristics in singing [42]. Furthermore, in polyphonic music, singing vocals are mixed with instrumental music; therefore, lyrics transcription in polyphonic music becomes more challenging than traditional speech recognition.

We begin by looking into multi-talker speech recognition to motivate our lyrics-chord multi-transcriber and then review the prior studies related to transformer-based solutions and multi-task learning to prepare for the formulation of the proposed multi-transcriber.

A. Decoding Mixed Signals

Polyphonic music audio mostly contains lyrics as well as chords. Humans are able to perceive the lyrical sentence as well as the chord progression together and individually while listening to music. The decoding of lyrics and chords resembles speech recognition of a two-talker speech utterance, i.e., two streams of overlapped speech. On the other hand, lyrics and chords are temporally correlated. For example, syllables are assigned to notes, while chord changes usually occur on strong beats. There is a temporal relationship between notes and beats in music. Therefore, joint modeling of two signals could potentially facilitate the decoding.

A successful engineering solution to multi-talker speech recognition [37] employs an attention-based encoder-decoder architecture with multiple decoders, one for each speaker. It assumes that the speech by multiple speakers shares the same vocabulary. In a two-talker situation, the order of speakers matters during training and testing. By making assumptions about the speech mixture, one may implement the decoding of mixed signals in a certain order, for example, by keeping a known speaker [43] on one side or ordering the speakers according to the speech energy [44]. In the case of recognizing speech from two random speakers, a *permutation invariant training* [37] strategy would be more suitable in a parallel decoding system. Permutation invariant training (PIT) [37] was first studied under the parallel framework, which considers all possible speaker permutations during training [37], [45]–[47]. As it assumes a fixed number of speakers in advance, it limits the scope of applications.

We note that lyrics-chord transcription is similar to two-talker speech recognition in terms of transcribing two overlapping signals. For example, we may design a parallel network to decode lyrics and chords in a fixed order, where lyrics and chords resemble two known speakers in a mixture. However, singing vocals and music differ much more than two speakers.

For example, they have different spectral energy distributions, and they do not share a common vocabulary. Nonetheless, the study of multi-talker speech recognition provides inspiration for our study of decoding mixed signals.

B. Multi-Task Learning

The instrumental accompaniment in polyphonic music is not random background noise. The accompaniment provides a rhythmic and harmonic support structure to the main melody of a song. It can be either described by metadata descriptors or modeled in a data-driven manner. For lyrics transcription to benefit from the musical structure, it is clear that we should not treat the instrumental music as background noise in the same way as in noisy speech recognition [48]–[50]. Chords are one of the fundamental building blocks of the tonal system [20] and can be described as musical words, analogous to the lexical words in lyrics. A sequence of chords outlines the harmonic flow of a piece of music [20]. Lyrics and chords are two essential pieces of information in polyphonic music. In this work, we investigate whether we can model the acoustic units of lyrics and chords in one system to improve the performance of lyrics transcription.

Multi-task learning (MTL) is an inductive transfer approach that uses domain information contained in the training signals of related tasks as an inductive bias. MTL is widely used in text classification [51], machine translation [52], language modeling [53], [54], and speech synthesis [55]–[57]. It offers multiple advantages over the single-task training paradigm, i.e., implicit data augmentation, attention focusing, eavesdropping, representation bias and regularization [58].

Drawing inspiration from MTL [58]–[60], we consider lyrics decoding and chord decoding as two single tasks that can be leveraged based upon one another. During training, we posit that chord transcription serves as an inductive bias to assist the main task of lyric transcription via inductive transfer from the musical domain to the lyrical domain.

C. Transformer-Based End-to-End Models

The traditional automatic speech recognition (ASR) implementation involves separate acoustic and language modeling [61]–[63]. Recently, end-to-end ASR systems have attracted great attention, where acoustic, lexical and linguistic data are modeled jointly in a single neural network that directly converts input speech to words [38]. The successful models include connectionist temporal classification (CTC) [64], [65] and sequence-to-sequence (S2S) models [66], [67]. The combination of CTC and S2S [68], [69] was also explored.

Transformer [70] is a popular encoder-decoder architecture that shows good performance in natural language processing [53], music generation [71], chord recognition [72], and speech recognition [73], [74]. Transformer-based multispeaker speech recognition shows better performance than RNN-based models in both single-channel and multi-channel scenarios [75]. One of the key components in the transformer is the self-attention layer, which computes the contributing information of the whole input sequence and implicitly learns the global time dependency while mapping the sequence into a vector at every time step [75]. This is highly desirable for lyrics transcription, wherein we would like to make use of the temporal context in singing vocals, beyond the current frame, to decode the lyrics, which will be the focus of this work. We are motivated to explore a transformer-based end-to-end solution to lyrics transcription.

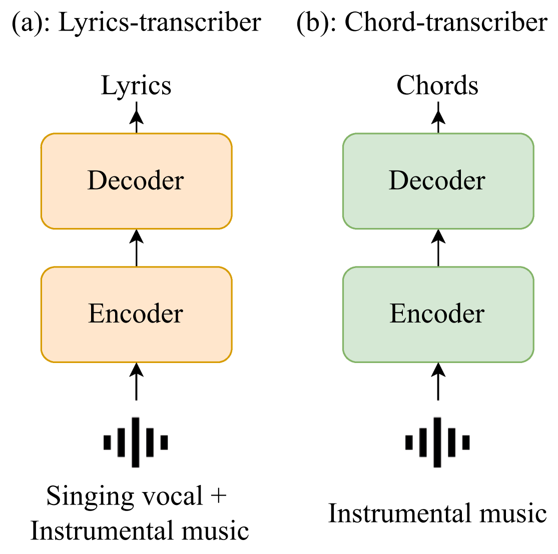


Fig. 1. An overview of the proposed (a) lyrics-transcriber and (b) chord-transcriber pipelines. The lyrics-transcriber is trained to decode lyrics from polyphonic music, and the chord-transcriber is trained to decode chords from instrumental music. The two transcribers are used as the pretrained model for the ultimate multi-transcriber, which decodes both lyrics and chords from polyphonic music in one instance.

III. Transformer-Based Lyrics and Chord Transcribers

Chords in music are defined as a group of notes played either simultaneously or in close succession [20] within a specified duration. Chord progression refers to the chords being played in succession that form the harmonic structure of the song. A sequence of chords played in a given song can be considered as a sequence of musical words. Lyrics, on the other hand, form the articulatory component of the singing vocals in a song. In this work, we first propose decoding either lyrics or chord sequences using a transformer-based direct modeling approach called a lyrics transcriber or chord transcriber. We adopt the same network architecture for the two transcribers; for brevity, we only describe a lyrics transcriber in detail.

The lyrics transcriber takes the mixed signal as input and generates lyrics as output. It can be considered as an end-to-end counterpart to the statistical approach [13]. The proposed lyrics transcriber is trained on polyphonic music with lyrics transcription [69], as shown in Fig. 1(a), and the chord transcriber is trained on instrumental music with chord transcription, as shown in Fig. 1(b). In either case, we do not seek to separate or extract the audio of interest but rather to model the signal to directly predict lyrics and chord transcription, respectively.

The lyrics-transcriber is based on joint encoder-decoder and connectionist temporal classification (CTC) architecture [69], where the encoder converts the input acoustic features to intermediate representations, and the decoder predicts lyrical

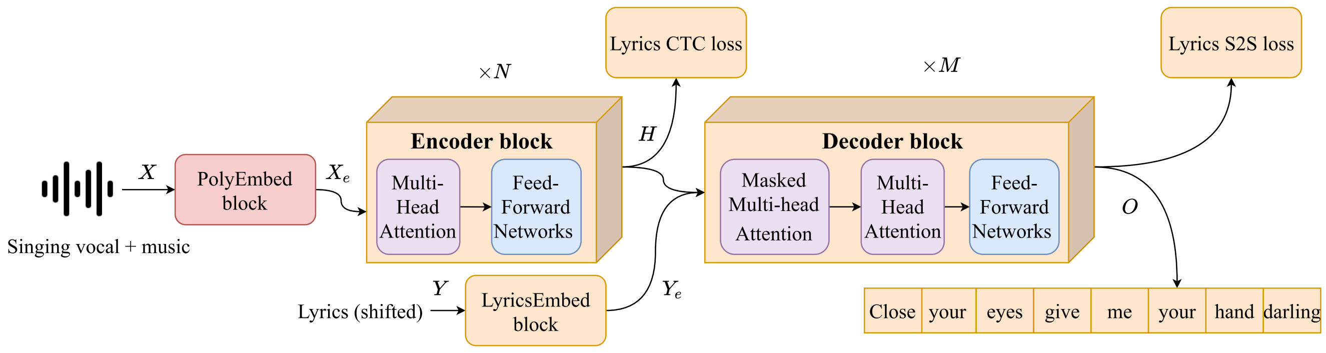


Fig. 2. A single-task lyrics-transcriber pipeline that learns to decode polyphonic music (singing vocal mixed with instrumental music) to lyrics. The same architecture is also used for chord-transcriber.

tokens, i.e. sub-words in this paper, one at a time given the intermediate representations and the previously predicted lyrical tokens in an autoregressive manner, as illustrated in Fig. 2.

The core module of the encoder and decoder is a multi-head attention (MHA) [70] that employs a self-attention mechanism to make use of the temporal context of the polyphonic music input sequence. Unlike the recurrent connections in RNN that require sequential iterations [76] or the use of local context with limited receptive fields in CNN [70], the advantage of using MHA is to jointly attend global polyphonic representation via different subspaces from multiple self-attention outputs.

1) Lyrics-Transcriber Encoder: The lyrics-transcriber encoder consists of an embedding block (PolyEmbed) and N identical encoder blocks, where each encoder block contains an MHA and a position wise feed-forward network (FFN). The input sequence $\mathbf{X}$ is first encoded into $\mathbf{X}_e$ by a PolyEmbed block using subsampling and positional encoding (PE) [70]. The encoder blocks then transform $\mathbf{X}_e$ into a hidden representation $\mathbf{H}$. Residual connection [77] and layer normalization [78] are employed inside each of the encoder blocks.

$$\begin{aligned}\mathbf{X}_e &= \text{PolyEmbed}(\mathbf{X}),\\ \mathbf{H} &= \text{EncoderBlocks}(\mathbf{X}_e)\end{aligned} \tag{1}$$

2) Lyrics-Transcriber Decoder: The lyrics-transcriber decoder consists of a textual embedding block (LyricsEmbed) and M identical decoder blocks, where each decoder block has a masked MHA, an MHA and an FFN. During training, $\mathbf{Y}$ represents the lyrical token history that is offset right by one position; however, during run-time inference, it represents the previous predicted token history. $\mathbf{Y}$ is first converted to lyrics token embedding $\mathbf{Y}_e$ via a LyricsEmbed block, which consists of an embedding layer and a positional encoding (PE) operation.

$$\begin{aligned}\mathbf{Y}_e &= \text{LyricsEmbed}(\mathbf{Y}),\\ \mathbf{O} &= \text{DecoderBlocks}(\mathbf{H}, \mathbf{Y}_e)\end{aligned} \tag{2}$$

The lyrics embedding $\mathbf{Y}_e$ is fed into the masked MHA that ensures causality, i.e., the predictions for the current position only depend on the past positions. The output of the masked MHA and the acoustic encoding $\mathbf{H}$ are then fed to the next MHA to capture the relationship between acoustic information $\mathbf{H}$ and textual information from the masked MHA. The residual connection [77] and layer normalization [78] are also employed inside each of the decoder blocks.

3) Lyrics-Transcriber Learning Objective: A combined CTC and S2S objective function is employed for model training. The CTC objective function helps supervise a monotonic alignment between the input polyphonic music, encoded into an acoustic representation at the output of the encoder, and the lyrical word sequence [69]. Another advantage of incorporating the CTC algorithm is to alleviate the need for explicit frame-level aligned lyrics, e.g., in the HMM model [69]. The network is trained to minimize both S2S and CTC losses jointly with an objective function $\mathcal{L}_{\text{lyric-trans}}$,

$$\begin{aligned}\mathcal{L}_{\text{lyric-trans}} &= \alpha\mathcal{L}^{\text{CTC}} + (1-\alpha)\mathcal{L}^{\text{S2S}},\\ \mathcal{L}^{\text{CTC}} &= \text{Loss}_{\text{CTC}}(\mathbf{G}_{ctc}, \mathbf{R}),\\ \mathcal{L}^{\text{S2S}} &= \text{Loss}_{\text{S2S}}(\mathbf{G}_{s2s}, \mathbf{R})\end{aligned} \tag{3}$$

where $\alpha \in [0, 1]$, and $\mathbf{R}$ is the ground-truth lyrical token sequence. The M decoder blocks are followed by linear projection and softmax layers that convert the decoder output $\mathbf{O}$ into a posterior probability distribution of the predicted lyrical token sequence $\mathbf{G}_{s2s}$. The S2S loss is the cross-entropy of $\mathbf{R}$ and $\mathbf{G}_{s2s}$. Additionally, a linear transform is applied on $\mathbf{H}$ to obtain the token posterior distribution $\mathbf{G}_{ctc}$. CTC loss is computed between $\mathbf{G}_{ctc}$ and $\mathbf{R}$ [69].

During run-time inference, the lyrics-transcriber directly converts input polyphonic acoustic features to output a lyrical token sequence without any explicit representation of phonetic or linguistic constructs, in a similar way as an end-to-end automatic speech recognition system does [69].

IV. Transformer-Based Multi-Transcriber

As discussed in Section II-B, lexical words (lyrics) and musical words (chords) can be considered analogous acoustic units in polyphonic music. Note that we consider the analogy of chords and lyrical words in a measurement sense compared to phonemes and letters intuitively, and we do not imply that a chord is strictly a harmonic word in music theory as a lyrical word is in language [35].

In this work, for the first time, we explore a joint vocabulary that covers both chords and lyrical words. This approach enables

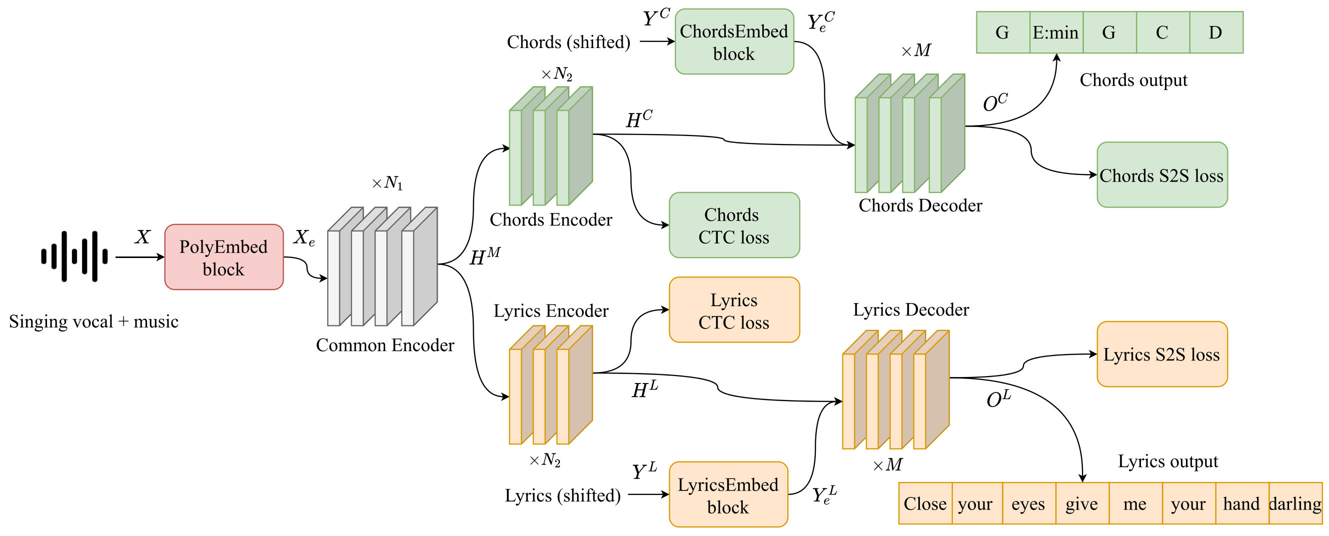


Fig. 3. The network architecture of the multi-transcriber with outputs in a fixed order (MT-FO), where we propose two separate encoder-decoder pathways for lyrics and chords.

the framework to decode both lyrics and chords from an input polyphonic music audio sample in the same way as a multi-talker ASR decodes overlapping speech from two speakers.

We propose two model implementations for lyrics transcription in which lyric and chord outputs are decoded concurrently. The models are based on a transformer network architecture referred to as a multi-transcriber. What motivates the multi-transcriber design is for the models to learn to disentangle lyrics and chord signals before respective decoding. If we compare the proposed multi-transcriber with the single-task transcriber in Section III, the multi-transcriber is equipped with a specific encoder-decoder pathway for each lyric and chord. Both pathways are trained under a multi-task joint supervised learning framework, while the single-task lyrics transcriber is not provided with chord supervision during training.

We consider two possible network architectures for a multi-transcriber, which takes polyphonic music as input and generates two output sequences, namely, lyric and chord sequences. One network architecture is designed to explicitly encode and decode chords and lyrics separately in a specific order. Another considers that the lyric or chord outputs are derived from one large vocabulary, therefore allowing for order permutation between the two output sequences during training and testing. We next investigate both network architectures.

A. *Multi-Transcriber With Outputs in a Fixed Order*

We propose a transformer-based multi-transcriber, as illustrated in Fig. 3, that generates two designated output streams for lyrics and chords in a fixed order; thus, it is denoted MT-FO. It consists of a common encoder, a lyrics encoder-decoder pathway, and a chord encoder-decoder pathway. The network architecture is motivated to benefit from the best aspects of both the direct modeling and extraction-transcription approaches.

The MT-FO network models singing vocals and music in two separate pathways to explicitly encode and decode lyrics and chords. Moreover, the fundamental frequency component and harmonic partials of instrumental accompaniment [20] are known to highly correlate with the unaccompanied singing both in time and frequency. Thus, we believe that musical chords provide useful information that would help in lyrics transcription.

1) Encoders: We incorporate a common encoder in the front end to benefit from this correlation. The common encoder is expected to perform holistic information extraction from the mixed input. It is trained on both lyrics and chord transcription tasks. What is learned for the chord transcription task is expected to aid the lyrics transcription task.

The encoder of the single-task lyrics transcription pipeline of Fig. 2 must accommodate the differences between singing vocals and nstrumental music for lyrics transcription. In the multi-transcriber framework, in addition to the common encoder, we introduce two output-specific encoders, one for lyrics and another for chords. The multi-head attention inside the encoder learns to encode the output-specific information via backpropagation training.

The multi-transcriber seeks to transcribe the input polyphonic acoustic feature sequence $\mathbf{X}$, the mixed input, to a lyric and chord sequence.

$$\begin{aligned}
\mathbf{X}_e &= \text{PolyEmbed}(\mathbf{X}),\\
\mathbf{Y}_e^L &= \text{LyricsEmbed}(\mathbf{Y}^L),\\
\mathbf{Y}_e^C &= \text{ChordsEmbed}(\mathbf{Y}^C),\\
\mathbf{H}^M &= \text{CommonEncoder}(\mathbf{X}_e),\\
\mathbf{H}^L &= \text{LyricsEncoder}(\mathbf{H}^M),\\
\mathbf{H}^C &= \text{ChordsEncoder}(\mathbf{H}^M),\\
\mathbf{O}^L &= \text{LyricsDecoder}(\mathbf{H}^L, \mathbf{Y}_e^L),\\
\mathbf{O}^C &= \text{ChordsDecoder}(\mathbf{H}^C, \mathbf{Y}_e^C)
\end{aligned} \tag{4}$$

where $\mathbf{Y}^L$ and $\mathbf{Y}^C$ are the lyrics label tokens and chords label tokens, respectively. The acoustic feature sequence $\mathbf{X}$ from polyphonic music input is first encoded to $\mathbf{X}_e$ via a PolyEmbed block for the common encoder to generate a global representation $\mathbf{H}^M$ of the mixed input. The musical representation $\mathbf{H}^C$ and the vocal representation $\mathbf{H}^L$ are then obtained using the chord encoder and the lyrics encoder, respectively.

2) Decoders: For lyrics transcription, as shown in Fig. 3, the lyrics decoder takes the hidden vocal representations $\mathbf{H}^L$ from the lyrics encoder and the previously predicted lyrical tokens $\mathbf{Y}^L$ to predict the subsequent lyrical tokens. $\mathbf{Y}^L$ is first converted to lyrics token embedding $\mathbf{Y}_e^L$ via the LyricsEmbed block. Then, the lyrics decoder is employed to predict the next tokens according to the previous lyrical token $\mathbf{Y}_e^L$ and the current vocal input $\mathbf{H}^L$.

Similarly, for chord transcription, the chord decoder takes the hidden music representations $\mathbf{H}^C$ from the chord encoder and the previously predicted chords tokens $\mathbf{Y}^C$ to predict the subsequent chords tokens. $\mathbf{Y}^C$ is first converted to chords token embedding $\mathbf{Y}_e^C$ via the ChordsEmbed block. The chord decoder is then employed to predict the next tokens according to the previous chords token $\mathbf{Y}_e^C$ and the current chord representation $\mathbf{H}^C$.

3) Multi-Task Training and Decoding: During the training stage, we use the ground-truth lyrics and chord labels, $\mathbf{Y}^L$ and $\mathbf{Y}^C$, in a teacher-forcing fashion. At the inference time, $\mathbf{Y}^L$ and $\mathbf{Y}^C$ are the previously predicted labels. The common encoder, the lyric encoder and the chord encoder contain $\mathbf{N}_1$, $\mathbf{N}_2$ and $\mathbf{N}_2$ identical encoder blocks, respectively. Both the lyrics decoder and the chord decoder have $\mathbf{M}$ identical decoder blocks. The encoder, decoder, PolyEmbed, and TokenEmbed blocks follow the same configuration as in Fig. 2.

Lyrics and chord transcription are jointly trained and optimized with fixed pathways for lyrics and chords, i.e., a fixed order using a multi-task learning objective $\mathcal{L}_{\text{MT-FO}}$, where what is learned in the chord transcription task is expected to aid the lyrics transcription task via the shared knowledge from the common encoder and the combined loss function. The musical and vocal representations are implicitly encoded and decoded from this integrated two-task framework to corresponding musical words and lyrical words. Specifically, we use MTL-based CTC and S2S objectives for training and decoding of each task as follows:

$$\begin{aligned}
\mathcal{L}_{\text{MT-FO}} &= \mathcal{L}_{\text{lyrics}} + \mathcal{L}_{\text{chords}},\\
\mathcal{L}_{\text{lyrics}} &= \alpha\mathcal{L}^{\text{CTC}}_{\text{lyrics}} + (1-\alpha)\mathcal{L}^{\text{S2S}}_{\text{lyrics}},\\
\mathcal{L}^{\text{CTC}}_{\text{lyrics}} &= \text{Loss}_{\text{CTC}}(\mathbf{H}^L, \mathbf{R}^L),\\
\mathcal{L}^{\text{S2S}}_{\text{lyrics}} &= \text{Loss}_{\text{S2S}}(\mathbf{G}^L, \mathbf{R}^L),\\
\mathcal{L}_{\text{chords}} &= \beta\mathcal{L}^{\text{CTC}}_{\text{chords}} + (1-\beta)\mathcal{L}^{\text{S2S}}_{\text{chords}},\\
\mathcal{L}^{\text{CTC}}_{\text{chords}} &= \text{Loss}_{\text{CTC}}(\mathbf{H}^C, \mathbf{R}^C),\\
\mathcal{L}^{\text{S2S}}_{\text{chords}} &= \text{Loss}_{\text{S2S}}(\mathbf{G}^C, \mathbf{R}^C)
\end{aligned} \tag{5}$$

where $\alpha \in [0,1]$, $\beta \in [0,1]$, $\mathbf{R}^L$ and $\mathbf{R}^C$ are the reference lyrics and chord token sequences, respectively. $\mathbf{G}^L$ and $\mathbf{G}^C$ are the lyrics and chord token sequences predicted by operating linear projection and softmax based on their corresponding decoded outputs $\mathbf{O}^L$ and $\mathbf{O}^C$, respectively.

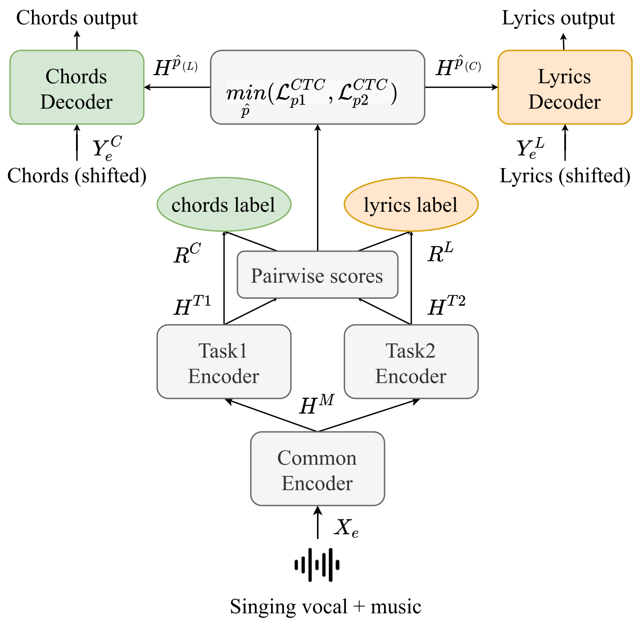


Fig. 4. The network architecture and training workflow of a multi-transcriber with permutation invariant outputs (MT-PI). The main difference between MT-PI and MT-FO lies in the fact that the former employs two task-interchangeable encoders, while the latter employs two task-specific encoders.

B. Multi-Transcriber with Permutation Invariant Outputs

Considering that the lyrics and chord tokens are derived from the same vocabulary, we do not explicitly model the acoustic models of lyrics and chords separately. The task is to decode the mixed input into two separate output streams, each of which contains either lyrics or chords, in a homogeneous fashion.

1) Network Architecture: This setup is similar to the classic problem of label ambiguity in multi-talker speech recognition, where we do not consider the order of speakers in the output streams [43], [44]. We call the encoding-decoding process of either lyrics or chords a task. We employ two encoders that are task-interchangeable but two decoders that are task-specific. In practice, we adopt a permutation invariant training (PIT) module [45] to address the permutation ambiguity during training, as shown in Fig. 4. The PIT module first determines the output-target assignment with the minimum error at the utterance level based on the forward-pass results. It then minimizes the error given the assignment and backpropagates.

To generate lyrics and chords in two separate output streams, we propose two task-specific decoders for lyrics and chords, respectively. In summary, the MT-PI model consists of two task-interchangeable encoders for lyrics and chord recognition for permutation selection and two task-specific decoders, each of which works for one of the tasks.

2) Multi-Task Training and Decoding: During the training stage, as the encoders are task interchangeable, we need to decide the exact encoder-decoder pathway for each input mixed signal. By comparing the two possible output-target pairs, as shown in Fig. 4, with outputs in the order of either lyrics-chords or chords-lyrics, we select the one with minimum CTC loss as the encoder-decoder pathway to optimize the model. The flow of data is further explained next.

$$\begin{aligned}
\mathbf{H}^{M} &= \text{CommonEncoder}(\mathbf{X}_e),\\
\mathbf{H}^{T1} &= \text{Task1Encoder}(\mathbf{H}^{M}),\\
\mathbf{H}^{T2} &= \text{Task2Encoder}(\mathbf{H}^{M}),\\
\mathcal{L}^{\text{CTC}}_{\text{p1}} &= \text{Loss}_{\text{CTC}}(\mathbf{R}^{L},\mathbf{H}^{T1}) + \text{Loss}_{\text{CTC}}(\mathbf{R}^{C},\mathbf{H}^{T2}),\\
\mathcal{L}^{\text{CTC}}_{\text{p2}} &= \text{Loss}_{\text{CTC}}(\mathbf{R}^{L},\mathbf{H}^{T2}) + \text{Loss}_{\text{CTC}}(\mathbf{R}^{C},\mathbf{H}^{T1}),\\
\mathcal{L}^{\text{CTC}}_{\text{minPIT}} &= \min_{\hat{p}}(\mathcal{L}^{\text{CTC}}_{\text{p1}},\mathcal{L}^{\text{CTC}}_{\text{p2}}),\\
\mathbf{O}^{L} &= \text{LyricsDecoder}(\mathbf{H}^{\hat{p}(L)},\mathbf{Y}^{L}_{e}),\\
\mathbf{O}^{C} &= \text{ChordsDecoder}(\mathbf{H}^{\hat{p}(C)},\mathbf{Y}^{C}_{e}),\\
\mathcal{L}_{\text{MT-PI}} &= \gamma\mathcal{L}^{\text{CTC}}_{\text{minPIT}} + (1-\gamma)(\mathcal{L}^{\text{S2S}}_{\text{lyrics}} + \mathcal{L}^{\text{S2S}}_{\text{chords}}) \qquad (6)
\end{aligned}$$

where $\gamma \in (0,1)$ and two task encoders transform inputs $\mathbf{H}^{M}$ into two task-related representations $\mathbf{H}^{T1}$ and $\mathbf{H}^{T2}$ for lyrics and chords.

We compute the CTC loss for two possible permutations of the output labels with $\mathcal{L}^{\text{CTC}}_{\text{p1}}$ and $\mathcal{L}^{\text{CTC}}_{\text{p2}}$, pairwise, and pick the one with the minimum CTC loss $\mathcal{L}^{\text{CTC}}_{\text{minPIT}}$ as the selected permutation $\hat{p}$ for backpropagation of gradients.

In Fig. 4, $H^{\hat{p}(L)}$ is the lyrics-related representation for lyrics decoding, and $H^{\hat{p}(C)}$ is the chords-related representation for chord decoding. The common encoder, task1 encoder, and task2 encoder contain N_1, N_2 and N_2 encoder blocks, respectively. Both the lyrics decoder and chord decoder consist of M blocks.

The task-specific MT-PI loss $\mathcal{L}_{\text{MT-PI}}$ is a combination of the CTC permutation loss $\mathcal{L}^{\text{CTC}}_{\text{minPIT}}$ and S2S objectives, where the S2S loss is calculated according to (5). By doing so, we ensure that the right labels are used as the supervision signal for each encoder-decoder pathway, regardless of the actual order between the lyric and chord output streams.

At the inference time, as we do not know which encoder works best for lyrics or chords, we use both encoder outputs as the inputs to each of the task-specific decoders to generate the lyrics and chords hypotheses. Finally, we evaluate the two encoder outputs and choose the one with the better score as the final pathway following the traditional PIT [45] implementation.

V. Experimental Setup

A. Dataset

We prepare a polyphonic music dataset and a solo-singing dataset, i.e. singing vocals without instrumental music, for the experiments.

1) Polyphonic Music Dataset: As shown in Table I, the polyphonic music training dataset, Poly-train, consists of the DALI-train [79] dataset and an NUS proprietary collection. The DALI-train dataset consists of 3,913 English polyphonic audio tracks.[1] The dataset is processed into 180,034 lyrics-transcribed audio lines with a total duration of 208.6 hours. The NUS collection dataset consists of 517 popular English songs. We obtain its line-level lyrics boundaries using the state-of-the-art audio-to-lyrics alignment system [13], leading to 26,462 lyrics-transcribed audio lines with a total duration of 27.0 hours. In this paper, English songs refer to songs of English lyrics that may come from a variety of origins.

TABLE I
A Description of a Polyphonic Music Dataset That Consists of DALI and NUS Collections

		# songs	# lines	duration
Poly-train	DALI-train	3,913	180,034	208.6 hours
	NUS	517	264,62	27.0 hours
Poly-dev	DALI-dev	100	5,356	3.9 hours
	NUS	70	2,220	3.5 hours
Poly-test	Hansen	10	212	0.5 hour
	Jamendo	20	374	0.9 hour
	Mauch	20	442	1.0 hour

TABLE II
A Description of the Solo-Singing Dataset

Name	# songs	# lines	duration
Solo-train	4,324	81,092	149.1 hours
Solo-dev	66	482	0.7 hours
Solo-test	70	480	0.8 hours

The Poly-dev dataset consists of the DALI-dev dataset of 100 songs from the DALI dataset [13] and 70 songs from an NUS proprietary collection. We adopt three widely used test sets – Hansen [80], Jamendo [12], and Mauch [36] – to form the Poly-test, as shown in Table I. The test datasets are English polyphonic songs that are manually segmented into line-level audio segments of an average of 8.126 seconds, each of which is called an audio line. We transcribe the lyrics line-by-line to avoid possible accumulated errors in the Viterbi decoding due to long audio clips in whole song audio samples [81], [82]. We have manually verified and ensured the correctness of these line-level segments and their corresponding transcriptions.[2]

2) Solo-Singing Dataset: We study the use of pre-training on solo-singing for lyrics transcription. A curated version [6] of the English solo-singing dataset $Sing!300 \times 30 \times 2$[3] is adopted, and the details are listed in Table II. A recent study [16] reports the state-of-the-art performance [6] on this dataset, which serves as a good performance reference. The training set Solo-train consists of 4,324 songs with 81,092 audio lines. The development set Solo-dev and the test set Solo-test contain 66 songs and 70 songs with 482 and 480 audio lines, respectively. The lyrics of all datasets are manually transcribed.

3) Chord Data: Chord transcriptions are not available in the training data. Therefore, we obtain pseudo chord labels of the polyphonic music dataset, as summarized in Table I,

[1]There are a total of 5,358 audio tracks in DALI, but we only have access to 3,913 English audio links.

[2]The line-level segmented Hansen, Mauch and Jamendo test sets will be made available to the public.

[3]The audio files can be accessed from https://ccrma.stanford.edu/damp/

TABLE III
THE AVERAGE NUMBER OF LYRICAL WORDS AND CHORDS PER LINE, AND THE AVERAGE DURATION PER LYRICAL WORD AND CHORD IN THE POLYPHONIC MUSIC DATASET

Statistics	Poly-train	Poly-dev	Poly-test
# lyrical words per line	5.30	5.43	12.90
# chords per line	2.89	2.78	3.96
Lyrical word duration (seconds)	0.91	0.73	0.72
Chord duration (seconds)	1.63	1.50	3.09

by applying an automatic chord recognition algorithm [72] to the source-separated instrumental music [17]. The chord recognition algorithm uses a bidirectional transformer and achieves good performance with a weighted chord symbol recall score of 83.1 for the major-minor chord label [72]. We use major-minor chord label types [72], [83] to form 25 chord characters (12 semitones for major and minor, and No chord) as the chord-transcriber token units. The statistics of the pseudo chord labels and lyrical words in our datasets are reported in Table III. We note that it would be helpful to have access to a realistic chord and lyrics ground-truth transcription database for future multi-task learning studies.

B. Comparison of Models

We use ESPnet [84] with PyTorch backend to build acoustic models for both single-task and multi-task frameworks, as shown in Table IV. The context and purpose of each of these models and the related experiments are elaborated in Section VI.

We extract 83-dimensional Filterbank features (fbank) with pitch from audio files with a window of 25 ms, shifting every 10 ms. We use sub-words as the lyrics-transcriber token units for the task of lyrics transcription, and 5,000 sub-words are generated using byte-pair encoding (BPE) for the lyrics-transcriber pipeline. These 5,000 sub-words and the 25 chord characters are combined to form the 5,025 character-BPE modeling units in a shared vocabulary for all the multi-transcriber models.

All models are trained with the Adam optimizer with a Noam learning rate decay, 25,000 warmup steps, 5,000,000 batch bins, and 100 epochs, as in [70]. The PolyEmbed block contains two CNN blocks with a kernel size of 3 and a stride size of 2. The interpolation factor between CTC loss and S2S loss is tuned for our task. Single-task models have the same configuration, where there are 12 encoder blocks and 6 decoder blocks ($N = 12$ and $M = 6$). For multi-task models, there are a total of 6 encoder blocks in the common encoder, 6 encoder blocks in domain-related encoders and 6 decoder blocks in domain-related decoders ($N_1 = N_2 = 6$ and $M = 6$).

Other parameters of transformer-based encoders and decoders follow the default setting in the published LibriSpeech model (LS)[4], where the attention dim is 512, the number of heads is 8 in MHA and the FFN layer dim is 2,048. We follow the default setting in ESPnet [84] to average the 5 best validated model checkpoints on the corresponding development set as in Table IV to obtain the final acoustic model. We follow the common joint decoding approach [69], [85], which takes CTC prediction and LM model score into account during decoding by setting different LM weights and CTC weights. During decoding for different lyrics transcription models, we use the same default parameter settings (penalty, beam width and CTC decoding weight are set to 0.0, 10 and 0.3, respectively).

[4]See the pretrained librispeech model "PyTorch large Transformer with specaug (4 GPUs) + Large LSTM LM" from the ESPNET github https://github.com/espnet/espnet/blob/master/egs/librispeech/asr1/RESULTS.md.

VI. EXPERIMENTS AND RESULTS

We study the effects of initialization, multi-task learning strategy and model size. We also conduct a music genre analysis to better understand the effects of music genres on lyric transcription. We also compare the proposed models with state-of-the-art systems on both solo-singing vocals and polyphonic music. We report the lyrics transcription performance in terms of the word error rate (WER), which is the ratio of the total number of insertions, substitutions, and deletions with respect to the total number of words.

A. Single-Task Models and Initialization

We first study the single-task lyrics transcriber (LT) and chord transcriber (CT), along with the effect of initialization. Sung lyrics can be considered as a variant of spoken words, while singing vocals in polyphonic music are noisy versions of singing vocals in solo-singing. It makes sense to initialize a lyrics transcription model with either a speech recognition model trained for spoken words or with a solo-singing transcription model trained on solo-singing vocals.

First, we obtain a publicly available pre-trained speech recognition model (LS) based on the LibriSpeech dataset. Second, we train a solo-singing transcription model on $Sing!300 \times 30 \times 2$, which is detailed in Table II.

1) LT Initialized by Speech Model: network architecture is shown in Fig. 2. As described in Table IV, LT-V and LT-V-Raw differ in that the LT-V model is pre-trained on the LibriSpeech database as a speech recognition model, while the LT-V-Raw model does not rely on any pre-training. Both models are trained using fbank features with solo-singing training audio files (Solo-train) as training set and Solo-dev as development set.

To reflect the proportional contributions between CTC loss and S2S loss, we evaluate different α values for the multi-objective learning loss in (3) with the LT-V model and report them in Fig. 5. The value for α is empirically set to 0.3 based on the development set (WER 16.50%), which is used in all other experiments hereafter.

The experiments for solo-singing lyrics transcription, with/without pretraining, are reported in Table V, which suggests that pretraining on the LibriSpeech dataset is beneficial to the lyrics transcription of solo-singing.

2) LT Initialized by the Solo-Singing Model: We further investigate the effect of pre-training on solo-singing data for polyphonic music transcription. The LT-P-Raw and LT-P models share the same network architecture, as shown in Fig. 2. They differ in that the LT-P model is initialized with the LT-V model, while the LT-P-Raw is not.

TABLE IV
A SUMMARY OF THE SINGLE-TASK AND MULTI-TASK MODELS AND THEIR LYRICS TRANSCRIPTION RESULTS (WER%) ON BOTH POLY-TEST AND POLY-DEV DATASETS

Model	Train data	Dev data	Model initialization	Single-task vs. multi-task	Poly-dev	Poly-test		
						Hansen	Jamendo	Mauch
Speech recognition (LS)	LibriSpeech	LibriSpeech	–	Single-task	105.96	106.03	92.00	108.39
Lyrics transcriber (LT-V-Raw)	Solo-train	Solo-dev	–	Single-task	89.60	77.22	88.65	86.14
Lyrics transcriber (LT-V)	Solo-train	Solo-dev	LS	Single-task	72.07	65.16	64.82	67.10
Lyrics transcriber (LT-P-Raw)	Poly-train	Poly-dev	–	Single-task	59.19	70.74	76.19	58.59
Lyrics transcriber (LT-P)	Poly-train	Poly-dev	LT-V	Single-task	44.12	39.87	44.26	36.80
Lyrics transcriber (LT-P-Large)	Poly-train	Poly-dev	LT-V-Large	Single-task	47.39	42.73	47.34	39.33
Multi-transcriber w PI (MT-PI)	Poly-train	Poly-dev	LT-P, LT-V, CT-M	Multi-task	42.22	38.51	44.64	32.38
Multi-transcriber w FO (MT-FO-A)	Poly-train	Poly-dev	LS, LT-V, CT-M	Multi-task	42.99	47.68	51.91	38.15
Multi-transcriber w FO (MT-FO-B)	poly-train	poly-dev	LS, CT-M	Multi-task	42.49	46.56	51.22	36.86
Multi-transcriber w FO (MT-FO-C)	poly-train	poly-dev	LT-V, CT-M	Multi-task	41.93	45.38	48.11	36.34
Multi-transcriber w FO (MT-FO-D)	poly-train	poly-dev	FreLT-P, LT-V, CT-M	Multi-task	48.49	43.04	44.80	41.14
Multi-transcriber w FO (MT-FO)	Poly-train	Poly-dev	LT-P, LT-V, CT-M	Multi-task	**41.40**	**36.34**	**43.44**	**31.82**

FreLT-P denotes that the common encoder is initialized by the LT-P model, but we freeze the weights of the LT-P model during training.

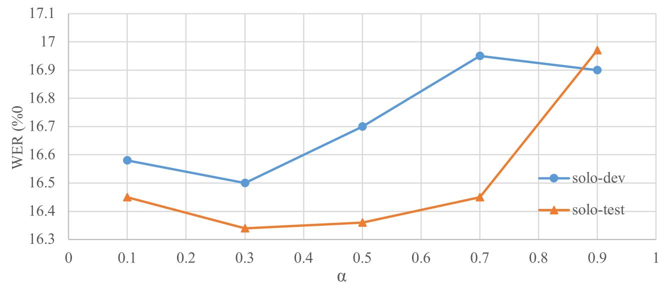


Fig. 5. Comparison of lyrics recognition (WER%) performance of the LT-V model with different α values for training. Solo-dev and Solo-test are used for evaluating LT-V as development and test sets, respectively.

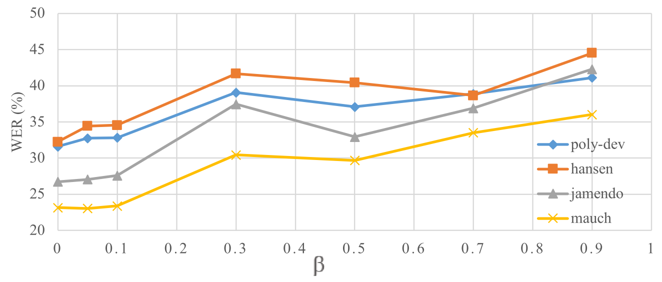


Fig. 6. Comparison of chord recognition (WER%) performance on a music model with different β for training.

TABLE V
COMPARISON OF SOLO-SINGING LYRICS RECOGNITION (WER%) PERFORMANCE WITH AND WITHOUT PRE-TRAINED MODEL

Solo-sing Model	Solo-dev	Solo-test
LT-V-Raw	22.70	21.70
LT-V	16.50	16.34

As shown in Table IV, the LT-P model significantly outperforms LT-P-Raw, which suggests that the adaptation from solo-singing to polyphonic music is helpful. Moreover, the LT-P model outperforms all other models, i.e. LS, LT-V-Raw and LT-V models, which also suggests significant mismatches between training and testing conditions for LS, LT-V-Raw and LT-V.

3) Single-Task Chord Transcriber: We develop a chord-transcriber CT-M, which is trained on instrumental music extracted from Poly-train. The network architecture of the chord transcriber follows the same configuration as in the LT-P model. We train the CT-M model by averaging the 5 best validated checkpoints on the pseudo Poly-dev set and evaluate the model on the pseudo chord labels of Poly-test.

To reflect the proportional contributions between CTC loss and S2S loss, we evaluate different β values for multi-objective learning loss and report them in Fig. 6, which shows that the best WER is achieved when $\beta = 0$. This suggests that the temporal alignment between chord labels and the acoustic features through CTC loss is not informative. This could be because the chords of long duration, as shown in Table III, are rendered in a structured manner. They can be decoded reasonably well without explicit temporal modeling. During decoding for chord recognition, we empirically set the penalty, beam width and CTC decoding weight to be 0.0, 10 and 0.0, respectively.

B. From Single-Task to Multi-Task

We now report the experiments on the proposed multi-transcriber (MT) with multi-task learning. The MT models are trained on both lyrics transcription and chord transcription tasks to improve the lyrics transcription performance. The results are summarized in Table IV.

1) Multi-Task Learning: We construct two multi-transcriber models: MT-FO, which generates output streams in a fixed order, and MT-PI, which generates permutation invariant outputs. Both models are trained on polyphonic music data.

The network architecture of MT-FO is illustrated in Fig. 3. Motivated by the findings regarding model initialization in Section III, we propose that the lyrics encoder and lyrics decoder are initialized by the LT-V model, the chord encoder and chord decoder are initialized by hlthe CT-M model, and the common encoder is initialized by the LT-P model. Following the setting in Section VI-A, we set $\alpha = 0.3$ and $\beta = 0.0$ in the MT-FO model training.

The network architecture of MT-PI is illustrated in Fig. 4, of which the task1 encoder and lyrics decoder are initialized by the LT-V model, the task2 encoder and chord decoder are initialized by the CT-M model, and the common encoder is initialized by the LT-P model. We empirically set $\gamma = 0.3$, as defined in (6), to reflect the contributions between CTC loss and S2S loss in the MT-PI model.

2) Multi-Task Vs. Single-Task: We now compare multi-task learning with the single-task learning strategy. As shown in Table IV, multi-transcribers, e.g., MT-FO and MT-PI, consistently outperform their single-task counterparts, e.g., LT-P and LT-P-Raw, across the development and test sets. In the multi-transcribers, we have lyrics transcription as the main task and chord transcription as the secondary task. The main task benefits from incorporating the secondary task under the multi-task learning framework. We observe that the MT-FO model outperforms the MT-PI model, which highlights the advantage of having two dedicated encoder-decoder pathways for lyrics and chords. The MT-FO model shows the best lyrics transcription performance among all the models.

3) Model Initialization for Ablation Study: We also evaluate four alternate initialization strategies for the ablation study of the MT-FO model, as presented in Table IV, where we set $\alpha = 0.3$ and $\beta = 0.0$ for the MT-FO-A, MT-FO-B, MT-FO-C and MT-FO-D models. Unlike in the MT-FO model, the common encoder of MT-FO-A is initialized by the LS model, and both the common encoder and lyrics encoder of MT-FO-B are initialized by the LS model. The common encoder and lyrics encoder of MT-FO-C are initialized by the LT-V model. For the MT-FO-D model, the common encoder is initialized by the LT-P model, and the rest are the same as the MT-FO model, but we freeze the weights of the common encoder during training to test the effectiveness of the common encoder.

We can determine that the MT-FO model consistently outperforms the MT-FO-A and MT-FO-B models, which shows the advantage of the initialization with the LT-P model over the LS model. Note that the LT-P model is a single task lyrics transcriber for polyphonic music that matches the test condition of MT-FO well, while the LS model is a speech recognition model. We observe that MT-FO outperforms MT-FO-C, which indicates that the common knowledge from LT-P is more beneficial for lyrics transcription of polyphonic music than the solo-singing knowledge from LT-V. We can see that MT-FO also performs better than MT-FO-D, which confirms the importance of incorporating the common encoder for multi-task learning.

4) Chord Transcription: We also report the results of chord transcription in Table VI. It is observed that the CT-M model outperforms all the multi-transcriber models. We observe that MT-FO significantly outperforms MT-FO-D, which suggests that the common encoder in MT-FO plays an important role in multitask learning for chord transcription. It is worth noting that all the proposed models are fine-tuned to optimize the lyrics transcription main task. We initialize the common encoder with the LT-P model, where chord decoding is incorporated only to improve the lyrics transcription main task. In future work, we will further study the interaction between the two tasks. Other aspects of music, such as beats and melody, also have a close relationship with rendering of the lyrics of a song, defining the syllable rate and the pitch of the singing vocals [86]. Thus, we would also like to explore lyrics transcription in an MTL setting with concomitant tasks such as drum transcription and note transcription [87]–[89] in the future.

TABLE VI
POLYPHONIC CHORD RECOGNITION (WER%) RESULTS USING THE SINGLE-TASK MODEL TRAINED ON INSTRUMENTAL MUSIC AND THE PROPOSED MODELS

Model	Poly-dev	Poly-test		
		Hansen	Jamendo	Mauch
CT-M	31.60	32.24	26.74	23.16
MT-PI	38.33	38.52	36.66	29.10
MT-FO-A	37.47	38.93	36.29	30.10
MT-FO-B	37.47	39.34	37.09	29.81
MT-FO-C	38.10	39.07	36.35	30.16
MT-FO-D	83.47	89.07	87.27	80.42
MT-FO	38.48	37.02	36.66	31.22

TABLE VII
COMPARISON OF SPEECH RECOGNITION AND LYRICS RECOGNITION (WER%) PERFORMANCES BETWEEN LARGE AND SMALL MODELS

Speech models	**LS**	**LS-Large**
dev-clean	3.29	3.64
dev-other	8.49	9.08
test-clean	3.63	3.89
test-other	8.35	8.90
Solo-sing models	**LT-V**	**LT-V-Large**
Solo-dev	16.50	16.55
Solo-test	16.34	17.47

C. Effect of Model Size

It is noted that the multi-transcriber (MT-FO) involves two pathways, each of which roughly follows the same network configuration as a single-task transcriber (LT-P). As a result, the MT-FO model has more parameters than the LT-P model. To answer the question of whether the performance gain of MT-FO over LT-P is due to the increased number of parameters or the multi-task learning strategy, we train a larger lyrics-transcriber model for LT-P, i.e., LT-P-large. The model size of LT-P-large is approximately twice that of LT-P and is therefore comparable with that of the MT-FO model, as shown in Table IV.

We first develop two large pre-trained models, i.e. LS-large and LT-V-large, each of which has the same number of model parameters as LT-P-large. The LS-large model is trained and tested on the LibriSpeech corpus [90] in the same way as the LS model is trained and tested, and dev-clean, dev-other, test-other and test-clean are standard development and test sets to evaluate the performance. The LT-V-large model is trained on the solo-singing data in the same way as the LT-V model. Specifically, LS-large is used to initialize LT-V-large, which is further used to initialize the LT-P-large model.

In particular, the LS/LT-V model has 512 nodes in the attention, 8 heads in the multi-head attention, and 2,048 nodes in the FFN layer. The LS-Large/LT-V-Large model has 1,024 attention nodes, 16 heads in the multihead attention, and 4,096 nodes in the FFN layer. In Table VII, we can observe that the LS-large model shows slightly worse performance than the LS model after 64 epochs of training, and the performance of LT-V-Large

TABLE VIII
Genre Distribution of Polyphonic Music Training Set and Test Sets

		# of songs	**Genre distribution**
Poly-train	DALI-train	3,913	hip-hop: 119, metal: 1,576, pop: 2,148, NA: 70
	NUS	517	hip-hop: 100, metal: 20, pop: 397
Poly-test	Hansen	10	hip-hop: 1, metal: 3, pop: 6
	Jamendo	20	hip-hop: 4, metal: 7, pop: 9
	Mauch	20	hip-hop: 0, metal: 8, pop: 12

TABLE IX
Lyrics Transcription Performance (WER%) by Music Genre on the Poly-Test Test Set for Three Models: LT-P, MT-FO, and MT-PI

Statistics	**metal**	**pop**	**hip-hop**
# songs in Poly-test	17	28	5
Ratio: # chords / # lyrics	0.36	0.42	0.23
Polyphonic models	**metal**	**pop**	**hip-hop**
LT-P	50.04	36.52	51.19
MT-FO	47.86	32.83	52.41
MT-PI	48.84	33.61	54.81

TABLE X
Comparison Between the Proposed End-to-End Solutions and Other Existing Competitive Solutions to Lyrics Transcription (WER%) of Solo-Singing

End-to-End Models	**Solo-dev**	**Solo-test**
LT-V	16.50	16.34
LT-V + RNN LM	13.86	14.92
LT-V + RNN LM-LS	**13.96**	**13.46**
Kaldi-based Models	**Solo-dev**	**Solo-test**
TDNN + 4-gramLM [6]	23.33	19.60
CTDNN + 4-gramLM [16]	21.08	17.70
CTDNN_SA + 4-gramLM [16]	20.38	17.01
CTDNN_SA + RNN LM rescore [16]	18.74	14.79
MSTRE-Net [14]	-	15.38

TABLE XI
Comparison Between the Proposed End-to-End Solutions and Other Existing Competitive Solutions to Lyrics Transcription (WER%) of Polyphonic Music With Respect to Whole Song Testing

Kaldi-based Models	**Hansen**	**Jamendo**	**Mauch**	**Dali-test**
RB1 [91]	83.43	86.70	84.98	-
DDA2 [92]	74.81	72.15	75.39	-
DDA3 [92]	77.36	73.09	80.66	-
CG [13]	-	59.60	44.00	-
GGL2 [18](+scoring)	48.11	61.22	45.35	-
GGL1 [18](+scoring)	45.87	56.76	43.76	-
MSTRE-Net [14]	**36.78**	**34.94**	37.33	42.11
End-to-End Models	**Hansen**	**Jamendo**	**Mauch**	**Dali-test**
DS [12]	-	77.80	70.90	-
LT-P	40.02	45.19	38.96	46.56
MT-PI	39.44	45.07	34.57	43.39
MT-FO	36.85	44.12	**33.69**	**40.20**

The Results of GGL1 (+scoring) and GGL2 (+scoring) are Obtained by the Standard Kaldi Recipe With Scoring, Which are Slightly Different From Those Found on the MIREX2020 Website

is also comparable with that of LT-V. The results suggest that a larger model does not lead to improved performance.

As shown in Table IV, LT-P-Large achieves similar performance as that of LT-P, and it does not outperform the MT-FO model. This confirms that the performance gain by multi-transcriber MT-FO over the single-task lyrics transcribers is due to the additional task of chord transcription, as opposed to the increased model size.

D. Music Genre Analysis

We analyze the performances for different music genres based on the polyphonic test sets – Hansen, Jamendo and Mauch. The music genre distribution in Poly-train and Poly-test is summarized in Table VIII. The Poly-test dataset consists of three genres – pop, hip-hop and metal, as given in [13]. We report the lyrics transcription performance by music genre for three high-performance models, namely, LT-P, MT-FO, and MT-PI, in Table IX.

We observe that MT-FO outperforms LT-P for pop and metal songs, while MT-FO is comparable with the LT-P model for hip-hop songs (only 5 hip-hop songs exist in Poly-test). One reason for this is that the total number of pop and metal songs in the training set (Poly-train) is considerably greater than that of hip-hop songs. The lyrics transcription performance for different music genres is also indicative of the situations for which the additional chord recognition task would not be beneficial for the lyrics transcription task. For example, hip-hop songs consist of segments with rap that ontain many words at a high syllable rate [13] and can be considered *lyrics-dominating* songs. The ratio of the number of chords to the number of words in a given utterance would be low for hip-hop songs compared to pop or metal songs, as shown by the statistics of the test dataset in Table IX. We would expect that such lyrics-dominating songs would not benefit from chord information as much as pop or metal songs would.

Moreover, as highlighted in [91], metal songs present louder accompaniments than jazz, country and pop songs in the detailed genre classes, and "Death Metal" receives a lyrics intelligibility score of zero. This may explain why pop songs show better lyrics transcription performance than metal songs. Hip-hop songs are also observed [91] to have a higher syllable rate and rapid vocalization, thereby exhibiting lower lyrics intelligibility than pop songs. This may be another reason why hip-hop songs have a higher word error rate than pop songs.

E. Comparison With the State-of-The-Art

We compare the proposed end-to-end solutions with the existing approaches for lyrics transcription. We report the results for solo-singing in Table X and polyphonic music in Table XI.

1) Lyrics Transcription of Solo-Singing: In this experiment, we evaluate our proposed end-to-end transformer-based approach for lyrics transcription of solo-singing audio with state-of-the-art reference models [6], [14], [16] in Table X. The proposed LT-V model, as described in Section VI-A1, is devised for lyrics transcription of solo-singing data with good performance, as shown in Table V. We would like to compare the LT-V model with three state-of-the-art reference models by Demirel *et al.* [16], Dabike *et al.* [6] and Demirel *et al.* [14], which present the state-of-the-art in the literature.

The reference models [6], [14], [16] in Table X are based on the Kaldi speech recognition engine, which involves several training steps, namely, GMM, HMM and factorized time-delay neural network (TDNN-F) [6], [14], [16]. Specifically, [6] adopts TDNN-F with a 4-gram language model named TDNN + 4-gramLM. [16] further incorporates CNN (CTDNN + 4-gramLM) and self-attention (CTNDD_SA + 4-gramLM) into TDNN-F and applies RNN LM (CTNDD_SA + RNN LM rescore), thereby achieving state-of-the-art performance for lyrics transcription of solo-singing. MSTRE-Net [14] is also introduced based on a multistream time-delay neural network (MTDNN) with 4-gram LM.

The end-to-end lyrics transcriber LT-V is described in Section VI-A1. Similar to [6], [16], the LT-V model is trained on Solo-train, validated on Solo-dev and tested on Solo-test. We observe from Table X that LT-V without LM outperforms the Kaldi models with 4-gramLM (TDNN + 4-gramLM, CTDNN + 4-gramLM and CTDNN_SA + 4-gramLM), which confirms the effectiveness of the transformer-based end-to-end models.

To make use of linguistic peculiarities of lyrics of songs as in [6], [16], an RNN-based language model is also developed using ESPnet [84] for lyrics. Two RNN language models are trained on the training data (Solo-train): a) RNN LM: an LM with random initialization and b) RNN LM-LS: an RNN LM trained by initializing weights with the LibriSpeech pretrained LM in the LS model. Our RNN LM architecture and training parameter settings are the same as those of the LibriSpeech pretrained LM model. We follow the default setting in ESPnet [84] to select the best validated checkpoint on the development set Solo-dev as the final language model. Our RNN LM is trained on Solo-train, validated on Solo-dev and tested on Solo-test, as shown in Table II. We set the LM weight to 0.3 for decoding in our RNN LM model.

We observe from Table X that LT-V with RNN LM outperforms LT-V, which suggests that the incorporation of RNN-based LM is helpful for lyrics transcription. The LT-V model with RNN LM-LS further improves performance over LT-V + RNN LM, which suggests that an LM pre-trained on a large corpus is beneficial. Overall, with an additional RNN LM, our proposed end-to-end model, LT-V + RNN LM-LS, outperforms all other state-of-the-art solo-singing models [6], [14], [16] as shown in Table X.

2) Lyrics Transcription of Polyphonic Music: We propose lyrics transcriber LT-P and multi-transcriber MT-FO and MT-PI models in this paper for polyphonic data with both single-task and multi-task learning strategies. We would like to compare them with the state-of-the-art reference models [12]–[14], [18], [92], [93]. The system of Stoller *et al.* [12] is based on the end-to-end Wave-U-Net framework, while the rest of the systems [13], [14], [18], [92], [93] are based on the traditional Kaldi-based ASR approach. A subset of these existing systems [18], [92], [93] were submitted to the lyrics transcription task in the 16th Music Information Retrieval Evaluation eXchange International Benchmarking Competition (MIREX 2020), and the system produced by Gao *et al.* [18] outperformed other submissions. The results of this challenge are publicly available[5].

In Table XI, we first report the lyrics transcription performance of all existing systems on the same test sets for whole song evaluation. We observe that MSTRE-Net [14] performs the best among all the Kaldi-based approaches. We also test on a larger database, DALI-test proposed in [14], which contains 240 whole-song polyphonic recordings.

We decode short nonoverlapping segments of songs in Hansen, Jamendo and Mauch using our proposed models and combine the transcriptions of these segments to report WER results for the complete songs, as displayed in Table XI. We applied automatic segmentation to the DALI-test dataset, where the segmentation can be achieved automatically by first extracting singing vocals by the state-of-the-art singing vocal extraction model [94] and then applying a vocal activity detector (VAD) [95] to the extracted vocals to detect the silent parts for segmentation. For VAD implementation [95], we merge the consecutive voiced segments if the length of the segment is less than 4 seconds, while we do not merge segments that are already more than 30 seconds long.

We observe that the LT-P model outperforms all previous End-to-End and Kaldi-based approaches except MSTRE-Net across all test data, which shows the general superiority of the end-to-end LT-P model over the conventional multistep ASR pipeline. Our proposed models outperform MSTRE-Net for the Mauch and Dali test sets, but do not perform better for Jamendo, and our models achieve comparable results with MSTRE-Net for the Hansen database.

We note that our proposed LT-P, MT-FO and MT-PI models do not employ any language model, while MSTRE-Net used 4-gram LM with a large lyrics corpus as external data. Compared with our E2E models, the Kaldi-based hybrid architecture is complicated and requires controlling the relative contribution from each part of the model [66], [73]. Specifically, Kaldi training involves multiple stages, beginning with a hybrid architecture containing GMM and HMM, which is used to generate per frame target states (forced alignment). It is followed by iterative training of an acoustic model (neural network) and re-estimation of the transition probabilities of the HMM [66]. Our proposed E2E model does not require such explicit alignment and complex training steps, thereby providing more flexibility in modeling. On the other hand, a known weakness of E2E transformer-based ASR models is its inability to handle long sequences because self-attention in each layer computes the output using the overall relationship between all input sequences [96]; therefore, decoding the whole song in one shot is not possible with our proposed transformer-based models.

We further report the results for line-level transcription. Inspired by the line-level testing of lyrics transcription of solo-singing [6], [14], [16], we prepare the segmented test sets of polyphonic music for the research community.[6] The details of line-level test data are described in Section V-A1.

[5][Online]. Available: https://www.music-ir.org/mirex/wiki/2020:Lyrics_Transcription_Results

[6][Online]. Available: https://github.com/xiaoxue1117/ALTP_chords_lyrics

TABLE XII
COMPARISON OF LYRICS TRANSCRIPTION (WER%) OF POLYPHONIC MUSIC ON SEGMENTED TEST SETS WITH POST-PROCESSING

Line-level test	**Hansen**	**Jamendo**	**Mauch**
LT-P	41.27	46.87	40.04
MT-PI	40.96	45.46	35.53
MT-FO	**38.31**	**44.37**	**34.68**
With post-processing	**Hansen**	**Jamendo**	**Mauch**
LT-P	39.87	44.26	36.80
MT-PI	38.51	44.64	32.38
MT-FO	**36.34**	**43.44**	**31.82**

For the line-level test, we observe from Table XII that the multi-transcriber MT-FO and MT-PI models consistently outperform the single-task LT-P model. We further perform post-processing of the test data by manually correcting the lexical inconsistency between the ground truth and the predicted lyrics, for example, by removing punctuation marks from lyrics and standardizing nonlexical items such as 'la' and 'lah'. We can consistently find that the multi-transcriber MT-FO and MT-PI models outperform the single-task LT-P model, as shown in Table XII. This again confirms the effectiveness of multitask learning over single-task learning with respect to lyrics transcription of polyphonic music.

VII. CONCLUSION

We advocate novel end-to-end network architectures for lyrics transcription that work for both solo-singing and polyphonic music. We also propose the idea of using chord recognition as a secondary task to strengthen the lyrics transcription main task for polyphonic music, which has been proven to be effective. We have shown that the proposed multi-transcriber framework outperforms single task frameworks for lyrics transcription through a comprehensive set of experiments on publicly available datasets.

REFERENCES

[1] D. Povey *et al.*, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2016, pp. 2751–2755.

[2] S. Sun, P. Guo, L. Xie, and M.-Y. Hwang, "Adversarial regularization for attention based end-to-end robust speech recognition," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 11, pp. 1826–1838, Nov. 2019.

[3] H. Fujihara and M. Goto, "Lyrics-to-audio alignment and its application," in *Multimodal Music Processing* (Dagstuhl Follow-Ups Series). Dagstuhl, Germany: Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, vol. 3, 2012.

[4] A. Mesaros, "Singing voice identification and lyrics transcription for music information retrieval invited paper," in *Proc. IEEE Conf. Speech Technol. Hum.- Comput. Dialogue*, 2013, pp. 1–10.

[5] C.-P. Tsai, Y.-L. Tuan, and L.-S. Lee, "Transcribing lyrics from commercial song audio: The first step towards singing content processing," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5749–5753.

[6] G. R. Dabike and J. Barker, "Automatic lyric transcription from karaoke vocal tracks: Resources and a baseline system," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 579–583.

[7] B. Sharma and Y. Wang, "Automatic evaluation of song intelligibility using singing adapted STOI and vocal-specific features," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 319–331, 2020.

[8] C. Gupta, B. Sharma, H. Li, and Y. Wang, "Automatic lyrics-to-audio alignment on polyphonic music using singing-adapted acoustic models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 396–400.

[9] A. Mesaros and T. Virtanen, "Automatic recognition of lyrics in singing," *EURASIP J. Audio, Speech, Music Process.*, vol. 2010, no. 1, 2010, Art. no. 546047.

[10] G. B. Dzhambazov and X. Serra, "Modeling of phoneme durations for alignment between polyphonic audio and lyrics," in *Proc. 12th Sound Music Comput. Conf.*, 2015, pp. 281–286.

[11] H. Fujihara, M. Goto, J. Ogata, and H. G. Okuno, "Lyricsynchronizer: Automatic synchronization system between musical audio signals and lyrics," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 6, pp. 1252–1261, Oct. 2011.

[12] D. Stoller, S. Durand, and S. Ewert, "End-to-end lyrics alignment for polyphonic music using an audio-to-character recognition model," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 181–185.

[13] C. Gupta, E. Yılmaz, and H. Li, "Automatic lyrics alignment and transcription in polyphonic music: Does background music help?," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 496–500.

[14] E. Demirel, S. Ahlbäck, and S. Dixon, "MSTRE-Net: Multistreaming acoustic modeling for automatic lyrics transcription," in *Int. Soc. Music Inf. Retrieval*, 2021.

[15] E. Demirel, S. Ahlbäck, and S. Dixon, "Low resource audio-to-lyrics alignment from polyphonic music recordings," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 586–590.

[16] E. Demirel, S. Ahlbäck, and S. Dixon, "Automatic lyrics transcription using dilated convolutional neural networks with self-attention," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2020, pp. 1–8.

[17] F.-R. Stöter, S. Uhlich, A. Liutkus, and Y. Mitsufuji, "Open-unmix-a reference implementation for music source separation," *J. Open Source Softw.*, vol. 4, no. 41, 2019, Art. no. 1667.

[18] X. Gao, C. Gupta, and H. Li, "Lyrics transcription and lyrics-to-audio alignment with music-informed acoustic models," *MIREX*, 2020.

[19] M. P. Ryynänen and A. P. Klapuri, "Automatic transcription of melody, bass line, and chords in polyphonic music," *Comput. Music J.*, vol. 32, no. 3, pp. 72–86, 2008.

[20] T. Cho, Improved techniques for automatic chord recognition from music audio signals, *Ph.D. dissertation*, Steinhardt Sch. Culture, Educ., Hum. Develop., New York University, New York, NY, USA, 2014.

[21] M. Mauch, "Automatic chord transcription from audio using computational models of musical context," *Ph.D. dissertation*, Sch. Electron. Eng. Comput. Sci., Queen Mary, University of London, London, U.K., 2010.

[22] S. Sadie, "The new grove dictionary of music and musicians," vol. 21, 2001.

[23] D. Deutsch, "Music recognition," *Psychol. Rev.*, vol. 76, no. 3, pp. 300–307, 1969.

[24] T. Cho and J. P. Bello, "On the relative importance of individual components of chord recognition systems," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 22, no. 2, pp. 477–492, Feb. 2014.

[25] T. Cho and J. P. Bello, "A feature smoothing method for chord recognition using recurrence plots," in *Proc. Int. Soc. Music Inf. Retrieval*, 2011, pp. 651–656.

[26] A. Sheh and P. W. Daniel Ellis, "Chord segmentation and recognition using EM-trained hidden Markov models," in *Proc. Int. Soc. Music Inf. Retrieval*, 2003, pp. 185–191.

[27] Y. Ueda, Y. Uchiyama, T. Nishimoto, N. Ono, and S. Sagayama, "HMM-based approach for automatic chord detection using refined acoustic features," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2010, pp. 5518–5521.

[28] E. J. Humphrey and J. P. Bello, "Rethinking automatic chord recognition with convolutional neural networks," in *Proc. IEEE 11th Int. Conf. Mach. Learn. Appl.*, 2012, vol. 2, pp. 357–362.

[29] E. J. Humphrey, T. Cho, and J. P. Bello, "Learning a robust tonnetz-space transform for automatic chord recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2012, pp. 453–456.

[30] E. J. Humphrey and J. P. Bello, "Four timely insights on automatic chord estimation.," in *Proc. Int. Soc. Music Inf. Retrieval*, 2015, vol. 10, pp. 673–679.

[31] X. Zhou and A. Lerch, "Chord detection using deep learning," in *Proc. Int. Soc. Music Inf. Retrieval*, 2015, vol. 53, pp. 52–58.

[32] Y. Wang, M.-Y. Kan, T. L. Nwe, A. Shenoy, and J. Yin, "Lyrically: Automatic synchronization of acoustic musical signals and textual lyrics," in *Proc. 12th Annu. ACM Int. Conf. Multimedia*, 2004, pp. 212–219.

[33] P. Juan Bello and J. Pickens, "A robust mid-level representation for harmonic content in music signals.," in *Proc. Int. Soc. Music Inf. Retrieval*, 2005, vol. 5, pp. 304–311.

[34] G. Brunner, Y. Wang, R. Wattenhofer, and J. Wiesendanger, "Jam-Bot: Music theory aware chord based generation of polyphonic music with LSTMs," in *Proc. IEEE 29th Int. Conf. Tools Artif. Intell.*, 2017, pp. 519–526.

[35] M. Mauch, D. Müllensiefen, S. Dixon, and G. Wiggins, "Can statistical language models be used for the analysis of harmonic progressions," in *Proc. 10th Int. Conf. Music Percep. Cogn.*, Sapporo, Japan, 2008.

[36] M. Mauch, H. Fujihara, and M. Goto, "Lyrics-to-audio alignment and phrase-level segmentation using incomplete internet-style chord annotations," in *Proc. 7th Sound Music Comput. Conf.*, 2010, pp. 9–16.

[37] X. Chang, Y. Qian, and D. Yu, "Monaural multi-talker speech recognition with attention mechanism and gated convolutional networks," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2018, pp. 1586–1590.

[38] S. Settle, J. L. Roux, T. Hori, S. Watanabe, and J. R. Hershey, "End-to-end multi-speaker speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 4819–4823.

[39] N. Kanda, Y. Gaur, X. Wang, Z. Meng, and T. Yoshioka, "Serialized output training for end-to-end overlapped speech recognition," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 2797–2801.

[40] X. Gao, X. Tian, R. K. Das, Y. Zhou, and H. Li, "Speaker-independent spectral mapping for speech-to-singing conversion," in *Proc. Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf.*, 2019, pp. 159–164.

[41] J. Merrill and P. Larrouy-Maestri, "Vocal features of song and speech: Insights from Schoenberg's Pierrot Lunaire," *Front. Psychol.*, vol. 82017, Art. no. 1108.

[42] A. M. Kruspe, "Bootstrapping a system for phoneme recognition and keyword spotting in unaccompanied singing," in *Proc. Int. Soc. Music Inf. Retrieval*, 2016, pp. 358–364.

[43] P. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Joint optimization of masks and deep recurrent neural networks for monaural source separation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 23, no. 12, pp. 2136–2147, Dec. 2015.

[44] C. Weng, D. Yu, M. L. Seltzer, and J. Droppo, "Deep neural networks for single-channel multi-talker speech recognition," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 23, no. 10, pp. 1670–1679, Oct. 2015.

[45] D. Yu, X. Chang, and Y. Qian, "Recognizing multi-talker speech with permutation invariant training," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2017, pp. 2456–2460.

[46] H. Seki, T. Hori, S. Watanabe, J. Le Roux, and J. R. Hershey, "A purely end-to-end system for multi-speaker speech recognition," in *Proc. Assoc. Comput. Linguistics*, 2018, pp. 2620–2630.

[47] X. Chang, Y. Qian, K. Yu, and S. Watanabe, "End-to-end monaural multi-speaker ASR system without pretraining," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 6256–6260.

[48] Y.-H. Tu, I. Tashev, S. Zarar, and C.-H. Lee, "A hybrid approach to combining conventional and deep learning techniques for single-channel speech enhancement and recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 2531–2535.

[49] Y.-H. Tu, J. Du, and C.-H. Lee, "DNN training based on classic gain function for single-channel speech enhancement and recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 910–914.

[50] K. Hermus, P. Wambacq, and H. Van Hamme, "A review of signal subspace speech enhancement and its application to noise robust speech recognition," *EURASIP J. Adv. Signal Process.*, vol. 2007, pp. 1–15, 2006.

[51] X. Liu, J. Gao, X. He, L. Deng, K. Duh, and Y.-Y. Wang, "Representation learning using multi-task deep neural networks for semantic classification and information retrieval," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2015, pp. 912–921.

[52] M.-T. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser, "Multi-task sequence to sequence learning,", 2015, *arXiv:1511.06114*.

[53] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2019, pp. 4171–4186.

[54] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A lite BERT for self-supervised learning of language representations,", 2019, *arXiv:1909.11942*.

[55] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, "Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 4460–4464.

[56] Q. Hu, Z. Wu, K. Richmond, J. Yamagishi, Y. Stylianou, and R. Maia, "Fusion of multiple parameterisations for DNN-based sinusoidal speech synthesis with multi-task learning," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2015, pp. 854–858.

[57] Y. Zhou, X. Tian, and H. Li, "Language agnostic speaker embedding for cross-lingual personalized speech generation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 3427–3439, 2021.

[58] S. Ruder, "An overview of multi-task learning in deep neural networks," 2017, *arXiv:1706.05098*.

[59] T. Evgeniou and M. Pontil, "Regularized multi–task learning," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2004, pp. 109–117.

[60] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, no. 1, pp. 41–75, 1997.

[61] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[62] T. N. Sainath, A.-R. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Proc. IEEE Int. Conf. Acoust., Speech and Signal Process.*, 2013, pp. 8614–8618.

[63] W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke, "The microsoft 2017 conversational speech recognition system," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5934–5938.

[64] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1764–1772.

[65] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2015, pp. 167–174.

[66] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent nn: First results," 2014, *arXiv:1412.1602*.

[67] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 4960–4964.

[68] S. Kim, T. Hori, and S. Watanabe, "Joint ctc-attention based end-to-end speech recognition using multi-task learning," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 4835–4839.

[69] S. Karita *et al.*, "Improving transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 1408–1412.

[70] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[71] C.-Z. A. Huang *et al.*, "Music transformer: Generating music with long-term structure," 2018, *arXiv:1809.04281*.

[72] J. Park, K. Choi, S. Jeon, D. Kim, and J. Park, "A bi-directional transformer for musical chord recognition," in *Proc. Int. Soc. Music Inf. Retrieval*, 2019, pp. 620–627.

[73] L. Dong, S. Xu, and B. Xu, "Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5884–5888.

[74] S. Zhang, E. Loweimi, P. Bell, and S. Renals, "On the usefulness of self-attention for automatic speech recognition with transformers," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2021, pp. 89–96.

[75] X. Chang, W. Zhang, Y. Qian, J. Le Roux, and S. Watanabe, "End-to-end multi-speaker speech recognition with transformer," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 6134–6138.

[76] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 4960–4964.

[77] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[78] J. Lei Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.

[79] G. Meseguer-Brocal, A. Cohen-Hadria, and G. Peeters, "DALI: A large dataset of synchronized audio, lyrics and notes, automatically created using teacher-student machine learning paradigm," in *Proc. Int. Soc. Music Inf. Retrieval*, 2018, pp. 431–437.

[80] J. K. Hansen, "Recognition of phonemes in a-cappella recordings using temporal patterns and mel frequency cepstral coefficients," in *Proc. 9th Sound Music Comput. Conf.*, 2012, pp. 494–499.

[81] P. J. Moreno, C. F. Joerg, J.-M. Van Thong, and O. Glickman, "A recursive algorithm for the forced alignment of very long audio segments.," in *Proc. Int. Conf. Spoken Lang. Process.*, 1998, vol. 98, pp. 2711–2714.

[82] C. Gupta, R. Tong, H. Li, and Y. Wang, “Semi-supervised lyrics and solo-singing alignment,” in *Proc. Int. Soc. Music Inf. Retrieval*, 2018, pp. 600–607.

[83] F. Korzeniowski and G. Widmer, “A fully convolutional deep auditory model for musical chord recognition,” in *Proc. IEEE Int. Workshop Mach. Learn. Signal Process.*, 2016, pp. 1–6.

[84] S. Watanabe *et al.*, “ESPNET: End-to-end speech processing toolkit,” in *Proc. Conf. Int. Speech Commun. Assoc.*, 2018, pp. 2207–2211.

[85] T. Hori, J. Cho, and S. Watanabe, “End-to-end speech recognition with word-based RNN language models,” in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2018, pp. 389–396.

[86] C. Gupta, J. Li, and H. Li, “Towards reference-independent rhythm assessment of solo singing,” in *Proc. IEEE Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf.*, 2021, pp. 912–919.

[87] J. Paulus and T. Virtanen, “Drum transcription with non-negative spectrogram factorisation,” in *Proc. IEEE 13th Eur. Signal Process. Conf.*, 2005, pp. 1–4.

[88] R. Vogl, M. Dorfer, and P. Knees, “Recurrent neural networks for drum transcription,” in *Proc. Int. Soc. Music Inf. Retrieval*, 2016, pp. 730–736.

[89] C.-W. Wu *et al.*, “A review of automatic drum transcription,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 26, no. 9, pp. 1457–1483, Sep. 2018.

[90] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, “Librispeech: An ASR corpus based on public domain audio books,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 5206–5210.

[91] N. Condit-Schultz and D. Huron, “Catching the lyrics: Intelligibility in twelve song genres,” *Music Perception: Interdiscipl. J.*, vol. 32, no. 5, pp. 470–483, 2015.

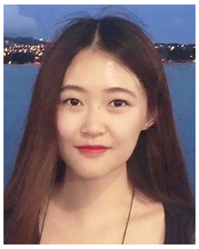
[92] G. R. Dabike and J. Barker, “The Sheffield university system for the MIREX 2020: Lyrics transcription task,” *MIREX*, 2021.

[93] E. Demirel, S. Ahlback, and S. Dixon, “A recursive search method for lyrics alignment,” *MIREX*, 2020.

[94] Q. Kong, Y. Cao, H. Liu, K. Choi, and Y. Wang, “Decoupling magnitude and phase estimation with deep resunet for music source separation,” 2021, *arXiv:2109.05418*.

[95] J. Kim and M. Hahn, “Voice activity detection using an adaptive context attention model,” *IEEE Signal Process. Lett.*, vol. 25, no. 8, pp. 1181–1185, Aug. 2018.

[96] Y. R. Oh, K. Park, and J. G. Park, “Fast offline transformer-based end-to-end automatic speech recognition for real-world applications,” *ETRI J.*, vol. 44, no. 3, pp. 476–490, 2021.



Xiaoxue Gao (Student Member, IEEE) received the B.Eng. degree in electronic information science and technology from Nanjing University, Nanjing, China, in 2017. She is currently working toward the Ph.D degree with the Human Language Technology Lab, Department of Electrical and Computer Engineering, National University of Singapore, Singapore. Her research interests include automatic lyrics transcription, lyrics-to-audio alignment, and speech-to-singing conversion.

Chitralekha Gupta (Member, IEEE) received the bachelor’s degree in engineering from Maharaja Sayajirao University Baroda, India in 2008, the master’s degree in engineering degree from the Indian Institute of Technology Bombay, India, in 2011, and the Ph.D. degree from the National University of Singapore (NUS),Singapore, in 2019. She is currently a Postdoctoral Research Fellow with NUS. She has previously was a Software Developer with Dell Research and Development, and as a Research Engineer with Airbus Defense and Space. Her research interests include singing voice analysis, applications of ASR in music, and neural audio synthesis. She was a Co-Captain with MIREX 2020 for the tasks lyrics-to-audio alignment and lyrics transcription. She was the recipient of the Start-Up Grant of Graduate Research Innovation Program NUS and has founded the music tech company MuSigPro Pte. Ltd. in Singapore in 2019. She was also the recipient of the NUS Dean’s Graduate Research Achievement Award 2018, NUS School of Computing Innovation Prize 2018, the Best Student Paper Award in APSIPA 2017, and prestigious NGS, NUS Graduate School for Integrative Sciences and Engineering Scholarship to pursue Ph.D. degree. She has played an active role in the organizing committees of ISMIR 2022, ICASSP 2022, SigDial 2021, ASRU 2019, ISCA-SAC 2018, and ISMIR 2017.

Haizhou Li (Fellow, IEEE) received the B.Sc., M.Sc., and Ph.D degrees in electrical and electronic engineering from the South China University of Technology, Guangzhou, China, in 1984, 1987, and 1990, respectively. He is currently a Professor with the School of Data Science, The Chinese University of Hong Kong, Shenzhen, China, and the Department of Electrical and Computer Engineering, National University of Singapore (NUS), Singapore. His research interests include automatic speech recognition, speaker and language recognition, and natural language processing. Prior to joining NUS, he was with the University of Hong Kong, Hong Kong, during 1988–1990 and South China University of Technology, Guangzhou, China, during 1990–1994. He was a Visiting Professor with CRIN, France, during 1994–1995, the Research Manager with Apple-ISS Research Centre during 1996–1998, the Research Director with Lernout & Hauspie Asia Pacific during 1999–2001, the Vice President with InfoTalk Corp. Ltd., during 2001–2003, and the Principal Scientist and Department Head of human language technology with the Institute for Infocomm Research, Singapore, during 2003–2016. He was the Editor-in-Chief of the IEEE ACM TRANSACTIONS ON AUDIO, SPEECH AND LANGUAGE PROCESSINGduring 2015–2018, and the Member of the Editorial Board of Computer Speech and Language during 2012–2018. He was an Elected Member of IEEE Speech and Language Processing Technical Committee during 2012–2014, the President of International Speech Communication Association during 2015–2017, the President of Asia Pacific Signal and Information Processing Association during 2015–2016, and the President of the Asian Federation of Natural Language Processing during 2017–2018. He was the General Chair of ACL 2012, INTERSPEECH 2014, ASRU 2019, and ICASSP 2022. Dr. Li is a Fellow of ISCA. He was the recipient of the National Infocomm Award 2002 and the President’s Technology Award 2013 in Singapore. He was named one of the two Nokia Visiting Professors in 2009 by the Nokia Foundation, and Bremen Excellence Chair Professor in 2019.