FUSING TRANSCRIPTION RESULTS FROM POLYPHONIC AND MONOPHONIC AUDIO FOR SINGING MELODY TRANSCRIPTION IN POLYPHONIC MUSIC

Bilei Zhu, Fuzhang Wu, Ke Li, Yongjian Wu, Feiyue Huang, Yunsheng Wu

Tencent Youtu AI Lab
bileizhu@tencent.com

ABSTRACT

This paper presents a new system for singing melody transcription from polyphonic songs. Instead of operating solely on polyphonic audio of each song to be processed (as most existing systems do), our system takes as inputs additionally multiple monophonic recordings of people singing the song. To transcribe the singing melody in a song, our system first tracks the singing pitch from polyphonic audio of the song by using a deep neural network (DNN)-based method, and then uses the estimated pitch series as reference to select the pitch sequences extracted from the multiple monophonic singing recordings. The selected monophonic pitch sequences, as well as the DNN pitch series from the polyphonic audio, are then transcribed separately, and their transcriptions results are fused to form the final note sequence. Experimental results show that, by introducing monophonic singings into transcription, the performance of singing melody transcription from polyphonic songs can be significantly improved.

Index Terms— Singing melody transcription, polyphonic audio, monophonic singing recordings, deep neural network (DNN), pitch sequence selection

1. INTRODUCTION

The aim of singing melody transcription is to automatically convert the lead voice in a polyphonic song into a series of notes each described by a pitch value (note number), an onset time and a duration. This involves two subtasks: one is to estimate the pitch of singing voice from polyphonic audio, and the other is to locate the boundaries of singing notes and assign each note with its corresponding pitch label. In literature, different combinations of singing pitch detection and note segmentation algorithms can be found (e.g., [1, 2, 3]). However, in spite of the continuing efforts of researchers, singing melody transcription still remains an unsolved problem. A major reason is that in polyphonic music, the spectral structure of singing voice overlaps with those of accompanying instruments, which makes singing pitch detection and note segmentation both difficult.

In contrast to singing transcription from polyphonic music, transcribing monophonic singing is generally considered less complex since there are no interfering instruments involved (e.g., [4, 5, 6]). In this paper, we introduce a singing melody transcription system for polyphonic songs. The originality of this system lies in that it fuses the transcription result from polyphonic audio by a method based on deep neural network (DNN), with the transcription results from a set of monophonic singing recordings, to extract the singing notes in a polyphonic song. The number of monophonic singing recordings for each song ranges from tens to hundreds, and these data are recorded without accompaniments from different users (mostly amateur singers) of a Karaoke APP named *WeSing*[1]. The recording process is unsupervised, and also the skills of Karaoke singers vary, leading to the existence of unclean and out-of-tune singing samples. To reduce the negative effects of these samples, our system is also equipped with a pitch sequence selection module, which uses the singing pitch series estimated by the DNN method as the selection reference.

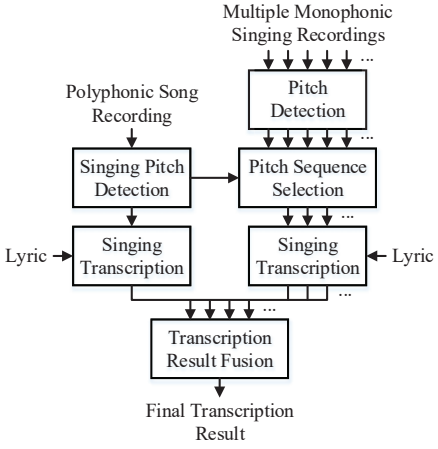

Fig. 1. System architecture.

Fig. 1 shows the architecture of our system. As we can see in the figure, our system takes as inputs for each song (1) a polyphonic audio file, and (2) multiple monophonic record-

[1] http://kg.qq.com/

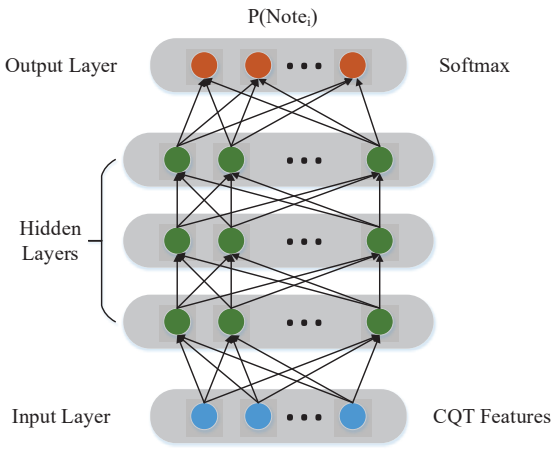

Fig. 2. DNN model for singing pitch detection.

ings of people singing the song. The system runs as follows. First, the polyphonic audio is fed to a DNN-based singing pitch detection module to provide an estimate of the singing pitch. This pitch series serves two purposes: (1) it is the input of a following singing transcription module; (2) it is also used as the reference for the selection of pitch sequences extracted from the multiple monophonic recordings. The selected pitch sequences are then transcribed. This is followed by a transcription result fusion module, where all the transcription results from polyphonic audio and monophonic recordings are fused to the final transcription result. Please note that in this paper we do not deal with the problem of automatic note segmentation. All the information of note boundaries are extracted from the lyric file.

2. ALGORITHM DESCRIPTION

2.1. Singing Transcription from Polyphonic Audio

The first module of our system performs singing pitch detection from polyphonic audio of the input song. This is done by using a supervised classification method as in [7]. However, in contrast to [7] which uses a shadow model (support vector machine to be specific), our system employs a DNN framework [8] for classification, which has been proved more powerful in many other tasks such as image recognition, speech recognition and natural language processing.

Fig. 2 illustrates our DNN model to classify the singing melody in polyphonic audio at frame level. The features for classification are derived from constant-Q transform (CQT) of the audio. CQT employs a logarithmic frequency scale, and compared with the well-known short-term Fourier transform (STFT) which has a linear frequency resolution, CQT is generally considered to match better with human perception of music. For CQT calculation, we use the *librosa* toolbox [9] with the hop length set to 512 samples (sampling rate = 44.1 kHz). The output of CQT is a set of time frames with CQT spectrums. For each frame, the feature is formed by stacking the CQT spectrum of the current frame with those of the previous and the next 20 frames.

As illustrated in Fig. 2, our DNN model is comprised of an input layer, 3 hidden layers and an output layer. The input layer takes as inputs the features calculated as above. The 3 hidden layers are fully-connected and stacked to lean an abstract and compact representation from the input features. Each hidden layer has 1024 hidden units and uses the rectified linear unit (ReLU) for activation. The output layer uses the *softmax* nonlinearity to obtain the output probability distribution. The number of outputs is set to correspond to the number of possible melodic notes that a human can sing. In our model, we use 60 notes corresponding to frequencies below 2 kHz.

To train our DNN model, features extracted from 2,246 songs in a training set (see Section 3) together with their corresponding pitch labels (in note number) are used. These labels are extracted from MIDI files, by decoding each MIDI file into a pitch series of which the time stamps match those of features. During the training, we use the standard stochastic gradient descent (SGD) algorithm to optimize the DNN parameters with min-batches of 4096 examples. Training is stopped after that the improvement in cross entropy falls below a threshold.

The trained DNN model is then utilized to decode each testing song into a pitch series, which is then transcribed to a note sequence by using the lyric file for note segmentation. Our lyric file contains the starting time and duration of each singing word in a song. We consider each singing word as a note, and the pitch of this note is calculated as the median of all the pitches falling in the time range of the note. Please note that here we do not consider the situation where each singing word contains multiple notes. This will be left for future work.

2.2. Singing Transcription from Monophonic Recordings

Our singing melody transcription system differs from existing systems mainly in that it employs the transcription results of multiple monophonic singing recordings to simulate the singing notes in the corresponding polyphonic music. The transcription of monophonic singing also starts with pitch detection. This is done by using YIN [10], an autocorrelation-based pitch estimator that has been found to be effective in many music transcription systems [6]. In our use of YIN, the frame length is set to 1024 samples with an overlap of 512 samples (sampling rate = 44.1 kHz), and the pitch range is set to 100∼800 Hz. The output of the pitch detection module is a set of pitch sequences, each from a monophonic singing recording.

However, as we have mentioned in Section 1, our monophonic singing data are collected from amateur singers via a Karaoke APP. Some of these singers may have poor singing

skills, and their performances may be seriously out of tune. Besides, since we have no supervision on where our singers use the APP (may be in a noisy environment) and whether our singers are focusing on singing the target song when using the APP (they may be talking, laughing or being silent during recording), there may be unclean singing samples in our data collection. Pitch sequences extracted from these out-of-tune and unclean singing samples may bring negative effects to our transcription, and thus they should be removed before further processing.

The unwanted pitch sequences are filtered out via a pitch sequence selection module. In this module, each of the pitch sequences extracted from monophonic singing recordings is compared with the singing pitch series extracted by the DNN model from polyphonic audio (see Section 2.1), to calculate the *overall accuracy* measure by using the *mir_eval* toolbox [11]. For each monophonic pitch sequence, overall accuracy is the proportion of frames where its pitch, measured in note number, equals to the DNN pitch. Please note that YIN outputs frequency values in Hertz, which should be converted to note numbers as follows,

$$n = [12 \times \log_2 \frac{f}{440} + 69], \quad (1)$$

where f is frequency in Hertz, n is the corresponding note number, and $[x]$ rounds x to the nearest integer.

We use the overall accuracy metric to measure the similarity of each monophonic singing with respect to the singing voice in polyphonic music, which is our target of transcription. The monophonic pitch sequences with overall accuracies lower than a threshold (hereafter denoted as θ) are considered to be originate from out-of-tune or unclean singing data, and thus they are removed. It worths mentioning that the singing pitch extracted from polyphonic audio itself contains errors. However, we argue that it is still a good reference for pitch sequence selection. This will be proved by our experimental results in Section 3.

After pitch sequence selection, each selected pitch sequence is transcribed to obtain a note sequence. Again, note segmentation is done with the lyric file, and the pitch value of each note is calculated as the median of the pitches falling in the time range of the note.

2.3. Transcription Result Fusion

The last module of our system takes as inputs and attempts to fuse all the transcription results obtained previously, including the note sequence extracted from polyphonic audio, and those from the multiple monophonic singing recordings. This is achieved in a straightforward way. Remind that in our system, the two singing transcription modules both use the lyric file to locate note boundaries (see Fig. 1). This means that all our transcription results obtained previously share the same note segmentation. We also use this segmentation in final transcription result. For each note in the final result, its pitch value is calculated as the median of pitches of the corresponding notes in the input note sequences (see Fig. 3).

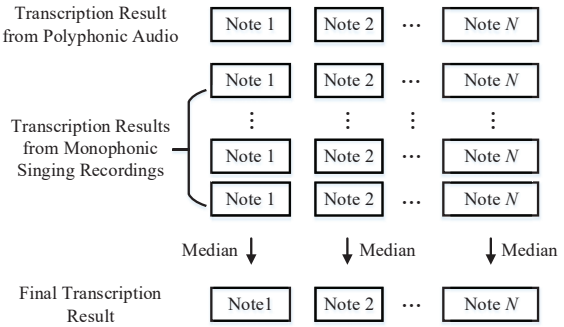

Fig. 3. Transcription result fusion.

3. EXPERIMENTAL RESULTS

3.1. Dataset

Our system was evaluated on a dataset of 2,773 polyphonic songs of various genres. All these songs contain a lead voice. For each of these song, we have a polyphonic audio recording with sampling rate of 44.1 kHz, and a MIDI file encoding the singing melody. The dataset was divided into two parts: a training set containing 2,246 songs and a testing set containing 527 songs. The training set was used to train the DNN model for singing pitch detection. For each song in the testing set, we collected via the Karaoke APP *WeSing* a set of monophonic recordings of unaccompanied singing. These monophonic recordings are sampled at 44.1 kHz, and their statistics are as follows: the total number of recordings is 33,736, the average, maximum and minimum number of recordings for each song is 64.02, 178 and 21 respectively.

3.2. Results of Singing Pitch Detection

The evaluation of pitch detection was carried out on the testing set. For each song in the set, we first decoded its corresponding MIDI file into a pitch series (i.e., the reference pitch series). The time stamps of the series were set to match the frames of 1024 samples in length with 50% overlap. During the evaluation, each singing pitch sequence we extracted was compared with its corresponding reference pitch series. Five measures were calculated using the *mir_eval* toolbox as evaluation metrics: *voicing recall rate*, *voicing false alarm rate*, *raw pitch accuracy*, *raw chroma accuracy* and *overall accuracy*. For details of these measures, please refer to [11]. Please note that in our experiments, an estimated pitch was deemed to be correct when its value, measured in note number, equals to the reference pitch. Pitch values in Hertz were transformed into note numbers by using Eq. 1.

Table 1 compares three algorithms for singing pitch detection: *Melodia* is a state-of-the-art method based on pitch

Table 1. Comparison of singing pitch detection algorithms.

	Voicing Recall	Voicing False Alarm	Raw Pitch	Raw Chroma	Overall
Melodia	**0.782±0.072**	0.376±0.118	0.449±0.119	0.501±0.100	0.514±0.091
DNN Model	0.710±0.156	0.181±0.099	**0.483±0.147**	**0.573±0.135**	0.612±0.097
Top-1 Singing	0.774±0.168	**0.111±0.067**	0.462±0.141	0.488±0.128	**0.627±0.095**

Table 2. Results of singing melody transcription.

	Precision	Recall	F-Measure
Melodia + Lyric	0.512±0.144	0.468±0.146	0.487±0.144
DNN Model + Lyric	0.527±0.145	0.482±0.148	0.501±0.146
Top-1 Singing + Lyric	0.507±0.164	0.465±0.165	0.483±0.164
Proposed System (θ = 0)	0.609±0.125	0.556±0.136	0.578±0.129
Proposed System (θ = 0.4)	**0.624±0.148**	**0.571±0.156**	**0.593±0.150**

contour characteristics [12], *DNN Model* represents our DNN model (see Section 2.1), and *Top-1 Singing* selects the pitch sequence of monophonic singing which has the highest overall accuracy with respect to the DNN pitch (i.e., lists top in pitch sequence selection, see Section 2.2). Each measure in the table is represented in the form of $a \pm b$, where a is the mean of the measure over all the 527 testing songs, and b is the standard deviation.

As we can see in Table 1, although our DNN model ranks last among the three methods for the voicing recall rate, it achieves the highest raw pitch accuracy and raw chroma accuracy. The *Top-1 Singing* method achieves the lowest voicing false alarm rate and ranks first for the overall accuracy, indicating the effectiveness of our pitch sequence selection strategy.

3.3. Results of Singing Melody Transcription

The evaluation of singing melody transcription was then performed, where for each song in the testing set, the transcribed note sequence was compared with the reference sequence extracted from the MIDI file to obtain three measures, i.e., *precision*, *recall* and *f-measure*. These three measures were calculated using the *mir_eval* toolbox, and the settings are as follows: the tolerance for the onset of an estimated note deviating from that of the reference is 0.15 s (as used in [3]), pitch deviation is not allowed between the estimated and reference notes (pitches are measured in note numbers), offsets are ignored.

Table 2 compares different stategies for singing melody transcription from polyphonic music. *Melodia + Lyric*, *DNN Model + Lyric* and *Top-1 Singing + Lyric* use the three methods in Table 1 to detect singing pitch respectively, and uses the lyric file for note segmentation. *Proposed System* denotes our proposed system, and θ is the parameter used in pitch sequence selection (θ = 0 means that all the monophonic pitch sequences are used without selection). Our final selection for θ is 0.4. We observed that a smaller θ will lead to the situation that many unwanted monophonic pitch sequences are left, and the transcription performance drops. In contrast, a θ larger than 0.4 will cause many occurrences that all monophonic pitch sequences for a song are filtered out and the transcription fails.

Table 2 shows that our system with θ = 0.4 ranks first for all the three measures. Especially, it outperforms the system with θ = 0, which obviously owes to the pitch sequence selection method we proposed. Moreover, for all the three measures, our system with the two settings of θ outperform *DNN Model + Lyric* by about 8% and 9% respectively. This indicates that, by introducing monophonic singings into transcription, the performance of singing melody transcription from polyphonic music can be significantly improved.

4. CONCLUSION

This paper presents our system for singing melody transcription from polyphonic songs. In our system, polyphonic audio and multiple monophonic singing recordings of each song are firstly transcribed separately, and their results are fused to form the final note sequence. Experiments were performed and the results show that our system outperform all the competitive methods. However, it has to be mentioned that our experimental results cannot be directly compared with those in other literatures. The reason is that our system uses the lyric file for note segmentation, and does not consider the problem where a single singing word contains several notes. Addressing this problem will be left for future work.

A possible use of our system in a real-world product has been under consideration. The Karaoke APP *WeSing* includes a singing scoring module, where user singings are compared with their corresponding MIDI files in the database to give scores measuring the singing performances. However, the number of MIDI files we have is limited, and therefore there are many songs that are not supported for scoring. We are considering to use our system to generate MIDI files for these songs.

5. REFERENCES

[1] M. Ryynänen and A. Klapuri, "Transcription of the singing melody in polyphonic music.," in *International Society for Music Information Retrieval*, 2006, pp. 222–227.

[2] E. Gómez, F. Cañadas, J. Salamon, J. Bonada, P. Vera, and P. Cabañas, "Predominant fundamental frequency estimation vs singing voice separation for the automatic transcription of accompanied flamenco singing.," in *International Society for Music Information Retrieval*, 2012, pp. 601–606.

[3] N. Kroher and E. Gómez, "Automatic transcription of flamenco singing from polyphonic music recordings," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 5, pp. 901–913, 2016.

[4] R. McNab, L. Smith, and I. Witten, "Signal processing for melody transcription," in *The 19th Australasian Computer Science Conference*, 1997, pp. 301–307.

[5] N. Adams, M. Bartsch, and G. Wakefield, "Note segmentation and quantization for music information retrieval," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 131–141, 2006.

[6] E. Molina, L. Tardón, A. Barbancho, and I. Barbancho, "Sipth: Singing transcription based on hysteresis defined on the pitch-time curve," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 2, pp. 252–263, 2015.

[7] G. Poliner and D. Ellis, "A classification approach to melody transcription," in *International Society for Music Information Retrieval*, 2005.

[8] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[9] B. McFee, C. Raffel, D. Liang, D. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *the 14th Python in Science Conference*, 2015.

[10] A. De Cheveigné and H. Kawahara, "Yin, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, pp. 1917–1930, 2002.

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

[12] J. Salamon and E. Gómez, "Melody extraction from polyphonic music signals using pitch contour characteristics," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 6, pp. 1759–1770, 2012.