

박사학위논문  
Ph.D. Dissertation

보컬 멜로디 추출을 위한 딥러닝

Deep Learning for Vocal Melody Extraction

2021

금상은 (琴湯恩 Kum, Sangeun)

한국과학기술원

Korea Advanced Institute of Science and Technology

박사학위논문

보컬 멜로디 추출을 위한 딥러닝

2021

김상은

한국과학기술원






문화기술대학원

# 보컬 멜로디 추출을 위한 딥러닝

금 상 은

위 논문은 한국과학기술원 박사학위논문으로  
학위논문 심사위원회의 심사를 통과하였음

2020년 11월 30일

심사위원장	남 주 한	(인)	
심 사 위 원	이 성 희	(인)	
심 사 위 원	이 경 면	(인)	
심 사 위 원	이 교 구	(인)	
심 사 위 원	Li Su	(인)	

# Deep Learning for Vocal Melody Extraction

Sangeun Kum

Advisor: Juhan Nam

A dissertation submitted to the faculty of  
Korea Advanced Institute of Science and Technology in  
partial fulfillment of the requirements for the degree of  
Doctor of Philosophy in Culture Technology

Daejeon, Korea  
December 22, 2020

Approved by



---

Juhan Nam

Professor of Graduate School of Culture Technology

The study was conducted in accordance with Code of Research Ethics<sup>1</sup>.

---

<sup>1</sup> Declaration of Ethical Conduct in Research: I, as a graduate student of Korea Advanced Institute of Science and Technology, hereby declare that I have not committed any act that may damage the credibility of my research. This includes, but is not limited to, falsification, thesis written by someone else, distortion of research findings, and plagiarism. I confirm that my thesis contains honest conclusions based on my own careful research under the guidance of my advisor.

DGCT            금상은. 보컬 멜로디 추출을 위한 딥러닝. 문화기술대학원 . 2021년.  
75+vi 쪽. 지도교수: 남주한. (영문 논문)  
Sangeun Kum. Deep Learning for Vocal Melody Extraction. Graduate  
School of Culture Technology . 2021. 75+vi pages. Advisor: Juhan Nam.  
(Text in English)

### 초록

이 논문에서는 보컬 멜로디 추출을 위한 다양한 딥 러닝 방법론을 제안한다. 보컬 멜로디 추출은 다양한 소리가 섞여 있는 음원에서 보컬 멜로디 라인의 음고를 식별하는 작업이다. 이전 연구는 스펙트럼 상에서 음고에 해당하는 특정 부분을 계산하거나 음원을 분리하는 방식을 사용했으나 다양한 음원에 대해서 최적의 결과를 얻는 데 한계가 있다. 최근에는 딥 러닝이 멜로디 추출에 적용되었으나 전반적인 정확도, 음악 관련 지식을 활용한 네트워크 구조, 그리고 데이터의 부족이라는 관점에서 여전히 한계가 존재한다. 우리는 효과적으로 멜로디의 음고를 추정하고 목소리를 감지할 수 있는 여러 딥 러닝 모델을 탐색하며, 또한 다양한 손실 함수에 대한 비교 및 분석을 제공한다. 우리는 멜로디 추출에서 서로 긴밀하게 연관되어 있는 두 가지 과제, 즉 멜로디 음고 예측과 보컬 존재 유무를 판별을 동시에 고려하도록 긴밀하게 결합된 새로운 모델과 손실 함수를 제안한다. 라벨링 된 데이터의 부족은 딥러닝 기반의 멜로디 추출 알고리즘의 근본적인 단점이다. 이 문제를 다루기 위해 우리는 멜로디 추출에 레이블이 없는 대량의 데이터를 활용할 수 있는 반지도 학습을 제안한다. 우리는 세 가지 선생-학생 모델과 무작위 데이터 증강 방법, 그리고 라벨이 없는 데이터의 구성에 따른 효과를 탐구하여 가장 효과적인 학습 방법을 제시한다. 또한 우리는 이 반지도 학습 방법을 보컬 탐지 모델을 학습하는데 적용하고, 이 방법이 레이블이 존재하는 데이터가 부족한 다른 음악 작업에도 확장될 수 있음을 보인다.

핵심 낱말 딥러닝, 보컬 멜로디 추출, 음성 구간 탐지, 반지도학습

### Abstract

In this thesis, we propose various deep learning (DL) based methods for vocal melody extraction. Vocal melody extraction is the task that identifies the melody pitch contour of the singing voice from multiple sources. Previous studies have been proposed as methods of calculating the pitch saliency from a spectrogram or isolating the melody source from the mixture. However, these methods have limitations in obtaining optimal outputs for various music. Although the performance of melody extraction has improved with the recent advances in DL, there are still limitations in terms of overall performance, the model using music-related knowledge and the lack of labeled data. Here we report the effective methods to estimate the pitch of melody and detect singing voice by introducing novel DL models and loss function. We also propose a multi-task network that allows pitch estimation and voice detection are tightly coupled. To address the lack of labeled data, we applied the semi-supervised learning that utilizes large amounts of unlabeled data. We explored the effects of three teacher-student model setups, data augmentation, and unlabeled data, and proposed the most effective learning method for vocal melody extraction. In addition, we apply semi-supervised learning to the singing vocal detection and show that it can be extended to other MIR tasks that suffer from lack of labeled data.

Keywords Deep Learning, Vocal Melody Extraction, Singing Voice Detection, Semi-Supervised Learning

# Contents

Contents . . . . .	i
List of Tables . . . . .	v
List of Figures . . . . .	vi
<b>Chapter 1. Introduction</b>	<b>1</b>
1.1 Contributions of this Dissertation . . . . .	2
<b>Chapter 2. Related Work</b>	<b>3</b>
2.1 Conventional approaches . . . . .	3
2.2 Deep learning based approaches . . . . .	3
<b>Chapter 3. Classification-based Singing Melody Extraction Using Deep Convolutional Neural Networks</b>	<b>6</b>
3.1 Introduction . . . . .	6
3.2 Proposed Methods . . . . .	7
3.2.1 DCNN Model Configuration for the Singing Pitch Extractor . . . . .	8
3.2.2 DCNN Model Configuration for the Singing Voice Activity Detector . . . . .	9
3.2.3 Voice False Alarm Detection for the SVAD . . . . .	10
3.2.4 Temporal Smoothing by HMM . . . . .	10
3.3 Dataset . . . . .	11
3.3.1 Training Datasets . . . . .	11
3.3.2 Test Datasets . . . . .	11
3.3.3 Evaluation . . . . .	12
3.4 Experiments . . . . .	12
3.4.1 Training Details of DCNNs . . . . .	12
3.4.2 Pitch Resolution and Ensemble Models . . . . .	13
3.4.3 HMM-based Postprocessing . . . . .	13
3.4.4 Singing Voice Activity Detector with VFAD . . . . .	13
3.5 Results . . . . .	13
3.5.1 DCNN Models of Melody Extraction . . . . .	13
3.5.2 HMM-based Post-processing . . . . .	14
3.5.3 Singing Voice Activity Detector for Melody Extraction . . . . .	14
3.6 Comparison to State-of-the-Art Melody Extraction Methods . . . . .	16

3.6.1	Evaluation Metrics . . . . .	16
3.6.2	Melodia vs. Proposed Method . . . . .	16
3.7	Conclusions . . . . .	18
<b>Chapter 4.</b>	<b>Comparison of loss functions for classification-based melody extraction</b>	<b>20</b>
4.1	Introduction . . . . .	20
4.2	Methods . . . . .	21
4.2.1	Learning Models . . . . .	21
4.2.2	Existing Loss Functions . . . . .	21
4.2.3	Proposed Loss Functions . . . . .	23
4.3	Experiments . . . . .	24
4.3.1	Dataset . . . . .	24
4.3.2	Training . . . . .	24
4.3.3	Evaluation Metrics . . . . .	24
4.4	Results & Discussion . . . . .	24
4.5	Conclusion . . . . .	26
<b>Chapter 5.</b>	<b>Joint Detection and Classification of Singing Voice Melody Using Convolutional Recurrent Neural Networks</b>	<b>27</b>
5.1	Introduction . . . . .	27
5.2	Proposed Method . . . . .	28
5.2.1	The Main Network . . . . .	28
5.2.2	Joint Detection and Classification Network . . . . .	30
5.3	Experiments . . . . .	32
5.3.1	Datasets . . . . .	32
5.3.2	Evaluation . . . . .	33
5.3.3	Training Detail . . . . .	33
5.3.4	Ablation Study . . . . .	33
5.4	Results and Discussion . . . . .	35
5.4.1	Comparison of Melody Extraction Performance . . . . .	35
5.4.2	Comparison of Voice Detection Performance . . . . .	37
5.4.3	Comparison with State-of-the-Art Methods for Melody Extraction . . . . .	38
5.4.4	Case Study of Melody Extraction on MedleyDB . . . . .	38
5.5	Conclusions . . . . .	40

<b>Chapter 6.</b>	<b>Semi-Supervised Learning Using Teacher-Student Models for Vocal Melody Extraction</b>	<b>42</b>
6.1	Introduction . . . . .	42
6.2	Related work . . . . .	43
6.3	Methods . . . . .	43
6.3.1	Model Architecture . . . . .	43
6.3.2	SSL in the Teacher-Student Framework . . . . .	44
6.3.3	Proposed Teacher-Student Models . . . . .	44
6.3.4	Data Augmentation . . . . .	45
6.3.5	Data Selection . . . . .	45
6.4	Datasets . . . . .	46
6.4.1	Labeled Data . . . . .	46
6.4.2	Unlabeled Data . . . . .	46
6.4.3	Test Data . . . . .	47
6.5	Experiments . . . . .	48
6.5.1	Experimental Setup . . . . .	48
6.5.2	Experiment 1: Teacher-Student Models . . . . .	48
6.5.3	Experiment 2: Joint Training vs. Fine-Tuning . . . . .	49
6.5.4	Experiment 3: Size of Training Data . . . . .	50
6.5.5	Experiment 4: Iterative Training . . . . .	50
6.5.6	Comparison with State-of-the-Arts . . . . .	50
6.6	Conclusion . . . . .	52
<b>Chapter 7.</b>	<b>Semi-Supervised Learning Using Teacher-Student Models for Singing Voice Activity Detection</b>	<b>53</b>
7.1	Introduction . . . . .	53
7.2	Related Work . . . . .	53
7.2.1	Singing Voice Activity Detection . . . . .	53
7.2.2	Semi-Supervised Learning . . . . .	54
7.2.3	Hard Negative Sampling . . . . .	54
7.3	Methods . . . . .	55
7.3.1	Model Architecture . . . . .	55
7.3.2	Teacher-Student Model . . . . .	55
7.3.3	Hard Negative Sampling . . . . .	56
7.4	Data sets . . . . .	56
7.4.1	Training data . . . . .	56
7.4.2	Test data . . . . .	57
7.5	Experiments . . . . .	57



7.5.1	Experiment 1: Teacher-Student Model and Iterative Training . . . . .	57
7.5.2	Experiment 2: Hard Negative Sampling . . . . .	58
7.6	Comparison with State-of-the-Arts . . . . .	59
7.7	Conclusions . . . . .	60
<b>Chapter 8.</b>	<b>Conclusions</b>	<b>61</b>
8.1	Summary . . . . .	61
8.2	Review on Current State and Future Work . . . . .	62
	<b>Bibliography</b>	<b>64</b>
	<b>Acknowledgments in Korean</b>	<b>73</b>
	<b>Curriculum Vitae</b>	<b>74</b>

## List of Tables

2.1	The algorithm architectures of 12 deep learning-based models for melody extraction . . . .	5
3.1	Configuration of DCNN for the singing pitch extractor . . . . .	8
3.2	configuration of DCNN for Singing Voice Activity Detector . . . . .	9
3.3	Comparison of the proposed SVAD performance with the VFAD according to theta value	16
3.4	Comparison of SVAD results on the Jamendo test dataset . . . . .	16
3.5	Comparison of Melody Extraction Results . . . . .	19
4.1	CNN and CRNN model Configurations. . . . .	22
4.2	Comparison of loss functions for melody extraction in the CNN and CRNN models. Note that the models were evaluated on voiced frames only to focus on the comparison. . . . .	25
5.1	Model configurations of the main and joint networks. ConvBlock and ResBlock have two convolutional layers; $[n \times n, k]$ denotes a convolutional operator of $n$ filters and a kernel size of $k$ . . . . .	30
5.2	Comparison of results of existing algorithms, $Main(SVD)$ , and $JDC_S(o_{sv})$ (average accuracy of 5 runs). Note that Jamendo is not included in the training set of $Main(SVD)$ and $JDC_S(o_{sv})$ . . . . .	38
5.3	Comparison of vocal melody extraction results. . . . .	39
6.1	Description of datasets. In FMA, The former and the latter are the total of the tracks with the vocal ratio above 0.3 and the total of the entire tracks respectively. . . . .	46
6.2	Vocal melody extraction results in terms of (RPA / OA) of the proposed and other methods on various test sets. The proposed model is iterated the self-training two times using the in-house dataset and $FMA_{Lv}$ . . . . .	51
6.3	Voicing detection results in terms of (VR / VFA) of the proposed and other methods on various test sets. . . . .	52
7.1	Description of training and test datasets. . . . .	57
7.2	Singing voicing detection results of the proposed and other methods on the Jamendo dataset. . . . .	59
7.3	Singing voicing detection results of the proposed and other methods on the RWC dataset. . . . .	60

## List of Figures

3.1	The diagram of our architecture for melody extraction including singing voice activity detector. . . . .	7
3.2	Raw pitch accuracies of test datasets (ADC04 and LabROSA) and classification accuracies of validation dataset according to the pitch resolution . . . . .	14
3.3	Comparison of raw pitch accuracy between SC-SPE and MC-SPE when the pitch resolution is increasing. $RPA_{SC}$ and $RPA_{MC}$ correspond to RPA of SC-SPE and MC-SPE, respectively. MC-SPE <sub>R</sub> is a network where a model that combines SC-SPE <sub>R</sub> , SC-SPE <sub>R/2</sub> and SC-SPE <sub>R/4</sub> . . . . .	14
3.4	Performance increment by HMM-based pitch smoothing on SC-SPE <sub>32</sub> . . . . .	15
3.5	An example of singing voice activity detection with VFAD: (1) The SPE predicts the pitch over all frames. (2) The SVAD determines the singing voice frames (blue box in the bottom) and removes non-vocal melody contours (dotted black). However, some melody lines are misidentified as singing voice (blue line). (3) To reduce the false alarm errors, the VFAD determines non-singing voice frames (red box in the bottom). (4) Finally, we obtain more elaborate melody contours (yellow line). (5) The ground truth (black line) is plotted 100Hz below the prediction for visual comparison. . . . .	15
3.6	Comparison of evaluation matrix of LabROSA according to pitch tolerance. The tolerance value used in the MIREX melody extraction task is 50 cents. . . . .	17
3.7	Comparison of pitch contours for the 'opera_fem4.wav' of ADC04. The reference pitch was plotted below 140 Hz for visual comparison. . . . .	17
4.1	CNN and CRNN architectures for melody extraction. The max-pooling operations in the convolutional blocks are conducted along the frequency axis only while preserving the temporal dimensionality. . . . .	21
5.1	Overview of the architecture and joint melody loss (a) CRNN architectures of the main network and joint detection and classification (JDC) network for melody extraction. ConvBlock is organized in the order of 'Conv-BN-leaky rectified linear unit (LReLU)-Conv', and PoolBlock is in the order of 'BN-LReLU-MaxPool'. We denote max-pooling and concatenation as "MP" and "C", respectively. The max-pooling is applied only to the frequency axis while preserving the time-wise dimensionality. (b) The diagram of the ResNet block (c) The block diagram of joint melody loss. $o_m$ is the softmax output of the melody from the main network. $o_{mv}$ and $o_v$ are the softmax output of the singing voice activity from the main network and auxiliary network, respectively. "V" and "NV" indicate voice and non-voice, respectively. . . . .	29
5.2	Architectures of melody extraction used for performance comparison in this paper. In the model name, the subscript refers to the type of JDC network, and parentheses refer to the source of voice detection output. The black and red solid arrows indicate the paths used for training and training/testing, respectively. The red dotted arrows indicate the path used for the test, although it was not used for training. . . . .	34

5.3	Performance of melody extraction on the <i>Main</i> , <i>Main(AUX)</i> , <i>Main(SVD)</i> , and JDC networks. The reported results were averaged across five training runs with different initializations. The “average” was calculated as the average score of all songs in all datasets. The band inside the box is the median, and the black triangle indicates the mean. RPA, raw pitch accuracy; RCA, raw chroma accuracy; VR, voicing detection rate; VFA, voicing false alarm rate. . . . .	36
5.4	Performance of singing voice detection for different evaluation metrics. “ACC” indicates the overall accuracy of voice detection. “PR” and “F1” indicate precision and F1 score, respectively. . . . .	37
5.5	Result of singing voice detection on Jamendo. . . . .	38
5.6	Case examples of singing melody extraction from the MedleyDB dataset: (a) good example (“MusicDelta-Gospel”) and (b,c) bad examples (“CelestialShore-DieForUs”, and “MatthewEntwistle-Lontano”). . . . .	41
6.1	Diagram of the three Teacher-Student models. . . . .	43
6.2	Comparison with supervised-learning model and three student models on three test sets. .	48
6.3	Comparison with pre-training, fine-tuning, and joint training methods on three test sets. .	49
6.4	Comparison with Noisy Students on varied sizes of unlabeled datasets. The subscript ‘ <i>v</i> ’ denotes a selected subset of FMA whose vocal ratio exceeds a threshold. We use the average of OA for ADC04, MIREX05, and MedleyDB to compare performances. . . . .	50
6.5	Effect of iteration training for Noisy Students. . . . .	51
7.1	Overview of the proposed architecture for singing voice detection . . . . .	55
7.2	Diagram of the Noisy Student of Teacher-Student framework for singing voice detection .	56
7.3	Comparison with supervised-only model (as a Baseline) and two Noisy Student models (as a SSL) on five test sets. The subscript ‘1’ and ‘2’ denotes a number of training iteration by teacher-student framework. . . . .	58
7.4	Comparison with Baseline model, semi-supervised model, and hard negative sampling model (as an HNS) that hard negative samples are added. . . . .	58
7.5	Comparison with Baseline, SSL, HNS and ‘HNS+SSL’ model that is a semi-supervised learning model with unlabeled data and hard negative samples . . . . .	59

# Chapter 1. Introduction

The most basic elements that constitute music are melody, rhythm, timbre, and harmony [1]. Among them, the melody is a significant factor influencing the distinction of songs. The definition of melody is very diverse, and the most commonly cited definition of the melody was proposed by Poliner et al. [2] as follows:

*“The melody is the single (monophonic) pitch sequence that a listener might reproduce if asked to whistle or hum a piece of polyphonic music, and that a listener would recognize as being the ‘essence’ of that music when heard in comparison”.*

As can be seen in the original Greek *melōidía* (“Sing” or “Chat”), the melody is often performed in the human voice [3]. From the definition and original terminology, we can see that ‘melody’ is a pitch sequence that is perceived through human cognitive processes and reproduced by human voice (or a musical instrument). It is revealed through various musical cognitive experiments that melody is not only recognized as a physical factor but is influenced by human cognition. Similar timbres or harmonics (i.e., with perceptual and conceptual similarities) have been observed to interfere with melody perception [4,5]. In addition, it has been found that a person considers a melody context while recognizing a melody using a pitch framework (e.g., key structure) that was previously implicitly learned [5]. These studies showed that a person is perceived through a complex perceptual process, not only physically distinguishing the pitch and melody.

The operation of automatically estimating the pitch trajectory of a melody from polyphonic sounds is called melody extraction in the field of MIR (Music Information Retrieval). The operation of extracting melody can be defined simply, but the actual execution involves implicitly complex processes, such as the process by which humans perceive melody. Therefore, it is academically important to devise a computational system for extracting melodies. In many previous studies, various algorithms have been proposed that utilize insights from human perception processes, based on salience functions or source separation methods [6]. Unfortunately, real music audios have a wide variety of variations by genre, instrument, timbre, and techniques of mixing and mastering. Furthermore, singing voices are more complex than musical instruments due to their acoustic characteristics (unstable vocalization and complex harmonics) [3]. Due to these hurdles, the proposed heuristic methods have limitations in obtaining optimal results for all cases [7].

In this thesis, we focus on the characteristic aspects of vocals and deal with the automatic extraction of vocal melodies from polyphonic music sources. In popular music, the singing voice is a central sound source that delivers melody, lyrics, and emotions. Since recent music services require more advanced solutions to search for songs or evaluate musical characteristics, automatic analysis of vocal has also been an important research topic [8]. To avoid the heuristic process of model design, the classification-based approach based on deep learning has recently attracted a lot of attention to melody extraction. Various domain-specific problems are being studied for more efficient model training as follows: input representations designed to effectively reveal melody features; methods related to output such as pitch resolution of label or loss function; process of leveraging lack of labeled data. We review the issues mentioned above and explore more effective strategies and systems for melody extraction.

## 1.1 Contributions of this Dissertation

This thesis contributes to the field of music information retrieval through studies on melody extraction and singing voice detection using deep learning. The main contributions of the thesis can be summarized as follows:

- **An investigation into the importance of pitch resolution to mitigate the pitch quantization problem that the classification-based approach has intrinsically.** Previous research handled only deep neural network (DNN) for melody extraction. We propose a singing melody extraction model using convolutional neural networks (CNN) and explore the importance of pitch resolution and combining multiple models with different pitch resolution in the singing pitch extractor.
- **Evaluation of conventional loss functions for melody extraction and proposal of variations of the modified loss functions.** We compare conventional loss functions for melody extraction using two classification models; one is based on CNN and the other is on Convolutional Recurrent Neural Network (CRNN). Through the performance comparison of the loss functions, we propose variations of the modified loss functions and discussed the directions to improve the melody extraction accuracy.
- **A novel method for melody extraction using joint detection and classification (JDC) network.** We propose a JDC network with a joint melody loss designed to combine the two tasks together (singing voice detection and pitch classification). This model uses a CRNN architecture with residual connections and bi-directional long short-term memory (Bi-LSTM) as the main network so that the model classifies the pitch with a high resolution. The auxiliary network that detects the singing voice is trained using multi-level features shared from the main network.
- **Applying the semi-supervised learning (SSL) methods to vocal melody extraction.** We propose a teacher-student model of SSL with the consistency of regularization for vocal melody extraction. It can leverage large-scale unlabeled music datasets with various audio data augmentation techniques. We investigate effective SSL strategies by exploring joint training, the size of unlabeled data, and the number of self-training iterations.
- **Applying the SSL methods to singing voice activity detection (SVAD).** We also apply our proposed SSL method for the SVAD and reveal that it can be effective for other MIR tasks that suffer from the lack of labeled data. We use a hard negative sampling technique and show that it helps to improve overall performance while reducing false-positive errors.

## Chapter 2. Related Work

### 2.1 Conventional approaches

Conventional melody extraction algorithms can be broadly classified as three categories according to approach type: saliency-based approach, source separation-based approach, and data-driven approach. The general methods of the saliency-based approach exploit saliency functions to calculate the pitch saliency from the mixture and to estimate possible pitch candidates [6,7,9,10]. The general processes of the salience-based approaches follows several steps. First of all, some pre-processing steps are applied to enhance the melody characteristics or harmonic-percussive source separation. Then apply a transform function such as STFT and find the spectral peak. Among them, find the peaks that could be candidates for the melody, and then determines the pitch contour. The final step is determining where is the melody part or not. Source separation-based approaches [11–13] have a strategy to separate the melody source from multiple source and use a monophonic pitch tracking algorithm to estimate the melody sequence. While the majority of previous works are associated with the first two approaches, the data-driven approaches have been rarely explored. This approach predicts a finite set of pitch labels from audio features. Ellis and Poliner [14] used a support vector machine classifier to predict a pitch label from the spectrogram. They were the first to demonstrate the utility of classifying pitch labels for melody extraction. Bittner et al. [15] proposed a method to distinguish a melody using a random forest classifier from among candidates of several pitch contours obtained from the salience function.

### 2.2 Deep learning based approaches

Classification-based approach based on deep learning has drawn much attention for melody extraction recently. Various model structures and learning techniques have been proposed so far. Unlike categorical labels where the distance between labels is non-linear, pitch labels are represented as a continuous scale. Considering the task-specific characteristics, several variations of categorical loss functions also have been proposed. In Table 2.1, we provide a summary of the deep learning-based algorithms. Kum et al. [16] presented a classification-based approach for melody extraction on vocal segments using multi-column fully-connected neural networks (FNN). In the proposed model, each of the neural networks is trained to predict a pitch label of singing voice from the spectrogram, but their outputs have different pitch resolutions. The final melody contour is inferred by combining the outputs of the networks and post-processing it with a hidden Markov model. In order to take advantage of the data-driven approach, they also augmented training data by pitch-shifting the audio content and modifying the pitch label accordingly. Rigaud and Radenen [17] also proposed an algorithm for estimating pitches of singing voice using FNN. They also presented singing voice activity detector (SVAD) using Bidirectional Long Short-Term Memory (BLSTM) with pre-processed features obtained by double-stage harmonic/percussive source separation (HPSS) Octave error is one of the frequently encountered problems in melody extraction because of the harmonic similarity of two pitches with octave relations. To reduce octave mismatch, Park and Yoo [18] proposed harmonic sum loss function by summing each loss taking into account all possible octave relationships. They used a long short-term memory recurrent neural network (LSTM-RNN) for extracting melody and voice detection. Basaran et al. [19] proposed

a convolutional recurrent neural network (CRNN) model to capture the temporal feature and spectral feature more effectively. To modeling the dominant melody and accompaniment of mixture and obtain a useful input salience representation, they proposed a source-filter non-negative matrix factorization (SF-NMF) as a pre-processing for input.

In the proposed algorithm, models using various input features or combining various network structures were used considering the characteristics of melody extraction rather than using a simple neural network architecture. Bittner et al. [20] replaced the existing salience function with CNN. The proposed network uses harmonic constant-Q transform (HCQT) as input representation. HCQT is a modified 3-dimensional representation of CQT, in which harmonic relation is added from the existing two-dimensional CQT representation. The network can implicitly learn to generalize many types of harmonic relationships using harmonic-related representations. They showed their model to be effective in learning the salience representation for multi-f<sub>0</sub> tracking as well as the prediction of f<sub>0</sub> of a single melody in polyphonic audio. Su [21] presented a novel time-frequency representation, Combined frequency and periodicity (CFP), for input feature which enhances the pitch contours and suppresses the harmonic components. From the CFP, the patch-based convolutional neural network (CNN) model classifies whether the pitch contour corresponding to the patch is a candidate for the song voice melody contour. Lu and Su [22] proposed semantic segmentation for melody extraction using a deep convolutional neural network (DCNN) with an U-Net architecture. They also adopted domain-adaptive transfer learning from large-scale symbolic data to audio using melody MIDI files. Hsieh et al. [23] proposed a model similar to the model structure proposed by [22], but proposed a model using up-pooling between the encoder and decoder instead of the skip connection to improve localization of melody. Also, encoded features are utilized to voice detection. Gao et al. [24] utilized MIDI files to extract melody at the semitone level and then refine the pitch using the salience function to a higher resolution. Chou et al. [25] proposed also combined two neural networks to imitate human pitch perception. They simulated the spectral model of pitch perception using CNN with a log frequency spectrogram. To mimic the temporal model, they used a time-domain autocorrelation through a bank of gammatone filters as the input of the DNN model. Chen et al. [26] also considered pitch perception to build a neural network model for melody extraction and proposed a multi-resolution end-to-end model by using 1-D and 2-D CNN with kernels. As such, various input features and model architectures have been proposed based on human hearing perception.



Table 2.1: The algorithm architectures of 12 deep learning-based models for melody extraction

Author	single/multiple-pitch estimation	Target source	Pre-processing	Input representation	Post-processing	Voicing	Model architecture	Loss function	Framework
<b>Kum et al. [16] (2016)</b>	single	vocal	re-sampling (8k)	STFT (log-amplitude)	Viterbi decoding	Energy threshold	multi-column FNN	BCE	Keras
<b>Rigaud &amp; Radenen [17] (2016)</b>	single	all	re-sampling (16k), Harmonic/Percussive Source Separation (HPSS)	1. pitch: STFT (log-amplitude) 2. voicing: Mel-spec	Viterbi decoding	Classification: voice detector	1.pitch : FNN, 2.voicing: BLSTM	CE	-
<b>Bittner et al. [20] (2017)</b>	multiple	all	-	HCQT	-	threshold	CNN	CE	Keras
<b>Park &amp; Yoo [18] (2017)</b>	single	all	re-sampling (16k)	STFT	-	Classification	LSTM-RNN	CE + harmonic sum loss	CNTK
<b>Su [21] (2018)</b>	single	vocal	re-sampling (16k)	CFP representation	-	threshold of output	CNN	CE	Keras
<b>Lu &amp; Su [22] (2018)</b>	single	vocal	re-sampling (16k)	CFP representation	-	threshold of output	encoder/decoder, progressive neural network (PNN)	Focal loss	Keras
<b>Basaran &amp; Essid [19] (2018)</b>	single	all	-	Source-Filter Nonnegative Matrix Factorization (SF-NMF) (salience representation)	-	Classification	CRNN	CE	Keras
<b>Chou et al. [25] (2018)</b>	single	all	re-sampling (16k)	1.STFT(log-frequency axis) 2.auditory filter bank (gammatone filters)	Viterbi decoding	Classification	FNN/CNN	CE	-
<b>Kum &amp; Nam [27] (2019)</b>	single	vocal	re-sampling (8k)	STFT (log-amplitude)	-	Classification	CRNN, joint network	CE + joint melody loss	Keras
<b>Hsieh et al. [28] (2019)</b>	single	vocal	re-sampling (16k)	CFP representation	-	Classification	encoder/decoder	BCE	PyTorch
<b>Chen et al. [26] (2019)</b>	single	vocal	re-sampling (16k)	Raw audio sample	-	Classification	1D,2D CNN	CE	-
<b>Gao et al. [24] (2019)</b>	single	vocal	-	1.Semitone-level pitch: CQT, 2.Salience-based: STFT	-	Classification	FNN + refinement	CE	-

# Chapter 3. Classification-based Singing Melody Extraction Using Deep Convolutional Neural Networks

## 3.1 Introduction

Melody extraction is the task of estimating the fundamental frequency that corresponds to the melodic line of a polyphonic music. Since melody is the essence of music from which listeners can identify the piece, melody extraction has been applied to various music information retrieval tasks such as query-by-humming [29] and cover song identification [30]. In popular music, melody is usually performed by singers and so the melody extraction task is often recast into detecting the presence of singing voices and estimating the dominant voice pitch when background music is accompanied. The expressive nature of the singing melody has been utilized for explaining characteristics of different music genres [31] or singers [32] as well. Furthermore, the continuous pitch curves have been incorporated in source separation algorithms to take vocal and background music apart [13].

A number of melody extraction algorithms, where some of them are particularly for singing voices, have been proposed so far. They can be broadly classified into three categories according to the approach type: salience-based, source separation-based, and classification-based ones [6]. While the majority of previous work are associated with the first two approaches, the data-driven approach based on classification, which predicts a finite set of pitch labels from audio features, have been rarely explored. An early work used a support vector machine classifier to predict a pitch label from spectrogram [14]. Since then, it had been no attempt until Bittner et al. proposed a random forest classifier that predicts pitch contours from pitch salience features [15].

The lack of the classification-based approach can be attributed to the following reasons. First, melodic pitch is a physically measurable value as opposed to abstract labels defined in high-level tasks such as genre or mood classification. Thus, it is more intuitive to directly leverage time-frequency representations where the patterns for pitch estimation are observable, as in the saliency-based or source separation-based approaches. Second, in the classification-based approach, the melodic pitch is supposed to be quantized to a certain resolution (e.g. semitone in [14]). While this discrete pitch may be useful for some applications that require a MIDI-level pitch notation, it loses detailed information about singing styles such as vibrato or note-to-note transition patterns. Third, the classification-based approach typically requires a sufficient amount of labeled data to achieve good performance. Manual extraction of melodic pitch in a frame-level is a highly tedious labor, particularly for mixed tracks. This has hindered the availability of labeled datasets.

In the recent past, however, there have been important changes that have encouraged the classification-based approach. First, multi-track audio recording data including singing voice as a separate track have been more available [33–35]. With the multi-track datasets, the melody labels can be obtained more easily by applying a monophonic pitch detector to the isolated vocal track. Second, deep learning, the powerful data-driven learning algorithm based on neural networks, has emerged and tremendously advanced, achieving a remarkable series of state-of-the-art results in numerous tasks. An indispensable element in the success of deep learning is the availability of large-scale labeled datasets.

Leveraging the datasets and recent advances in deep learning, several classification-based methods using neural networks have been recently attempted. Rigaud and Radenen proposed to use two types of

neural networks. One is for detecting singing voice activity built with 3-Bidirectional Long Short-Term Memory (BLSTM) layers following [36]. The other is for extracting singing pitch composed of 2-hidden fully connected layers and a softmax layer that discriminates up to an eighth of semi-tone [17]. Comparing the state-of-the-art salience-based system, *Melodia* [7], they show significantly improved results. Kum and Nam proposed multi-column deep neural networks (MC-DNN) where each column network is trained to predict a pitch label with a different pitch resolution and the outputs are combined [16]. The results showed that the ensemble method achieves better performance than a single model and also it returns a high pitch resolution. Park and Yoo presented a LSTM-based melody classification algorithm where they added harmonic sum loss to the objective function to incorporate the harmonic structure in melodic tone [37]. They showed that the harmonic sum loss makes the model more robust to octave mismatch and interference from background music.

In this work, we propose a singing melody extraction model using deep convolutional neural networks (DCNN). While DCNN-based models have been shown to achieve state-of-the-art results in many music information retrieval (MIR) tasks including singing voice detection [38], polyphonic piano transcription [39], chord recognition [40] and music-auto tagging [41], to the best of our knowledge, they have been not applied to singing melody extraction yet. In particular, we investigate the importance of pitch resolution in the singing pitch extractor (SPE). Also, we suggest to use the output of pitch prediction in the SPE as a mean to suppress voice false alarm errors from the result of the singing voice activity detection (SVAD). Using several public datasets, we show that the proposed method significantly outperforms our previous work and the overall results are comparable to state-of-the-arts.

## 3.2 Proposed Methods

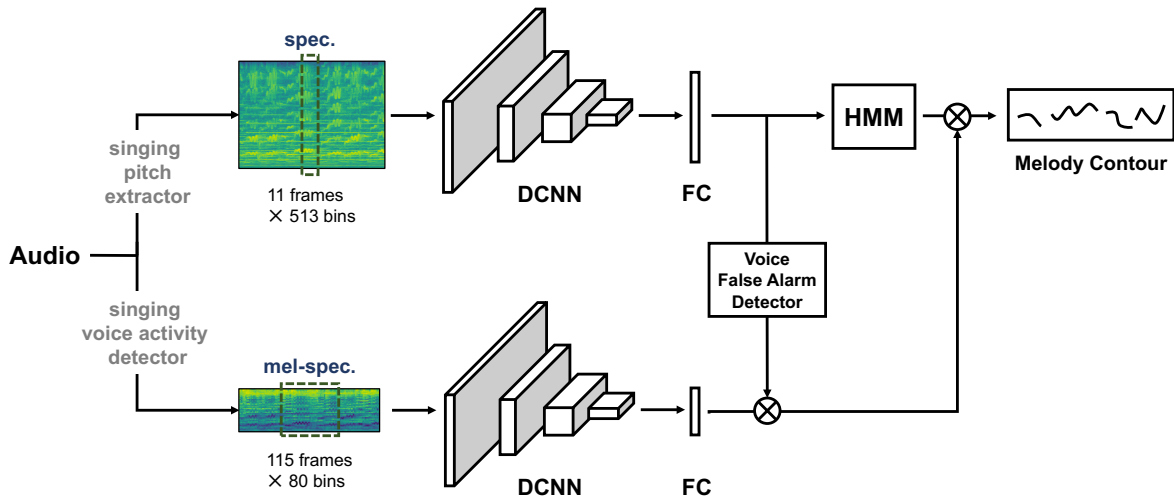


Figure 3.1: The diagram of our architecture for melody extraction including singing voice activity detector.

The proposed melody extraction method is illustrated in Figure 3.1. It is composed by two main parts. The SPE extracts melody features and predicts its pitch from a short segment of spectrograms

(11 frames). Then, the output is temporally smoothed by a hidden Markov model (HMM) based post-processing. The SVAD serves to distinguish singing voice frames from a long segment of mel-spectrograms (115 frames) and removes melodic contours of the non-voice segments. In addition, the voice false alarm detector reduces the false positive from the SVAD results by exploiting the output of SPE.

### 3.2.1 DCNN Model Configuration for the Singing Pitch Extractor

Table 3.1: Configuration of DCNN for the singing pitch extractor

input: SPEC. 11(frames) $\times$ 513(bins)				
	$R_8$	$R_{16}$	$R_{32}$	output
block1	64	conv1 (3 $\times$ 3)		64 $\times$ 11 $\times$ 513
	64	conv2 (3 $\times$ 3)		64 $\times$ 11 $\times$ 513
		average-pool (2 $\times$ 1)		64 $\times$ 5 $\times$ 513
		max-pool (1 $\times$ 3)		64 $\times$ 5 $\times$ 171
block2	128	conv3 (3 $\times$ 3)		128 $\times$ 5 $\times$ 171
	128	conv4 (3 $\times$ 3)		128 $\times$ 5 $\times$ 171
		average-pool (2 $\times$ 1)		128 $\times$ 2 $\times$ 171
		max-pool (1 $\times$ 3)		128 $\times$ 2 $\times$ 57
block3	256	conv5 (3 $\times$ 3)		256 $\times$ 2 $\times$ 57
	256	conv6 (3 $\times$ 3)		256 $\times$ 2 $\times$ 57
		average-pool (2 $\times$ 1)		256 $\times$ 1 $\times$ 57
		max-pool (1 $\times$ 3)		256 $\times$ 1 $\times$ 14
block4	512	conv7 (3 $\times$ 3)		512 $\times$ 1 $\times$ 14
	256	conv8 (3 $\times$ 3)		256 $\times$ 1 $\times$ 14
		max-pool (1 $\times$ 7)		256 $\times$ 1 $\times$ 2
FC	512	1024	2048	-
softmax	361	721	1441	-

The architecture of SPE are summarized in Table 1. The SPE is configured with four convolutional blocks and one fully connected layer. Each block contains two convolutional layers and two pooling layers except the last one. The convolution filters have a filter size of 3 $\times$ 3 and the number of filters in the convolutional blocks gradually increases as 64, 128, 256 and up to 512. Then, average-pooling is applied to the time axis and max-pooling is to the frequency axis. The intuition behind this setting is that singing pitch is typically continuous and so temporal smoothing by the average maintain the pitch information better than max-pooling. Experimentally, we confirmed that this actually worked better than using max-pooling on both axes. We apply batch normalization on each convolutional layer and use the Leaky ReLU as an activation function for the non-linearity. We include dropout to the end of each block and use softmax activation function for the output layer. The pitch labels cover from D2 (73.416 Hz) to B5 (987.77 Hz). We quantized the pitch labels on MIDI scale but with high resolutions.

$R_t$  denotes the resolution in  $1/t$  semitone unit. For example,  $R_1$  indicates pitch resolution in semitone unit.  $R_2, R_4, R_8, R_{16}$  and  $R_{32}$  indicate progressively higher resolutions than semitone by a factor of 2.

We used spectrogram as input for the SPE. We first resampled audio clips to 8 kHz and merged stereo channels into mono. We then computed spectrogram with Hann window of 1024 samples and hop size of 80 samples. We compressed the magnitude of the spectrogram in a log scale and used 513 bins from 0 Hz to 4000 Hz. As in the previous work [16], we took multiple frames of spectrogram as input to capture contextual information from neighboring frames and use the pitch label at the center position of the context window. We also experimented with different sizes of input frames and obtained the best results at 11 frames in SPE as well. Thus, we fix the input size to 11 frames for all experiments.

### 3.2.2 DCNN Model Configuration for the Singing Voice Activity Detector

Table 3.2: configuration of DCNN for Singing Voice Activity Detector

input : Mel-Spec 115 (frames) $\times$ 80 (bins)			
	stride	architecture	output
block1	1	64 conv1 (3 $\times$ 3)	64 $\times$ 115 $\times$ 80
	1	64 conv2 (3 $\times$ 3)	64 $\times$ 115 $\times$ 80
	2	maxpool (3 $\times$ 3)	64 $\times$ 57 $\times$ 39
block2	1	128 conv3 (3 $\times$ 3)	128 $\times$ 57 $\times$ 39
	1	128 conv4 (3 $\times$ 3)	128 $\times$ 57 $\times$ 39
	2	maxpool (3 $\times$ 3)	128 $\times$ 28 $\times$ 19
block3	1	256 conv5 (3 $\times$ 3)	256 $\times$ 28 $\times$ 19
	1	256 conv6 (3 $\times$ 3)	256 $\times$ 28 $\times$ 19
	2	maxpool (3 $\times$ 3)	256 $\times$ 13 $\times$ 9
block4	1	256 conv7 (3 $\times$ 3)	256 $\times$ 13 $\times$ 9
	1	512 conv8 (3 $\times$ 3)	512 $\times$ 13 $\times$ 9
	2	maxpool (3 $\times$ 3)	512 $\times$ 6 $\times$ 4
1 $\times$ 1 conv block	1	128 conv (1 $\times$ 1)	128 $\times$ 6 $\times$ 4
	1	2 conv (1 $\times$ 1)	2 $\times$ 6 $\times$ 4
	-	global average-pool	2
Total #params : 2,983,746			

The architecture of SVAD is summarized in Table 3.2. The SVAD is configured with four convolutional blocks and 1 $\times$ 1 convolutional layer. Each convolution block contains two 3 $\times$ 3 convolutional layers followed by batch normalization. The number of channels in the blocks gradually increases as 64, 128, 256 and up to 512. 3 $\times$ 3 max-pooling layers with 2 $\times$ 2 stride are used at the end of each convolutional block. At the final stage of the DCNN model, we used 1 $\times$ 1 convolution and global average pooling instead of fully connected layer. The architecture was inspired by the *Network In Network* model [42], which has the advantage of avoiding the problem of overfitting and greatly reducing the amount of computation without degrading performance. After the DCNN predicts the output, we used a median filter of 110ms as the final step to perform temporal smoothing [38].

The SVAD takes 115 frames of mel-spectrogram as input to capture contextual information over

a long time span following [38]. We resampled audio signals to 16kHz and merged stereo channels to mono. We extracted mel-spectrogram with 80 triangular filters between 0 and 8 kHz, a frame length of 1024, hop size of 160 samples. We compressed the magnitude by a log scale.

### 3.2.3 Voice False Alarm Detection for the SVAD

The SVAD takes a long segment (115 frames or 1.15 seconds) as input. We observed that the setting often produces false positive errors around the boundary of melody contours or short pauses between two melody contours. It is because long input frames taken from the boundary regions contain singing voice in part and so the SVAD is likely to predict the existence even if there is no voice at the center position. Therefore, we need to have additional means to minimize the false positive errors by taking a smaller size of input frames. For this purpose, a method of reducing the errors by detecting sub-semitone fluctuations has been previously attempted [43]. In this work, we propose a novel method that utilizes the output of the SPE.

We empirically found that, when the SPE takes non-voiced frames as input and predicts the pitch, the output was not dominant at a particular class and tends to have a low probability for each class. This is probably because the model was trained only with voice frames and so the model cannot make a prediction with high confidence for the unseen input. By exploiting the observation, we add a voice false alarm detector (VFAD) based on the SPE as follows:

$$S_{VFAD}(n) = \begin{cases} 1 & \text{if } \mathit{argmax}(y_{SPE}(n)) > \theta \\ 0 & \text{if } \mathit{argmax}(y_{SPE}(n)) < \theta \end{cases}$$

where  $y_{SPE}(n)$  is the softmax output of SPE at  $n$  frame and  $\theta$  is a threshold. We obtain the final result of singing voice activity,  $S(n)$  by incorporating the VFAD into the SVAD:

$$S(n) = S_{SVAD}(n) \cdot S_{VFAD}(n) \tag{3.1}$$

where  $S_{SVAD}(n)$  is the result of SVAD that returns one for voiced frames or zero for unvoiced frames.

### 3.2.4 Temporal Smoothing by HMM

After predicting the output in the SPE, we conduct temporal smoothing for the frame-wise pitch prediction. The procedure was basically borrowed from the Viterbi decoding based on HMM in [14]. The HMM state corresponds to each of the melody pitch values and the prior probabilities and the transition matrix are computed from the ground-truth of the training set. As posterior probabilities, the prediction from the combined output of SPE is used. To generate the prior and transition probabilities, we counted the number of occurrences and all pitch-to-pitch transition per pitch label, respectively. In addition, we normalized the transition matrix by replacing each element with the average of its corresponding diagonal. This alleviates the sparsity problem in the transition matrix obtained from a limited training set by assuming that all adjacent pitch transitions depend only on their interval rather than absolute pitch value.

However, even with the normalization, the diagonal components of the transition matrix are still dominant. Thus, when the pitch difference between consecutive melodies is small, the result of smoothing tends to keep the same pitch. This leads to the loss of detail changes in the pitch contours. To deal

with the problem, we add more weights to off-diagonal elements by multiplying a penalty matrix to the transition matrix as follows:

$$P = e^\lambda D + I \quad (3.2)$$

$$\tilde{T} = PT \quad (3.3)$$

where  $D$  is the off-diagonal matrix of the transition matrix  $T$ , whose diagonal elements are zeros.  $I$  is identity matrix. By increasing the value of the off-diagonal component, it adjusts the sensitivity to small pitch changes during the smoothing process.

### 3.3 Dataset

#### 3.3.1 Training Datasets

We used the RWC and MedleyDB datasets to train the SPE. To train the SVAD model, we used the Jamendo dataset in addition to the two. We divided them into training and validation splits to tune the network parameters. To avoid overfitting and select the best performing model, we chose songs such that genre and gender are evenly distributed over the splits and also the songs of the same singer are not divided over the splits.

- **RWC** [44]: 80 Japanese popular songs and 20 American popular songs with singing voice melody annotations. We divided the dataset into two splits, 85 songs for training and the remaining 15 songs for validation. The total length of the dataset is 407 minutes.
- **MedleyDB** [34]: 122 songs with a variety of musical genres and 70 of them including vocals with melody annotations. Among them, we chose 60 songs that are dominated by vocal melody. We divided the dataset into two splits, 47 songs for training and the remaining 13 songs for validation. The total length of the dataset is about 200 minutes.
- **Jamendo** [45]: 93 songs designed for the evaluation of singing voice detection. The training, validation and test set splits are designated as 61, 16 and 16 songs, respectively. The total length of the dataset used for training is about 360 minutes.

We also augmented the three datasets to obtain more generalized models. Pitch shifting has proven to be an effective way to increase data and improve results for singing voice activity detection [38] and melody extraction [16] as well. To this end, instead of resampling that modifies the pitch and length of audio clips at the same time [2], we used a phase vocoder method that conducts pitch-shifting independent of time-stretching [46]. We augmented the training set by applying the pitch-shifting by  $\pm 1, 2$  semitones, thereby increasing the data size by five times.

#### 3.3.2 Test Datasets

To evaluate the proposed model, we use publicly available datasets: ADC2004, LabROSA, MIR1k and iKala. Synthesized sounds or instrument sounds (e.g. 'train13MIDI.wav' in LabROSA or 'midi1.wav' in the ADC04 dataset) were excluded from the training data so that both SPE and SVAD focus on singing voice in polyphonic music. Thus, we used only singing voice songs as test data among the whole datasets.

- **LabROSA**<sup>1</sup>: 13 excerpts that contain Rock, R&B, pop, and jazz songs, as well as audio generated from a MIDI file. We evaluated our algorithm using 9 songs out of a total of 13 songs.
- **ADC2004**<sup>1</sup>: 20 excerpts of 20 seconds that contain pop, jazz and opera songs, as well as synthesized singing and audio from MIDI files. Jazz and MIDI songs were excluded from the evaluation.
- **iKala** [35]: 262 Chinese songs clips of 30 seconds performed by 6 professional singers.
- **MIR-1k** [33]: 1000 songs clips with the total duration of 133 minutes. 19 amateur singers (11 males and 8 females) participated in the recording.

### 3.3.3 Evaluation

We evaluated the proposed method in terms of five metrics, including overall accuracy (OA), raw pitch accuracy (RPA), raw chroma accuracy (RCA), voicing detection rate (VR) and voicing false alarm rate (VFA), as detailed in [6]. We compute them using *mir-eval*, a Python library designed for objective evaluation in MIR tasks [47]. The evaluation consists of two main parts: voice detection determining whether a voice is included in a particular time frame (VR and VFA) and pitch detection determining the most accurate melody pitch for each time frame (RPA, RCA, and OA). We convert the pitch labels, which were quantized to MIDI scale, back to frequency scale (Hz) to compare them with the ground truth.

$$f = 2^{(m-69)/12} \cdot 440(\text{Hz}) \quad (3.4)$$

where  $m$  is the estimated pitch label. The pitch of the frame is considered correct if the difference between the estimated pitch frequency and the ground-truth is within  $\pm 50$  cents (0.5 semitone). In addition, we progressively reduced the pitch tolerance to  $\pm 25$ ,  $\pm 12.5$  cents. We report the results in order to show the performance under more strict conditions.

## 3.4 Experiments

Given the SPE and SVAD models and training data, we conducted several experiments to figure out the effect of different settings in the models. In the followings, we describe the experimental setup.

### 3.4.1 Training Details of DCNNs

We randomly initialized the network parameters using He uniform initialization [48] and trained them with stochastic gradient descent with Nesterov momentum which was set to 0.9. We iterated it over all the training data up to 100 epochs. The initial learning rate was set to 0.02. To prevent overfitting, we applied a dropout ratio of 0.3 after all max-pooling layers. By means of early-stopping strategy, if the validation accuracy does not increase after 20 iterations, we reset the learning rate to 1/2 of the initial learning rate and repeated the training. We iterate this process five times. For fast computing, we ran the code using Keras [49], a deep learning library in Python, on a computer with two GPUs.

---

<sup>1</sup>We obtained the LabROSA dataset from the website, <http://labrosa.ee.columbia.edu/projects/melody/>. This was used for part of the 2005 MIREX melody extraction task. In our previous work [16], we referred to it as MIREX05.



### 3.4.2 Pitch Resolution and Ensemble Models

Our first experiment is to figure out the maximum pitch resolution of the SPE. High pitch resolution allows the SPE to predict continuous pitch curves, mitigating the pitch quantization problem that the classification-based approach has intrinsically. In our previous work [16], we progressively increased the pitch resolution and observed that the performance saturates before  $R_8$ . With the DCNN-based model, we conduct the same experiment and find the pitch resolution that provides the best performance.

We also combine multiple neural networks with different pitch resolutions as we conducted in [16]. We denote SC-SPE $_r$  as a single-column DCNN with a pitch resolution  $R_r$  and MC-SPE $_r$  as an ensemble model that combines SC-SPE $_r$ , SC-SPE $_{r/2}$  and SC-SPE $_{r/4}$ . We evaluated all the models on two test sets (ADC2004, LabROSA) and compared the accuracy.

### 3.4.3 HMM-based Postprocessing

We conducted temporal smoothing of the pitch prediction using the Viterbi decoding. The prior probabilities and transition matrix were estimated from the ground-truth of the training set. To increase the value of the off-diagonal components, we set the  $\lambda$  according to Equation 3.2. We used a set of  $\lambda$  and empirically found that  $\lambda$  of 1 yielded the best results.

### 3.4.4 Singing Voice Activity Detector with VFAD

As mentioned in Chapter 3.2.3, we use the VFAD to reduce false positive frames after the SVAD. If the maximum softmax output of SPE does not exceed a specific threshold  $\theta$ , it is assumed that the frame is not a singing voice. The threshold  $\theta$  was set to a value between 0 and 0.05 to find the proper threshold. We used the songs from the ADC04 and LabROSA datasets. We evaluated the performance in terms of VR, precision, F1 score and VFA. We also compared the performance of the SVAD with those from state-of-the-art algorithms. We reported the results on the Jamendo dataset as unseen test data. To evaluate the performance of the SVAD, we compute three common evaluation metrics: VR, precision, F1 score [50].

## 3.5 Results

### 3.5.1 DCNN Models of Melody Extraction

Figure 3.2 shows the RCA on the two test datasets and the classification accuracy of the validation set with varying pitch resolution. We conducted the experiment by increasing the pitch resolution until the accuracy becomes saturated. The result shows that the higher the pitch resolution, the lower the classification accuracy. This is because it is not easy to predict the exact class corresponding to the reference pitch as the melody extractor has more classes to predict. On the other hand, the RPA on the test datasets tend to increase as the pitch resolution is higher. Compared to the DNN model which was saturated at  $R_4$  [16], the DCNN model has the best results at  $R_{32}$ . This indicates that the DCNN is more capable of handling high pitch resolutions. However, we should note that higher resolutions require more network parameters.

Figure 3.3 shows that the MC-SPE models perform better than the SC-SPE models in general, validating that combining multiple models with different pitch resolutions is more effective [16]. However, the effect of using the multi-column models becomes less significant as the pitch resolution increases.

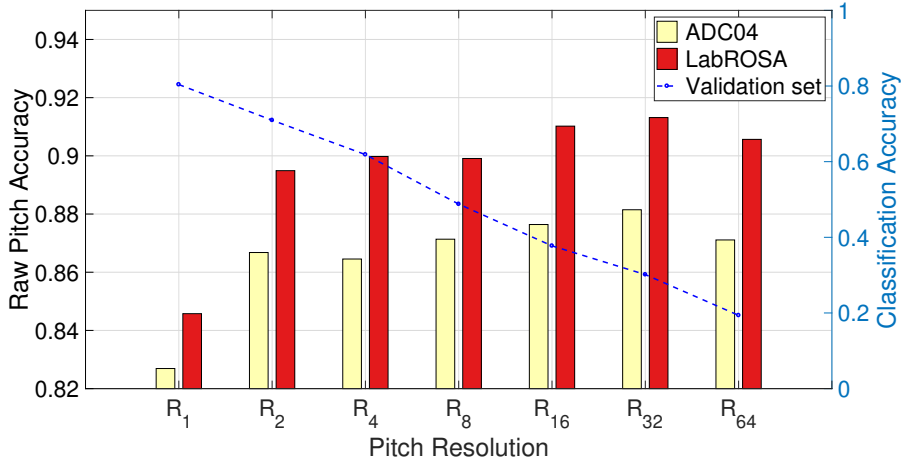


Figure 3.2: Raw pitch accuracies of test datasets (ADC04 and LabROSA) and classification accuracies of validation dataset according to the pitch resolution .

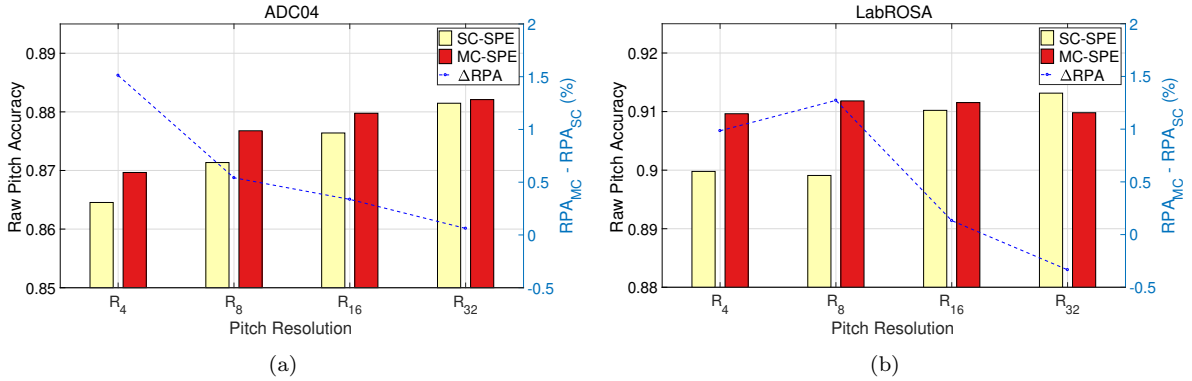


Figure 3.3: Comparison of raw pitch accuracy between SC-SPE and MC-SPE when the pitch resolution is increasing.  $RPA_{SC}$  and  $RPA_{MC}$  correspond to RPA of SC-SPE and MC-SPE, respectively.  $MC-SPE_R$  is a network where a model that combines  $SC-SPE_{R/2}$  and  $SC-SPE_{R/4}$ .

This is clearly indicated by  $\Delta RPA$ , the difference of RPA between the two models. For  $R_{32}$  on the LabROSA, the SC-SPE model is even better the MC-SPE model, achieving the best accuracy. Thus, considering the ensemble model requires as many parameters as the model size, SC-SPE is seen to be a more practical choice in the DCNN-based approach.

### 3.5.2 HMM-based Post-processing

Figure 3.4 show that both RPA and RCA increase by more than 1% on both datasets after the temporal smoothing. Comparing the difference between RPA and RCA, we can observe that the octave error decreases significantly. This indicates that the abrupt rise and fall of pitch contours are suppressed.

### 3.5.3 Singing Voice Activity Detector for Melody Extraction

We compared the performance of the SVAD with VFAD on ADC04 and LabROSA. Table 3.3 shows the evaluation matrix when  $\theta$  is 0, 0.03, and 0.05. The larger the value of  $\theta$ , the smaller the VFA. If the

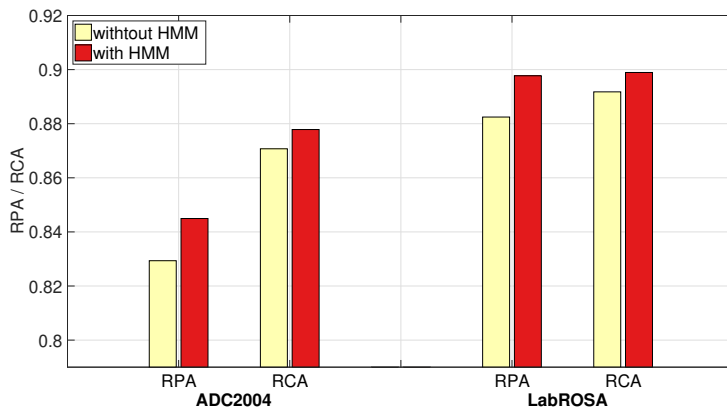


Figure 3.4: Performance increment by HMM-based pitch smoothing on SC-SPE<sub>32</sub>.

threshold is set too high, the F1 score is lowered. Using VFAD does not significantly reduce VFA because the number of frame is relatively small to be removed by VFAD among the voice frames detected by the SVAD. However, this process makes it possible to provide more natural melody contours. As shown in Figure 3.5, the effect of VFAD can be seen by comparing the blue line (obtained from SVAD) with the yellow line (obtained by further using VFAD). Based on the results from Table 3.3, we set 0.03 as a trade-off.

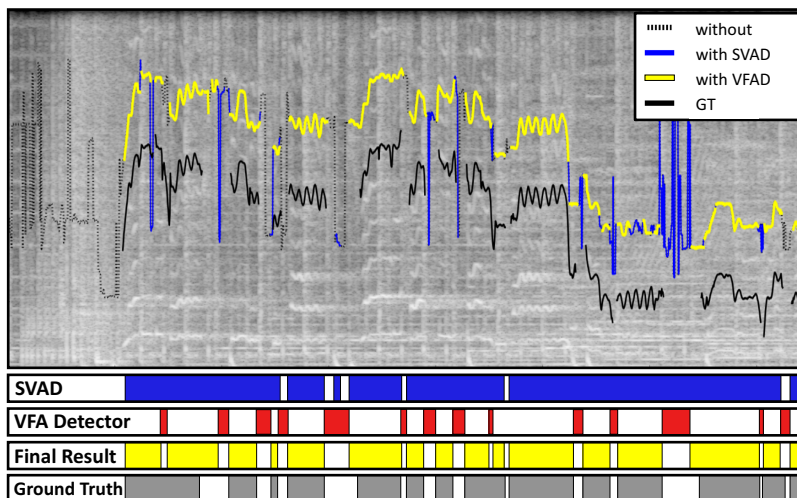


Figure 3.5: An example of singing voice activity detection with VFAD: (1) The SPE predicts the pitch over all frames. (2) The SVAD determines the singing voice frames (blue box in the bottom) and removes non-vocal melody contours (dotted black). However, some melody lines are misidentified as singing voice (blue line). (3) To reduce the false alarm errors, the VFAD determines non-singing voice frames (red box in the bottom). (4) Finally, we obtain more elaborate melody contours (yellow line). (5) The ground truth (black line) is plotted 100Hz below the prediction for visual comparison.

Table 3.4 compares the proposed method to two state-of-the-arts algorithms on 16 songs in the Jamendo test dataset. The Lehner algorithm is based on LSTM-RNN and the Fluctogram feature to reduce false positives [51] and the Leglaive algorithm is on BLSTM-RNN [36]. We did not show other state-of-the-arts algorithm using DCNN due to different evaluation metrics, for example, [38]. The

Table 3.3: Comparison of the proposed SVAD performance with the VFAD according to theta value

$\theta$	ADC04			LabROSA		
	0	0.03	0.05	0	0.03	0.05
VR	<b>0.861</b>	0.858	0.846	<b>0.891</b>	0.891	0.887
Precision	0.976	0.976	<b>0.977</b>	0.933	0.939	<b>0.945</b>
F1 score	<b>0.915</b>	0.914	0.907	0.912	0.914	<b>0.915</b>
VFA	0.113	0.112	<b>0.106</b>	0.121	0.109	<b>0.097</b>

proposed method has higher precision and lower voice recall than the two. This conservative result in detecting the voice activity is attributed to the VFAD. However, when we evaluate them according to the F1 score, the proposed method slightly outperformed the two compared algorithms.

Table 3.4: Comparison of SVAD results on the Jamendo test dataset

	VR	Precision	F1 score
Lehner [51]	0.906	0.898	0.902
Leglaives [36]	<b>0.926</b>	0.895	0.910
<b>Proposed</b>	0.893	<b>0.933</b>	<b>0.913</b>

## 3.6 Comparison to State-of-the-Art Melody Extraction Methods

### 3.6.1 Evaluation Metrics

Table 3.5 compares the proposed method with state-of-the-art algorithms on the four test datasets. The model used for the final test is SC-SPE<sub>32</sub>. The melody extraction results of the compared algorithms were obtained from MIREX<sup>2</sup>. For ADC04 and MIREX05, we should note that our results are not exactly comparable to them because we used only songs with vocal and also the LabROSA dataset is a subset of MIREX05. Also, we did not list recently reported results based on deep learning [17, 37] because they have different test settings.

The proposed method significantly outperforms our previous multi-column DNN model [16] for three datasets. The overall accuracy of the proposed model is above 80% for all datasets except MIR-1K. This might be because the audio files in MIR-1K have poor recording quality. Compared to the results from MIREX, the proposed method achieved better accuracy except those from Dressler on ADC04 [52]. A notable result is that the proposed method has significantly low voice false alarm. This may be attributed to the proposed SVAD that is supported by the VFAD.

### 3.6.2 Melodia vs. Proposed Method

In general, classification-based approach to melody extraction produces discrete pitch contours, losing detailed singing information. However, the proposed method can generate nearly continuous

<sup>2</sup><http://www.music-ir.org/mirex/wiki/MIREX.HOME>

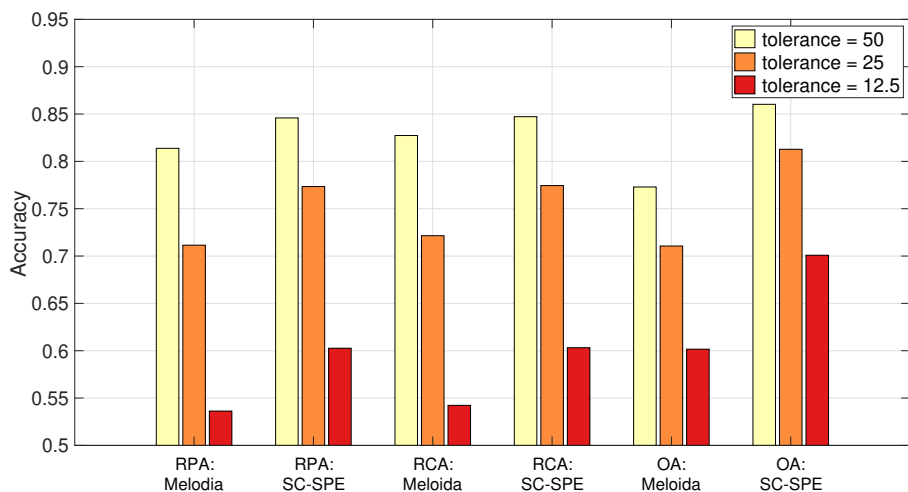


Figure 3.6: Comparison of evaluation matrix of LabROSA according to pitch tolerance. The tolerance value used in the MIREX melody extraction task is 50 cents.

curves by increasing the output resolution up to  $R_{32}$ . Therefore, they preserve natural singing styles such as vibrato or note-to-note transition patterns. In order to confirm the continuity, we obtained the evaluation results by reducing the pitch tolerance to  $\pm 25$ ,  $\pm 12.5$  cents, and compared them with the results from Melodia, a saliency-based algorithm that generates the continuous pitch curves [7]. Figure 3.6 shows that the proposed method (SC-SPE) achieves 5 to 10% higher than Melodia although the performance becomes worse for smaller tolerance. Figure 3.7 compares an example of pitch contours, each from Melodia and the proposed method. This illustrates more intuitively that the proposed method produces highly continuous curves that are similar to the ground-truth in Hz.

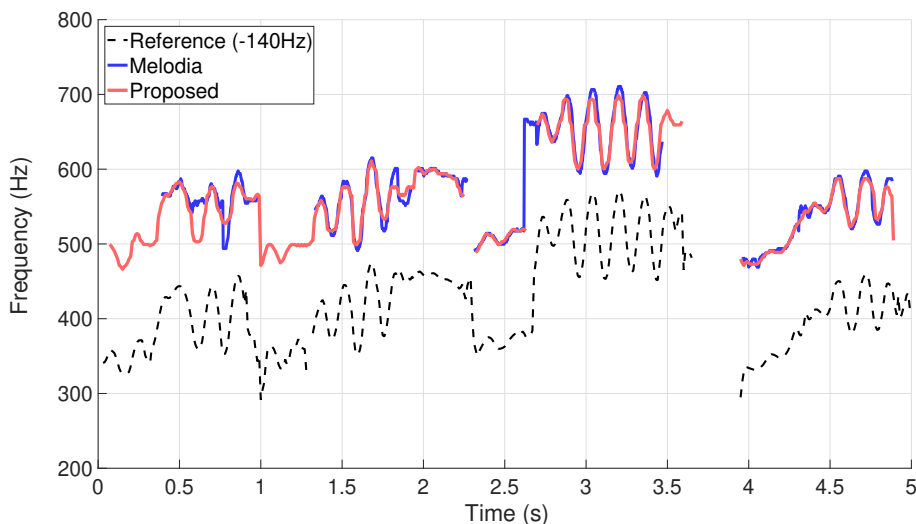


Figure 3.7: Comparison of pitch contours for the 'opera\_fem4.wav' of ADC04. The reference pitch was plotted below 140 Hz for visual comparison.

## 3.7 Conclusions

We proposed a novel melody extraction algorithm composed of singing pitch detector and singing voice activity detector using deep convolution neural networks. We have shown that the SPE can effectively extract melody features and classify pitch classes. Since the pitch can be predicted with a high resolution, the classification-based algorithm can produce nearly continuous curves. The multi-column method for predicting pitches of various resolutions can improve performance in DCNN, but the effect becomes less significant as the pitch resolution is higher. We propose a high performance SVAD with VFAD to minimize false positive errors. Finally, we compared our melody extraction model to previous state-of-the-arts methods on several public test dataset and showed that the results are comparable to those from the best.

Table 3.5: Comparison of Melody Extraction Results

(a) ADC04					
Method	OA	RPA	RCA	VR	VFA
Dressler [52]	<b>0.853</b>	<b>0.883</b>	<b>0.889</b>	<b>0.901</b>	0.158
Arora et al. [10]	0.690	0.814	0.859	0.765	0.235
Bosch et al. [53]	0.697	0.767	0.799	0.776	0.202
Salamon et al. [7]	0.740	0.772	0.794	0.806	0.152
Ikemiya et al. [13]	0.719	0.814	0.846	0.849	0.435
Kum et al. [16]	0.731	0.758	0.783	0.889	0.412
<b>Proposed</b>	<b>0.811</b>	<b>0.798</b>	<b>0.802</b>	<b>0.859</b>	<b>0.112</b>

(b) MIREX05					
Method	OA	RPA	RCA	VR	VFA
Dressler [52]	0.715	0.770	0.806	0.831	0.300
Arora et al. [10]	0.634	0.692	0.765	0.810	0.344
Bosch et al. [53]	0.637	0.688	0.7338	0.791	0.385
Salamon et al. [7]	0.676	0.698	0.769	0.776	0.239
Ikemiya et al. [13]	0.674	0.764	0.815	0.945	0.557

LabROSA					
Method	OA	RPA	RCA	VR	VFA
Kum et al. [16]	0.684	0.776	0.786	0.870	0.490
<b>Proposed</b>	<b>0.859</b>	<b>0.842</b>	<b>0.844</b>	<b>0.891</b>	<b>0.109</b>

(c) MIR-1k					
Method	OA	RPA	RCA	VR	VFA
Kum et al.	0.613	<b>0.726</b>	<b>0.770</b>	<b>0.934</b>	0.658
<b>Proposed</b>	<b>0.741</b>	0.718	0.749	0.817	<b>0.196</b>

(d) iKala					
Method	OA	RPA	RCA	VR	VFA
<b>Proposed</b>	0.811	0.769	0.773	0.849	0.085

## Chapter 4. Comparison of loss functions for classification-based melody extraction

### 4.1 Introduction

Melody extraction is the task of estimating the fundamental frequency of melodic source in music. In popular music, melody is usually performed by the main vocal and thus the task is often recast into estimating the pitch of the singer when background music is accompanied. The melodic source is usually mixed to be louder than other instruments in music production [54]. Thus, on a time-frequency representation of the mixture, it appears to have predominant harmonic patterns. Many of previous melody extraction algorithms focused on leveraging the prior knowledge, for example, by computing the pitch salience and tracking the melodic source using a set of rules or classifiers on it [7, 21].

On the other hand, the black-box approach that predicts the melodic pitch by data-driven learning algorithms has drawn much attentions recently, as deep learning has become successful [16, 20, 37]. They typically form the melody extraction task as a multi-class classification problem that choose one out of the possible pitch values. Specifically, the continuous pitch range of the melodic source is quantized such that the resolution is sufficiently high and the ground truth pitch is represented as an one-hot vector representation after the quantization.

Unlike categorical labels such as music genres or image objects, however, pitch is a scale value that continuously increases or decreases, and thus small deviations (e.g., less than 50 cents) in pitch prediction can be more acceptable than large differences. Also, the distance in pitch scale is highly non-linear. For example, harmonically related pitch ratios, such as 1:2 (octave) or 1:4 (double octave) are closer than other ratios that have less absolute differences (e.g. the triton). In fact, octave errors in pitch detection is very common due to the overlap of harmonics patterns in a time-frequency representation [6]. Considering the unique characteristics of pitch label, some of classification-based melody extraction (or pitch estimation) algorithms modified the loss functions. For example, Bitnner *et al.* [20] used a Gaussian-blurred one-hot label that mitigates the loss for small pitch deviations around the ground truth. Park and Yoo [37] proposed the harmonic sum loss that includes an additional term that penalizes harmonically close pitch predictions more to reduce the octave errors.

Another issue is that the pitch distribution of melodic sources is not even in music. In general, the mid range has a higher level than the low or high range in pitch distribution, typically having a Gaussian bell shape [14]. This label imbalance is common in other domains as well. This issue is often addressed by putting more weights for less frequent labels. Recently, Lin *et al.* proposed the focal loss to solve the class imbalance problem in object detection and showed improved performance [55]. The focal loss was applied to melody extraction as well [56].

Although the modified loss functions are claimed to be more effective than the standard categorical cross-entropy, there has been no study that compares them together in the same model and experimental setting. In this paper, we evaluate each of the loss functions using two classification models; one is based on Convolutional Neural Network (CNN) and the other is on Convolutional Recurrent Neural Network (CRNN). In addition, we propose variations of the modified loss functions. Through the performance comparison of the loss functions, we will discuss the directions to improve the melody extraction accuracy in different training and test scenarios.



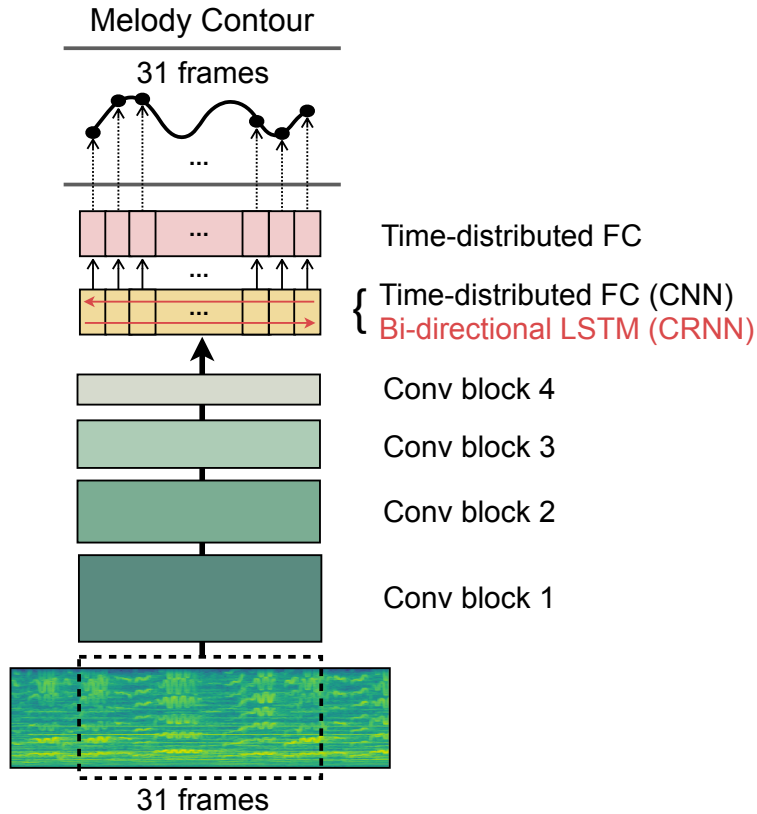


Figure 4.1: CNN and CRNN architectures for melody extraction. The max-pooling operations in the convolutional blocks are conducted along the frequency axis only while preserving the temporal dimensionality.

## 4.2 Methods

### 4.2.1 Learning Models

The architectures of the CNN and CRNN models for melody extraction are shown in Figure 4.1 and Table 4.1. We designed the convolutional blocks in a VGGNet style [57]. That is, the networks are configured with four convolutional blocks and each block contains two convolutional layers and the size of the filter is  $3 \times 3$ . However, we should note that max-pooling is applied only to the frequency axis with pooling size  $(1 \times 4)$  while preserving the time-wise dimensionality. A time-distributed fully connected layer (TD-FC) and a Bi-directional Long Short Term Memory (Bi-LSTM) on top of the convolutional blocks still preserves the input size. Finally, we compute the loss function separately on each frame of the time-distributed softmax layer output and the pitch labels, and then add them up to obtain the final loss function. We used batch normalization on each convolutional layer and the Leaky ReLU as an activation function. We also included the dropout at the end of the block to alleviate overfitting.

### 4.2.2 Existing Loss Functions

#### CE with Gaussian-blurred one-hot label vector ( $CE_G$ )

When the categorical Cross-Entropy (CE) is used as a loss function, the one-hot vector representation of pitch labels returns the same loss unless the predicted pitch exactly matches the pitch label with a

	Output size	CNN	CRNN
input	31×513	-	
conv1	31×128	[3×3,64]×2	
conv2	31×32	[3×3,128]×2	
conv3	31×8	[3×3,256]×2	
conv4	31×2	[3×3,256]×2	
TD-FC	31 × 2048	1024	-
Bi-LSTM	31 × 512	-	256
TD-FC	31×721	721	
softmax	31×721	721	

Table 4.1: CNN and CRNN model Configurations.

high pitch resolution. To address the issue, attenuating the penalty for near-correct predictions has been introduced [20, 58]. They implemented it by conducting Gaussian-blurring on the one-hot vectors. Specifically, they defined the CE between the prediction  $\hat{\mathbf{y}}$  and the Gaussian-blurred labels  $\mathbf{y}_g$  as below:

$$L = L_{CE}(\mathbf{y}_g, \hat{\mathbf{y}}) \quad (4.1)$$

$$y_g(i) = \begin{cases} \exp(-\frac{(c_i - c_{true})^2}{2\sigma_g^2}) & \text{if } |c_i - c_{true}| \leq M, \\ 0 & \text{otherwise,} \end{cases} \quad (4.2)$$

where  $L_{CE}(\mathbf{y}, \hat{\mathbf{y}})$  is the cross-entropy loss for the pitch prediction  $\hat{\mathbf{y}}$ .  $c_{true}$  is the constant index of the true pitch and  $c_i$  is a variable index.  $M$  determines the number of non-zero elements. We set  $M$  to 2 in our experiment.

### Harmonic Sum Loss (HSL)

The octave error is very common in pitch estimation because two pitches with octave relations are harmonically close to each other. The harmonic sum loss was proposed to reduce such octave mismatch [18]. They considered all possible octave relations (e.g. 2, 1/2, 4, 1/4, ...) and implemented it by adding a separate loss as below:

$$L = L_{CE}(\mathbf{y}, \hat{\mathbf{y}}) + \lambda_h L_{HS}(\mathbf{y}_h, \hat{\mathbf{y}}) \quad (4.3)$$

$$L_{HS}(\mathbf{y}_h, \hat{\mathbf{y}}) = \mathbf{y}_h^T \cdot \hat{\mathbf{y}} \quad (4.4)$$

$$y_h(i) = \begin{cases} 1 & \text{if } \text{mod}(c_i, L) = \text{mod}(c_{true}, L) \\ & \text{and } c_i \neq c_{true}, \\ 0 & \text{otherwise,} \end{cases} \quad (4.5)$$

where  $L$  is an integer by which one octave is apart on the target label,  $\lambda_h$  is a balancing weight, and  $\mathbf{y}_h$  is the one-hot vector with a value of 1 for all octave frequencies of  $\mathbf{y}$ .  $c_{true}$  is the constant index of the true pitch and  $c_i$  is a variable index.

### Focal loss (FL)

The focal loss was proposed to focus on training minor classes by down-weighting dominant examples in imbalanced datasets [55]. This imbalance is typical in pitch distributions so that it can be effective in melody extraction [56]. The focal loss  $L_f$  is defined as below:

$$L_f(\mathbf{y}, \hat{\mathbf{y}}) = -\alpha(1 - \hat{\mathbf{y}})^\gamma \mathbf{y}^T \cdot \log(\hat{\mathbf{y}}) \quad (4.6)$$

where  $\alpha$  is a weighting factor and we set  $\alpha = 0.25$ ,  $\gamma = 2$  in this work.  $(1 - \hat{\mathbf{y}})^\gamma$  is a modulating factor to suppress high prediction probabilities.

### 4.2.3 Proposed Loss Functions

#### Gaussian-blurred Pitch Loss (GPL)

The Gaussian-blurred pitch loss is a modified version of  $CE_G$ . This implements the loss attenuation of nearly-correct pitch predictions as a separate loss terms as done in HSL. The attenuation loss term is defined as below:

$$L = L_{CE}(\mathbf{y}, \hat{\mathbf{y}}) - \lambda_p L_{GP}(\mathbf{y}_g, \hat{\mathbf{y}}) \quad (4.7)$$

$$L_{GP}(\mathbf{y}_g, \hat{\mathbf{y}}) = \mathbf{y}_g^T \cdot \frac{\hat{\mathbf{y}}}{\|\hat{\mathbf{y}}\|_2} \quad (4.8)$$

where  $\mathbf{y}_g$  is the Gaussian-blurred one-hot vector defined in Equation 4.2,  $\lambda_p$  is a balancing weight and  $\|\mathbf{y}\|_2$  indicates L2 normalization.

#### Gaussian-blurred Octave Loss (GOL)

Another variation is based on combining  $CE_G$  and HSL. That is, we discourage pitch predictions with octave errors considering nearby pitches as well. In this loss, however, we counted only one octave up or down with regard to the target pitch as errors with more than two octaves did not make much contributes in our preliminary experiment. Therefore, we implemented the Gaussian-blurred octave loss as below:

$$L = L_{CE}(\mathbf{y}, \hat{\mathbf{y}}) + \lambda_o L_{GO}(\mathbf{y}_o, \hat{\mathbf{y}}) \quad (4.9)$$

$$L_{GO}(\mathbf{y}_o, \hat{\mathbf{y}}) = \mathbf{y}_o^T \cdot \frac{\hat{\mathbf{y}}}{\|\hat{\mathbf{y}}\|_2} \quad (4.10)$$

$$y_o(i) = \begin{cases} \exp\left(-\frac{(c_i - c_{true'})^2}{2\sigma_o^2}\right) & \text{if } |c_i - c_{true'}| \leq M, \\ & \text{and } c_i \neq c_{true'}, \\ 0 & \text{otherwise,} \end{cases} \quad (4.11)$$

$$c_{true'} \in \{c_{true}, c_{true_{oct+}}, c_{true_{oct-}}\} \quad (4.12)$$

where  $y_o$  is the Gaussian-blurred there-hot vector where the left and the right correspond to octave-up ( $c_{true_{oct+}}$ ) and -down ( $c_{true_{oct-}}$ ) of the target pitch ( $c_{true}$ ).  $\lambda_o$  is a balancing weight.

## 4.3 Experiments

### 4.3.1 Dataset

We used the RWC music database [44] as a primary dataset to evaluate the models and loss functions. It has 80 Japanese popular songs and 20 American popular songs with singing voice melody annotations. We divided it into three splits, 70 songs for training, 15 songs for validation and the remaining 15 songs for test. We carefully selected the songs so that genres and genders are evenly distributed for each split and also the songs of the same singer are not included in the other splits. To train the models, we augmented the training set by pitch-shifting  $\pm 1, 2$  semitones. We also used ADC2004 and LabROSA<sup>1</sup> to test the models on unseen datasets. To focus on vocal melody extraction, we excluded non-vocal audio clips.

### 4.3.2 Training

We first resampled the audio files to 16 kHz and merged into mono channel in the time domain. We then computed spectrogram using a 1024 point Hann window and a hop size of 160 samples (10ms) and compressed the magnitude of the spectrogram in a log scale. To train the models, we selected only the segments where all frames have corresponding pitch values. The pitch labels range from D2 to B5. We quantized the pitch range such that one semitone is divided into 16 steps. As a result, each pitch value is represented as a 721-dimensional one-hot vector. We initialized the network parameters using the He technique [48] and minimized the loss functions using Adam. We set the initial learning rate to 0.001, and adopted the early-stopping strategy. We conducted our experiment using Keras on a machine with two GTX 1080 Ti GPUs.

### 4.3.3 Evaluation Metrics

To compare the loss functions, we tested the models only for voiced frames. Therefore, we primarily used the raw pitch accuracy (RPA), raw chroma accuracy (RCA), and difference between RPA and RCA as evaluation metrics. After predicting pitch for all frames, we converted the quantized pitch labels to frequency scale  $f$  (Hz) to compare them to the ground truth pitch. The predicted pitch is considered correct if the difference between the prediction and the ground-truth is within  $\pm 50$  cents (0.5 semitone). We calculated them using *mir-eval*, a Python library designed for objective evaluation in Music Information Retrieval (MIR) tasks [47].

## 4.4 Results & Discussion

Table 4.2 shows the melody extraction results for the evaluated loss functions, including the baseline loss (i.e., the standard CE with one-hot vector) on the test datasets. All conditions were identical during training, and only the loss functions were changed for fair comparison. The overall results show that the modified loss functions generally outperform the baseline indicating that any of the modifications is toward a right direction to improve the performance. CRNN generally outperforms CNN regardless of the loss functions and the difference between RCA and RPA is also smaller in CRNN. This confirms that learning temporal dependency in the upper layer is effective for estimating pitch as well as for reducing the octave error by bolstering temporal continuity. Interestingly, the performance gaps among the loss

<sup>1</sup><http://labrosa.ee.columbia.edu/projects/melody/>

	RWC			ADC04			LabROSA		
	RPA	RCA	diff(%)	RPA	RCA	diff(%)	RPA	RCA	diff(%)
<b>CE</b>	0.8390	0.8781	3.91	0.7737	0.8228	4.91	0.8518	0.8806	2.88
<b>CE<sub>G</sub></b>	0.8726	0.8918	1.92	<b>0.8292</b>	0.8498	2.06	0.8786	0.8891	1.05
<b>GPL</b>	0.8658	0.8859	2.00	0.8156	0.8558	4.01	0.8855	0.9010	1.55
<b>HSL</b>	0.8471	0.8817	3.46	0.7774	0.8229	4.55	0.8543	0.8801	2.58
<b>GOL</b>	0.8657	0.8865	2.08	0.8187	<b>0.8577</b>	3.90	0.8868	<b>0.9052</b>	1.84
<b>FL</b>	<b>0.8829</b>	<b>0.8961</b>	<b>1.32</b>	0.8283	0.8497	<b>2.15</b>	<b>0.8926</b>	0.9014	<b>0.88</b>

(a) CNN model

	RWC			ADC04			LabROSA		
	RPA	RCA	diff(%)	RPA	RCA	diff(%)	RPA	RCA	diff(%)
<b>CE</b>	0.8858	0.8992	1.34	0.8145	0.8371	2.26	0.8909	0.9016	1.08
<b>CE<sub>G</sub></b>	0.8956	<b>0.9082</b>	1.26	0.8412	0.8597	1.85	0.9039	0.9113	0.74
<b>GPL</b>	0.8903	0.9006	1.03	<b>0.8455</b>	0.8598	1.42	0.9053	0.9092	0.39
<b>HSL</b>	0.8867	0.9007	1.40	0.8381	0.8560	1.79	0.9013	0.9059	0.46
<b>GOL</b>	0.8820	0.8912	0.92	0.8394	0.8556	1.62	<b>0.9115</b>	<b>0.9142</b>	<b>0.27</b>
<b>FL</b>	<b>0.8978</b>	0.9050	<b>0.72</b>	0.8320	<b>0.8644</b>	2.24	0.9035	0.9093	0.58

(b) CRNN model

Table 4.2: Comparison of loss functions for melody extraction in the CNN and CRNN models. Note that the models were evaluated on voiced frames only to focus on the comparison.

functions are also reduced in the CRNN model. In the followings, we discuss more detail about how the loss functions influence the performance.

- **Gaussian-blurring:** CE<sub>G</sub> and GPL include the Gaussian-blurred target label but with different implementations. While both of them improve the accuracies, the results are slightly different depending on the models and whether the training set and test set are from the same dataset. In RWC, CE<sub>G</sub> generally performs better than GPL. In the unseen datasets (ADC04, LabROSA), however, GPL achieves higher accuracies than CE<sub>G</sub> especially in the CRNN model.
- **Penalty loss for octave errors:** HSL and GOL have an additional penalty for octave errors. The results in Table 4.2 show that the octave loss functions help the discriminative learning in both CNN and CRNN models. HSL achieves higher accuracies than GOL in RWC with the CRNN model. In the unseen dataset, however, GOL are generally better than HSL.
- **Weighting of imbalanced dataset:** The results of the focal loss (FL) show notable improvements in both CNN and CRNN models. Particularly, in RWC where the training set and test set are from the same dataset, FL outperforms other loss functions in RPA and the difference between RPA and RCA. On the other hand, in the unseen test sets, FL is not consistently prominent. This may be explained by the fact that FL was designed based on the statistical property of data whereas other loss functions were based on domain knowledge in melody extraction. In other words, the statistically-driven loss is more effective when the training set and the test set are homogeneous, whereas the knowledge-driven losses better overcome the discrepancy when the two sets are heterogeneous. This indicates that the focal loss is a recommended choice if the size of

training set with pitch labels is sufficiently large. Otherwise, the knowledge-driven loss functions (GPL, GOL, and  $CE_G$ ) can be better alternatives.

## 4.5 Conclusion

In this paper, we have reviewed the loss functions ( $CE_G$ , HSL, FL) used in the melody extraction task so far and also proposed the two variations (GPL, GOL). We showed that modified loss functions are generally better than the standard cross-entropy loss. There was no single best one for all test sets. However, if the training data with pitch label is sufficiently large, the statistically-driven loss (FL) is recommended. Otherwise, the knowledge-driven loss (GPL, GOL or  $CE_G$ ) can be better choices.

# Chapter 5. Joint Detection and Classification of Singing Voice Melody Using Convolutional Recurrent Neural Networks

## 5.1 Introduction

Melody extraction is estimating the fundamental frequency or pitch corresponding to the melody source. In popular music, the singing voice is commonly the main melodic source; thus, extracting the pitch of the singing voice in polyphonic music is the most common task. The extracted melody is useful in many ways. For example, the result can be directly applied to melody-based retrieval tasks such as query-by-humming [59] or cover song identification [30]. The pitch contour that contains the unique expressiveness of individual songs can be used as a feature for high-level tasks such as different music genre classification [31] or singer identification [32].

Singing voices are usually mixed to be louder than background music played with musical instruments in music production [54]. Furthermore, singing voices generally have different characteristics from those of music instruments; they have expressive vibrato and various formant patterns unique to vocal singing. A number of previous methods exploit the dominant and unique spectral patterns for melody extraction leveraging prior knowledge and heuristics. For example, they include calculating the pitch salience [6, 9, 53, 60, 61] or separating the melody source [11–13] to estimate the fundamental frequencies of melody. In contrast, a data-driven approach using machine learning [2] has also been proposed. Recently, deep learning has been the main approach, as it has proven to be very successful in a wide variety of fields. Researchers have attempted various deep neural network architectures for melody extraction. Examples include fully-connected neural networks (FNN) [16, 17], convolutional neural networks (CNN) [20, 62], recurrent neural networks (RNN) [37], convolutional recurrent neural networks (CRNN) [19], and encoder-decoder [22, 28].

Singing melody extraction involves detecting voice segments because the melodic source is not always active in the music track. Voice detection is a very important task that affects the performance of melody extraction. The methods for singing voice detection can be roughly divided into three approaches. One approach is detecting the voice segments by thresholding the likelihood of pitch estimation [16, 20, 62]. This method is very simple, but more inaccurate because the optimal threshold value may vary depending on the level ratio between the voice and background sound. Another approach is constructing a separate model for singing voice detection [17, 61]. Since the model is dedicated to the voice detection, it can achieve better performance. However, it is obvious that it takes more effort to train the model separately, and the complexity of the whole melody extraction algorithm increases. The last approach is adding an explicit “non-voice” label to a list of target pitch outputs. This is particularly valid in classification-based melody extraction methods [19, 25, 37]. When singing voice is present, one of the target pitches is chosen as an output class. Otherwise, the “non-voice” label is chosen as another output class. This “none of the above” labeling method is also known to have a regularization effect [63].

Our proposed melody extraction model is based on the last approach. However, we pay attention to the discrepancy of the abstraction level between voice detection and pitch classification. Voice detection is mainly based on vibrato or formant modulation patterns that distinguish it from other musical instruments. This requires a much wider context than identifying pitch from voice segments, which can be estimated even from a single frame of audio. That is, the voice/non-voice discrimination is a higher-

level task than the pitch classification. We handle this dual-target problem using a joint detection and classification (JDC) network. JDC can be regarded as multi-task learning [64]. It has been shown to improve generalization by combining task-specific information contained in network parameters for each related task. In the area of music or audio, it has been applied to tasks such as source separation [65, 66] and bird audio detection [67]. In both tasks, detecting the source of interest, that is vocal or bird sound, is an important subtask that affects the performance of the main task. An additional network was built to detect vocal or bird sound and was integrated with the main network to improve the performance. Likewise, we also added an auxiliary network (*AUX*) trained to detect singing voice on top of the main melody extraction network.

In summary, the contributions of this paper are as follow: first, we propose a CRNN architecture with residual connections and bi-directional long short-term memory (Bi-LSTM) as the main network so that the model classifies the pitch with a high resolution. Second, beyond the melody classification model, we propose a JDC network that can perform two tasks in melody extraction (singing voice detection and pitch classification) independently. The auxiliary network that detects the singing voice is trained using multi-level features shared from the main network. Third, we propose a joint melody loss designed to combine the two tasks together for the JDC network and to be effective for generalization of the model. Finally, by comparing the results on public test sets, we show that the proposed method outperforms state-of-the-art algorithms. Code and the pre-trained model used in this paper are available at [https://github.com/keums/melodyExtraction\\_JDC](https://github.com/keums/melodyExtraction_JDC).

## 5.2 Proposed Method

Figure 5.1 illustrates the overall architecture of the proposed singing melody extraction model. This section describes the details.

### 5.2.1 The Main Network

#### Architecture

The main network is the central part that extracts singing melody from polyphonic music audio. The architecture was built with 1 ConvBlock, 3 ResBlocks, 1 PoolBlock, and a bi-directional long short-term memory (Bi-LSTM) layer. The diagram of the architecture is shown in Figure 5.1a. The parameters and output sizes of the modules are listed in Table 5.1. ConvBlock is a module consisting of two  $3 \times 3$  convolutional (Conv) layers, with a batch normalization (BN) layer [68] and a leaky rectified linear unit (LReLU) between them [69].

ResBlock is a variation of ConvBlock that has an additional BN/LReLU, a max-pooling (MaxPool) layer with a pooling size of four, and a skip connection, inspired by ResNet of a full pre-activation version [70]. The max-pooling was only conducted in the frequency axis throughout all blocks; therefore, it preserved the input size (31 frames) in the time axis. The skip connection had an  $1 \times 1$  convolution layer to match the dimensionality between two feature maps. PoolBlock was another module that consists of BN, LReLU, and MaxPool. The dropout rate of 50% was added in the end of the PoolBlock to alleviate overfitting. Finally, the Bi-LSTM layer took 31 frames of features ( $2 \times 256$ ) from the convolution blocks (note that the max-pooling preserved the input size) and predicted pitch labels via the softmax function in a sequence-to-sequence manner.

We experimented with different kernel sizes of 1D or 2D convolution, but 2D-convolution with



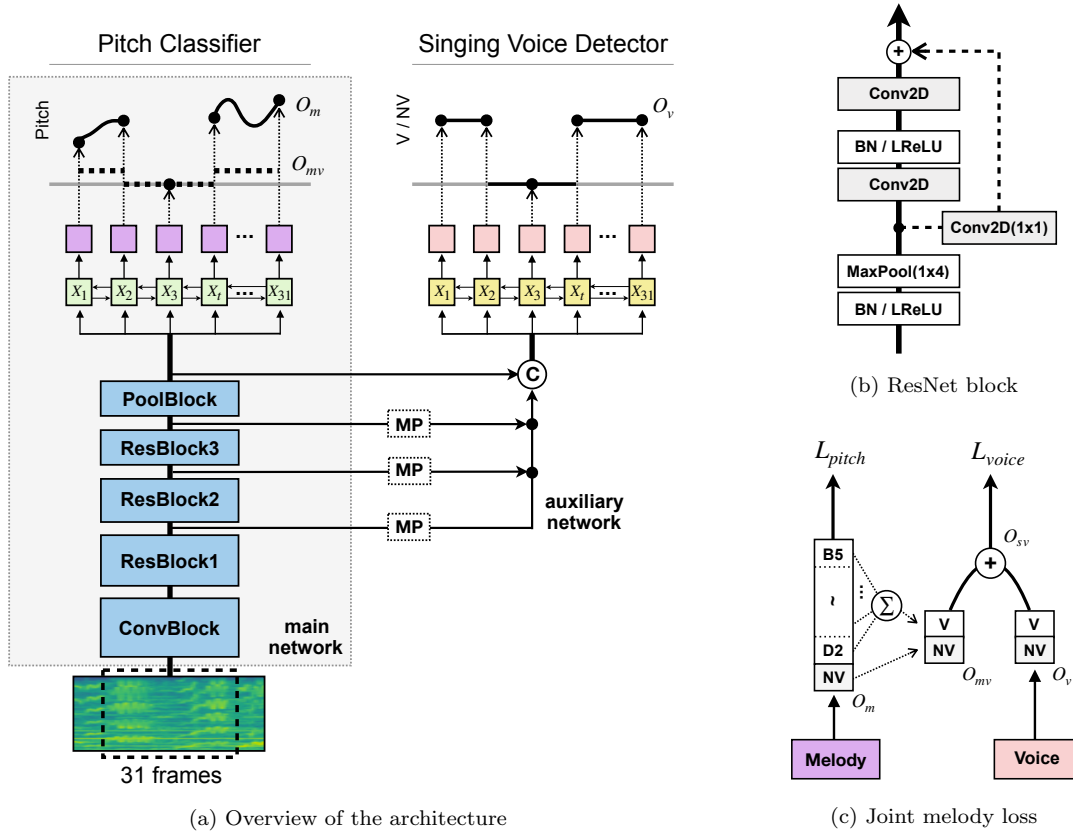


Figure 5.1: Overview of the architecture and joint melody loss (a) CRNN architectures of the main network and joint detection and classification (JDC) network for melody extraction. ConvBlock is organized in the order of ‘Conv-BN-leaky rectified linear unit (LReLU)-Conv’, and PoolBlock is in the order of ‘BN-LReLU-MaxPool’. We denote max-pooling and concatenation as “MP” and “C”, respectively. The max-pooling is applied only to the frequency axis while preserving the time-wise dimensionality. (b) The diagram of the ResNet block (c) The block diagram of joint melody loss.  $o_m$  is the softmax output of the melody from the main network.  $o_{mv}$  and  $o_v$  are the softmax output of the singing voice activity from the main network and auxiliary network, respectively. “V” and “NV” indicate voice and non-voice, respectively.

$3 \times 3$  filters was the most effective for melody extraction. Veit [71] showed that residual networks can be viewed as a collection of many paths. The skip connection allowed the output of each layer to be input into all subsequent blocks connected, and it made a residual network ensemble system. In our experiments, ResNet model achieved better results compared to the VGGNet [57] model. We used a total of three ResBlocks in this model. Reducing the number of ResBlocks resulted in lower performance. On the other hand, when we increased the number of ResBlocks, we could not see any noticeable performance improvement. This shows that very high-level features are not required to predict the pitch of the frame in the spectrogram.

The pitch labels ranged from D2 (73.416 Hz) to B5 (987.77 Hz) with a resolution of 1/16 semitone (i.e., 6.25 cents). Furthermore, “non-voice” (or “zero-pitch”) was added to the pitch labels. This special label was active when the singing voice was not present. Therefore, the total number of labels (the size of the output layer) became 722. The main network used spectrograms as input data. Specifically, we merged audio files into a mono channel and down-sampled those to below 8 kHz, in which the majority

Table 5.1: Model configurations of the main and joint networks. ConvBlock and ResBlock have two convolutional layers;  $[n \times n, k]$  denotes a convolutional operator of  $n$  filters and a kernel size of  $k$ .

	Components		Output Size	
	<i>Main</i>	<i>Main + AUX</i>	<i>Main</i>	<i>Main + AUX</i>
Input	-		$31 \times 513$	
Conv block	$[3 \times 3, 64] \times 2$		$31 \times 513, 64$	
ResNet Block 1	$[3 \times 3, 128] \times 2$		$31 \times 128, 128$	
ResNet Block 2	$[3 \times 3, 192] \times 2$		$31 \times 32, 192$	
ResNet Block 3	$[3 \times 3, 256] \times 2$		$31 \times 8, 256$	
Pool block	-		$31 \times 2, 256$	
Bi-LSTM	256	$256 + 32$	$31 \times 512$	$31 \times (512 + 64)$
FC	722	$722 + 2$	$31 \times 722$	$31 \times (722 + 2)$

of the singing voice spectrum is distributed. We used a 1024-point Hann window and a hop size of 80 samples (10 ms) to compute the spectrogram and compressed the magnitude in a log scale. Finally, we used 513 bins from 0 Hz–4000 Hz and 31 consecutive frames as the input of the main network.

### Loss Function

We quantized the continuous scale of the pitch range into a discrete set of values to form the output layer in a classification setting. They are often represented as a one-hot vector to incorporate it into the categorical cross entropy (CE) loss function [16, 17]. One problem of the loss function is that, unless the predicted pitch is close enough to the ground-truth pitch within the quantization size (6.25 cents in our case), it is regarded as the “wrong class”. In order to mitigate excessive loss for neighboring pitches of the ground truth, a Gaussian-blurred version of one-hot vector was proposed [20, 58]. We also adopted this version, so the loss function of the main network ( $L_{pitch}$ ) was defined between the prediction  $\hat{\mathbf{y}}$  and the Gaussian-blurred labels  $\mathbf{y}_g$  as below:

$$L_{pitch} = CE(\mathbf{y}_g, \hat{\mathbf{y}}) \quad (5.1)$$

$$y_g(i) = \begin{cases} \exp\left(-\frac{(c_i - c_{true})^2}{2\sigma_g^2}\right) & \text{if } c_{true} \neq 0 \text{ and } |c_i - c_{true}| \leq M, \\ 0 & \text{otherwise,} \end{cases} \quad (5.2)$$

where  $CE(\mathbf{y}_g, \hat{\mathbf{y}})$  is the cross-entropy loss for the pitch prediction.  $c_{true}$  is the constant index of the true pitch, and  $c_i$  is a variable index.  $M$  determines the number of non-zero elements. We set  $M$  to three and  $\sigma_g$  to one in our experiment.

### 5.2.2 Joint Detection and Classification Network Architecture

The output layer of the main network was formed with pitch labels and a special non-voice label. These labels were handled as a set of classes in the same level. Although voice detection and pitch estimation in the melody extraction task have a close relationship, they require different levels of abstraction, as mentioned in Section 5.1. In pitch estimation, the network predicts a continuously-varying

value of pitch at each frame, although it uses contextual information from neighboring frames. In voice detection, the network predicts the sustained binary status of voice from textures that can be obtained from a wider context, such as vibrato or formant modulation. Therefore, simply adding a non-voice label to the target pitch labels has limitations in extracting the characteristics of voice activity.

In order to address the discrepancy between the pitch estimation and voice detection, we propose a joint detection and classification (JDC) network. We set up the JDC network so that the two tasks shared the modules. Instead of building it with new modules, we maintained the existing main network and added a branch for dedicated singing voice detection. As shown in Figure 5.1a, the JDC network shared ConvBlock, ResBlock, and PoolBlock in the bottom, but had a separate Bi-LSTM module for each task. In particular, the voice detection task took the combined features of the shared modules from the main network. The use of multi-level features stemmed from the idea that voice detection may require observing diverse textures in different levels of abstraction. The outputs of ResBlock were max-pooled to match the output size, and then, they were concatenated. The Bi-LSTM layer predicted the probabilities that there was a singing voice from the concatenated features in a sequence-to-sequence manner via the softmax function. We call this an auxiliary network. The features from convolutional blocks were learned jointly using the main and auxiliary network, and the loss function was combined with that derived from the main network to form the final loss function. The detail is described in the next section.

### Joint Loss Function

The JDC model was optimized by minimizing a joint melody loss that combines the two loss functions from the main network and the auxiliary network, respectively, as illustrated in Figure 5.1c. We detected the singing voice using both networks. That is, we summed the 721 pitch predictions in the output of the main network,  $o_m$ , and converted them to a single “voice” prediction. This resulted in the voice detection output from the main network,  $o_{mv}$ . We then added this to the output of the auxiliary network,  $o_v$ , to make a decision for voice detection as below:

$$o_{sv} = o_{mv} + o_v \quad (5.3)$$

Then, the loss function for voice detection was defined as the cross-entropy between the sum of the voice output and the ground truth  $\mathbf{v}_{gt}$ :

$$L_{voice} = CE(\text{softmax}(o_{sv}), \mathbf{v}_{gt}) \quad (5.4)$$

Finally, the joint melody loss function was given by combining the loss function for voice detection ( $L_{voice}$ ) and the loss function for pitch estimation ( $L_{pitch}$ ):

$$L_{joint} = L_{pitch} + \alpha L_{voice} \quad (5.5)$$

where  $\alpha$  is a balancing weight, and we used  $\alpha = 0.5$  in our experiment (In our initial experiment, we tried three different values of  $\alpha$  (0.1, 0.5, and 1) and achieved the best overall accuracy with 0.5 on the test datasets. Then, we fixed  $\alpha = 0.5$  in the rest of the experiments. This might not be optimal, and selecting an optimal value could improve the result.).

## 5.3 Experiments

### 5.3.1 Datasets

#### Train Datasets

We used the following three datasets to train the models. The songs were carefully selected so that genres and singer genders were evenly distributed for each split and songs from the same singer were not included in the other splits.

- **RWC** [44]: 80 Japanese popular songs and 20 American popular songs with singing voice melody annotations. We divided the dataset into three splits: 70 songs for training, 15 songs for validation, and the remaining 15 songs for testing.
- **MedleyDB** [34]: 122 songs with a variety of musical genres. Among them, we chose 61 songs that are dominated by vocal melody. We divided the dataset into three splits: 37 songs for training, 10 songs for validation and 12 songs for testing. For a comparison of results under the same conditions, we selected the training sets according to [62].
- **iKala** [35]: 262 Chinese songs clips of 30 s performed by six professional singers. We divided the dataset into two splits: 235 songs for training and 27 songs for validation.

We augmented the training sets by conducting pitch-shift on the original audio files by  $\pm 1, 2$  semitones to obtain more generalized models. Pitch-shifting has proven to be an effective way to increase data and improve results for singing voice activity detection [38], as well as melody extraction [16]. To this end, we used an algorithm based on a phase vocoder that conducts pitch-shifting independent of time-stretching [46].

#### Test Datasets

Four datasets (ADC04, MIREX05 (<http://labrosa.ee.columbia.edu/projects/melody/>), MedleyDB, and RWC) to evaluate the performance of melody extraction were used. Non-vocal audio clips were excluded to focus on vocal melody extraction. To compare the voice detection performance, our models were also evaluated using Jamendo [45], a public dataset that is mainly used for singing voice detection.

- **ADC04**: 20 excerpts of 20 s that contain pop, jazz, and opera songs, as well as synthesized singing and audio from MIDI files. Jazz and MIDI songs were excluded from the evaluation.
- **MIREX05**: 13 excerpts that contain rock, R&B, pop, and jazz songs, as well as audio generated from a MIDI file. We used 12 songs out of a total of 20, excluding jazz and MIDI files for evaluation.
- **MedleyDB**: 12 songs not included in the training set. For a comparison of results under the same conditions, we selected the test sets according to [62].
- **RWC**: 15 songs not included in the training set. This was used internally to evaluate the performance of the proposed models.
- **Jamendo**: 93 songs designed for the evaluation of singing voice detection. Among them, only 16 songs designated as a test set to measure the performance of singing voice detection were used.

### 5.3.2 Evaluation

We evaluated the proposed method in terms of five metrics, including overall accuracy (OA), raw pitch accuracy (RPA), raw chroma accuracy (RCA), voicing detection rate (VR), and voicing false alarm rate (VFA). The measures are defined as below:

$$\text{RPA, RCA} = \frac{\#\{\text{voice frames for which (pitches, chromas) are predicted correctly}\}}{\#\text{ of voiced frames}} \quad (5.6)$$

$$\text{OA} = \frac{\#\{\text{frames for which pitches and voicing are predicted correctly}\}}{\#\text{ of all frames}} \quad (5.7)$$

$$\text{VR} = \frac{\#\{\text{voiced frames for which voicing are predicted correctly}\}}{\#\text{ of voiced frames}} \quad (5.8)$$

$$\text{VFA} = \frac{\#\{\text{frames that are predicted as voiced, but not actually voiced}\}}{\#\text{ of unvoiced frames}} \quad (5.9)$$

The evaluation consists of two main parts: voice detection determining whether the voice is included in a particular time frame (VR and VFA) and pitch estimation determining the melody pitch for each time frame (RPA, RCA). OA is the combined accuracy of pitch estimation and voice detection. We converted the quantized pitch labels in 6.25 cents (1/16 semitone) to frequency scales (Hz) to compare them with the ground truth.

$$f = 2^{(m-69)/12} \times 440 \text{ (Hz)} \quad (5.10)$$

where  $m$  is the MIDI note number estimated by the main network for melody extraction. The pitch at each frame was considered correct if the difference between the estimated pitch and the ground-truth pitch was within a tolerance of  $\pm 50$  cents. We computed them using *mir-eval*, a Python library designed for objective evaluation in MIR tasks [47].

### 5.3.3 Training Detail

We randomly initialized the network parameters using He uniform initialization [48] and trained them with the Adam optimizer. We repeated it over all the training data up to 45 epochs. The initial learning rate was set to 0.002. If the validation loss did not decrease within three epochs, the learning rate was reset to 80% of the previous value. Furthermore, if the validation loss did not decrease during seven epochs, the training stopped. For fast computing, we ran the code using Keras [49], a deep learning library in Python, on a computer with two GPUs.

### 5.3.4 Ablation Study

We conducted an ablation study to verify the effectiveness of the proposed model. We experimented with the following settings to investigate the model. The architectures of each model used in the experiments are illustrated in Figure 5.2.

- **The effect of the auxiliary network:** The proposed JDC network model was compared to the main network only (without the auxiliary network). They are denoted by  $JDC_S$  and  $Main$ , respectively.

- **The effect of the combined voice detection in calculating the loss function:** The proposed model used the sum of the outputs from both the main and auxiliary networks (Equation (5.3)) in calculating the voice loss function. This was compared to the case where only the output of auxiliary network,  $o_v$ , was used in calculating the loss function, which is denoted by  $JDC_A$ .
- **The effect of the combined voice detection in predicting singing voice:** The JDC network can detect singing voice with three possibilities in the test phase:  $o_{mv}$  from the main network,  $o_v$  from the auxiliary network, and  $o_{sv}$  the sum of the two outputs. We compared the performance of the three melody extraction outputs and evaluated them for each of the two loss functions above. As a result, we had a total of six outputs, which are denoted by  $JDC_S(o_{mv})$ ,  $JDC_S(o_v)$ ,  $JDC_S(o_{sv})$ ,  $JDC_A(o_{mv})$ ,  $JDC_A(o_v)$ , and  $JDC_A(o_{sv})$ .

To demonstrate the effectiveness of the JDC network, we also examined the performances of the model  $Main(AUX)$  and  $Main(SVD)$ . In both models, the networks for pitch estimation and singing voice detector (SVD) were trained separately. Both networks for melody extraction were identical to  $Main$ , but only the labels corresponding to the pitches were used as the target labels. The architecture of SVD for  $Main(AUX)$  was identical to the auxiliary network. Following [38], we implemented the SVD for  $Main(SVD)$ . For fair comparison, the same training datasets were used in all training phases.

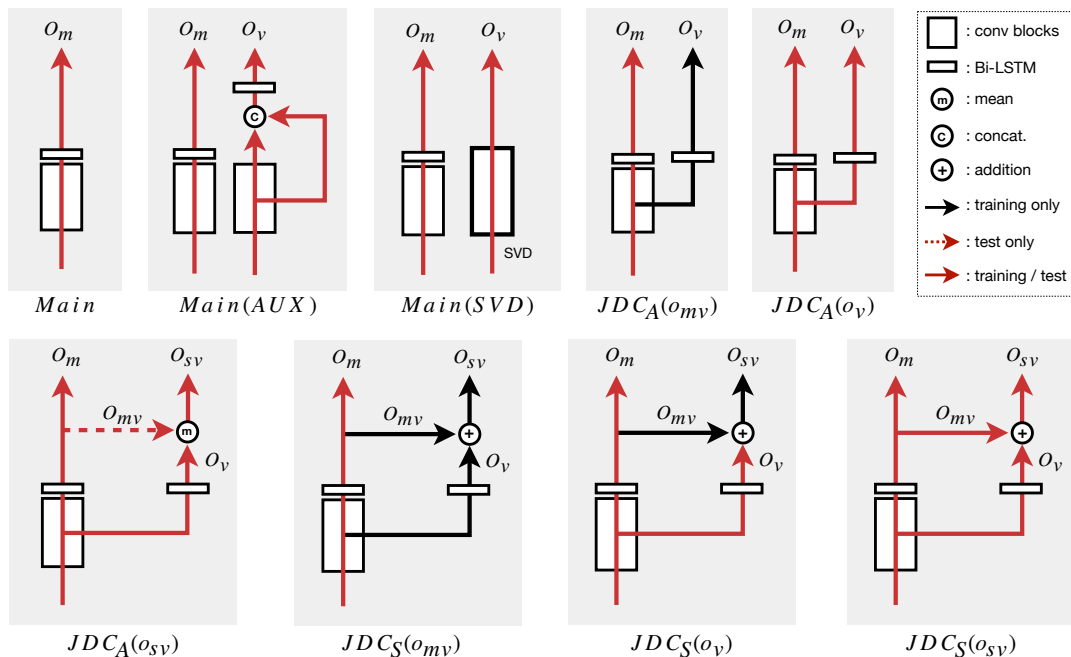


Figure 5.2: Architectures of melody extraction used for performance comparison in this paper. In the model name, the subscript refers to the type of JDC network, and parentheses refer to the source of voice detection output. The black and red solid arrows indicate the paths used for training and training/testing, respectively. The red dotted arrows indicate the path used for the test, although it was not used for training.

The models and voice detection outputs on the four test datasets were evaluated for melody extraction as mentioned in Section 5.3.1. We carried out five runs of training with different initializations and analyzed the results across different metrics of melody extraction evaluation and different test sets,

respectively. To further validate the effectiveness of the models, a  $t$ -test between the *Main* and each JDC network to compute the statistical significance on the results (the results were normally distributed (Shapiro–Wilk test:  $p \geq 0.05$ )) of the five trials was performed.

In addition, we examined their singing voice detection capability using the Jamendo dataset, which was dedicated to the voice detection task (there was no annotation on the pitch of the melody). Because this was for the voice detection task, we report only VR and VFA. In addition to the models and voice detection outputs in the ablation study, we also compared out best model to other state-of-the-art singing voice detection algorithms [36, 38, 51].

## 5.4 Results and Discussion

### 5.4.1 Comparison of Melody Extraction Performance

Figure 5.3a shows the results of the five melody extraction evaluation metrics for the compared models and outputs. In general, *JDC* networks were superior to *Main* in terms of OA, and among the *JDC* networks, *JDC<sub>S</sub>* networks that used the sum of the two outputs in the loss function were more accurate than *JDC<sub>A</sub>* that used only the output of the auxiliary network.

Both RPA and RCA increased significantly in all *JDC* networks, especially *JDC<sub>S</sub>(*o<sub>v</sub>*)* and *JDC<sub>S</sub>(*o<sub>sv</sub>*)*. This is mainly attributed to the increase in VR. That is, the *JDC* networks detected the activity of singing voice more responsively, having fewer missing errors. The average RPA and RCA of *Main* were 76.1% and 78.1%, respectively, while those of *JDC<sub>S</sub>(*o<sub>v</sub>*)* were 84.7% and 86.0%, respectively ( $p$ -value  $\leq 0.01$ ). However, both VR and VFA were high due to their aggressiveness, and this led to degradation in OA. On the other hand, *JDC<sub>S</sub>(*o<sub>mv</sub>*)* predicted the voice activity more reliably by significantly reducing VFA. The average VFA of *JDC<sub>S</sub>(*o<sub>sv</sub>*)* was 17.7%, but that of *JDC<sub>S</sub>(*o<sub>mv</sub>*)* was 9.0%. As a result, *JDC<sub>S</sub>(*o<sub>mv</sub>*)* achieved the highest average OA (85.7%,  $p$ -value  $\leq 0.01$ ), outperforming the two networks.

This result indicates that the voice detection output of the main network was more conservative than the output of the auxiliary network. This is true because the main network had more classes (i.e., pitch labels) with which to compete. However, comparing *JDC<sub>S</sub>(*o<sub>mv</sub>*)* to *JDC<sub>A</sub>(*o<sub>mv</sub>*)*, the main network in *JDC<sub>S</sub>(*o<sub>mv</sub>*)* became more sensitive to voice activity due to the influence of the auxiliary network. This reveals that combining *o<sub>mv</sub>* with *o<sub>v</sub>* in calculating the voice detection loss function (Equation (5.4)) contributed to driving more tightly-coupled classification and detection, thereby improving the performance of melody extraction.

The overall performance of *Main(AUX)* was generally higher than that of *Main*, but it did not outperform *JDC<sub>S</sub>(*o<sub>mv</sub>*)*. The average OA of *Main(SVD)* was comparable to *Main*, and the performance was lower than that of *Main(AUX)*. Experimental results also showed that the deviations of RPA and RCA of the proposed models were high, except for *Main(AUX)* and *Main(SVD)*. Since the proposed models were trained for both pitch estimation and voice detection at different levels of abstraction, they were sensitive to initialization.

Figure 5.3b shows the results of overall accuracy (OA) on the four test sets for the compared models and outputs. The performance gap varied by up to 10% depending on the dataset, indicating that the models were affected by the characteristic that each test set had (e.g., genre). Again, we see that the performances of the *JDC* networks were generally superior to that of *Main* for all test datasets.

Comparing *JDC<sub>A</sub>* to *JDC<sub>S</sub>* in each of the three cases (*o<sub>mv</sub>*, *o<sub>v</sub>*, and *o<sub>sv</sub>*), the average of OA for three *JDC<sub>A</sub>* and *JDC<sub>S</sub>* networks were 83.5% and 84.9%, respectively. *JDC<sub>S</sub>* networks were generally superior to *JDC<sub>A</sub>* networks. The average OA of *JDC<sub>S</sub>(*o<sub>mv</sub>*)* was improved by 3.17% over that of *Main*. With

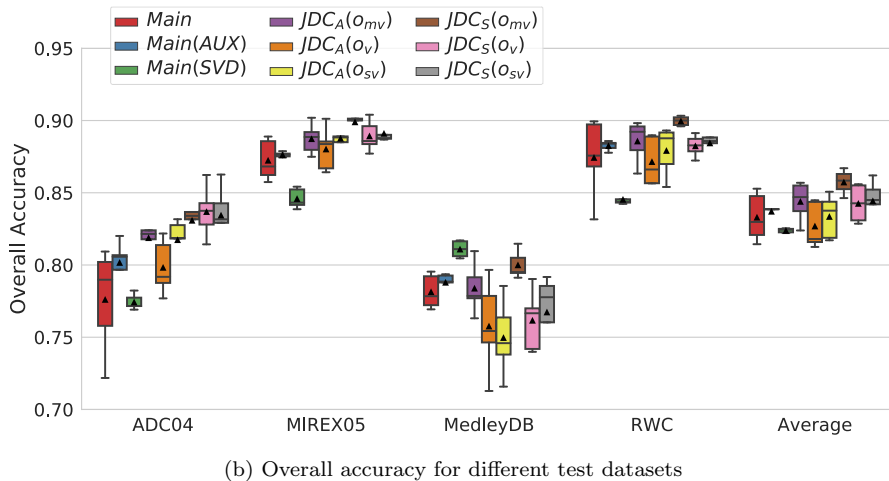
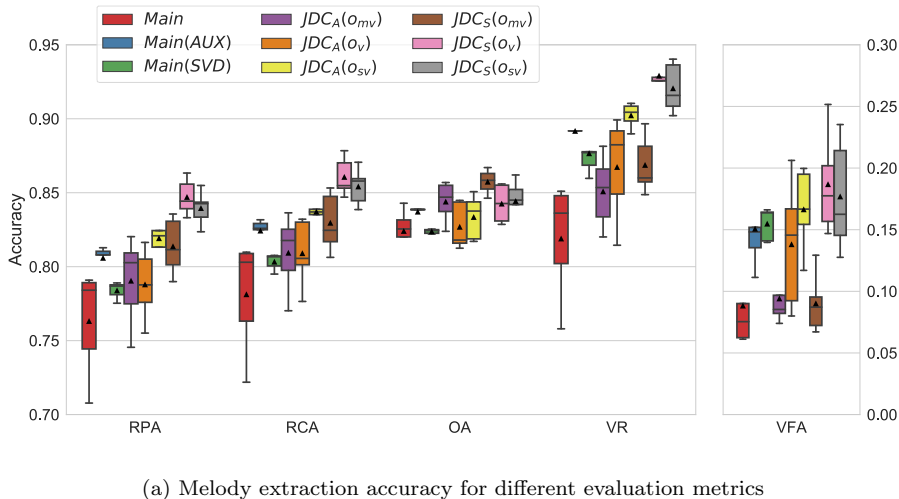


Figure 5.3: Performance of melody extraction on the *Main*, *Main(AUX)*, *Main(SVD)*, and JDC networks. The reported results were averaged across five training runs with different initializations. The “average” was calculated as the average score of all songs in all datasets. The band inside the box is the median, and the black triangle indicates the mean. RPA, raw pitch accuracy; RCA, raw chroma accuracy; VR, voicing detection rate; VFA, voicing false alarm rate.

regard to OA, a *t*-test revealed a statistical significance between *Main* and  $JDC_S(o_{mv})$ . The results are as follows: ADC04 (0.025), MIREX05 (0.01), MedleyDB (0.027), and RWC (0.043).  $JDC_S(o_{mv})$  increased the average OA with respect to *Main* for ADC04, which is an especially challenging dataset. The average overall accuracy of  $JDC_S(o_{mv})$  is 83.7%, which was 6.1% higher than that of 77.6% of *Main*.

To summarize, in the training phase, the most effective models were  $JDC_S$  networks that used both the main and auxiliary outputs for voice detection in the loss function. In the inference stage, the most effective output was  $o_{mv}$ , which used only the output of the main network. As a result, the best performance was obtained by  $JDC_S(o_{mv})$ . The overall performances of *Main(AUX)* and *Main(SVD)* were lower than the JDC networks. The JDC network had only 3.8 M parameters, while *Main(AUX)* and *Main(SVD)* had 7.6 M and 5.3 M parameters, respectively. It also shows that the JDC network is an efficient architecture for melody extraction.



## 5.4.2 Comparison of Voice Detection Performance

Figure 5.4 shows the average performances of singing voice detection for the *Main*, *Main(AUX)*, *JDC<sub>A</sub>*, and *JDC<sub>S</sub>* networks evaluated on the four test sets. *JDC<sub>S</sub>(O<sub>mv</sub>)* achieved the best voice detection performance, leading to improved melody extraction performance. The F1 score of *Main* was 91.0%, and that of *JDC<sub>S</sub>(O<sub>mv</sub>)* was 93.3% ( $p$ -value  $\leq 0.05$ ). F1 scores of other JDC networks were higher than *Main*, but there were no significant differences. For *Main(AUX)*, *Main(SVD)*, voice detection performance was significantly lower (the F1 scores were 87.5% and 88.9%, respectively). This seems to be due to the fact that the used training set had a higher percentage of voice segments than non-voice segments. If enough data can be used for model training, there is a possibility that the performance of SVD may be further improved.

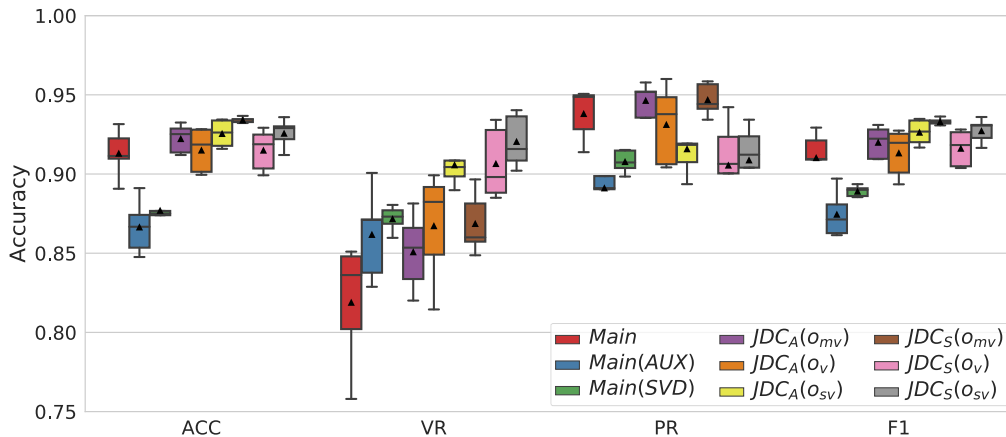


Figure 5.4: Performance of singing voice detection for different evaluation metrics. “ACC” indicates the overall accuracy of voice detection. “PR” and “F1” indicate precision and F1 score, respectively.

Figure 5.5 displays the performances of the proposed networks evaluated on the Jamendo dataset, which is dedicated to singing voice detection and unseen in training the models. As observed in the melody extraction results, the voice detection output of the main network was more conservative. This led to a low VR and VFA. On the other hand, the *JDC* networks that had the separate singing voice detector became more responsive, having higher VR and VFA. When comparing the two families of JDC networks, *JDC<sub>S</sub>* was more conservative than *JDC<sub>A</sub>* as the voice loss function contained the voice output from the main network. A similar result was found among the voice detection outputs. That is, JDC with  $o_{mv}$  had lower VR and VFA than JDC with  $o_v$  or  $o_{sv}$ . While the JDC networks returned comparable results, the best performance in terms of accuracy was obtained by *JDC<sub>S</sub>(o<sub>sv</sub>)*. The average of VR of *JDC<sub>S</sub>(o<sub>sv</sub>)* was 18.3% higher than that of *Main*, maintaining a low VFA of 22.6%.

In Table 5.2, we compare the voice detection result with other state-of-the-art algorithms. Lee et al. [72] reproduced each algorithm using the Jamendo dataset as the training data under the same conditions, and we used the results for comparison. The performance of *JDC<sub>S</sub>(o<sub>sv</sub>)* was lower; however, considering that the compared models were in fact trained with the same Jamendo dataset (by using different splits for training and testing), the result from our proposed model was highly encouraging, showing that it generalized to some extent.

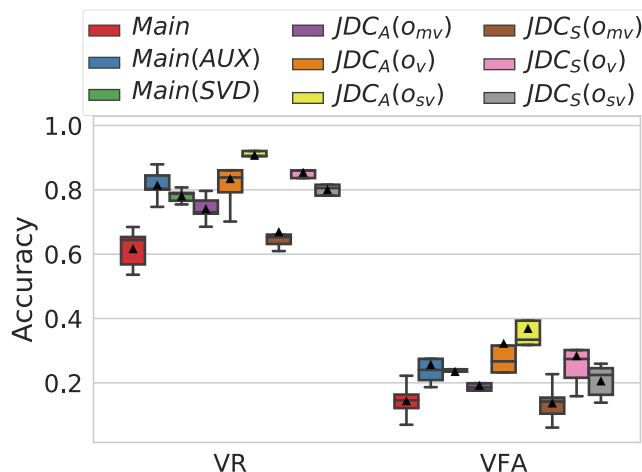


Figure 5.5: Result of singing voice detection on Jamendo.

Table 5.2: Comparison of results of existing algorithms,  $Main(SVD)$ , and  $JDC_S(o_{sv})$  (average accuracy of 5 runs). Note that Jamendo is not included in the training set of  $Main(SVD)$  and  $JDC_S(o_{sv})$ .

	Accuracy	VR	Precision	F1 Score
Lehner [51]	87.9	91.7	83.8	87.6
Schlüter [38]	86.8	89.1	83.7	86.3
Leglaives [36]	87.5	87.2	86.1	86.6
$Main(SVD)$	77.4	79.7	76.2	78.3
$JDC_S(o_{sv})$	80.0	80.2	79.1	79.2

### 5.4.3 Comparison with State-of-the-Art Methods for Melody Extraction

We compared our best melody extraction model,  $JDC_S(o_{mv})$ , with state-of-the-art methods using deep neural networks [20, 22, 28, 62]. For a comparison of results under the same conditions, the test sets were ADC04, MIREX05, and MedleyDB for comparing other methods as mentioned in Section 5.3.1. Table 5.3 lists the melody extraction performance metrics on three test datasets. The best score in each column is highlighted in bold. The pre-trained model and code of Bittner et al. [20] are publicly available online, and the results in Table 5.3 were reproduced by [22] for vocal melody extraction. The results show that the proposed method had high VR and low VFA, leading to high RPA and RCA, and it outperformed the state-of-the-art methods. In addition, we confirmed that the proposed method had stable performance over all datasets compared to other state-of-the-art methods. It also showed that combining two tasks of melody extraction, i.e., pitch classification and singing voice detection, through the proposed JDC network and loss function was helpful for performance improvement.

### 5.4.4 Case Study of Melody Extraction on MedleyDB

We evaluated the models with a tolerance of one semi-tone, following the standard melody extraction evaluation rule. However, we should note that our proposed model can predict the pitch with a higher resolution (1/16 semi-tone). Figure 5.6a shows the spectrogram of an audio clip (top) and the corresponding melodic pitch prediction along with the ground truth (bottom). Our proposed model can

Table 5.3: Comparison of vocal melody extraction results.

(a) ADC04 (Vocal)					
Method	VR	VFA	RPA	RCA	OA
Bittner et al. [20]	<b>92.9</b>	50.5	77.1	78.8	70.8
Su [62]	90.1	41.3	74.7	75.7	72.4
Lu and Su [22]	73.8	<b>3.0</b>	71.7	74.8	74.9
Hsieh et al. [28]	91.1	19.2	84.7	86.2	83.7
Proposed	88.9	11.4	<b>85.0</b>	<b>87.1</b>	<b>85.6</b>
(b) MIREX05 (Vocal)					
Method	VR	VFA	RPA	RCA	OA
Bittner et al.	<b>93.6</b>	42.8	76.3	77.3	69.6
Su	95.1	41.1	83.1	83.5	74.4
Lu & Su.	87.3	7.9	82.2	82.9	85.8
Hsieh et al.	84.9	13.3	75.4	76.6	79.5
Proposed	90.9	<b>2.4</b>	<b>87.0</b>	<b>87.5</b>	<b>90.7</b>
(c) MedleyDB (Vocal)					
Method	VR	VFA	RPA	RCA	OA
Bittner et al.	<b>88.4</b>	48.7	72.0	74.8	66.2
Su	78.4	55.1	59.7	63.8	55.2
Lu & Su	77.9	22.4	68.3	70.0	70.0
Hsieh et al.	73.7	<b>13.3</b>	65.5	68.9	79.7
Proposed	80.4	15.6	<b>74.8</b>	<b>78.2</b>	<b>80.5</b>
(d) RWC					
Method	VR	VFA	RPA	RCA	OA
Proposed	92.4	5.4	85.4	86.2	90.0

track nearly continuous pitch curves, preserving natural singing styles such as pitch transition patterns or vibrato.

While the proposed model achieved improved performance in singing melody extraction, the overall accuracy was still below 90%. We found that errors occurred more frequently in particular cases. Figure 5.6b,c gives the examples of bad cases where VR and RPA were less than 60%. In both examples, the failures were mainly attributed to voice detection errors.

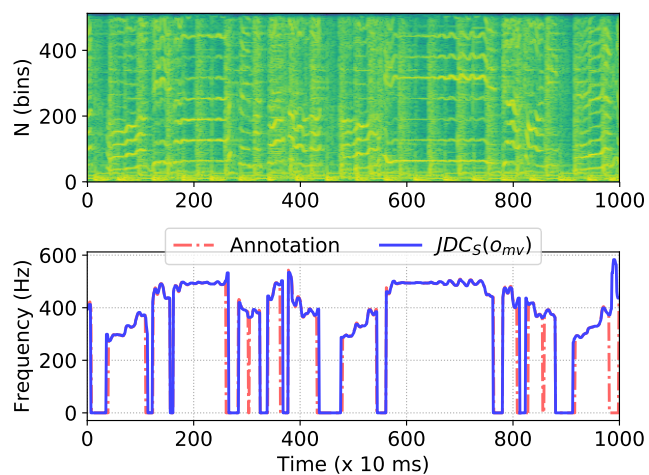
In Figure 5.6b, the harmonic patterns of the vocal melody were not clearly distinguished from background music because the vocal track was relatively softer than the accompanying music track. This weak vocal volume was investigated as a cause of bad singing detection in [72]. Since our melody extraction model was trained in a data-driven way, this could be addressed to some degree by augmenting the training data, for example adjusting the mixing of vocal gains (if they are in separate tracks).

In Figure 5.6c, a strong reverberation effect was imposed on the singing voice; thus, the harmonic patterns of the singing voice appeared even after the voice became silent. The algorithm then detected

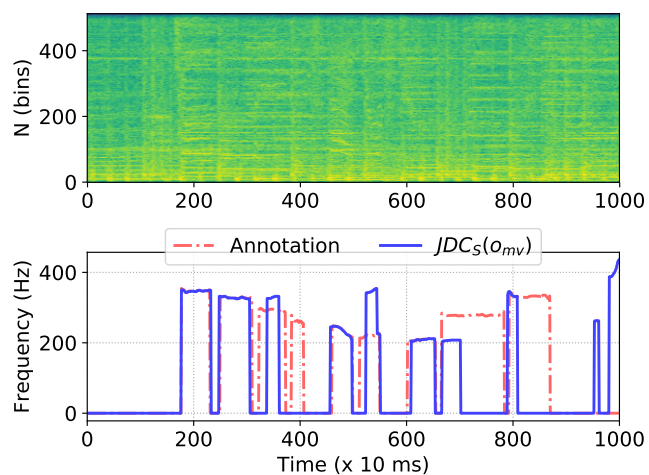
the reverberated tone as vocals and predicted the pitch from it. This case is somewhat controversial because this could be seen as a problem of the ground truth notation. When we excluded these types of heavily-processed audio clips in MedleyDB (“PortStWillow-StayEven” and “MatthewEntwistle-Lontano”), we observed a significant increase in performance (about 5% in OA on MedleyDB).

## 5.5 Conclusions

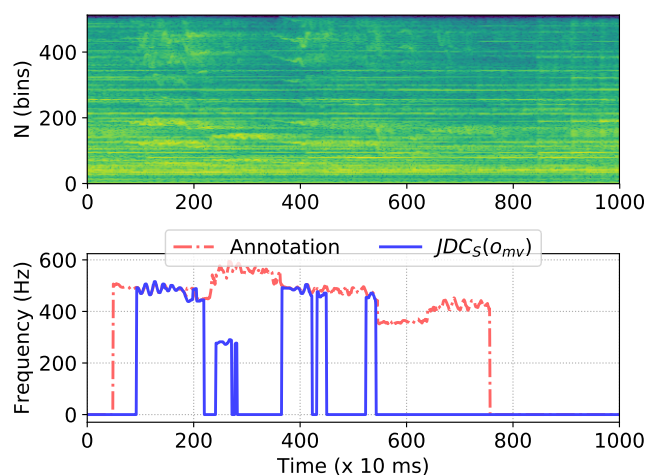
We presented a joint detection and classification (JDC) network that performs singing voice detection and pitch estimation simultaneously. The main network uses a CRNN architecture that consists of convolutional layers with residual connections and Bi-LSTM layers. The main network is trained to classify the input spectrogram into pitch labels quantized with a high resolution or a special non-voice label. The auxiliary network is trained to detect the singing voice using only multi-level features shared with the main network. We also examined the joint melody loss function that optimizes the JDC network to combine more tightly the tasks and the three different voice detection outputs from the two networks. Through the experiment, we provided a better understanding of how the main network and auxiliary network work for voice detection. We also showed that the knowledge sharing between two networks helps perform the melody extraction task more effectively. We showed that the proposed JDC network has consistently high performance for all test datasets and outperforms previous state-of-the-art methods based on deep neural networks. Finally, we illustrated failure cases, which may provide ideas for future work to improve the melody extraction performance further.



(a) Good example



(b) Bad Example #1



(c) Bad Example #2

Figure 5.6: Case examples of singing melody extraction from the MedleyDB dataset: (a) good example (“MusicDelta-Gospel”) and (b,c) bad examples (“CelestialShore-DieForUs”, and “MatthewEntwistle-Lontano”).

# Chapter 6. Semi-Supervised Learning Using Teacher-Student Models for Vocal Melody Extraction

## 6.1 Introduction

One of the key elements in the success of deep learning is a large amount of labeled data. However, when the labeled data is scarce in a given task, it can be a bottleneck in leveraging the power of deep neural networks. The issue has been found in many music information retrieval (MIR) tasks as well. Among others, melody extraction research has suffered from it as pitch labeling requires experienced annotators to handle the annotation tool and the process is extremely labor-intensive [73].

The lack of labeled data in melody extraction research has been tackled in several different ways. A popular method to alleviate the issue is data augmentation which increases labeled data by transforming the input audio, for example, using pitch-shifting [16, 22, 27]. Data augmentation, however, has the limitation in covering the diversity in the input space. Another approach is using multi-track audio data [34, 35, 74]. This allows to use monophonic pitch tracking algorithms for the melodic source and therefore it expedites laborious the pitch labeling. However, multi-track recording datasets often maintain individual tracks as stem files where multiple similar sound sources can be mixed (e.g., main vocal and backing vocal). Therefore, obtaining clean pitch labels from multi-track audio can be not straightforward [20, 75]. Recently, melody MIDI files, which are more easily accessible, have been utilized to guide melody extraction from audio with transfer learning techniques from the symbolic to audio domain [22, 24]. MIDI data exhibit greater flexibility than audio on data augmentation, but still face limitations on representing natural pitch contours of singing voice, which usually contain subtle variations such as vibrato and portamento.

Semi-supervised learning (SSL) is another but more general strategy to address the lack of labeled data. SSL uses a large amount of unlabeled data, which is usually easy to collect, jointly with labeled data. A popular class of SSL methods is based on self-training in the teacher-student framework. Recent works have combined random data augmentation with the SSL methods to encourage the model to produce robust output even when input is perturbed. This approach has achieved state-of-the-art performance on image classification [76–78], speech recognition [79], and audio classification [80]. There are a few MIR researches that used the teacher-student framework to address the lack of labeled data, for example, in automatic drum transcription [81] and singing voice detection [82, 83]. However, to the best of our knowledge, recent advances in SSL methods that leverage the power of deep neural networks and random data augmentation in the teacher-student framework have been not studied yet in the music domain.

In this paper, we apply the SSL methods to vocal melody extraction with the following contributions. First, we present the SSL methods for vocal melody extraction leveraging large-scale unlabeled music datasets. This prevents the model from overfitting to small labeled data and improve the performance. Second, we compare three setups of teacher-student models along with various audio data augmentation techniques. We show the model with the consistency regularization is most effective. Third, we investigate effective SSL strategies by exploring joint training, the size of unlabeled data, and the number of self-training iterations. Fourth, we show that the proposed teacher-student training method enables a baseline convolutional recurrent neural network model to achieve performance comparable to state-of-the-arts. Finally, apart from the SSL method, we propose large-scale testing data artificially generated

from unlabeled data using an analysis-synthesis framework, considering the lack of labeled data even at the testing stage. Evaluation on the diverse and sizable test set will reinforce the effectiveness of the proposed method. For reproducibility, the source code and pre-trained model used in this paper are available at [https://github.com/keums/melodyExtraction\\_SSL](https://github.com/keums/melodyExtraction_SSL).

## 6.2 Related work

The teacher-student framework has been previously studied in several MIR tasks to address the lack of labeled data. Wu and Lerch applied the approach to automatic drum transcription [81]. They used multiple teacher models based on non-negative matrix factorization (NMF) trained with different datasets and a student model based on deep neural network trained with labels from the teachers. They showed that the student model outperforms the teacher models. However, it was not a self-training setting where the teacher model is repeatedly replaced with an improved student model. Schlüter explored the self-training for singing voice detection [82]. They first trained a convolutional neural network (CNN) on the original labels with low-granularity, then a second network on pseudo-labels with high-granularity from the first network, and a third network on the summarized saliency maps from the second network. They showed this self-improvement worked up to the third network. However, they conducted the self-training on weakly-labeled data in the context of multiple-instance learning and did not use any unlabeled data. Recently, Meseguer-Brocal et al. used the teacher-student paradigm for singing voice detection to create a large-scale time-aligned vocal melody and lyrics dataset [83]. They consistently improved the teacher model by increasing the correlation between the prediction of the model and the time-aligned lyrics annotation.

## 6.3 Methods

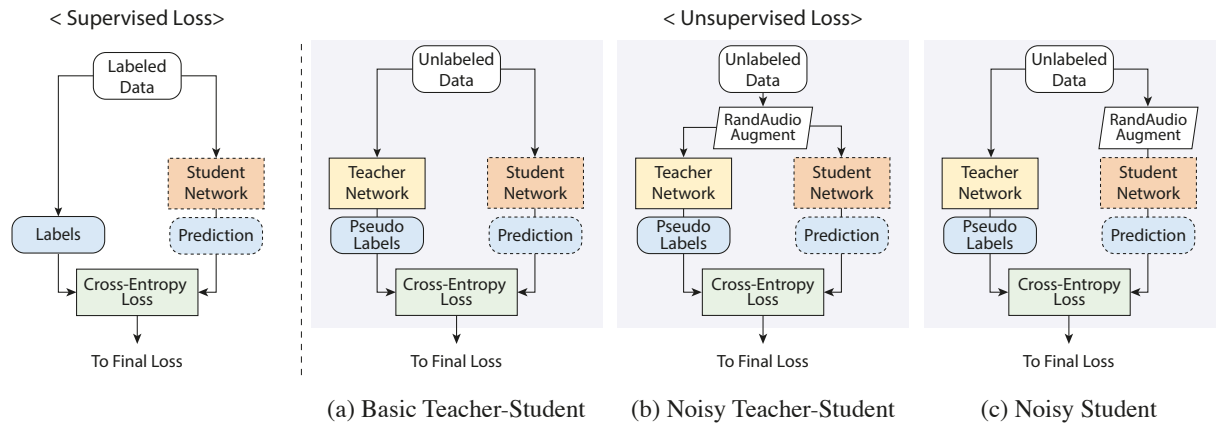


Figure 6.1: Diagram of the three Teacher-Student models.

### 6.3.1 Model Architecture

Recent melody extraction algorithms have used CNN [20, 26, 62] and its variants [23, 25, 27] as a standard architecture. Since we focus on the effectiveness of SSL in this paper, we employ a previously proposed convolutional recurrent neural network (CRNN) which was a baseline architecture in [27].

---

**Algorithm 1:** Train Teacher-Student Models in SSL

---

Train a teacher network  $T_1$  on labeled data  $\mathcal{D} = \{(x_d, y_d) : d \in (1, \dots, N)\}$ ;  
Generate augmented data  $\tilde{\mathcal{U}} = \{\tilde{x}_u = RAA(x_u) : u \in (1, \dots, M)\}$  from unlabeled data  
 $\mathcal{U} = \{x_u : u \in (1, \dots, M)\}$ ;  
**for**  $i = 1$  **to**  $k$  **do**  
    Use  $T_i$  to generate pseudo labels for  $\mathcal{U}$  (or  $\tilde{\mathcal{U}}$ );  
    Train student network  $S_i$  using both  $\mathcal{D}$  and  $\mathcal{U}$  (or  $\tilde{\mathcal{U}}$ ) as training data;  
     $T_{i+1} = S_i$ ;  
**end**

---

The CRNN architecture consists of 4 ResNet blocks and a bi-directional long short-term memory layer. We first merge the audio waveforms into a mono channel and downsample them to 8 kHz. We then calculate the logarithmic-magnitude spectrogram using short-time Fourier transform with a 1024-point Hann window and an 80-point hop size. The CRNN architecture takes 31 consecutive frames of the spectrogram as input and predicts a pitch label quantized with a resolution of 1/8 semitone and ranged from E2 (82.4 Hz) to B6 (1975.7 Hz). The size of the output layer is 442, including a non-vocal label.

### 6.3.2 SSL in the Teacher-Student Framework

Our SSL method is based on self-training in the teacher-student framework where the teacher model is first trained with labeled data and then the student model is trained with artificial labels generated from the teacher model using unlabeled data. The artificial labels can be the prediction distribution vector [76, 77] or one-hot vector determined by the class with a highest confidence [78, 84]. We formally describe the overall procedure in Algorithm 1. We first train the initial teacher model  $T_1$  using only labeled data  $\mathcal{D}$  where  $x_d$  are labeled examples and  $y_d$  are one-hot reference labels. For unlabeled data  $\mathcal{U}$  where  $x_u$  are unlabeled examples, we use random data augmentation to generate noisy input data  $\tilde{\mathcal{U}}$  where  $\tilde{x}_u$  are noisy unlabeled examples. RandAudioAugment (RAA) is an audio version of random data augmentation method which is described in Section 6.3.4. While it is more effective to use random data augmentation on the student model only in image classification [77], we also try applying it for both teacher and student models for ablation study. Once we train the student model jointly with the labeled data and unlabeled data (with pseudo labels), we replace the teacher model with the student model. We repeat the same pseudo labeling and the training with a new student model.

### 6.3.3 Proposed Teacher-Student Models

Our proposed TS models are illustrated in Figure 6.1. The supervised loss  $\mathcal{L}_D$  is computed with labeled data and defined as:

$$\mathcal{L}_D = \frac{1}{N} \sum_{d=1}^N H(y_d, p(y|x_d; \theta_s)) \quad (6.1)$$

where  $H(\cdot)$  denotes the cross-entropy between the pitch label  $y_d$  and pitch prediction  $p(y|x)$ , and  $\theta_s$  denotes a set of parameters of the student model. The supervised loss is a common loss term of the three investigated TS models. Each of them are explained below.

- **Basic Teacher-Student** is a fundamental teacher-student framework that uses the unlabeled data  $\mathcal{U}$  but trains the student network with the pseudo labels generated from the teacher network. The



final loss of Basic Teacher-Student  $\mathcal{L}_B$  is defined as

$$\mathcal{L}_B = \mathcal{L}_D + \frac{1}{M} \sum_{u=1}^M H(y_u, p(y|x_u; \theta_s)) \quad (6.2)$$

where  $y_u$  is the pseudo labels on  $\mathcal{U}$  generated by the teacher network, i.e.  $y_u = p(y|x_u; \theta_t)$  where  $\theta_t$  to denote the parameters of teacher network. The basic teacher-student model is illustrated in Figure 6.1(a).

- **Noisy Teacher-Student** takes noisy unlabeled data  $\tilde{\mathcal{U}}$  for both of the teacher and student networks using RAA and the rest is the same as the basic teacher-student model. The final loss of Noisy Teacher-Student  $\mathcal{L}_N$  is defined as

$$\mathcal{L}_N = \mathcal{L}_D + \frac{1}{M} \sum_{u=1}^M H(\tilde{y}_u, p(y|\tilde{x}_u; \theta_s)) \quad (6.3)$$

where  $\tilde{y}_u$  is a prediction on  $\tilde{\mathcal{U}}$  generated by the teacher network, i.e.  $\tilde{y}_u = p(y|\tilde{x}_u; \theta_t)$ . The noisy teacher-student model is illustrated in Figure 6.1(b).

- **Noisy Student** takes noisy unlabeled data  $\tilde{\mathcal{U}}$  only for the student network while the teacher network takes unnoised input  $\mathcal{U}$  to generate the pseudo labels. The idea is that the student should produce consistent outputs that minimize the difference from the teacher even though the input is perturbed [77]. This notion is also similar to consistency regularization [85, 86]. The final loss of Noisy Student  $\mathcal{L}_C$  is defined as

$$\mathcal{L}_C = \mathcal{L}_D + \frac{1}{M} \sum_{u=1}^M H(y_u, p(y|\tilde{x}_u; \theta_s)) \quad (6.4)$$

The noisy student model is illustrated in Figure 6.1(c).

### 6.3.4 Data Augmentation

We conducted pitch-shift by  $\pm 1,2$  semitone on the labeled data  $\mathcal{D}$  (audio and corresponding labels). In the melody extraction task, it has shown that pitch-shifting can improve the generality and performance of the model by increasing the amount of audio and label pairs for different  $f_0$  [16, 87]. For data augmentation of unlabeled data  $\mathcal{U}$ , we propose RandAudioAugment (RAA) inspired by RandAugment [88], which is a method of randomly applying different kinds of transformations to increase image data. RAA converts audio by randomly selecting multiple audio effects as follows: audio equalizer (low-shelf, high-shelf), filters (low-pass, high-pass), overdrive, phaser, and reverb. Here, we use *pysndfx* that is a Python library designed for applying effects to audio files <sup>1</sup>. We sampled a random magnitude of each transformation from a pre-defined range. The implementation details for RAA are also described in the source code.

### 6.3.5 Data Selection

The SSL algorithm using large-scale unlabeled data may suffer from labeling noise. Unlabeled data are highly likely to have audio without vocals. Filtering only high-confidence examples or the top-K examples in image classification has demonstrated to be an effective method to handle the labeling

<sup>1</sup><https://github.com/carlthome/python-audio-effects>

noise [77, 89]. Likewise, we performed data selection so that only the tracks with vocal ratios exceeding a threshold were used for training. To estimate the ratio of vocals included in the track, we used our singing voice detector<sup>2</sup> based on CNN based on [38]. Considering the distribution of vocal ratio in the FMA, we set the threshold to 0.3.

## 6.4 Datasets

	Dataset	Number of Tracks	Total Length
Training (Labeled)	RWC	100	6h 47m
	MedleyDB	61	2h 39m
	iKala	262	2h 6m
Training (Unlabeled)	In-house	535	6h 21m
	FMA_small	3,521 / 8,000	25 / 60h
	FMA_medium	10,639 / 25,000	89h / 208h
	FMA_large	40,505 / 106,574	337h / 888h
Test	ADC04	12	4m
	MIREX05	9	4m
	MedleyDB	12	43m
	AST218	218	14h 53m

Table 6.1: Description of datasets. In FMA, The former and the latter are the total of the tracks with the vocal ratio above 0.3 and the total of the entire tracks respectively.

Table 6.1 shows the simple statistics of the labeled and unlabeled training datasets and test datasets.

### 6.4.1 Labeled Data

We used the three labeled datasets (RWC [44], MedleyDB [34], and iKala [35]) and split them into a train and validation set following [20]. We augmented the training data by pitch-shifting with  $\pm 1, 2$  semitone. The total length of the labeled training data amounts to about 55 hours after the data augmentation.

### 6.4.2 Unlabeled Data

As to unlabeled data, we used an in-house dataset crawled from YouTube and the Free Music Archive (FMA) [90]. The in-house dataset is pop songs with vocals recorded in a variety of environments. It includes both public-released and user-uploaded tracks. FMA is a large-scale open dataset containing up to 106,574 tracks and covers 161 genres of music. We used FMA for performance comparison on data scalability. The FMA has three different subsets depending on the number of the track and genre included: FMA\_small (FMA<sub>S</sub>), FMA\_medium (FMA<sub>M</sub>), and FMA\_large (FMA<sub>L</sub>). We selected vocal tracks from them as described in Section 6.3.5 and denote the selected versions as FMA<sub>Sv</sub>, FMA<sub>Mv</sub>, and FMA<sub>Lv</sub>, respectively. We augmented the unlabeled datasets via RAA during training as described in Section 6.3.4.

<sup>2</sup><https://github.com/keums/SingingVoiceDetection>

### 6.4.3 Test Data

#### Public Test Sets

We used three public test sets (ADC04, MIREX05<sup>3</sup>, and MedleyDB) to evaluate the performance of vocal melody extraction. In this study, we excluded non-vocal tracks from ADC04 and MIREX05, and used songs not included in training data for MedleyDB. To obtain the ground truth for singing voice in MedleyDB, we adopted its 'MELODY2' annotations. These three datasets have been commonly used to compare the performance of melody extraction. However, the number of tracks and the total length are very limited as shown in Table 6.1.

#### Proposed Large-Scale Test Set

To make up the scarcity of testing data for evaluating singing voice extraction algorithms, we propose a new test set composed of DSD100 [91] and MusDB18 [92]. The two multitrack datasets were originally designed for source separation. Each track has four isolated stems: vocals, drums, bass, and others. Following the analysis/synthesis framework [75], the singing melodies for 218 selected tracks<sup>4</sup> were synthesized with automatically generated  $f_0$  contours. In detail, for each song, we extracted the melody of the vocals with five different pitch trackers, and each  $f_0$  information along with the vocal audio was fed into the WORLD [93] (D4C edition [94]) vocoder to reproduce five monophonic variations of the vocal stem. The original vocal audio was parameterized into harmonic and aperiodic spectral envelopes, and then resynthesized with provided pitch contours. Then a mask was applied to filter intervals without  $f_0$  information. For remixing, the amplitude of the synthesized vocal was weighted to that of the original vocal stem, and the rest stems were directly summed up as accompaniments, then mixed with the weighted synthesized vocal that perfectly matched the  $f_0$  annotation. These 1,090 polyphonic mixtures with accurate and automatic annotations constitute the proposed analysis/synthesis test set, AST218<sup>5</sup>.

Each track in AST218 has five variations whose vocal melody was annotated separately with five different pitch estimators: CREPE [58] (with confidence threshold of 0.5 and 0.7), pYIN [95], and Lu&Su [22] (with time step of 10 and 20ms), as they have different merits. Since there is no exact way to pinpoint a common optimal confidence threshold across the entire dataset, we chose two different threshold values for CREPE: one is 0.5, suffering from high false positive (FP) but preserving details; the other threshold is 0.7, acceptable FP though sacrificing some recall. pYIN was chosen for it has even lower FP while producing stable and continuous melodic lines when the vocal stem is monophonic. However, it is not stable in the polyphonic scenario, which is universal in DSD100 and MusDB18. In need of other polyphonic-based melody estimators to balance the  $f_0$  quality, we chose two time step setups of the Lu&Su model: 20ms, at which this model is optimized; and 10ms, which provides more continuous predictions and offers alternative pitch contours when encountering multiple melodic vocal lines.

The analysis/synthesis framework has been practiced successfully in evaluating monotonic pitch trackers [58]. As a sanity check, we evaluated several patchCNN [62] setups on the original and resynthesized ADC04, MIREX05, and MedleyDB. The differences of OA are within  $\pm 2\text{--}5\%$ , which is acceptable, meaning this framework is also applicable for polyphonic test set generation.

<sup>3</sup><http://labrosa.ee.columbia.edu/projects/melody/>

<sup>4</sup>Songs that appear in MedleyDB were excluded for they were part of the training data, but songs in MusDB18 having counterparts in DSD100 were not removed for they are not exactly identical. Additionally, 12 songs that do not have discernible vocal melodies were also excluded.

<sup>5</sup><https://sites.google.com/view/mctl/resource>

When evaluating vocal extraction algorithms on AST218, we averaged the scores from the five variations. Our pilot study shows that these five pitch contours reach consensus over a majority of frames, while the estimations differ for tricky frames. Rather than manually check on the estimated  $f_0$ , we used AST218 in an ensemble manner, fully leveraging the spirit of automatic pitch annotation.

## 6.5 Experiments

### 6.5.1 Experimental Setup

#### Training Details

We used the CRNN architecture with residual connections and bi-directional long short-term memory in all experiments. The implementation of the model was consistent with that of the main network of [27]. We trained our models using Adam optimizer for 70 epochs on 2 GPUs. The initial learning rate was set to 0.003 in all the experiments. We used a learning rate schedule that reduces the learning rate by 0.7 times if validation accuracy did not increase within three epochs. The model and the training procedures were implemented using Keras<sup>6</sup> [49].

#### Evaluation

To evaluate the performance of melody extraction, we mainly used overall accuracy (OA) which combines the accuracy of pitch estimation with voice detection. We also used three metrics raw pitch accuracy (RPA) for pitch estimation, and voicing recall (VR) and voicing false alarm (VFA) for voice detection [6]. These metric are computed by *mir\_eval* [47] library designed. This metric is computed by the *mir\_eval* [47] library designed for evaluation in MIR tasks with pitch tolerance of 50 cents. We converted the pitch label quantized in 12.5 cents (1/8 semitones) to frequency scales (Hz) and compared them to the ground truth.

### 6.5.2 Experiment 1: Teacher-Student Models

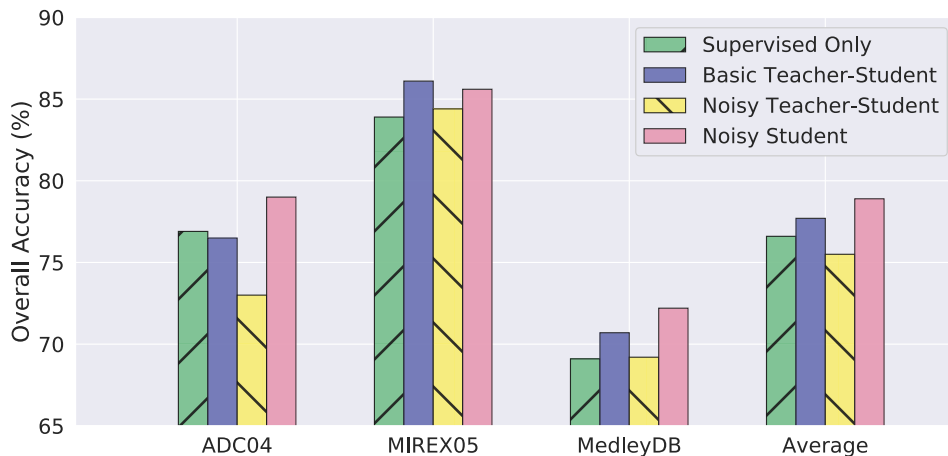


Figure 6.2: Comparison with supervised-learning model and three student models on three test sets.

<sup>6</sup>We used Keras 2.3.0, Accessed: 15 May 2020

Our first experiment is to demonstrate the efficacy of the proposed Teacher-Student models for SSL. In this experiment, we trained three Teacher-Student models described in Section 6.3.3 using an in-house dataset as unlabeled data. We evaluated the performance of each model on ADC04, MIERX05, and MedleyDB, which have been used as standard test sets for evaluation. As shown in Figure 6.2, the basic teacher-student model can achieve 1.1% higher average OA than the supervised-only model which has 77.7% average OA. This confirms the possibility of using unlabeled data to improve the performance of melody extraction. Our experiment also shows that the noisy student model outperforms all the others, having 78.9% average OA.

The noisy student model increases OA by 3.1% with respect to the supervised-only model in MedleyDB, which is especially a challenging dataset because it contains tracks that are difficult to distinguish between vocals and background music, or tracks with excessive audio effects. The results indicate that the student network can be trained reliably using the noisy student model, even if the initial teacher network is not robust to diverse noise. Meanwhile, the performance of the noisy teacher-student has deteriorated, being worse than the supervised-only model. This degradation is probably because the noised teacher model is not generating reliable pseudo labels.

### 6.5.3 Experiment 2: Joint Training vs. Fine-Tuning



Figure 6.3: Comparison with pre-training, fine-tuning, and joint training methods on three test sets.

The training methods of the teacher-student framework can be divided into three approaches depending on how  $\mathcal{D}$  and  $\mathcal{U}$  are used for training: pre-training on only  $\mathcal{U}$  and then fine-tuning on  $\mathcal{D}$ ; joint-training on both  $\mathcal{U}$  and  $\mathcal{D}$  simultaneously. Figure 6.3 compares the results among pre-training, fine-tuning, and joint training for the noisy student model. The jointly trained model achieves 0.8% higher average OA than the fine-tuned model, with the highest results on MedleyDB. This indicates that joint training on unlabeled data and labeled data would help the networks produce a decision boundary that better reflects real music [96]. Interestingly, the average OA of the pre-trained model only on unlabeled data is higher than that of the supervised learning model. This suggests that the distribution of unlabeled data is similar to that of labeled data. Considering that the in-house dataset consists of pop songs with vocals, the in-house dataset can be seen as having a similar tendency to the labeled data. It provides insight into the data selection in the next experiment.

### 6.5.4 Experiment 3: Size of Training Data

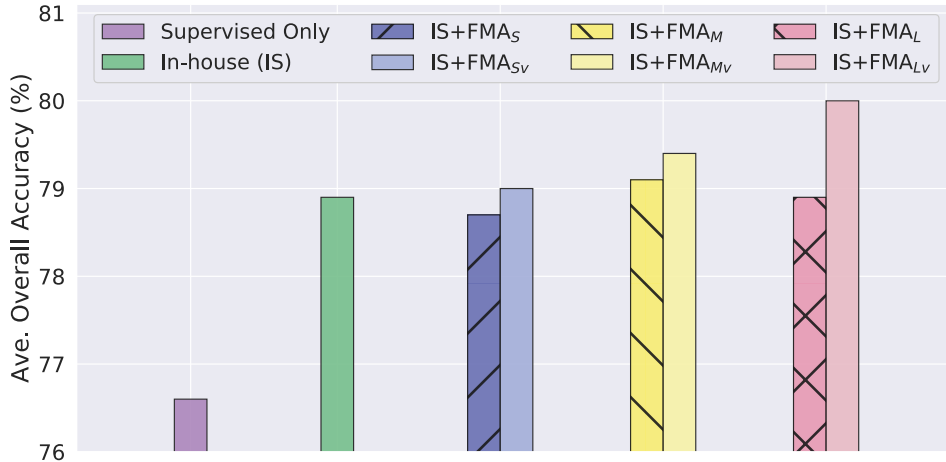


Figure 6.4: Comparison with Noisy Students on varied sizes of unlabeled datasets. The subscript ‘*v*’ denotes a selected subset of FMA whose vocal ratio exceeds a threshold. We use the average of OA for ADC04, MIREX05, and MedleyDB to compare performances.

We investigated the importance of the size and validity of unlabeled data. To explore the effect of the size of unlabeled data, we started with the in-house dataset as training data for the noisy student model and progressively included larger subsets of FMA. The results can be seen in Figure 6.4. Although the FMA data set contains more numerous tracks than the in-house dataset, the average OA of  $FMA_S$  and  $FMA_L$  is lower than that of the model trained only with the in-house dataset. Note that the proposed model focuses only on vocal melodies. As a result, teacher models may suffer from labeling noise generated by numerous instrument tracks included in the FMA. In addition, all labels on the instrumental track are classified as non-vocal pitch, resulting in data imbalance.

To confirm the validity of the dataset, we performed data selection for each FMA subset as mentioned in Section 6.3.5 and used them to train each student model. Interestingly, as the size of the  $\mathcal{U}$  increases, the performance of each model tends to be significantly improved. For example,  $FMA_{Lv}$  achieves an average OA of 80.2%, which is 3.6% higher than the supervised-only model. This indicates that effective SSL requires a large amount of  $\mathcal{U}$  with a similar distribution for  $\mathcal{D}$ .

### 6.5.5 Experiment 4: Iterative Training

We iterated the self-training 4 times for the noisy student model using the in-house dataset and  $FMA_{Lv}$ . The results are illustrated in Figure 6.5. We observe that the performance continuously increases up to 2 iterations achieving the highest average OA of 81.1%. Generally, self-training tends to amplify the error caused by labelling noise during training. However, the noisy student model trained on large-scale unlabeled data can help overcome this difficulty. Nevertheless, increasing the number of training iterations three or more times does not improve performance, and rather slightly lower the accuracy.

### 6.5.6 Comparison with State-of-the-Arts

We compared the supervised-only model (as a baseline) and proposed the noisy student model (NS) with four recent melody extraction algorithms based on deep neural networks: the patch-based

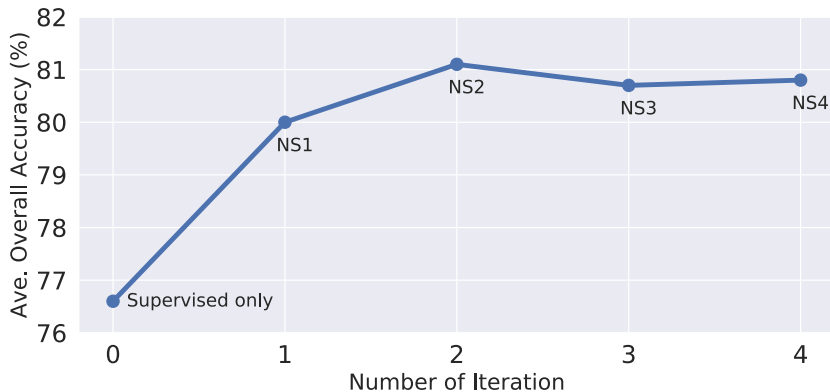


Figure 6.5: Effect of iteration training for Noisy Students.

Methods	ADC04	MIREX05	MedleyDB	AST218
PatchCNN [62]	76.9 / 72.9	69.7 / 73.8	44.0 / 59.3	42.3 / 59.7
DSM [20]	89.2 / 72.2	87.7 / 80.1	80.6 / 75.4	38.9 / 68.3
SegNet [23]	88.7 / 83.3	82.6 / 80.0	70.6 / 75.5	41.5 / 68.1
JDC [27]	<b>90.6 / 83.5</b>	<b>91.4 / 87.4</b>	72.7 / 78.1	55.8 / <b>75.4</b>
Baseline	78.7 / 76.8	79.9 / 81.5	57.2 / 70.7	<b>56.3</b> / 69.7
Proposed (NS)	90.4 / 82.2	90.4 / 85.9	<b>76.3 / 79.2</b>	54.2 / 74.2

Table 6.2: Vocal melody extraction results in terms of (RPA / OA) of the proposed and other methods on various test sets. The proposed model is iterated the self-training two times using the in-house dataset and  $FMA_{Lv}$ .

CNN (patchCNN) [62], the deep salience map (DSM) [20], the streamlined encoder/decoder network (segNet) [23], and the joint detection and classification model (JDC) [27], which have open-sourced codes with vocal mode. Each method was run with its default parameters, and then evaluated on the three conventional test sets and the newly introduced AST218. Besides, we report the frame-level scores instead of song-level ones to settle uneven song lengths.

Table 6.2 and Table 6.3 list the evaluation results of each method on the four test sets. In general, performances of the proposed NS model are comparable to other supervised-learning-based methods and even outperforms others in MedleyDB, and it effectively improves the OA of the baseline by 4.5–8.5%. The overall rankings of VR and VFA vary across the test sets, but the behavior converges in terms of OA. One can also observe that the AST218 is the most challenging in the majority of cases. In such a challenging dataset, the performances of the NS model shows that the proposed method is robust to large-scale evaluation. However, the NS model improves the baseline except for VR and RPA in AST218. This result might be because the simple rule-based remixing of vocal and accompaniment tracks in AST218 is different from the artistic practice of mixing engineers, which can affect voicing detection and, in turn, RPA.

Methods	ADC04	MIREX05	MedleyDB	AST218
PatchCNN	91.8 / 46.1	80.3 / <b>11.6</b>	60.1 / 22.4	61.6 / 26.0
DSM	95.7 / 61.1	93.9 / 29.4	<b>85.4</b> / 26.6	44.6 / 7.7
SegNet	95.2 / 38.5	92.2 / 24.0	78.8 / 21.7	51.7 / 10.0
JDC	96.7 / 40.2	<b>97.5</b> / 18.5	80.5 / 18.3	64.7 / <b>8.6</b>
Baseline	92.6 / <b>33.8</b>	89.1 / 15.2	71.0 / <b>16.7</b>	<b>72.0</b> / 19.2
Proposed (NS)	<b>97.4</b> / 42.1	97.3 / 20.4	83.3 / 19.1	61.6 / 9.4

Table 6.3: Voicing detection results in terms of (VR / VFA) of the proposed and other methods on various test sets.

## 6.6 Conclusion

This study provides a framework of semi-supervised learning using the teacher-student model for vocal melody extraction. We compared three setups of teacher-student models and revealed that the noisy student model is the most effective and robust to real-world music where various noises can be present. We showed that large-scale unlabeled data is effective when they are properly selected. We found that iterative training for the teacher-student model helps improve performance. We also confirmed the effectiveness of the proposed method by evaluating it on artificial large-scale test data generated from automatically annotated multitrack data. Although these findings are based only on vocal melody extraction, we believe our method can be extended to other MIR tasks that suffer from the lack of labeled data such as automatic music transcription and chord recognition.



# Chapter 7. Semi-Supervised Learning Using Teacher-Student Models for Singing Voice Activity Detection

## 7.1 Introduction

Singing Voice Activity Detection (SVAD) is to identify a singing voice among multiple tracks and determine where a singing voice is included in a particular time frame. Identifying the singing voice is the most fundamental step in music analysis because it is a crucial role in delivering melody, lyrics, and emotions [8]. Modern music services require advanced solutions for searching for songs or evaluating musical characteristics such as singer identification, song voice separation, and melody transcription [97]. SVAD can be used as a basic pre-processing step for these applications.

Early works for SVAD are mainly based on vibrato or formant modulation patterns that distinguish it from other musical instruments [43,51,98]. In recent years, SVAD systems have a recent trend toward a data-driven approach using deep neural networks [36,38,82,99]. Although the proposed algorithms show high performance, they still struggle to solve repetitive and common errors caused by pitch-fluctuating instruments [72]. Identifying a singing voice requires a much wider context than estimating pitch from voice segments. That is, the voice/non-voice discrimination is a higher-level task than the pitch classification. It is more difficult to distinguish the singing voice in mixed audio because instruments play harmoniously related notes with the singing voice. Furthermore, complex vocal characteristics such as unstable vocalization and complex harmonics make it hard to identify a singing voice from mixture audio. Various model structures and methods have been proposed, but the reason the above problems still have not been solved may be due to the lack of labeled data for training.

The scarcity of data continues to be mentioned as one of the major hurdles to be solved in the field of MIR [34,73], however collecting and labeling music data requires a daunting task. One of the general methods for utilizing large-scale unlabeled data is semi-supervised learning (SSL) based on self-training in the teacher-student framework. Our previous study for this framework for vocal melody extraction [100] showed that that large-scale unlabeled data is effective when they are properly selected. With these findings, we believe that our method can be extended to the task of singing voice detection.

In this paper, based on the method as mention in Chapter 6, we apply the SSL methods to SVAD with the following contributions. First, we reveal that our proposed SSL method also can be effective for other MIR tasks that suffer from the lack of labeled data by applying it to SVAD. Second, we explore a hard negative sampling technique as a kind of data selection and show that it helps to improve overall performance while reducing false-positive errors. Finally, we show that our proposed model outperforms other state-of-the-arts.

## 7.2 Related Work

### 7.2.1 Singing Voice Activity Detection

In the early method for singing voice detection, a diverse set of various features are used such as MFCC, LPC, Perceptually derived LPC, zero-crossing rate, sharpness, harmonic coefficient, and pitch fluctuation. With these various features as inputs, classifiers based on SVM, HMM, and GMM were

utilized to detect singing voice [45, 101, 102]. Unfortunately, these input features have the disadvantage of being very heuristic and expensive. Lehner et al. [98] proposed a simple method with optimized MFCC according to the length of the observation window, and achieve a comparable result to state-of-the-arts. Lehner et al. [43] also proposed Fluctogram, a new feature that reflects a representation of characteristic pitch fluctuations to reduce frequently occurring false-positive errors.

Various deep learning strategies proposed for different tasks have also been applied for an effective model for SVAD. Schlüter and Grill [38] proposed a CNN architecture inspired by an image recognition task. The network is trained with 115 frame mel-spectrogram excerpts paired with labels indicating the presence of a singing voice in the center frame. It is designed to take context into account using a wider window. In particular, various data augmentation technologies were explored to find the most effective strategy. To address the sound level invariance, Schlüter and Lehner [103] also proposed a zero-mean convolution that parameterized the filter of the first convolutional layer to zero-average. It is effective to stabilize the prediction of changes in the input gain and improve training on different volumes of data. These methods achieved high performance but had the disadvantage that it was difficult to perform frame-level classification and temporal smoothing simultaneously. The recurrent network allows the model to learn the inherent sequential aspect of the singing voice. RNN architectures [36, 51] and CRNN architectures [99] are proposed to make it easier for the model to take on temporal characteristics.

### 7.2.2 Semi-Supervised Learning

It is an extremely laborious task to collect a large amount of data and labels. To alleviate the need for labels, semi-supervised learning (SSL) is used as one of the strategies for leveraging a large amount of unlabeled data. The recent works in SSL using the teacher-student model have achieved state-of-the-art performance on image classification [76–78, 86, 89]. In addition, data augmentation techniques (CTAugment [78, 104] and RandAugment [77, 86]) are combined with the teacher-student framework to improve the robustness of the student model by producing robust output even when input is added noise or transformed.

Recently, recent advances in SSL methods used in the field of image classification has been little studied in the MIR domain. However, it was recently applied to melody extraction and showed the potentiality of application in the music field. Kum et al. [100] present SSL methods for vocal melody extraction leveraging large-scale unlabeled music datasets with three teacher-student models. It shows the effectiveness of SSL strategies with consistency regularization and random audio augmentation. They also investigated effective SSL strategies by exploring joint training, the size of unlabeled data, and the number of self-training iterations. In this study, we use the teacher-student framework of Noisy Student proposed in [100] to train singing voice detection model using a large amount of unlabeled data with random audio augmentation.

### 7.2.3 Hard Negative Sampling

Imbalance data often causes the model to learn to ignore certain classes of errors. One intuitive solution is to consider the imbalanced distribution of the data. We can sample the data into a balanced distribution by oversampling the minority class data and downsampling the majority class data. The most common method is considering a balance between positive examples and negative examples. Easy negative samples may not contribute to better training even if the number increase. However, hard-negative samples could provide more information to discriminate each class. By controlling the ratio of

hard examples rather than a number of easy examples that can be easily identified, it can make a more robust classifier can be made [105, 106]. In particular, hard negative sampling (or hard negative mining) is mainly proposed as a method to reduce false-positive errors. Rather than randomly selecting negative examples, it is common to use negative examples with the highest confidence score in the training set together with positive examples randomly selected for learning [107, 107, 108].

## 7.3 Methods

### 7.3.1 Model Architecture

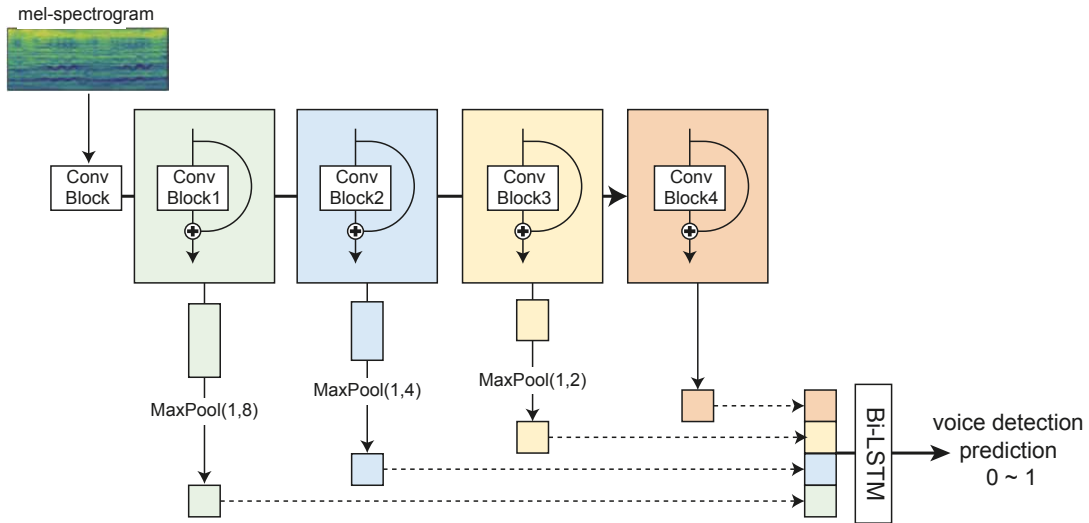


Figure 7.1: Overview of the proposed architecture for singing voice detection

The diagram of the architecture is shown in Figure 7.1. This structure was inspired by the auxiliary network for singing voice detection of the JDC network [27]. The architecture was configured with 4 ConvBlock with residual connection and a bi-directional long short-term memory (Bi-LSTM) layer. ConvBlock is organized in the order of two ‘ $3 \times 3$  convolutional layer - batch normalization layer - leaky rectified linear’, and followed by a  $1 \times 2$  max-pooling layer that preserved the input size in the time axis. The number of channels in the convBlocks gradually increases as 64, 128, 128, and up to 256. We used a concatenated multi-level features stemmed from each convBlock to observe diverse textures in different levels of abstraction. We used a Bi-LSTM layer to predict probabilities of the presence of the singing voice in a sequence-to-sequence manner from concatenated features via the softmax function. We use the binary cross-entropy between the prediction and the ground truth as a loss function.

The SVAD takes 75 frames of mel-spectrogram as input to capture contextual information over a long time. We resampled audio signals to 16kHz and merged stereo channels to mono. We extracted mel-spectrogram with 80 triangular filters between 0 and 8 kHz, a frame length of 1024, hop size of 160 samples. We compressed the magnitude by a log scale.

### 7.3.2 Teacher-Student Model

In this experiment, we used Noisy Student as teacher-student framework proposed by [100] for leveraging unlabeled large datasets. The diagram of the Noisy Student is shown in Figure 7.2. The noisy

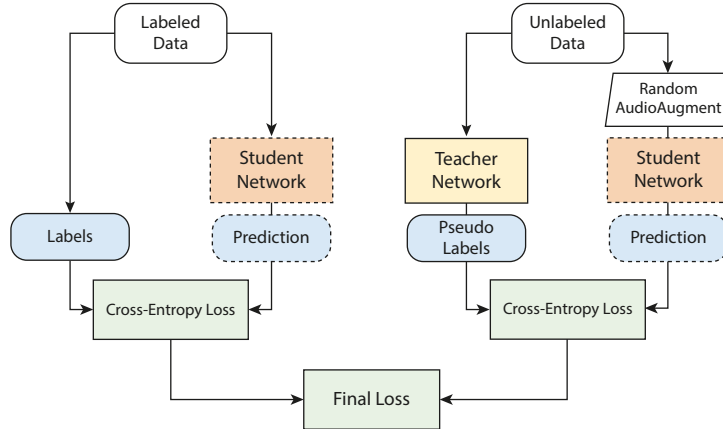


Figure 7.2: Diagram of the Noisy Student of Teacher-Student framework for singing voice detection

unlabeled data is used as input to the student network to be trained. On the other hand, the original data is used as input to a pre-trained teacher network to generate pseudo labels. The loss of Noisy Student  $\mathcal{L}_C$  is defined as

$$\mathcal{L}_C = \frac{1}{N} \sum_{d=1}^N H(y_d, p(y|x_d; \theta_s)) + \frac{1}{M} \sum_{u=1}^M H(y_u, p(y|\tilde{x}_u; \theta_s)) \quad (7.1)$$

where  $H(\cdot)$  denotes the cross-entropy.  $y_d$  and  $y_u$  are ground-truth on labeled data and pseudo labels on unlabeled data, respectively.  $\tilde{x}_u$  is sample of noisy unlabeled data.  $p(y|x; \theta_s)$  denotes pitch prediction of student model whose set of parameters is  $\theta_s$ .

### 7.3.3 Hard Negative Sampling

Some Instrument such as violin, saxophone, trumpet, and electric guitar have similar characteristics to singing voices, considering the pitch range, harmonic structure, and temporal dynamics (vibrato), and they often cause confusion in detecting singing voice. In this study, as described in the Section 7.4, we selected hard negative samples from an in-house dataset that played melodies only with musical instruments, and all tracks are annotated as ‘0’ (unvoiced). First, we trained a teacher network (HNS) using hard negative samples and unvoiced labels along with existing labeled data. Then we trained the student network (HNS+SSL) with all the unlabeled data using the Noisy Student setup as mentioned in Section 7.3.2.

## 7.4 Data sets

### 7.4.1 Training data

We used the four labeled datasets (Jamendo [45], RWC [44], MedleyDB [34], and iKala [35]). We divided them into training and validation set to tune the network parameters following [100]. We also augmented labeled data sets by applying the pitch-shifting by  $\pm 1, 2$  semitones. Pitch shifting has proven to be an effective method to increase data and improve results for singing voice activity detection [38].

As to unlabeled data, we used Free Music Archive (FMA) [90]. FMA is a large-scale open dataset containing up to 106,574 tracks and covers 161 genres of music. We used FMA for performance comparison on data scalability. The FMA has three different subsets depending on the number of the track and

	Dataset	# of songs	Total length
<b>Training (Labeled)</b>	RWC	100	6h 47m
	MDB	61	2h 39m
	iKala	262	2h 6m
	Jamendo	77	5h
<b>Training (Unlabeled)</b>	FMA_large	106,574	888h
	In-house	171	52h
<b>Test</b>	Jamendo	16	1h
	RWC	60	4h 21m
	ADC04	12	4m
	MIREX05	9	4m
	Orchset	64	24m

Table 7.1: Description of training and test datasets.

genre included. According to [100], it was confirmed that the larger the size of the unlabeled dataset has a positive effect on the improvement of the model performance. Therefore, in this study, we use the FMA\_large data for training the model without conducting a comparative experiment on the data size.

As to hard negative sampling, we used in-house datasets collected from jazz, blues, and classical songs. This dataset contains songs that are played by instruments only, so all pseudo-labels are annotated unvoiced. The harmonic curve corresponding to the melody is wavy like a human voice and its level is dominant, so the SAVD model often mispredicts these unvoiced frames as voice frames. The total length of the in-house dataset amounts to about 52 hours. For all unlabeled data, we augmented the unlabeled datasets via random audio augmentation during training as described in [100].

## 7.4.2 Test data

We used three public test sets (Jamendo, RWC, ADC04, and MIREX05<sup>1</sup>) to evaluate the performance of vocal melody extraction. In this study, we excluded non-vocal tracks from these data sets. Orchset [109] is used for testing to check how many false-positive errors occur. It only contains audio clips of symphonic music without vocal tracks, so we annotated all frames as non-voice.

## 7.5 Experiments

### 7.5.1 Experiment 1: Teacher-Student Model and Iterative Training

To verify the effectiveness of the proposed teacher-student model in SVAD, we compared the supervisory-only model (baseline) and the proposed semi-supervised model (SSL). In this experiment, we trained Noisy Student model described in Section 7.3.2 using several labeled datasets and FMA\_large as unlabeled data. We also iterated the self-training 2 times for the model described in Section 6.5.5 We evaluated the performance of each model on RWC, ADC04, MIERX05 and Orchset, including Jamendo, which is used as a standard data set to evaluate SVAD performance.

As shown in Figure 7.3, SSL<sub>1</sub> outperforms the Baseline on all test datasets and achieve 2.45% higher average accuracy than Baseline. In addition, SSL<sub>2</sub> achieves 0.45% higher average accuracy than SSL<sub>1</sub>.

<sup>1</sup><http://labrosa.ee.columbia.edu/projects/melody/>(<http://labrosa.ee.columbia.edu/projects/melody/>)

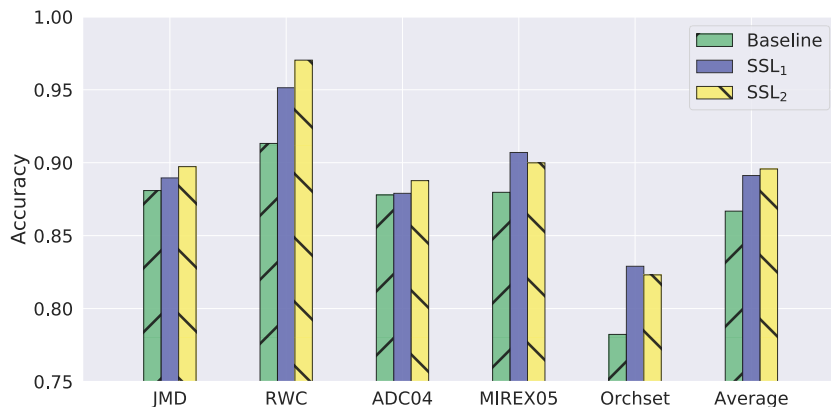


Figure 7.3: Comparison with supervised-only model (as a Baseline) and two Noisy Student models (as a SSL) on five test sets. The subscript ‘1’ and ‘2’ denotes a number of training iteration by teacher-student framework.

These indicate that that teacher-student model and self-training using the noisy student can help improve the performance of singing voice detection by leveraging a large-scale unlabeled dataset.

### 7.5.2 Experiment 2: Hard Negative Sampling

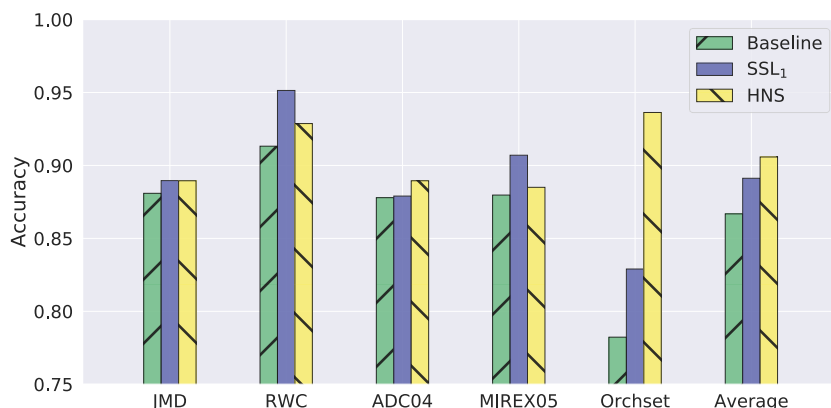


Figure 7.4: Comparison with Baseline model, semi-supervised model, and hard negative sampling model (as an HNS) that hard negative samples are added.

We investigated the effectiveness of the hard negative sampling by comparing the HNS model with Baseline and SSL models. The results are illustrated in Figure 7.4. Orchset consists of audio clips of symphonic music with distinct melody lines, so it is prone to generate false-positive errors in SVAD results. Therefore, the baseline model achieves an accuracy of 78.2% for Orchset. This result is more than 10% less accurate than the other test datasets. On the other hand, the performance of HNS improved for all datasets compared to the baseline model, with a significant performance improvement of 15.4% for Orchset. The results shows that the hard negative sampling is particularly effective to reduce false positives caused by melodies played by the musical instrument while improving overall performance. It also indicates that the hard negative mixing strategy would help the model learn representations that could discriminate between the voice and the instrument. However, the performance of the HNS model was lower compared to the SSL model except for ADC04. It shows that simply treating negative samples

within the mini-batch could hurt discriminative training.

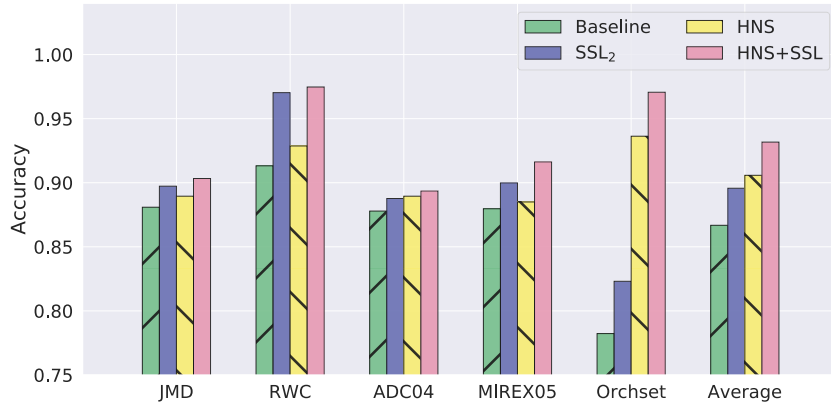


Figure 7.5: Comparison with Baseline, SSL, HNS and ‘HNS+SSL’ model that is a semi-supervised learning model with unlabeled data and hard negative samples

To take advantage of both hard negative sampling and SSL, we trained the student model (as a HNS+SSL) by using the HNS model as a teacher and additionally using hard negative samples as unlabeled data. As shown in Figure 7.5, HNS+SSL has a 93.2% average accuracy and outperforms other models. In particular, There is a only 3% false-positive in Orchset, which is 18.8% higher than the baseline model. This results show that large-scale unlabeled data and hard negative samples available over SSL are effective to prevent overfitting and would help the networks produce a decision boundary.

## 7.6 Comparison with State-of-the-Arts

We compared the proposed model with other singing voice detection algorithms: Mauch [102], Lehner-1 [98], Lehner-2 [43], Lehner-3 [51], Leglaive [36], Schlüter [38], Zhang [99]. Each algorithm was evaluated on the RWC and Jamendo datasets, and the comparison results are shown in Table 7.2 and Table 7.3.

	Accuracy	Precision	Recall	F1
Lehner-1	0.848	-	-	0.846
Lehner-2	0.882	0.880	0.862	0.871
Lehner-3	0.894	0.898	0.906	0.902
Schlüter	<b>0.923</b>	-	0.903	-
Leglaive	0.915	0.895	<b>0.926</b>	<b>0.910</b>
Zhang	0.888	0.865	0.920	0.892
Proposed	0.903	<b>0.934</b>	0.884	0.904

Table 7.2: Singing voicing detection results of the proposed and other methods on the Jamendo dataset.

Some performances were not listed in these tables as no experimental results were reported. The performance of the proposed model is generally comparable to other algorithms on the Jamendo dataset and outperforms other algorithms on the RWC dataset. The results show that the SSL through the noisy student architecture and applying hard negative sampling together helps to make the model more robust.

	<b>Accuracy</b>	<b>Precision</b>	<b>Recall</b>	<b>F1</b>
Mauch	0.872	0.887	0.921	0.904
Lehner-1	0.868	0.879	0.906	0.892
Lehner-2	0.875	0.875	0.926	0.900
Lehner-3	0.923	0.938	0.934	0.936
Schlüter	0.927	-	0.935	-
Zhang	0.916	0.926	0.934	0.930
Proposed	<b>0.975</b>	<b>0.973</b>	<b>0.973</b>	<b>0.974</b>

Table 7.3: Singing voicing detection results of the proposed and other methods on the RWC dataset.

## 7.7 Conclusions

In this study, we also reveal that the SSL help improves performance singing voice detection as well as vocal melody extraction leveraging large-scale unlabeled music datasets and hard negative samples. Large-scale unlabeled data and hard negative samples available over SSL are effective to prevent overfitting. By comparing the performance of each model in Orchset, we found that hard negative sampling was effective in reducing false-positive errors. We also revealed that rather than simply adding hard negative samples to the training data, it is helpful to improve the performance of the model by utilizing them for training through SSL with large-scale unlabeled data. However, the limitation of this study is that the hard negative samples were selected heuristically from songs whose melody was played as an instrument. Future studies should explore a general method for automatically selecting hard negative samples from training data instead of the heuristic method.



## Chapter 8. Conclusions

### 8.1 Summary

Vocal melody extraction is the task that identifies the melody pitch contour of singing voice from polyphonic music. We have presented our various works on vocal melody extraction using deep learning methods, such as various models, loss functions, and learning methods.

Chapter 2 briefly introduced conventional algorithms and reviewed various deep learning-based models in terms of input/output feature, pre/post-processing, model architecture, and loss function.

In Chapter 3, we presented a classification-based singing melody extraction model using CNN. The proposed model consists of a singing pitch extractor (SPE) and a singing voice activity detector (SVAD). The SPE is trained to predict a high-resolution pitch label of singing voice from a short segment of a spectrogram. This allows the model to predict highly continuous curves. The melody contour is smoothed further by post-processing the output of the melody extractor. SVAD was trained to use the mel-spectrogram as input to detect if the corresponding frames contain singing voices. This often produces a voice false alarm error. However, we could reduce the errors around the boundary of singing segments by exploiting the output of the SPE.

Chapter 4 reviewed the existing loss functions ( $CE_G$ , HSL, FL) and other proposed variants (GPL, GOL). Unlike categorical labels such as music genre, the pitch label is represented as a continuous scale and also the distance between two pitches is highly nonlinear. Considering the task-specific characteristics, we proposed several variations of categorical loss functions. Through an experiment comparing the loss function, we discussed the direction to improve the melody extraction performance.

In Chapter 5, we introduced a joint detection and classification (JDC) network that conducts the singing voice detection and the pitch estimation simultaneously. The JDC network is composed of the main network that predicts the pitch contours of the singing melody and an auxiliary network that facilitates the detection of the singing voice. The main network is built with a CRNN with residual connections and predicts pitch labels that cover the vocal range with a high resolution, as well as non-voice status. The auxiliary network is trained to detect the singing voice using multi-level features shared from the main network. The two optimization processes are tied with a joint melody loss function. The experiments demonstrate how the auxiliary network and the joint melody loss function improve melody extraction performance. Furthermore, the results show that our method outperforms state-of-the-art algorithms on the datasets.

Chapter 6 presented the SSL method using teacher-student (TS) models for vocal melody extraction. The lack of labeled data is a major obstacle in melody extraction, where labeling is extremely laborious or costly. SSL provides a solution to alleviate the issue by leveraging a large amount of unlabeled data. The teacher model is pre-trained with labeled data and guides the student model to make identical predictions given unlabeled input in a self-training setting. We examined three setups of TS models with different data augmentation schemes and loss functions. Also, considering the scarcity of labeled data in the test phase, we artificially generated large-scale testing data with pitch labels from unlabeled data using an analysis-synthesis method. The results show that the SSL method significantly increases the performance against supervised learning only and the improvement depends on the TS models, the size of unlabeled data, the number of self-training iterations, and other training details. We also found that

it is essential to ensure that the unlabeled audio has vocal parts. Finally, we showed that the proposed SSL method enables a baseline CRNN model to achieve performance comparable to state-of-the-arts.

Chapter 7 revealed that the SSL method also effective to improve performance singing voice detection as well as vocal melody extraction leveraging large-scale unlabeled music datasets and hard negative samples. These results show that the proposed SSL method is not limited to melody extraction or singing voice detection, but can be extended to other MIR operations without labeled data such as chord detection and automatic music recording.

## 8.2 Review on Current State and Future Work

In this thesis, we focused on vocal melody extraction and discussed various methods and training details. However, there are still several limitations and a large margin of improvement in melody extraction. We need to discuss the scope of the ‘vocal melody’ to evaluate the performance of the algorithm more reasonably. The MedleyDB dataset divides the definition of melody into three based on the number of target source and melody line and provides ground-truth labels according to these definitions. However, there is still no consensus on the definition of the term ‘vocal melody’. For example, it should be discussed how to define the melody in the part where singers sing harmony, or in the part where they sing like a background like ‘woo—ah—’. This problem had an effect on the devaluation of the actual algorithm performance. This problem affected the devaluation of the actual algorithm performance. Therefore, it is essential to clarify the definition of a vocal melody and properly label the pitch according to that definition.

Weak or unclear vocal sounds often give rise to missing or false-positive error, because it is difficult to clearly distinguish between the vocal melody and the harmonic pattern in the background. With separate vocal tracks, we may mitigate the errors related to level by adjusting the mixing of vocal gains and augmenting the training data. On the other hand, excessive effects applied to the mastering process produce a fundamental problem that makes the singing voice obscure. Until a method that can effectively attenuate these sound effects is proposed, it could still remain a challenging task to accurately determine estimating pitch and detecting voicing.

An interesting area of future work is the in-depth discussion of new metrics that reflect the continuity and accuracy of the pitch. The need for new metrics related to continuity comes from the fact that existing evaluation metrics are based on frame-level accuracy and do not deal with continuity. Furthermore, the objective test results evaluated for each frame are not necessarily related to the perceptual test results. We may evaluate pitch continuity by introducing consecutive frame-level accuracy instead of single frame-level accuracy. In addition, it is necessary to study how the perceived results correlate with the existing evaluation metrics focused on the quantitative evaluation of the algorithm. We can also propose another new evaluation metric that can indicate precision power by aggregating performance measures across all possible pitch tolerance. For evaluation of melody extraction algorithm, the pitch tolerance is typically fixed at  $\pm 50$  cents, which can make the evaluation metric sensitive to a specific threshold. Inspired by area under a ROC curve [110], we also expect to propose a new evaluation by plotting according to pitch tolerance from 0 to 1 semitone and calculating the area under the curve. It can summarize the overall model performance for all possible thresholds, avoiding the subjectivity assumed in the pitch tolerance selection process.

Another of our next goals is to propose applications that help users find the music they are interested in by automatically analyzing and comparing the melody of the song. Modern music search services

require more advanced solutions to search for songs or evaluate musical characteristics. Since the melody is one of the musical aspects that often does not change with other performances, melody extraction is the most basic step in music analysis. We believe that our high-performance algorithms and strategies for melody extraction can attribute to the application such as cover song detection and query-by-humming.

## Bibliography

- [1] Meyer, L. B., *Style and music: Theory, history, and ideology*. University of Chicago Press, 1996.
- [2] Poliner, G. E., Ellis, D. P., Ehmann, A. F., Gómez, E., Streich, S., and Ong, B. (2007). “Melody transcription from music audio: Approaches and evaluation.” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 15, pp. 1247–1256.
- [3] Müller, M., Gómez, E., and Yang, Y.-H., “Computational methods for melody and voice processing in music recordings (dagstuhl seminar 19052).” in *Dagstuhl Reports*, vol. 9, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.
- [4] Schulkind, M. D. (2004). “Conceptual and perceptual information both influence melody identification.” *Memory & cognition*, 32, pp. 841–851.
- [5] Warrier, C. M., and Zatorre, R. J. (2002). “Influence of tonal context and timbral variation on perception of pitch.” *Perception & psychophysics*, 64, pp. 198–207.
- [6] Salamon, J., Gómez, E., Ellis, D. P., and Richard, G. (2014). “Melody extraction from polyphonic music signals: Approaches, applications, and challenges.” *IEEE Signal Processing Magazine*, 31, pp. 118–134.
- [7] Salamon, J., and Gómez, E. (2012). “Melody extraction from polyphonic music signals using pitch contour characteristics.” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 20, pp. 1759–1770.
- [8] Kim, K., Lee, J., Kum, S., Park, C. L., and Nam, J. (2020). “Semantic tagging of singing voices in popular music recordings.” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, pp. 1656–1668.
- [9] Dressler, K., “An auditory streaming approach for melody extraction from polyphonic music..” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 19–24, 2011.
- [10] Arora, V., and Behera, L. (2013). “On-line melody extraction from polyphonic audio using harmonic cluster tracking.” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 21, pp. 520–530.
- [11] Durrieu, J.-L., Richard, G., David, B., and Févotte, C. (2010). “Source/filter model for unsupervised main melody extraction from polyphonic audio signals.” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 18, pp. 564–575.
- [12] Tachibana, H., Ono, T., Ono, N., and Sagayama, S., “Melody line estimation in homophonic music audio signals based on temporal-variability of melodic source.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 425–428, 2010.

- [13] Ikemiya, Y., Itoyama, K., and Yoshii, K. (2016). “Singing voice separation and vocal f0 estimation based on mutual combination of robust principal component analysis and subharmonic summation.” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24, pp. 2084–2095.
- [14] Ellis, D. P. W., and Poliner, G. E. (2006). “Classification-based melody transcription.” *Machine Learning*, 65, pp. 439–456.
- [15] Bittner, R. M., Salamon, J., Essid, S., and Bello, J. P., “Melody extraction by contour classification.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 500–506, 2015.
- [16] Kum, S., Oh, C., and Nam, J., “Melody extraction on vocal segments using multi-column deep neural networks.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 819–825, ISMIR, 2016.
- [17] Rigaud, F., and Radenen, M., “Singing voice melody transcription using deep neural networks.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 737–743, 2016.
- [18] Park, H., and Yoo, C. D., “Melody extraction and detection through lstm-rnn with harmonic sum loss.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2766–2770, 2017.
- [19] Basaran, D., Essid, S., and Peeters, G., “Main melody extraction with source-filter nmf and crnn.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 82–89, 2018.
- [20] Bittner, R. M., McFee, B., Salamon, J., Li, P., and Bello, J. P., “Deep salience representations for f0 estimation in polyphonic music.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, 2017.
- [21] Su, L., “Vocal melody extraction using patch-based cnn.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 371–375, 2018.
- [22] Lu, W.-T., and Su, L., “Vocal melody extraction with semantic segmentation and audio-symbolic domain transfer learning.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 521–528, 2018.
- [23] Hsieh, T.-H., Su, L., and Yang, Y.-H., “A streamlined encoder/decoder architecture for melody extraction.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 156–160, 2019.
- [24] Gao, Y., Zhu, B., Li, W., Li, K., Wu, Y., and Huang, F., “Vocal melody extraction via dnn-based pitch estimation and salience-based pitch refinement.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1000–1004, 2019.
- [25] Chou, H., Chen, M.-T., and Chi, T.-S., “A hybrid neural network based on the duplex model of pitch perception for singing melody extraction.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 381–385, 2018.

- [26] Chen, M.-T., Li, B.-J., and Chi, T.-S., “Cnn based two-stage multi-resolution end-to-end model for singing melody extraction.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1005–1009, 2019.
- [27] Kum, S., and Nam, J. (2019). “Joint detection and classification of singing voice melody using convolutional recurrent neural networks.” *Applied Sciences*, 9, p. 1324.
- [28] Hsieh, T.-H., Su, L., and Yang, Y.-H. (2018). “A streamlined encoder/decoder architecture for melody extraction.” *arXiv preprint arXiv:1810.12947*.
- [29] Ghias, A., Logan, J., Chamberlin, D., and Smith, B. C., “Query by humming: musical information retrieval in an audio database.” in *In Proceedings of the ACM international conference on Multimedia*, pp. 231–236, ACM, 1995.
- [30] Serra, J., Gómez, E., and Herrera, P. (2010). “Audio cover song identification and similarity: Background, approaches, evaluation, and beyond..” *Advances in Music Information Retrieval*, 274, pp. 307–332.
- [31] Salamon, J., Rocha, B., and Gómez, E., “Musical genre classification using melody features extracted from polyphonic music signals.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 81–84, 2012.
- [32] Kako, T., Ohishi, Y., Kameoka, H., Kashino, K., and Takeda, K., “Automatic identification for singing style based on sung melodic contour characterized in phase plane..” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 393–398, 2009.
- [33] Hsu, C.-L., and Jang, J.-S. R. (2010). “On the improvement of singing voice separation for monaural recordings using the mir-1k dataset.” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 18, pp. 310–319.
- [34] Bittner, R. M., Salamon, J., Tierney, M., Mauch, M., Cannam, C., and Bello, J. P., “Medleydb: A multitrack dataset for annotation-intensive mir research..” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, vol. 14, pp. 155–160, 2014.
- [35] Chan, T.-S., Yeh, T.-C., Fan, Z.-C., Chen, H.-W., Su, L., Yang, Y.-H., and Jang, R., “Vocal activity informed singing voice separation with the ikala dataset.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 718–722, 2015.
- [36] Leglaive, S., Hennequin, R., and Badeau, R., “Singing voice detection with deep recurrent neural networks.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 121–125, 2015.
- [37] Park, H., and Yoo, C. D., “Melody extraction and detection through LSTM-RNN with harmonic sum loss.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2766–2770, 2017.
- [38] Schlüter, J., and Grill, T., “Exploring data augmentation for improved singing voice detection with neural networks..” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 121–126, 2015.

- [39] Kelz, R., Dorfer, M., Korzeniowski, F., Böck, S., Arzt, A., and Widmer, G., “On the potential of simple framewise approaches to piano transcription.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 475–481, 2016.
- [40] Korzeniowski, F., and Widmer, G., “A fully convolutional deep auditory model for musical chord recognition.” in *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, 2016.
- [41] Lee, J., Park, J., Kim, K. L., and Nam, J., “Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms.” in *In Proceedings of the Sound and Music Computing Conference (SMC)*, pp. 220–226, 2017.
- [42] Lin, M., Chen, Q., and Yan, S. (2013). “Network in network;.” *arXiv preprint arXiv:1312.4400*.
- [43] Lehner, B., Widmer, G., and Sonnleitner, R., “On the reduction of false positives in singing voice detection.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7480–7484, 2014.
- [44] Goto, M., Hashiguchi, H., Nishimura, T., and Oka, R., “Rwc music database: Popular, classical and jazz music databases..” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, vol. 2, pp. 287–288, 2002.
- [45] Ramona, M., Richard, G., and David, B., “Vocal detection in music with support vector machines.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1885–1888, 2008.
- [46] Laroche, J., “Time and pitch scale modification of audio signals.” in *In Proceedings of the Applications of digital signal processing to audio and acoustics*, pp. 279–309, Springer, 2002.
- [47] Raffel, C., McFee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., Ellis, D. P., and Raffel, C. C., “mir\_eval: A transparent implementation of common mir metrics.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, Citeseer, 2014.
- [48] He, K., Zhang, X., Ren, S., and Sun, J., “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.” in *In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1026–1034, 2015.
- [49] Chollet, F., “Keras.” <https://github.com/fchollet/keras>, 2015.
- [50] Sokolova, M., and Lapalme, G. (2009). “A systematic analysis of performance measures for classification tasks.” *Information Processing & Management*, 45, pp. 427–437.
- [51] Lehner, B., Widmer, G., and Bock, S., “A low-latency, real-time-capable singing voice detection method with lstm recurrent neural networks.” in *In Proceeding of the Signal Processing Conference (EUSIPCO)*, pp. 21–25, IEEE, 2015.
- [52] Dressler, K., “Pitch estimation by the pair-wise evaluation of spectral peaks.” in *In Proceedings of the AES Conference on Semantic Audio*, Audio Engineering Society, 2011.
- [53] Bosch, J. J., Bittner, R. M., Salamon, J., and Gómez, E., “A comparison of melody extraction methods based on source-filter modelling..” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 571–577, 2016.

- [54] Schlüter, J., and Lehner, B., “Zero-Mean Convolutions for Level-Invariant Singing Voice Detection.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, 2018.
- [55] Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2018). “Focal loss for dense object detection.” *IEEE transactions on pattern analysis and machine intelligence*.
- [56] Lu, W.-T., and Su, L., “Vocal melody extraction with semantic segmentation and audio-symbolic domain transfer learning.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, 2018.
- [57] Simonyan, K., and Zisserman, A. (2014). “Very deep convolutional networks for large-scale image recognition.” *arXiv preprint arXiv:1409.1556*.
- [58] Kim, J. W., Salamon, J., Li, P., and Bello, J. P., “CREPE: A convolutional representation for pitch estimation.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.
- [59] Molina, E., Tardón, L. J., Barbancho-Perez, I., Barbancho-Perez, A. M., et al., “The importance of f0 tracking in query-by-singing-humming.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, 2014.
- [60] Reddy, M. G., and Rao, K. S. (2017). “Predominant melody extraction from vocal polyphonic music signal by time-domain adaptive filtering-based method.” *Circuits, Systems, and Signal Processing*, pp. 1–23.
- [61] Zhang, W., Chen, Z., Yin, F., and Zhang, Q. (2018). “Melody extraction from polyphonic music using particle filter and dynamic programming.” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 26, pp. 1620–1632.
- [62] Su, L., “Vocal melody extraction using patch-based cnn.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 371–375, 2018.
- [63] Zhang, X., and LeCun, Y., “Universum prescription: regularization using unlabeled data.” in *AAAI*, pp. 2907–2913, 2017.
- [64] Ruder, S. (2017). “An overview of multi-task learning in deep neural networks.” *arXiv preprint arXiv:1706.05098*.
- [65] Huang, P.-S., Kim, M., Hasegawa-Johnson, M., and Smaragdis, P. (2015). “Joint optimization of masks and deep recurrent neural networks for monaural source separation.” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 23, pp. 2136–2147.
- [66] Stoller, D., Ewert, S., and Dixon, S., “Jointly detecting and separating singing voice: A multi-task approach.” in *International Conference on Latent Variable Analysis and Signal Separation*, pp. 329–339, Springer, 2018.
- [67] Kong, Q., Xu, Y., and Plumbley, M. D., “Joint detection and classification convolutional neural network on weakly labelled bird audio detection.” in *In Proceedings of the Signal Processing Conference (EUSIPCO)*, pp. 1749–1753, 2017.



- [68] Ioffe, S., and Szegedy, C. (2015). “Batch normalization: Accelerating deep network training by reducing internal covariate shift.” *arXiv preprint arXiv:1502.03167*.
- [69] Maas, A. L., Hannun, A. Y., and Ng, A. Y., “Rectifier nonlinearities improve neural network acoustic models.” in *In Proceedings of the International Conference on Machine Learning (ICML)*, vol. 30, 2013.
- [70] He, K., Zhang, X., Ren, S., and Sun, J., “Identity mappings in deep residual networks.” in *In Proceedings of the European conference on computer vision*, pp. 630–645, Springer, 2016.
- [71] Veit, A., Wilber, M. J., and Belongie, S., “Residual networks behave like ensembles of relatively shallow networks.” in *Advances in neural information processing systems*, pp. 550–558, 2016.
- [72] Lee, K., Choi, K., and Nam, J., “Revisiting singing voice detection: a quantitative review and the future outlook.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 506–513, 2018.
- [73] Salamon, J., “What’s broken in music informatics research? three uncomfortable statements.” in *Proc. of the 36th International Conference on Machine Learning, PMLR 97*, 2019.
- [74] Hsu, C.-L., and Jang, J.-S. R. (2009). “On the improvement of singing voice separation for monaural recordings using the mir-1k dataset.” *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 18, pp. 310–319.
- [75] Salamon, J., Bittner, R. M., Bonada, J., Bosch, J. J., Gómez, E., and Bello, J. P., “An analysis/synthesis framework for automatic f0 annotation of multitrack datasets..” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 71–78, 2017.
- [76] Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., and Raffel, C. A., “Mixmatch: A holistic approach to semi-supervised learning.” in *Advances in Neural Information Processing Systems*, pp. 5050–5060, 2019.
- [77] Xie, Q., Hovy, E., Luong, M.-T., and Le, Q. V. (2019). “Self-training with noisy student improves imagenet classification.” *arXiv preprint arXiv:1911.04252*.
- [78] Sohn, K., Berthelot, D., Li, C.-L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., and Raffel, C. (2020). “Fixmatch: Simplifying semi-supervised learning with consistency and confidence.” *arXiv preprint arXiv:2001.07685*.
- [79] Mošner, L., Wu, M., Raju, A., Parthasarathi, S. H. K., Kumatani, K., Sundaram, S., Maas, R., and Hoffmeister, B., “Improving noise robustness of automatic speech recognition via parallel data and teacher-student learning.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6475–6479, 2019.
- [80] Lu, K., Foo, C.-S., Teh, K. K., Tran, H. D., and Chandrasekhar, V. R. (2019). “Semi-supervised audio classification with consistency-based regularization.” *Proc. Interspeech*, pp. 3654–3658.
- [81] Wu, C.-W., and Lerch, A., “Automatic drum transcription using the student-teacher learning paradigm with unlabeled music data.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 613–620, 2017.

- [82] Schlüter, J., “Learning to pinpoint singing voice from weakly labeled examples..” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 44–50, 2016.
- [83] Meseguer-Brocal, G., Cohen-Hadria, A., and Peeters, G., “Dali: A large dataset of synchronized audio, lyrics and notes, automatically created using teacher-student machine learning paradigm.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, 2018.
- [84] Lee, D.-H., “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks.” in *Workshop on challenges in representation learning, ICML*, vol. 3, p. 2, 2013.
- [85] Sajjadi, M., Javanmardi, M., and Tasdizen, T., “Regularization with stochastic transformations and perturbations for deep semi-supervised learning.” in *Advances in neural information processing systems*, pp. 1163–1171, 2016.
- [86] Xie, Q., Dai, Z., Hovy, E., Luong, M.-T., and Le, Q. V. (2019). “Unsupervised data augmentation for consistency training.” *arXiv preprint arXiv:1904.12848*.
- [87] Bittner, R. M., McFee, B., and Bello, J. P. (2018). “Multitask learning for fundamental frequency estimation in music.” *arXiv preprint arXiv:1809.00381*.
- [88] Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. (2019). “Randaugment: Practical data augmentation with no separate search.” *arXiv preprint arXiv:1909.13719*.
- [89] Yalniz, I. Z., Jégou, H., Chen, K., Paluri, M., and Mahajan, D. (2019). “Billion-scale semi-supervised learning for image classification.” *arXiv preprint arXiv:1905.00546*.
- [90] Defferrard, M., Benzi, K., Vandergheynst, P., and Bresson, X., “FMA: A dataset for music analysis.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, 2017.
- [91] Liutkus, A., Stöter, F.-R., Rafii, Z., Kitamura, D., Rivet, B., Ito, N., Ono, N., and Fontcave, J., “The 2016 signal separation evaluation campaign.” in *Latent Variable Analysis and Signal Separation - 12th International Conference*, pp. 323–332, Springer, 2017.
- [92] Rafii, Z., Liutkus, A., Stöter, F.-R., Mimitakis, S. I., and Bittner, R., “The MUSDB18 corpus for music separation.” 2017.
- [93] Morise, M., Yokomori, F., and Ozawa, K. (2016). “WORLD: a vocoder-based high-quality speech synthesis system for real-time applications.” *IEICE TRANSACTIONS on Information and Systems*, 99, pp. 1877–1884.
- [94] Morise, M. (2016). “D4C, a band-aperiodicity estimator for high-quality speech synthesis.” *Speech Communication*, 84, pp. 57–65.
- [95] Mauch, M., and Dixon, S., “pYIN: A fundamental frequency estimator using probabilistic threshold distributions.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 659–663, 2014.
- [96] Oliver, A., Odena, A., Raffel, C. A., Cubuk, E. D., and Goodfellow, I., “Realistic evaluation of deep semi-supervised learning algorithms.” in *Advances in Neural Information Processing Systems*, pp. 3235–3246, 2018.

- [97] Khine, S. Z. K., Nwe, T. L., and Li, H., “Singing voice detection in pop songs using co-training algorithm.” in *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1629–1632, 2008.
- [98] Lehner, B., Sonnleitner, R., and Widmer, G., “Towards light-weight, real-time-capable singing voice detection..” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 53–58, 2013.
- [99] Zhang, X., Yu, Y., Chen, X., and Li, W. (2020). “Comparison for improvements of singing voice detection system based on vocal separation.” *arXiv preprint arXiv:2004.04040*.
- [100] Kum, S., Lin, J.-H., Su, L., and Nam, J., “Semi-supervised learning using teacher-student models for vocal melody extraction.” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, ISMIR, 2020.
- [101] Rocamora, M., and Herrera, P., “Comparing audio descriptors for singing voice detection in music audio files.” in *Brazilian symposium on computer music, 11th. san pablo, brazil*, vol. 26, p. 27, 2007.
- [102] Mauch, M., Fujihara, H., Yoshii, K., and Goto, M., “Timbre and melody features for the recognition of vocal activity and instrumental solos in polyphonic music..” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 233–238, 2011.
- [103] Schlüter, J., and Lehner, B., “Zero-mean convolutions for level-invariant singing voice detection..” in *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 321–326, 2018.
- [104] Berthelot, D., Carlini, N., Cubuk, E. D., Kurakin, A., Sohn, K., Zhang, H., and Raffel, C. (2019). “Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring.” *arXiv preprint arXiv:1911.09785*.
- [105] Schroff, F., Kalenichenko, D., and Philbin, J., “Facenet: A unified embedding for face recognition and clustering.” in *In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 815–823, 2015.
- [106] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C., “Ssd: Single shot multibox detector.” in *In Proceedings of the European conference on computer vision*, pp. 21–37, Springer, 2016.
- [107] Shrivastava, A., Gupta, A., and Girshick, R., “Training region-based object detectors with online hard example mining.” in *In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 761–769, 2016.
- [108] Zhang, X., Fang, Z., Wen, Y., Li, Z., and Qiao, Y., “Range loss for deep face recognition with long-tailed training data.” in *In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5409–5418, 2017.
- [109] Bosch, J. J., Marxer, R., and Gómez, E. (2016). “Evaluation and combination of pitch estimation methods for melody extraction in symphonic classical music.” *Journal of New Music Research*, 45, pp. 101–117.

- [110] Hanley, J. A., and McNeil, B. J. (1982). "The meaning and use of the area under a receiver operating characteristic (roc) curve.." *Radiology*, 143, pp. 29-36.

## Acknowledgments in Korean

'수고했네 금 박사'. 디펜스를 마친 후 건내는 지도 교수님의 악수는 그저 가볍지만 애틍습니다. 끝이 안보이던 이 길을 드디어!! 마무리 하는 단계이 서 있지만 마음은 그저 홀가분 하지 애틍습니다. 쉽게 잠들 줄 알았던 그날 밤은 지난 7여년간의 기쁨과 슬픔이 영화처럼 지나가 쉽사리 잠들지 못했습니다. 지금까지도 왠지 모를 무거움과 부담을 느끼는 까닭은, 여전히 스스로 보기에 부족하다고 느끼기 때문입니다. 나 홀로 무언가를 해낸 것이 정말 하나도 없기 때문입니다. '거인의 어깨에 올라서서 더 넓은 세상을 바라보라' 라는 아이작 뉴턴의 말처럼 저는 그동안 정말 많은 사람들의 어깨를 빌려 왔습니다. 그 분들에게 이 작은 공간을 빌려 감사의 말씀을 전하고자 합니다.

가장 먼저는 저를 독립적인 연구자로 스스로 연구를 할 준비를 마칠 수 있도록 힘껏 도와주신 남주한 교수님께 큰 감사의 말씀을 전해드립니다. 인생에 있어서 가장 소중한 만남이 부모님, 배우자 그리고 스승님 이라고 하는데, 저는 최고의 스승님 밑에서 학문을 정진할 수 있는 큰 기쁨을 누렸습니다. 교수님의 가르침을 잊지 않고 계속해서 이어나가도록 하겠습니다.

그리고 가장 오랜 시간동안 함께 한 맥랩의 동료 모두에게 감사의 말을 전합니다. 기쁘고 슬픈 모든 순간을 함께 한 여러분들이 없었다면 절대! 여기까지 올 수 없었을 것입니다. 뛰어난 여러분과 함께 할 수 있었던 것은 저에게 큰 도움이 되었습니다. 정말 감사합니다. 그리고 학업 이후의 여정을 함께 할 Neutune 멤버들, 힘차게 우리의 비전을 향해 나아가 봅시다!! Thanks to Professor Li Su and Jing-Hua Lin from Academia Sinica, Taiwan. Your deep insights and enthusiasm have encouraged me a lot in my research.

무엇보다 물심양면으로, 기도로 응원해 주신 부모님께 깊은 감사의 말씀을 전해드립니다. 부모님이 저의 부모님이어서 저는 정말 하나님께 감사합니다. 지금까지 몸소 보여주신 사랑을 그대로 본받아 사랑 가득한 예쁜 가정 꾸려가겠습니다. 새롭게 만나게 된 저의 또 다른 부모님인 예비 장인어른, 장모님. 항상 따뜻하게 예비 사위 맞아 주셔서 끝까지 논문을 지치지 않고 마무리 할 수 있었습니다. 정말 감사합니다. 마지막으로 끝없이 사랑을 표현해 주고 응원해 준 하진! 정말 고맙고 사랑해!! 덕분에 이 힘든 과정을 기쁨으로 끝까지 나아갈 수 있었어. 앞으로의 여정도 지금까지 그래 왔듯이 하나님의 인도하심 따라 함께 동행해보자.

저의 석사 논문의 감사의 글의 끝에 나누었던 구절을 인용하면서, 저의 다짐으로 감사의 글을 마무리 하고자 합니다. 근심 속에 있지만 항상 기뻐하며, 아무 것도 없어 보이지만 많은 사람을 부요하게 하는 인생의 멋진 연주를 계속해 나가겠습니다. 감사합니다.

## Curriculum Vitae

Name : 금 상 은  
Date of Birth : June 18, 1987  
E-mail : keums@kaist.ac.kr

### Educations

2003. 3. – 2006. 2. Young Shin High School  
2007. 3. – 2014. 8. College of IT Engineering, Kyungpook National University (B.S.)  
2014. 9. – 2016. 8. Graduate School of Culture Technology, KAIST (M.S.)  
2016. 9. – 2021. 2. Graduate School of Culture Technology, KAIST (Ph.D)

### Career

2018. 7. – 2018. 8. Research Intern at Naver Labs, Naver Corporation, Korea

### Publications

1. Jongpil Lee, Tae Hyoung Kim, **Sangeun Kum**, Keunhyoung Luke Kim, Changheun Oh, Juhan Nam, "Investigation on Vocal Tags and Singer Similarity of K-pop", 한국음향학회, 춘계학술대회 발표논문집, 2016
2. **금상은**, 남주한, "멜로디 추출 알고리즘으로 살펴보는 음악정보검색", 대한전자공학회, 전자공학회지 43(5), pp. 41-49, 2016
3. **Sangeun Kum**, Changheun Oh, Juhan Nam, "Melody Extraction on vocal segments using Multi-Column Deep Neural Networks", in Proceedings of ISMIR, 2016
4. Jongpil Lee , Jiyoung Park , **Sangeun Kum** , Youngho Jeong , Juhan Nam. *Combining multi-scale features using sample-level deep convolutional neural networks for weakly supervised sound event detection*. Proc. DCASE, 2017, 69-73.
5. KeunHyoung Luke Kim, **Sangeun Kum**, Chae Lin Park, Jongpil Lee, Jiyoung Park, Juhan Nam (2017, October). *Building k-pop singing voice tag dataset: A progress report*. In Late Breaking Demo in the International Society for Music Information Retrieval Conf.
6. Jiyoung Park, Donghyun Kim, Jongpil Lee, **Sangeun Kum**, Juhan Nam (2018). A hybrid of deep audio feature and i-vector for artist recognition. Joint Workshop on Machine Learning for Music, ICML
7. **Sangeun Kum**, and Juhan Nam. "Joint detection and classification of singing voice melody using convolutional recurrent neural networks." Applied Sciences 9.7 (2019): 1324.
8. **Sangeun Kum**, Jing-Hua, Li Su, Juhan Nam, "Semi-supervised learning using teacher-student models for vocal melody extraction", in Proceedings of ISMIR, 2020

9. Keunhyoung Luke Kim , Jongpil Lee , **Sangeun Kum**, Chae Lin Park, and Juhan Nam (2020). *Semantic Tagging of Singing Voices in Popular Music Recordings*. IEEE/ACM Transactions on Audio, Speech, and Language Processing.