MAJL: A Model-Agnostic Joint Learning Framework for Music Source Separation and Pitch Estimation

Anonymous Authors

ABSTRACT

Music source separation and pitch estimation are two vital tasks in music information retrieval. Typically, the input of pitch estimation is obtained from the output of music source separation. Therefore, existing methods have tried to perform these two tasks simultaneously, so as to leverage the mutually beneficial relationship between both tasks. However, these methods still face two critical challenges that limit the improvement of both tasks: the lack of labeled data and joint learning optimization. To address these challenges, we propose a Model-Agnostic Joint Learning (MAJL) framework for both tasks. MAJL is a generic framework and can use variant models for each task. It includes a two-stage training method and a dynamic weighting method named *Dynamic Weights on Hard Samples* (DWHS), which addresses the lack of labeled data and joint learning optimization, respectively. Experimental results on public music datasets show that MAJL outperforms state-of-the-art methods on both tasks, with significant improvements of 0.92 in Signal-to-Distortion Ratio (SDR) for music source separation and 2.71% in Raw Pitch Accuracy (RPA) for pitch estimation. Furthermore, comprehensive studies not only validate the effectiveness of each component of MAJL, but also indicate the great generality of MAJL in adapting to different model architectures.

CCS CONCEPTS

• **Applied computing** → **Sound and music computing**.

KEYWORDS

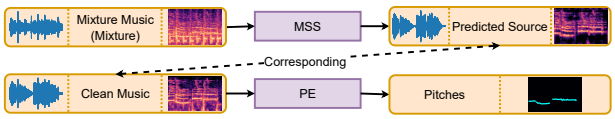
music source separation, pitch estimation, joint learning

1 INTRODUCTION

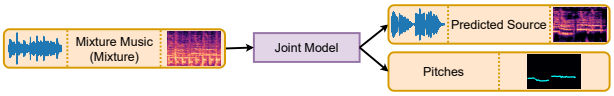
The digital music industry has been growing rapidly in the last few years due to the mass publication of music through smartphone apps, enabling hundreds of millions of people to access a song via large music platforms. This has created huge music streaming companies such as Spotify (worth $37B) and QQ Music (worth $10B). The digital music industry in the US has a market of close to $10B in 2022 and has been growing at more than 10% in each of the last five years [39] and that in China has a market of $5B and has been growing at 30~50% in each of the last five years [26].

Music Information Retrieval (MIR) is a pivotal research domain that supports the functionality of large music platforms. Within MIR, music source separation (MSS) and pitch estimation (PE) emerge as critical tasks with far-reaching implications. MSS facilitates several downstream tasks, such as lyrics extraction [15] and music transcription [2], accentuating its pivotal role. PE has great significance for various MIR applications, including content-based music recommendation and query/search by singing [60]. Notably, real-world musical compositions are often mixture music, typically as .wav or .mp3 files, which do not inherently provide the pitch information of music data. To extract target sources along with their corresponding pitches, simultaneous execution of both MSS and PE tasks becomes imperative.

The MSS task entails generating isolated stems for vocals, bass, or drums from raw audio or spectrograms of mixture music, as shown in the top row of Figure 1(a). The PE task involves extracting fundamental frequencies ($f0$) from clean music audio or spectrograms, as shown in the second row of Figure 1(a). Following previous studies [18, 32, 61], we default to using pitch estimation to refer to single pitch estimation unless explicitly stated otherwise in this paper. It is important to note that the clean music in the PE task corresponds to the predicted source in the MSS task because both are music data from a specific instrument.



(a) The flow chart of pipeline methods for MSS and PE.



(b) The flow chart of joint learning methods for MSS and PE.

Figure 1: Existing methods for MSS and PE.

Music source separation and pitch estimation are closely related in music information retrieval, where the input of pitch estimation is typically obtained from the output of music source separation, as shown in Figure 1(a). Consequently, various studies have aimed to simultaneously tackle these two tasks, leveraging on their mutually beneficial relationship. One line of studies (e.g., DNN+UPDUDP [15] and HR-ED [16]) uses *pipeline methods*, as shown in Figure 1(a). However, these methods train MSS and PE models independently on different music datasets, leading to a mismatch between the data distributions at training and testing time. This mismatch limits the improvement of pitch estimation from mixture music. Another line of studies (e.g., HWJH [23], HS-W$_p$ [44], and S→P→S→P [28]) employs *joint learning methods*, as shown in Figure 1(b). Although these joint models combine MSS and PE tasks by summing the respective losses, the distinct objectives of each task pose a challenge to achieve simultaneous improvements. Moreover, these models are often designed for specific tasks, lacking scalability, thus limiting

Unpublished working draft. Not for distribution.

ACM MM, 2024, Melbourne, Australia

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

their ability to improve performance even when better MSS or PE models become available. Although the above studies have tried to perform MSS and PE tasks simultaneously, there are still two major challenges have to be addressed as described below.

Challenge 1: Lack of labeled data. The field of music information retrieval suffers from a large scarcity of annotated datasets, particularly due to the laborious and professional nature of obtaining target sources and corresponding pitches, even for experienced musicians. As a result, only a small amount of music datasets (e.g., MIR-1K [22] and MedleyDB[1] [4]), offer both target sources and corresponding pitches. But the total duration of MIR-1K is only 2.25 h, and the total duration of MedleyDB is 3.21 h. We call such kind of dataset as *fully-labeled dataset*. In contrast, datasets specific to MSS or PE are much larger than the fully-labeled dataset. For example, MUSDB18 [50] is a music source separation dataset and the total duration of MUSDB18 is about 5 times as that of MIR-1K. While MIR_ST500 [58] is a dataset for pitch estimation from mixture music and the total duration of MIR_ST500 is about 14 times as that of MIR-1K. We call such kind of datasets as *single-labeled dataset*. The small amount of fully-labeled datasets prevent models from effectively learning the relationship between MSS and PE tasks.

Challenge 2: Joint learning optimization. Existing joint learning methods (e.g., HS-W$_p$ [44] and S→P→S→P [28]) design joint models and perform joint learning by simply summing up the losses of both tasks. However, these methods do not address the following problems: (i) *Error propagation.* Due to the cascade relationship between MSS and PE, poor predictions of music source separation can lead to poor pitch estimation results. This error propagation problem is critical in joint learning of both tasks. (ii) *Misalignment between different objectives.* The objectives of MSS and PE differ from each other. As a result, simply adding the losses of these two tasks cannot guarantee simultaneous improvements in MSS and PE. We believe that existing joint learning methods failed to align different objectives because they treated all samples equally. These two problems make the joint learning of MSS and PE challenging.

To address these challenges, we propose a Model-Agnostic Joint Learning (MAJL) framework for music source separation and pitch estimation. Our framework can adopt existing MSS and PE models and improve the performance of both tasks when better models become available. MAJL contains a two-stage training method and the Dynamic Weights on Hard Samples (DWHS), which is designed to address Challenge 1 and Challenge 2, respectively.

To address Challenge 1, we design a two-stage training method to leverage large single-labeled datasets. This method comprises an initialization stage (Stage I) and a semi-supervised training stage (Stage II). In Stage I, the model is trained using the available fully-labeled data, then utilizing this trained model to generate pseudo labels and corresponding confidence values for the single-labeled data. The confidence value reflects the quality of pseudo labels. In Stage II, we retrain the model from scratch, using the music data that consists of fully-labeled data, single-labeled data, and pseudo labels generated during Stage I. Additionally, a threshold-based filter is applied to exclude low-confidence single-labeled data, ensuring the quality of pseudo labels. This two-stage training method effectively leverages extensive single-labeled datasets and high-quality pseudo labels, effectively addressing the lack of labeled data.

For Challenge 2, we design a dynamic weighting method called Dynamic Weights on Hard Samples (DWHS), to solve the problems of error propagation and misalignment between different objectives in joint learning. Addressing the error propagation problem entails identifying the module making poor predictions in our framework. Furthermore, to solve the problem of misalignment, we need to identify hard samples for both the MSS and PE tasks. The DWHS entails extracting pitch results from target and predicted sources, termed *target_source2Pitch* and *predicted_source2Pitch*, respectively. By comparing these pitch results, we not only identify modules producing poor predictions but also identify hard samples for both tasks. This allows us to allocate appropriate weights to hard samples across tasks, thereby mitigating error propagation and aligning different objectives within joint learning.

In this paper, our contributions are summarized as follows:

- We propose a Model-Agnostic Joint Learning (MAJL) framework for music source separation and pitch estimation. MAJL is a generic framework, which can further improve the performance of both tasks when better MSS or PE models are available. Moreover, our framework outperforms previous methods, leading to significant improvements in the performance of both tasks.
- We design a two-stage training method to solve the lack of labeled data. The two-stage training method combines music source separation and pitch estimation tasks at the data aspect. By leveraging large single-labeled datasets, our method extends fully-labeled data, allowing the model to better learn the relationship between both tasks.
- To address the challenge of joint learning optimization, we design a novel dynamic weighting method, named Dynamic Weights on Hard Samples (DWHS). The DWHS can handle error propagation and misalignment between different objectives by identifying hard samples and setting appropriate weights for these samples.

2 RELATED WORK

Music Source Separation (MSS) is a crucial task in MIR, aiming to isolate individual sources from mixture music. Many deep learning methods have been proposed for MSS, generally categorized into common models and side-information informed models.

Common models operate solely on hidden features extracted from the time or frequency domains. For example, Spleeter [21], U-Net [29], CWS-PResUNet [40] and ResUNetDecouple+ [36] predict target sources using frequency domain features. In contrast, Demucs [10], Wave-U-Net [57] and its follow-ups [41, 47] leverage time domain features. Other methods such as KUIELAB-MDX-Net [33], Hybrid Demucs [9] and HT Demucs [53] fuse both domain features to enhance the performance of MSS. In contrast, the side information informed models use additional information, such as lyrics, pitches, or spatial information to improve MSS. For example, JOINT3 [54] employs phoneme-level lyrics alignment, while Soundprism [12, 14] and SPAIN-NET [48] leverage pitches and spatial

[1]The MedleyDB dataset comprises a total duration of 5.56 h and encompasses a wide range of musical instruments. Within this dataset, a subset of 3.21 h contains clean vocals along with their corresponding pitches. Notably, the duration of recordings for each other instrument is less than 1 h. Thus, we focus on clean vocals here.

information, respectively. These methods exclusively address the MSS task, which do not support simultaneous PE task.

Pitch Estimation (PE) is a fundamental task in MIR, aimed at extracting fundamental frequency ($f0$). PE can be broadly classified into two sub-tasks: PE from clean music and PE from mixture music.

For clean music, existing methods encompass heuristic-based and data-driven methods. Heuristic-based methods like ACF [13], YIN [8], SWIPE [5], and pYIN [42] leverage candidate-generating functions to predict pitches. Conversely, data-driven methods, including CREPE [32], DeepF0 [56], and HARMOF0 [61], rely on supervised training of models for PE. While these methods achieve accurate pitch results from clean music, their performance is constrained when applied to mixture music due to the presence of other existing sources. For mixture music, existing methods comprise pipeline and end-to-end methods. Pipeline methods involve utilizing MSS models (e.g., Spleeter [21] and U-Net [29]) to extract target sources from the mixture music, and then using PE models to predict corresponding pitches. However, a mismatch between the data distributions at training and testing times often limits the performance of PE from mixture music. End-to-end methods (e.g., DSM-HCQT [3], CNN-Raw [11], and JDC [37]) are designed to directly predict pitches from mixture music. Nevertheless, these methods encounter performance limitations as a result of the presence of other sources in mixture music.

3 PROBLEM FORMULATION

In this section, we formulate the problems of Music Source Separation (MSS) and Pitch Estimation (PE), along with a previously proposed naive joint learning method referred to as the Joint Cascade Framework (JCF) in this paper.

Music Source Separation (MSS). The task of MSS is to extract a target source from mixture music signals. The mixture music signals can be represented as either the raw audio waveform x or its corresponding spectrogram $X_{T\times F}$, where T is the number of audio frames and F is the number of frequency bins. It should be noted that the spectrogram is computed using the short-time Fourier transform (STFT) as a feature representation of the original music signal. The output of MSS is the target source s, and the spectrogram of target source is represented as $S_{T\times F}$. Thus, the MSS task can be formulated as $\mathcal{F}_{mss} : x(X_{T\times F}) \rightarrow s$.

Pitch Estimation (PE). The task of PE aims to estimate the pitch sequence of clean music from a raw audio waveform or its spectrogram representation. Following previous studies (e.g., 360 in CREPE [32] and 352 in HARMOF0 [61]), each pitch is typically represented as a N-dimensional one-hot vector y. As a result, the output of PE is a sequence of pitch vectors $Y_{T\times N}$, where N is the number of pitch values. Furthermore, the input of PE is typically obtained from the output of MSS. Thus, the PE task can be formulated as $\mathcal{F}_{pe} : s(S_{T\times F}) \rightarrow Y_{T\times N}$.

Joint Cascade Framework (JCF). The JCF is designed to leverage the cascade relationship between MSS and PE, enabling joint learning of both tasks. It (cf. Figure 3) comprises a Music Source Separation Module (MSS Module) and a Pitch Estimation Module (PE Module). Firstly, the features of mixture music are input into the MSS Module to obtain predicted sources. Then, the PE Module extracts corresponding pitches from the predicted sources.

In line with previous studies [29, 32, 36, 59, 61], the training of JCF involves using the Mean Absolute Error (MAE) loss for MSS and the Binary Cross Entropy (BCE) loss for PE, respectively. Therefore, the loss function for MSS is defined as:

$$\mathcal{L}_{mss}(s, \hat{s}) = \sum_{i=0}^{L} |s_i - \hat{s_i}| \tag{1}$$

where s is the target sources, $\hat{s}$ is the predicted sources and L is the length of mixture music. And the loss function for PE is defined as:

$$\mathcal{L}_{pe}(y, \hat{y}) = -\sum_{i=0}^{N} (y_i \log \hat{y_i} + (1 - y_i) \log(1 - \hat{y_i})) \tag{2}$$

where N is the number of pitch values, y is the ground truth of pitch results and $\hat{y}$ is the predicted pitch value. Thus, the total loss for naive joint learning of MSS and PE is:

$$\mathcal{L}_{total} = \mathcal{L}_{mss} + \mathcal{L}_{pe} \tag{3}$$

The JCF is unable to solve the two challenges mentioned in Section 1. Therefore, we propose our Model-Agnostic Joint Learning (MAJL) framework for both tasks, building upon and extending the JCF.

4 METHOD

As shown in Figure 2, the Model-Agnostic Joint Learning (MAJL) framework contains two important components: two-stage training method and Dynamic Weights on Hard Samples (DWHS). Besides, the music source separation module (MSS Module) and pitch estimation module (PE Module) within MAJL can be easily replaced with existing MSS and PE models.

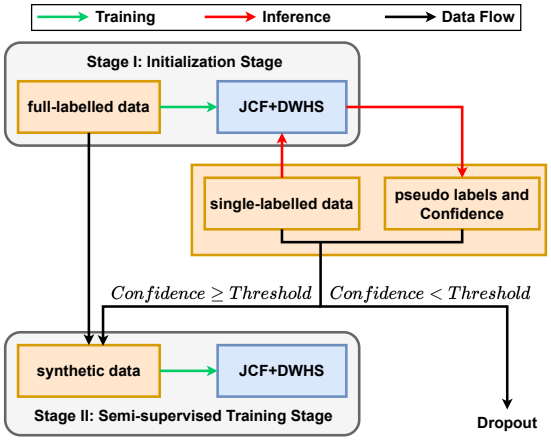


Figure 2: The overall structure of Model-Agnostic Joint Learning (MAJL) framework. Details of JCF and DWHS are shown in Figure 3. The synthetic data contains fully-labeled data, single-labeled data with generated pseudo labels.

4.1 Two-Stage Training Method

To address the limited availability of fully-labeled datasets and leverage large single-labeled datasets, we design a two-stage training method within our framework. This method comprises an initialization stage (Stage I) and a semi-supervised training stage (Stage II), as shown in Figure 2.

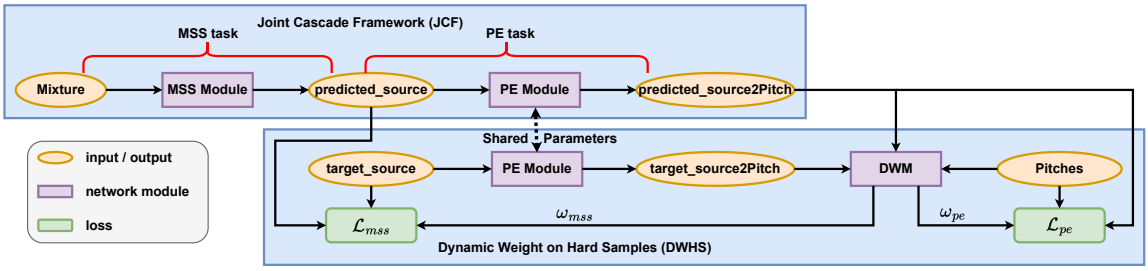


Figure 3: Details of JCF and DWHS. The MSS Module and the PE Module used existing MSS and PE models respectively. The details of Dynamic Weight Module (DWM) are shown in Section 4.2.2.

Initialization Stage (Stage I): During Stage I, our framework is trained using fully-labeled music data (e.g., MIR-1K or MedleyDB). Then the trained framework is employed to generate pseudo labels for target sources or corresponding pitches. Additionally, we compute confidence values for each pitch result and each frame of target sources. For predicted pitches, a pitch value is considered present when $max(\hat{y}) \geq 0.5$; otherwise, a pitch value is considered absent. Then the confidence value (confi) is defined as:

$$\text{confi} = \begin{cases} max(\hat{y}) & max(\hat{y}) \geq 0.5 \\ 1 - max(\hat{y}) & max(\hat{y}) < 0.5 \end{cases} \tag{4}$$

It should be noted that this confidence value can also be applied to predicted sources due to the inherent cascade relationship between music source separation and pitch estimation.

To maintain a consistent format for confidence values across different dataset types (fully-labeled and single-labeled datasets), we set the confidence values of true labels to 1. Therefore, for fully-labeled datasets (e.g., MIR-1K and MedleyDB), the confidence values of MSS (confi_{mss}) and PE (confi_{pe}) are defined as:

$$\text{confi}_{mss} = 1 \quad \text{confi}_{pe} = 1 \tag{5}$$

Then, for MSS datasets (e.g., MUSDB18), which belongs to single-labeled datasets, the confidence values of MSS (confi_{mss}) and PE (confi_{pe}) are defined as:

$$\text{confi}_{mss} = 1 \quad \text{confi}_{pe} = \text{confi} \tag{6}$$

While for PE datasets (e.g., MIR_ST500), which also belongs to single-labeled datasets, the confidence values of MSS (confi_{mss}) and PE (confi_{pe}) are defined as:

$$\text{confi}_{mss} = \text{confi} \quad \text{confi}_{pe} = 1 \tag{7}$$

Semi-supervised Training Stage (Stage II): After the initialization stage, we obtain pseudo labels and confidence values from single-labeled datasets. We then combine fully-labeled music data, single-labeled music data, and pseudo-labels to create a synthetic dataset. To filter pseudo-labels from single-labeled music data, we set a threshold (th). The detailed filtering process involves applying weights for MSS and PE in the loss computation. Subsequently, we retrain our framework from scratch using the synthetic dataset. Thus, the loss function of stage II is written as:

$$\mathcal{L}_{total} = \text{confi}_{mss} \times \mathcal{L}_{mss} + \text{confi}_{pe} \times \mathcal{L}_{pe} \tag{8}$$

Here, confi_{mss} equals 1 if $\text{confi}_{mss} \geq th$, and 0 otherwise. confi_{pe} is calculated using the same way as confi_{mss}.

4.2 Dynamic Weights on Hard Samples (DWHS)

4.2.1 Analysis of Different Cases in DWHS. The naive joint learning method can not ensure simultaneous improvements in both tasks due to the problems of error propagation and misalignment between distinct objectives. To address these problems concurrently, we should identify hard samples and assign appropriate weights to each sample in both tasks. This can be achieved by comparing the predicted pitches from target sources with those from predicted sources (c.f. Figure 3). By this comparison, we can determine which module is delivering poor predictions and identify samples that are hard for either MSS or PE. A comprehensive analysis of different cases arising from this comparison is summarized in Table 1. Detailed explanations of each case in Table 1 are provided as follows.

For *Case 1*, both the predicted pitches from predicted sources and those from target sources are correct. This result indicates that there is no issue with the MSS Module, the PE Module or the quality of data. For *Case 2*, the predicted pitches from predicted sources are correct, while those from target sources are incorrect. This result indicates the presence of noisy pitch labels in the data. To mitigate the impact of noisy labels, the weights of such samples for PE should be within the range of 0 to 1. For *Case 3*, the predicted pitches from predicted sources are incorrect, while those from target sources are correct. This result indicates that the predicted sources are quite different from the original target sources, making the data hard for the MSS. To emphasize learning on hard samples, the weights assigned to these samples in the MSS should be greater than 1. For *Case 4*, both the predicted pitches from target sources and those from predicted sources are incorrect. This result highlights poor predictions by the PE Module, indicating that the music data is hard for the PE. To emphasize learning on hard samples, the weights assigned to these samples in the PE should be greater than 1.

4.2.2 Module of DWHS. By leveraging the above analysis, we can assign different weights to each sample based on identified cases, thereby aligning the focus of two tasks during joint learning. The most direct method involves setting different weights for different cases as outlined in Table 1. However, this approach incurs high training costs due to the difficulty of manually determining proper weights for each case. Therefore, we introduce the DWHS, which automatically extracts the appropriate weights for different cases.

The DWHS utilizes a simple network called the Dynamic Weight Module (DWM) to determine dynamic weights for the MSS and PE tasks under different cases. As illustrated in Figure 4, the inputs for the DWM consist of *predicted_source2Pitch*, *Pitches*, and *target_source2Pitch* from Figure 3, maintaining the same format as

Table 1: Analysis for different cases in DWHS and the weights that should be set for different cases by the DWHS.

Case	predicted_source2Pitch	target_source2Pitch	Analysis			The Weights Should be Assigned	
			MSS Module	PE Module	Data	ω_{mss}	ω_{pe}
1	Correct	Correct	✓	✓	✓	$\omega_{mss} = 1$	$\omega_{pe} = 1$
2	Correct	Incorrect	✓	✓	×	$\omega_{mss} = 1$	$0 \leq \omega_{pe} < 1$
3	Incorrect	Correct	×	✓	✓	$\omega_{mss} \geq 1$	$\omega_{pe} = 1$
4	Incorrect	Incorrect	✓	×	✓	$\omega_{mss} = 1$	$\omega_{pe} \geq 1$

$Y_{T\times N}$. These inputs are concatenated and passed through two CNN layers with ReLU activation, configured as depicted in Figure 4 with 3×3 kernels. Following the CNN layers, there is a flatten layer, a fully connected layer with ReLU activation, and finally, a fully connected layer with sigmoid activation that generates dynamic weights for both tasks. This process enables dynamic assignment of weights without the need for manual specification of specific weights. Specifically, the weights (ω_{mss} and ω_{pe}) corresponding to different cases outlined in Table 1 are automatically extracted by the Dynamic Weight Module (DWM) of the DWHS.

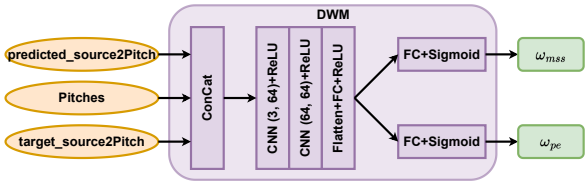


Figure 4: The model structure of dynamic weight module (DWM) in Dynamic Weights on Hard Samples (DWHS).

4.2.3 Loss of DWHS. To ensure the weights extracted by DWM for different cases align with the specified weights shown in Table 1, we design the loss function for the DWM of DWHS to address four specific cases. For *Case 1*, the Mean Absolute Error (MAE) loss is employed to ensure the predicted weights are around 1. Then the loss function for DWM of DWHS in Case 1 is defined as:

$$\mathcal{L}_{dwhs\_1} = |\omega_{mss} - 1| + |\omega_{pe} - 1| \tag{9}$$

For *Case 2*, the MAE loss is utilized to ensure the predicted weights of MSS are around 1. For the predicted weights of PE, the Bayesian Personalized Ranking (BPR) loss [52] is employed to ensure lower weights for noisy music data. Then the loss function for DWM of DWHS in Case 2 is defined as:

$$\mathcal{L}_{dwhs\_2} = |\omega_{mss} - 1| - \ln \sigma(1 - \omega_{pe}) \tag{10}$$

For *Case 3*, the BPR loss is used to ensure hard samples for MSS receive higher weights. Simultaneously, the MAE loss is used to ensure the predicted weights of PE are around 1. Then the loss function for DWM of DWHS in Case 3 is defined as:

$$\mathcal{L}_{dwhs\_3} = -\ln \sigma(\omega_{mss} - 1) + |\omega_{pe} - 1| \tag{11}$$

For *Case 4*, the MAE loss is employed to ensure the predicted weights of MSS are around 1. Additionally, the BPR loss is used to ensure hard samples for PE receive higher weights. Then the loss function for DWM of DWHS in Case 4 is defined as:

$$\mathcal{L}_{dwhs\_4} = |\omega_{mss} - 1| - \ln \sigma(\omega_{pe} - 1) \tag{12}$$

Thus, the total loss of DWHS is written as:

$$\mathcal{L}_{dwhs} = \mathcal{L}_{dwhs\_1} + \mathcal{L}_{dwhs\_2} + \mathcal{L}_{dwhs\_3} + \mathcal{L}_{dwhs\_4} \tag{13}$$

It is important to note that if there is no music data belonging to a specific case, the loss function for that case is automatically set to 0. For example, if there are no music data belonging to Case 1, then the $\mathcal{L}_{dwhs\_1}$ becomes 0.

Thus, with the DWHS, the loss function of stage I is written as:

$$\mathcal{L}_{total} = \omega_{mss} \times \mathcal{L}_{mss} + \omega_{pe} \times \mathcal{L}_{pe} + \mathcal{L}_{dwhs} \tag{14}$$

And the loss function of stage II is written as:

$$\mathcal{L}_{total} = \text{confi}_{mss} \times \omega_{mss} \times \mathcal{L}_{mss} + \text{confi}_{pe} \times \omega_{pe} \times \mathcal{L}_{pe} + \mathcal{L}_{dwhs} \tag{15}$$

where confi_{mss} and confi_{pe} are the same as those in Eq. 8.

The above two-stage training method and Dynamic Weights on Hard Samples are two important components of MAJL aimed at addressing the **lack of labeled data (Challenge 1)** and the **joint learning of optimization (Challenge 2)**, respectively. The effectiveness of MAJL and each component is evaluated in Section 6.

5 EXPERIMENTAL SETUP

Datasets. We evaluate our framework using four public datasets: MIR-1K [22], MedleyDB [4], MIR_ST500 [58], and MUSDB18 [50]. MIR-1K and MedleyDB provide both mixture and clean vocal tracks, along with pitch labels for vocal parts, making them fully-labeled datasets. In contrast, MIR_ST500 and MUSDB18 provide PE and MSS labels, respectively, classifying them as single-labeled datasets. It should be noted that all pitch labels are transformed into frequency bins represented in Hz format the same as MIR-1K.

Evaluation Metrics. Following previous studies, we use four metrics to evaluate our framework. For MSS, we use Signal-to-Distortion Ratio (SDR) [36], Global Normalized Signal-to-Distortion Ratio (GNSDR) [22] to evaluate the quality of predicted sources. A higher SDR or a higher GNSDR indicates better separation results, and vice versa. For PE, we use Raw Pitch Accuracy (RPA) and Raw Chroma Accuracy (RCA) [32] to evaluate the accuracy of predicted pitches.

Implementation Details. The raw audio is sampled at 16kHz and then transformed into spectrograms using the short-time Fourier transform (STFT) with a Hann window size of 2048 and a hop length of 320 (20ms). During the training of MAJL, we use a batch size of 16 and the Adam optimizer [34]. The learning rate is initialized to 0.001 and then reduced by 0.98 of the previous learning rate every 10 epochs. For MIR-1K [22] and MedleyDB [4] datasets, we randomly split these datasets into training (80%) and testing (20%) sets. The splitting way for MIR_ST500 and MUSDB18 datasets is introduced in [58] and [50], respectively. During experiments, we only consider the target source of vocals due to the lack of fully-labeled data from other sources such as bass and drums.

6 EXPERIMENTAL RESULTS

In this section, we present the experimental results of our framework to show the superiority of MAJL. We firstly compare MAJL

Table 2: Performance comparison for MSS and PE tasks on the MIR-1K [22] and MedleyDB [4] datasets. "Extra" indicates the extra single-labeled music data used at the training time. "Both" means MUSDB18 and MIR_ST500.

Methods	Extra	MIR-1K				MedleyDB			
		MSS		PE (%)		MSS		PE (%)	
		SDR	GNSDR	RPA	RCA	SDR	GNSDR	RPA	RCA
End-to-End Methods									
CNN-Raw [11]	✗	——	——	81.70	90.90	——	——	64.33	66.42
JDC [37]	✗	——	——	87.47	88.00	——	——	69.58	77.15
Pipeline Methods w/i CREPE [32]									
U-Net [29]	✗	11.43	8.48	89.28	90.41	5.06	8.75	72.65	74.98
ResUNetDecouple+ [36]	✗	12.06	9.13	91.40	92.07	5.54	10.31	74.62	76.29
Pipeline Methods w/i HARMOF0 [61]									
U-Net [29]	✗	11.43	8.48	87.95	88.57	5.06	8.75	71.24	73.78
ResUNetDecouple+ [36]	✗	12.06	9.13	90.21	90.61	5.54	10.31	73.38	75.90
Joint Learning Methods									
HS-W$_p$ [44]	✗	9.80	6.87	85.04	85.32	4.32	7.56	68.44	70.03
S→P→S→P [28]	✗	11.70	8.72	86.62	86.94	5.14	9.00	70.83	73.79
MAJL-Stage I	✗	12.33	9.36	93.17	93.65	6.04	11.12	76.07	78.28
MAJL	MUSDB18	12.81	9.86	93.38	93.88	6.91	11.87	76.91	79.43
MAJL	MIR_ST500	12.55	9.59	93.67	94.08	6.39	11.43	77.78	80.11
MAJL	Both	**12.98**	**9.99**	**94.11**	**94.38**	**7.18**	**12.14**	**78.38**	**83.21**

with baselines on different datasets in Section 6.1. Then we explore the generality of MAJL through an investigation of its various modules in Section 6.2. Following these experiments, we visualize and analyze the weights of DWHS to understand the effectiveness of DWHS in Section 6.3. Finally, a study is conducted to investigate the threshold (th) used in the two-stage training method in Section 6.4.

6.1 Overall Performance

In this experiment, we conduct a comprehensive comparison of MAJL with several baselines, encompassing End-to-End methods, pipeline methods, and joint learning methods. We evaluate the performance of MAJL on both fully-labeled datasets and single-labeled datasets. These experimental results not only demonstrate the effectiveness of MAJL in joint learning of both tasks, but also highlight its superiority in either the MSS task or the PE task.

6.1.1 Results on Fully-labeled Dataset. To show the superiority of our framework, we perform a comparison with several baselines. For the MSS and PE modules, we choose ResUNetDecouple+ and CREPE, respectively, since they achieve the best performance among all other MSS and PE modules, as shown in Table 5.

Our framework shows the effectiveness in both tasks, achieving state-of-the-art performance for both tasks, as summarized in Table 2. **(i):** Compared to the End-to-End methods, our framework tackles both tasks simultaneously, leading to significant improvement in the PE task. **(ii):** Moreover, our framework outperforms existing joint learning and pipeline methods. Specifically, MAJL-Stage I outperforms the previous best model (Pipeline method with ResUNetDecouple+ and CREPE) by 0.27 in SDR and 1.77% in RPA on the MIR-1K dataset. These results demonstrate that our DWHS is effective in enhancing the performance of both tasks. **(iii):** Furthermore, by incorporating additional single-labeled music data, the performance of both tasks is further improved. For example, MAJL demonstrates a 0.65 improvement in SDR and 0.94% in RPA over MAJL-Stage I when incorporating these additional datasets (MUSDB18 and MIR_ST500) on the MIR-1K dataset, validating the effectiveness of our two-stage training method. **(iv):** Besides, the experimental results on the MedleyDB dataset is similar with those on the MIR-1K dataset, showing the effectiveness of MAJL. Further analysis of the DWHS and the two-stage training method of our framework is described in Section 6.3 and Section 6.4, respectively.

6.1.2 Results on Single-labeled Datasets. To further validate the effectiveness of MAJL in enhancing both tasks, we conduct an evaluation on the test sets of single-labeled datasets. The model under evaluation in this experiment is the same as the one discussed in Section 6.1.1. In the Stage II of training MAJL, additional single-labeled music data from both MUSDB18 and MIR_ST500 is utilized. Additionally, the test sets of the MSS and PE tasks are derived from MUSDB18 and MIR_ST500, respectively. The results on MUSDB18 and MIR_ST500 are summarized in Table 3 and Table 4, respectively.

Table 3: Performance comparison for the music source separation task on the test set of MUSDB18.

Methods	SDR	GNSDR
U-Net [29]	6.72	13.38
HT Demucs [53]	7.93	14.58
Hybrid Demucs [9]	8.13	14.96
CWS-PResUNet [40]	8.92	15.49
ResUNetDecouple+ [36]	8.96	15.59
KUIELAB-MDX-Net [33]	9.00	15.64
MAJL (Stage I with MedleyDB)	9.46	15.94
MAJL (Stage I with MIR-1K)	**10.13**	**16.51**

The results indicate that our framework effectively learns the relationship between both tasks, enhancing the performance of each individual task. The results on the MUSDB18 dataset, presented in Table 3, show that our framework achieves state-of-the-art performance for the MSS task. Specifically, MAJL trained on the MIR-1K dataset in the Stage I outperforms the previous best MSS model

Table 4: Performance comparison for the pitch estimation from mixture music on the test set of MIR_ST500.

Methods	RPA(%)	RCA(%)
CNN-Raw [11]	76.58	77.26
JDC [37]	80.09	80.38
U-Net [29] with CREPE [32]	81.61	82.36
ResUNetDecouple+ [36] with CREPE [32]	81.96	82.43
U-Net [29] with HARMOF0 [61]	82.00	82.35
ResUNetDecouple+ [36] with HARMOF0 [61]	82.78	82.99
MAJL (Stage I with MedleyDB)	83.44	83.91
MAJL (Stage I with MIR-1K)	**84.49**	**85.26**

(KUIELAB-MDX-Net) by 1.13 in SDR. This result illustrates our framework has the capability to effectively leverage the PE task to improve the performance of the MSS task. Similarly, the results on the MIR_ST500 dataset, as shown in Table 4, demonstrate that our framework achieves state-of-the-art performance for the pitch estimation from mixture music. In particular, MAJL trained on the MIR-1K dataset in the Stage I outperforms the previous best method (ResUNetDecouple+ with HARMOF0) by 1.71% in RPA. This result indicates that our framework can effectively leverage the MSS task to enhance the performance of the PE task. In summary, these results highlight that our framework effectively leverages the mutually beneficial relationship between both tasks, thereby improving their individual performances.

6.2 Experiments With Different Modules

In this experiment, we investigate the effect of using different modules in our framework to demonstrate its generality. Our framework can employ various model architectures for the MSS Module and the PE Module. Therefore, we evaluate the performance of our framework using different combinations of these modules, including ResUNetDecouple+ [36], U-Net [29], HARMOF0 [61] and CREPE [32]. For this experiment, we utilize the MIR-1K dataset in Stage I because MAJL-Stage I, trained with the MIR-1K dataset, demonstrates superior performance on the MUSDB18 and MIR_ST500 datasets, thereby enabling the generation of better pseudo-labels. The results obtained with different combinations of these MSS and PE modules are summarized in Table 5. According to the results in this table, we can find three main observations as follows.

Firstly, this experiment shows the great generality of our framework. As shown in Table 5, MAJL consistently outperforms both the corresponding pipeline and naive joint learning methods across all module combinations. Particularly, when utilizing U-Net and CREPE, our framework achieves an improvement of 0.94 in SDR and 2.95% in RPA compared to the corresponding pipeline method. Similarly, our framework outperforms the corresponding naive joint learning method by 0.59 in SDR and 2.54% in RPA. These results validate the model-agnostic nature of our framework, showing its consistent performance enhancement across various music source separation and pitch estimation models.

Secondly, the two-stage training method and DWHS are robust with different MSS and PE modules. Specifically, MAJL using ResUNetDecouple+ and CREPE achieves the best performance. Notably, compared to MAJL-Stage I, MAJL achieves improvements of 0.65 in SDR and 0.94% in RPA, demonstrating the effectiveness of the two-stage training method in leveraging large single-labeled music data. Moreover, MAJL-Stage I outperforms the naive joint learning method by 0.42 in SDR and 1.29% in RPA, indicating that the DWHS effectively aligns different objectives and enhances the performance of both tasks. Furthermore, similar performance trends are observed with MAJL and MAJL-Stage I utilizing alternative MSS and PE modules. These results demonstrate that both the two-stage training method and the DWHS are model-agnostic and robust, further confirming the model-agnostic nature of our framework.

Table 5: Performance with different combinations of modules on the MIR-1K dataset. U, R, H, C represents the model architecture U-Net [29], ResUNetDecouple+ [36], HARMOF0 [61] and CREPE [32], respectively. MAJL here uses both MIR_ST500 and MUSDB18 as single-labeled data.

MSS	PE	Joint Method	MSS		PE(%)	
			SDR	GNSDR	RPA	RCA
U	H	Pipeline	11.43	8.48	87.95	88.57
		Naive Joint Learning	11.05	8.11	88.96	89.20
		MAJL-Stage I	12.05	9.11	90.62	91.57
		MAJL	12.25	9.29	91.09	91.85
R	H	Pipeline	12.06	9.13	90.21	90.61
		Naive Joint Learning	12.04	9.08	91.46	91.61
		MAJL-Stage I	12.28	9.33	92.16	92.76
		MAJL	12.60	9.64	92.51	93.16
U	C	Pipeline	11.43	8.48	89.28	90.41
		Naive Joint Learning	11.78	8.84	89.69	90.44
		MAJL-Stage I	12.16	9.19	91.78	92.40
		MAJL	12.37	9.41	92.23	92.79
R	C	Pipeline	12.06	9.13	91.40	92.07
		Naive Joint Learning	11.91	8.92	91.88	92.15
		MAJL-Stage I	12.33	9.36	93.17	93.65
		MAJL	**12.98**	**9.99**	**94.11**	**94.38**

Lastly, our framework effectively learn the relationship between MSS and PE tasks, making both tasks beneficial for the each other. For example, MAJL using ResUNetDecouple+ and CREPE outperforms MAJL using U-Net and CREPE by 0.61 in SDR and 1.88% in RPA. This improvement arises from the better performance of ResUNetDecouple+ in the MSS task, where the effectiveness of the MSS task is beneficial for the PE task. Similarly, MAJL using ResUNetDecouple+ and CREPE outperforms MAJL using ResUNetDecouple+ and HARMOF0 by 0.38 in SDR and 1.60% in RPA. This is because CREPE performs better than HARMOF0 at the PE task, and the PE task is beneficial for the MSS task. Thus, our framework also has the potential to further improve the performance of both tasks when better MSS and PE models become available.

The above results demonstrate the robustness and great generality of our framework, since our framework achieves the best performance across all module combinations. Moreover, our MAJL framework effectively learns the relationship between MSS and PE tasks, resulting in enhanced performance for both tasks.

6.3 Visualization and Analysis of Dynamic Weights

To provide an intuitive representation of the weights extracted by the DWHS, we visualize the changes in these weights over iterations. In this experiment, we employ ResUNetDecouple+ as the MSS Module and CREPE as the PE Module, consistent with the previous experiment detailed in Section 6.1. In addition, the dynamic weights extracted by Dynamic Weights on Hard Samples (DWHS)

are obtained from our framework, specifically MAJL-Stage I, to exclusively investigate the weight results of DWHS and eliminate potential interference, such as single-labeled music data.

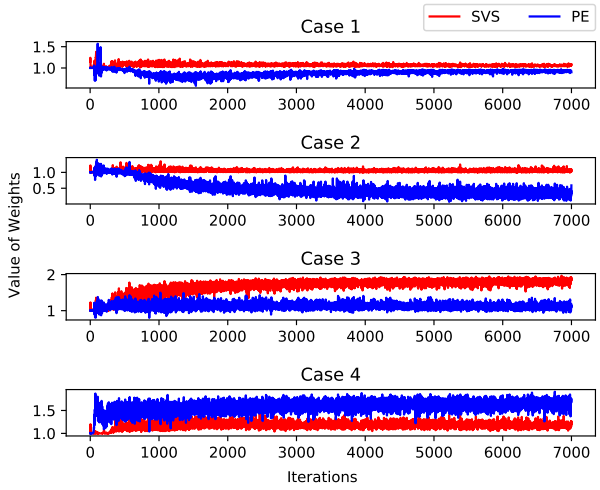


Figure 5: Dynamic weights extracted by the DWHS.

Figure 5 illustrates the dynamic weights extracted by the DWHS. In this figure, we observe that the weight assigned to the PE task for noisy music data is set close to 0, effectively mitigating the negative impact on the PE task. Additionally, for Case 3 and Case 4, the weights assigned to MSS and PE exceed 1, emphasizing the importance of handling hard samples. Other weights are approximately 1. All these weights are automatically set based on the analysis presented in Table 1. Furthermore, the results presented in Table 2 demonstrate that the DWHS method outperforms the corresponding naive joint learning method. These findings highlight that the DWHS method can adaptively determine appropriate weights for both noisy and hard samples, leading to enhanced performance in both MSS and PE tasks.

6.4 Threshold in Two-Stage Training Method

In this experiment, we use the MUSDB18 and MIR_ST500 datasets as single-labeled music data to explore the impact of the threshold (th) used in the two-stage training method. The ResUNetDecouple+ is used as the MSS Module and CREPE is used as the PE Module, consistent with the previous experiment in Section 6.1. The MAJL is initially trained on the MIR-1K dataset in Stage I, as it demonstrates superior performance compared to MAJL trained on the MedleyDB dataset, as evidenced by the results presented in Table 3 and Table 4.

As shown in Figure 6, the results corresponding to different thresholds demonstrate the effectiveness of the two-stage training method, and that the chosen threshold influences the performance of MSS and PE tasks. Specifically, when incorporating both datasets (MUSDB18 and MIR_ST500) to the fully-labeled music data, the performance of both tasks firstly improved and then decreased as the threshold increased. This trend indicates that the quality of pseudo-labels affects the performance of both tasks, with lower-quality pseudo-labels leading to a decrease in the performance of both tasks. In addition, incorporating the MUSDB18 or MIR_ST500 dataset to the fully-labeled music data follows a similar trend in the performance of both tasks. Moreover, when the threshold was set

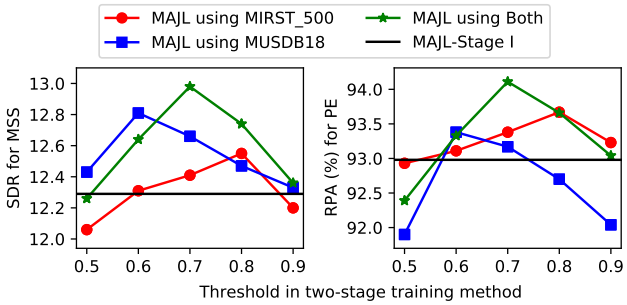


Figure 6: Performance with different values of threshold (th) in two-stage training method on MIR-1K dataset. MAJL-Stage I represents Stage I in the two-stage training method.

to 0.7, adding both datasets (MUSDB18 and MIR_ST500) to the fully-labeled music data achieves the best performance, outperforming MAJL-Stage I by 0.65 in SDR and 0.94% in RPA. These results show that the two-stage training method can leverage large single-labeled music data to enhance the performance of both tasks.

7 FUTURE WORK

Our experiments have primarily focused on vocals, given that vocals are a common target source for both MSS and PE tasks , as highlighted in previous studies [28, 44]. Furthermore, the availability of music data containing other target sources and corresponding pitches was limited. However, it is important to note that our framework is applicable to a variety of musical instruments, including drums, bass and so on.

Expanding our framework to encompass other instruments requires the acquisition or creation of annotated data for both MSS and PE tasks. Overcoming this challenge may involve adapting transfer learning techniques from related domains or exploring further unsupervised training methods. In conclusion, the future of our research holds the potential to adapt and expand our Model-Agnostic Joint Learning (MAJL) framework to encompass a broader range of musical instruments. These directions align with the ongoing evolution of music production techniques and computational audio analysis, making our framework important in advancing the field of music information retrieval.

8 CONCLUSION

In this paper, we have proposed a model-agnostic joint learning (MAJL) framework to address the challenges in joint learning of music source separation and pitch estimation. MAJL is generic for both tasks in music information retrieval, offering the capability to adapt improved models for both tasks, further improving their performance. By designing a two-stage training method and a dynamic weighting method named *Dynamic Weights on Hard Samples* (DWHS), our framework effectively addresses the challenges of the lack of labeled data and the joint learning optimization, respectively. Through leveraging extensive single-labeled music data, MAJL learns the mutually beneficial relationship between music source separation and pitch estimation tasks, leading to improved performance for both tasks. Our experimental results show that the proposed framework achieves a significant improvement in both tasks, with 0.92 in SDR for the music source separation task and 2.71% in RPA for the pitch estimation task.

REFERENCES

[1] S Abdali and Babak NaserSharif. 2017. Non-negative matrix factorization for speech/music separation using source dependent decomposition rank, temporal continuity term and filtering. *Biomedical Signal Processing and Control* (2017), 168–175.

[2] Emmanouil Benetos, Simon Dixon, Zhiyao Duan, and Sebastian Ewert. 2018. Automatic Music Transcription: An Overview. *IEEE Signal Processing Magazine* (2018), 20–30.

[3] Rachel M Bittner, Brian McFee, Justin Salamon, Peter Li, and Juan Pablo Bello. 2017. Deep Salience Representations for F0 Estimation in Polyphonic Music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 63–70.

[4] Rachel M Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Pablo Bello. 2014. Medleydb: A multitrack dataset for annotation-intensive mir research.. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 155–160.

[5] A. Camacho and J. G. Harris. 2008. A sawtooth waveform inspired pitch estimator for speech and music. In *The Journal of the Acoustical Society of America*. 1638–1652.

[6] Ke Chen, Xingjian Du, Bilei Zhu, Zejun Ma, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. 2022. Zero-shot Audio Source Separation through Query-based Learning from Weakly-labeled Data. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 4441–4449.

[7] Kin Wai Cheuk, Keunwoo Choi, Qiuqiang Kong, Bochen Li, Minz Won, Amy Hung, Ju-Chiang Wang, and Dorien Herremans. 2022. Jointist: Joint Learning for Multi-instrument Transcription and Its Applications. *arXiv preprint arXiv:2206.10805* (2022).

[8] Alain De Cheveigné and Hideki Kawahara. 2002. YIN, a fundamental frequency estimator for speech and music. In *The Journal of the Acoustical Society of America*. 1917–1930.

[9] Alexandre Défossez. 2021. Hybrid spectrogram and waveform source separation. *arXiv preprint arXiv:2111.03600* (2021).

[10] Alexandre Défossez, Nicolas Usunier, Léon Bottou, and Francis Bach. 2019. Music source separation in the waveform domain. *arXiv preprint arXiv:1911.13254* (2019).

[11] Mingye Dong, Jie Wu, and Jian Luan. 2019. Vocal Pitch Extraction in Polyphonic Music Using Convolutional Residual Network. In *INTERSPEECH*. 2010–2014.

[12] Zhiyao Duan and Bryan Pardo. 2011. Soundprism: An online system for score-informed source separation of music audio. *IEEE Journal of Selected Topics in Signal Processing* (2011), 1205–1215.

[13] John Dubnowski, Ronald Schafer, and Lawrence Rabiner. 1976. Real-time digital hardware pitch detector. *IEEE Transactions on Acoustics, Speech, and Signal Processing* (1976), 2–8.

[14] Sebastian Ewert, Bryan Pardo, Meinard Muller, and Mark D. Plumbley. 2014. Score-Informed Source Separation for Musical Audio Recordings: An overview. *IEEE Signal Processing Magazine* (2014), 116–124.

[15] Zhe-Cheng Fan, Jyh-Shing Roger Jang, and Chung-Li Lu. 2016. Singing Voice Separation and Pitch Extraction from Monaural Polyphonic Audio Music via DNN and Adaptive Pitch Tracking. In *2016 IEEE Second International Conference on Multimedia Big Data (BigMM)*. 178–185.

[16] Yongwei Gao, Xulong Zhang, and Wei Li. 2021. Vocal melody extraction via hrnet-based singing voice separation and encoder-decoder-based f0 estimation. *Electronics* (2021), 298.

[17] Josh Gardner, Ian Simon, Ethan Manilow, Curtis Hawthorne, and Jesse Engel. 2022. MT3: Multi-Task Multitrack Music Transcription. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[18] Beat Gfeller, Christian Frank, Dominik Roblek, Matt Sharifi, Marco Tagliasacchi, and Mihajlo Velimirović. 2020. SPICE: Self-supervised pitch estimation. In *Proceedings of the IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*. 1118–1128.

[19] Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse H. Engel, Sageev Oore, and Douglas Eck. 2018. Onsets and Frames: Dual-Objective Piano Transcription. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 50–57.

[20] Curtis Hawthorne, Ian Simon, Rigel Swavely, Ethan Manilow, and Jesse Engel. 2021. Sequence-to-sequence piano transcription with transformers. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*.

[21] Romain Hennequin, Anis Khlif, Felix Voituret, and Manuel Moussallam. 2020. Spleeter: a fast and efficient music source separation tool with pre-trained models. *Journal of Open Source Software* (2020), 2154.

[22] Chao-Ling Hsu and Jyh-Shing Roger Jang. 2010. On the Improvement of Singing Voice Separation for Monaural Recordings Using the MIR-1K Dataset. In *Proceedings of the IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*. 310–319.

[23] Chao-Ling Hsu, DeLiang Wang, Jyh-Shing Roger Jang, and Ke Hu. 2012. A tandem algorithm for singing pitch extraction and voice separation from music accompaniment. In *Proceedings of the IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*. 1482–1491.

[24] Po-Sen Huang, Scott Deeann Chen, Paris Smaragdis, and Mark Hasegawa-Johnson. 2012. Singing-voice separation from monaural recordings using robust principal component analysis. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 57–60.

[25] Yun-Ning Hung, Gordon Wichern, and Jonathan Le Roux. 2021. Transcription Is All You Need: Learning To Separate Musical Mixtures With Score As Supervision. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 46–50.

[26] iResearch. 2019. China's Digital Music Market Will Face A New Round of Competition. In *iResearch*. http://www.iresearchchina.com/content/details7_53675.html

[27] Sanjeev N Jain and Chandrashekhar Rai. 2012. Blind source separation and ICA techniques: a review. *International Journal of Engineering Science and Technology* (2012), 1490–1503.

[28] Andreas Jansson, Rachel M. Bittner, Sebastian Ewert, and Tillman Weyde. 2019. Joint Singing Voice Separation and F0 Estimation with Deep U-Net Architectures. In *European Signal Processing Conference (EUSIPCO)*. 1–5.

[29] Andreas Jansson, Eric J. Humphrey, Nicola Montecchio, Rachel M. Bittner, Aparna Kumar, and Tillman Weyde. 2017. Singing Voice Separation with Deep U-Net Convolutional Networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*.

[30] Rainer Kelz, Sebastian Böck, and Gerhard Widmer. 2019. Deep polyphonic adsr piano note transcription. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 246–250.

[31] Jong Wook Kim and Juan Pablo Bello. 2019. Adversarial learning for improved onsets and frames music transcription. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*.

[32] Jong Wook Kim, Justin Salamon, Peter Li, and Juan Pablo Bello. 2018. CREPE: A convolutional representation for pitch estimation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 161–165.

[33] Minseok Kim, Woosung Choi, Jaehwa Chung, Daewon Lee, and Soonyoung Jung. 2021. Kuielab-mdx-net: A two-stream neural network for music demixing. *arXiv preprint arXiv:2111.12203* (2021).

[34] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[35] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley. 2020. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. In *Proceedings of the IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*. 2880–2894.

[36] Qiuqiang Kong, Yin Cao, Haohe Liu, Keunwoo Choi, and Yuxuan Wang. 2021. Decoupling magnitude and phase estimation with deep resunet for music source separation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*.

[37] Sangeun Kum and Juhan Nam. 2019. Joint Detection and Classification of Singing Voice Melody Using Convolutional Recurrent Neural Networks. *Applied Sciences* (2019), 1324.

[38] Liwei Lin, Qiuqiang Kong, Junyan Jiang, and Gus Xia. 2021. A Unified Model for Zero-shot Music Source Separation, Transcription and Synthesis. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 381–388.

[39] Sebastian Lindlahr. 2021. Forecast of Digital Music revenue by segment in the United States from 2017 to 2025. In *Statista*. https://www.statista.com/forecasts/460034/digital-music-revenue-in-the-united-states-forecast

[40] Haohe Liu, Qiuqiang Kong, and Jiafeng Liu. 2021. CWS-PResUNet: Music source separation with channel-wise subband phase-aware resunet. *arXiv preprint arXiv:2112.04685* (2021).

[41] Craig Macartney and Tillman Weyde. 2018. Improved speech enhancement with the wave-u-net. *arXiv preprint arXiv:1811.11307* (2018).

[42] Matthias Mauch and Simon Dixon. 2014. pYIN: A fundamental frequency estimator using probabilistic threshold distributions. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 659–663.

[43] Brian McFee, Colin Raffel, Dawen Liang, Daniel P Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. 2015. librosa: Audio and music signal analysis in python. In *Proceedings of the python in science conference (SciPy)*. 18–25.

[44] Tomoyasu Nakano, Kazuyoshi Yoshii, Yiming Wu, Ryo Nishikimi, Kin Wah Edward Lin, and Masataka Goto. 2019. Joint singing pitch estimation and voice separation based on a neural harmonic structure renderer. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. 160–164.

[45] Alexey Ozerov, Pierrick Philippe, Frdric Bimbot, and Rmi Gribonval. 2007. Adaptation of Bayesian models for single-channel source separation and its application to voice/music separation in popular songs. In *Proceedings of the IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*. 1564–1578.

[46] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019.

Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).

[47] Joaquin Perez-Lapillo, Oleksandr Galkin, and Tillman Weyde. 2020. Improving Singing Voice Separation with the Wave-U-Net Using Minimum Hyperspherical Energy. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 3272–3276.

[48] Darius Petermann and Minje Kim. 2022. Spain-Net: Spatially-Informed Stereophonic Music Source Separation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 106–110.

[49] Colin Raffel, Brian McFee, Eric J Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, Daniel PW Ellis, and C Colin Raffel. 2014. mir_eval: A transparent implementation of common MIR metrics. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 367–372.

[50] Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, and Rachel Bittner. 2017. The MUSDB18 corpus for music separation. https://doi.org/10.5281/zenodo.1117372

[51] Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, Derry FitzGerald, and Bryan Pardo. 2018. An overview of lead and accompaniment separation in music. In *Proceedings of the IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*. 1307–1335.

[52] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. 452–461.

[53] Simon Rouard, Francisco Massa, and Alexandre Défossez. 2023. Hybrid Transformers for Music Source Separation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 1–5.

[54] Kilian Schulze-Forster, Clement S. J. Doire, Gaël Richard, and Roland Badeau. 2021. Phoneme Level Lyrics Alignment and Text-Informed Singing Voice Separation. In *Proceedings of the IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*. 2382–2395.

[55] Jonathon Shlens. 2014. A tutorial on independent component analysis. *arXiv preprint arXiv:1404.2986* (2014).

[56] Satwinder Singh, Ruili Wang, and Yuanhang Qiu. 2021. DeepF0: End-To-End Fundamental Frequency Estimation for Music and Speech Signals. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 61–65.

[57] Daniel Stoller, Sebastian Ewert, and Simon Dixon. 2018. Wave-u-net: A multi-scale neural network for end-to-end audio source separation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*.

[58] Jun-You Wang and Jyh-Shing Roger Jang. 2021. On the Preparation and Validation of a Large-Scale Dataset of Singing Transcription. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 276–280.

[59] Haojie Wei, Xueke Cao, Tangpeng Dan, and Yueguo Chen. 2023. RMVPE: A Robust Model for Vocal Pitch Estimation in Polyphonic Music. In *INTERSPEECH*. 5421–5425.

[60] Haojie Wei, Jun Yuan, Rui Zhang, Yueguo Chen, and Gang Wang. 2023. JEPOO: Highly Accurate Joint Estimation of Pitch, Onset and Offset for Music Information Retrieval. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. 4892–4902.

[61] Weixing Wei, Peilin Li, Yi Yu, and Wei Li. 2022. HarmoF0: Logarithmic Scale Dilated Convolution for Pitch Estimation. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*. 1–6.

[62] Svante Wold, Kim Esbensen, and Paul Geladi. 1987. Principal component analysis. *Chemometrics and intelligent laboratory systems* (1987), 37–52.

[63] Yujia Yan, Frank Cwitkowitz, and Zhiyao Duan. 2021. Skipping the Frame-Level: Event-Based Piano Transcription With Neural Semi-CRFs. *Advances in Neural Information Processing Systems* (2021), 20583–20595.

A RELATED WORK DETAILS

A.1 Music Source Separation

Music source separation (MSS) is a crucial task in music information retrieval (MIR), involving the decomposition of music into its constitutive components, such as isolating vocals, bass, and drums [51]. When focusing specifically on clean vocals and accompaniment without further separating the accompaniment into individual instruments, it becomes a singing voice separation task [45], a universal and specialized form of MSS. Various deep learning methods address the MSS task, broadly categorized into three types: the traditional methods, the deep learning-based methods, and the side information informed methods.

A.1.1 Traditional Methods. Traditional methods are based on digital signal processing and mathematical statistics, include Independent Components Analysis (ICA) [55], Principal Component Analysis (PCA) [62], and Non-negative Matrix Factorization (NMF) [1]. For instance, ICA is initially employed to solve the blind source separation task [27], akin to the MSS task. The PCA-based method [24] utilizes robust principal component analysis to represent accompaniment through a separated low-rank matrix and singing voices through a separated sparse matrix. And the NMF-based methods [1] model each source with a dictionary, capturing source signals within the non-negative span of this dictionary. Although these traditional methods are interpretable and accomplish the music source separation task to some extent, they often lack the ability to distinguish between different instruments.

A.1.2 Deep Learning-Based Methods. Deep learning-based methods fall into three categories: methods in the frequency domain, methods in the time domain, and hybrid methods in both domains. For example, U-Net [29], Spleeter [21], CWS-PResUNet [40], and ResUNetDecouple+ [36] belong to methods in the frequency domain, which process mixture music in the frequency domain (e.g., spectrogram) to predict masks or spectrograms of target sources. Alternatively, Demucs [10], Wave-U-Net [57], and its follow-ups [41, 47] are the methods in the time domain. They directly process raw audio using one-dimensional convolutional networks to predict target sources in the time domain. Additionally, other methods such as KUIELAB-MDX-Net [33], Hybrid Demucs [9], and HT Demucs [53] are hybrid methods in both domains. They combine features from both domains for improved performance of music source separation. However, the above methods focus only on the features extracted from raw audio, ignoring the important role of auxiliary information such as melody, rhythm, lyrics, and so on in the music source separation task.

A.1.3 Side Information Informed Methods. Side information informed methods leverage additional information, such as lyrics, music scores (pitches), or spatial details, to improve the performance of music source separation. For instance, JOINT3 [54] utilizes phoneme-level lyrics alignment to improve the performance of music source separation. SPAIN-NET [48] and Soundprism [12, 14] use music scores and spatial information to enhance the performance of music source separation. However, these methods treat side information as auxiliary features and lack the capability to perform highly related tasks in music information retrieval simultaneously.

A.2 Pitch Estimation

Pitch estimation (PE) is a fundamental task in MIR, playing a crucial role in various downstream applications. Following previous studies [18, 32, 61], we default to using pitch estimation to refer to single pitch estimation (SPE), unless explicitly stated otherwise in this paper. Then we will introduce both single pitch estimation (SPE) and multi-pitch estimation (MPE) tasks in detail.

A.2.1 Single Pitch Estimation (SPE). The SPE task involves predicting no more than one pitch at any timestamp. It can be further categorized into SPE from clean music and SPE from mixture music.

For SPE from clean music, there are two primary methods: the heuristic-based methods and the data-driven methods. The heuristic-based methods, such as ACF [13], YIN [8], SWIPE [5] and pYIN [42], leverage candidate-generating functions to predict pitches. In contrast, the data-driven methods, such as CREPE [32], DeepF0 [56] and HARMOF0 [61], employ supervised training of models for SPE. While these methods achieve a good performance on clean music, they prove to be less effective in the mixture music due to their limited robustness to accompaniments.

For SPE from mixture music, there are mainly two approaches. The first approach is pipeline methods, which involves using MSS models (e.g., Spleeter [21] and U-Net [29]) to extract target sources from the mixture music, and then using PE models to estimate the pitch of target sources. These models are trained separately, resulting in a mismatch between the data distributions at training and testing times, which limits the performance of PE from mixture music. The second approach is end-to-end methods (e.g., DSM-HCQT [3], CNN-Raw [11] and JDC [37]), which are designed to directly estimate pitches from mixture music. However, these methods are also limited in performance due to the presence of other sources in the mixture music.

A.2.2 Multi Pitch Estimation (MPE). The MPE task entails predicting multiple pitches at any timestamp, also known as music transcription in the field of music information retrieval. Given its increased complexity compared to SPE, our focus in this paper is exclusively on the single pitch estimation task. Methods for music transcription are usually classified into two categories: frame-level transcription methods and note-level transcription methods.

The frame-level transcription methods, such as OAF [19], ADSRNet [30] and Non-Saturating GAN [31], employing CNN and LSTM to predict pitches for each frame. These frame-level pitches are then transformed into sequential notes, achieving the MPE task. On the contrary, the note-level transcription methods, such as seq-to-seq [20], MT3 [17] and EBPT [63] treat notes as events, predicting note-level pitch results and achieving the MPE task at the note level. However, the above MPE methods often focus on specific instruments and may perform inadequately on the SPE task, lacking robustness across different instruments.

A.3 Joint Learning For MSS and PE

With the development of joint learning, several research tasks in MIR have shown the potential for mutual improvement through multi-tasks joint learning. Among these tasks, music source separation and pitch estimation are closely related, leading to the

exploration of methods that take advantage of their mutually beneficial relationship. Several existing methods have attempted to exploit the mutually beneficial relationship between music source separation and pitch estimation tasks.

For example, JDC [37] is a joint learning approach that addresses voice detection and pitch estimation. However, it mainly employs voice detection as an auxiliary task for accompaniment processing. Other methods, such as [25] and [6], utilize transcription as an auxiliary task, incorporating joint transcription and source separation training for a limited number of instruments. HS-W$_p$ [44] and Joint-UNet [28] represent joint learning methods specifically designed for music source separation and pitch estimation tasks. Furthermore, models like MSI-DIS [38] and JOINTIST [7] aim to use a unified model for both music source separation and transcription tasks. It is worth noting that these models often require training on datasets containing both pitch labels and target sources. While these existing joint learning approaches provide valuable insights, practical challenges arise when combining music source separation and pitch estimation tasks, especially in scenarios where training data with both pitch labels and target sources is limited.

B EXPERIMENTAL SETUP DETAILS

B.1 Datasets

To compare with previous MSS models and PE models fairly, we use three public datasets for our experiments: MIR-1K [22], MedleyDB [4], MIR_ST500 [58] and MUSDB18 [50].

MedleyDB [4] dataset consists of 122 songs, 108 of the original multi-tracks include melody annotations. Based on the melody definition "The f0 curve of the predominant melodic line drawn from a single source", the melody annotation is the pitch of the stem with the most predominant melodic source. Thus, the MedleyDB dataset has both the target sources and corresponding pitches, making it the fully-labeled dataset. This dataset has various musical instruments, including vocals, guitar, violin, dizi, and so on. The total duration of vocals music data is about 3.21 h, while the total duration of each other instruments is less than 0.69 h. We focus only on vocal music in this paper since the amount of data for each other musical instruments is limited.

MIR-1K [22] dataset is designed for singing voice separation and contains 1000 song clips extracted from 110 karaoke songs sung by researchers from the MIR lab. The dataset provides both the mixture track and the clean vocals track, as well as pitch labels for the vocal parts, making it a fully-labeled dataset. The total length of MIR-1K is 133 minutes, and each clip ranges from 4 to 13 seconds.

MIR_ST500 [58] is a singing voice transcription dataset containing 500 Chinese pop songs. It provides the YouTube URL of the mixture music and the note labels of vocal parts, which can be used for PE from mixture music. Thus, it is considered a single-labeled dataset. This dataset is divided into a train dataset (400 songs) and a test dataset (100 songs), with the total duration of about 32 hours.

MUSDB18 [50] is a dataset of music source separation consisting of 150 full-length music tracks of different European and American genres, such as pop, rap, and heavy metal. Each track is composed of isolated drums, bass, vocals, and other stems. Thus, it is considered a single-labeled dataset. The dataset is divided into a train folder with 100 songs and a test folder with 50 songs, with a total duration of about 10 hours.

B.2 Evaluation Metrics

The following evaluation metrics are used to evaluate the performance of the music source separation and pitch estimation tasks, as described in previous studies on music source separation [22, 36, 45] and pitch estimation [32, 56, 61], respectively. These metrics are computed using the mir_eval [49] library:

Raw Pitch Accuracy (RPA) [32] computes the proportion of melody frames in the reference for which the predicted pitch is within ±50 cents of the ground truth pitch.

Raw Chroma Accuracy (RCA) [32] computes the raw pitch accuracy after mapping the estimated and reference frequency sequences onto a single octave. It measures the raw pitch accuracy ignoring the octave errors.

Signal-to-Distortion Ratio (SDR) [36] measures the quality of the predicted sources with respect to the original target sources. It is defined as follows:

$$SDR(s, \hat{s}) = 10 \times \log_{10} \frac{||s||^2}{||\hat{s} - s||^2} \tag{16}$$

where $\hat{s}$ is the predicted sources and s is the target sources. A higher SDR indicates better separation results, and vice versa. A perfect separation would result in infinite SDR.

Global Normalized Signal-to-Distortion Ratio (GNSDR) [45] is calculated as follows:

$$GNSDR = \frac{\sum_{i=1}^{i=N} l_i NSDR(s, \hat{s}, x)}{\sum_{i=1}^{i=N} l_i} \tag{17}$$

where i is the index of a song, N is the total number of songs, l_i is the length of the ith song, and $NSDR(s, \hat{s}, x)$ is the normalized SDR. The $NSDR(s, \hat{s}, x)$ [45] is defined as follows:

$$NSDR(s, \hat{s}, x) = SDR(s, \hat{s}) - SDR(s, x) \tag{18}$$

where x is the mixture music. The NSDR is the improvement in SDR between the mixture and the predicted sources.

B.3 Implementation Details

The raw audio is sampled at 16kHz and then transformed into a spectrogram using the short-time Fourier transform (STFT) with a Hann window size of 2048 and a hop length of 320 (20ms). We use the librosa [43] and torchlibrosa [35] to perform the audio processing. During training of the model-agnostic joint learning (MAJL) framework, we use a batch size of 16 and the Adam optimizer [34]. The learning rate is initialized to 0.001 and is reduced by 0.98 of the previous learning rate every 10 epochs. In our framework, there are mainly three hyper-parameters, ω_{noise}, $upper\_bound$ and threshold (th). The hyper-parameters $upper\_bound$ and ω_{noise} only used for the naive DWHS in additional methods (Section C), where $upper\_bound$ ranges from 1 to 10 and ω_noise ranges from 0 to 1. While the hyper-parameter threshold (th) is used to filter pseudo labels, ranging from 0.5 to 1.

Each training audio is divided into segments of 2.56 seconds. For the MIR-1K [22] dataset, we randomly split the dataset into training (80%) and testing (20%) sets. We treat MIR_ST500 [58] and MUSDB18 [50] datasets as single-labeled datasets since they

Table 6: Analysis for different cases in DWHS and corresponding weights calculated by the naive DWHS. The $\hat{y}_t$ is defined as $max(y) \times max(\hat{y}) + (1 - max(y)) \times (1 - max(\hat{y}))$. The $upper\_bound$ and ω_{noise} are two hyper-parameters for the naive DWHS. The $Clamp(1/\hat{y}_t, 1, upper\_bound)$ represents restrict $1/\hat{y}_t$ to be between 1 and $upper\_bound$.

Case	predicted_source2Pitch	target_source2Pitch	Analysis			Naive DWHS	
			MSS Module	PE Module	Data	ω_{mss}	ω_{pe}
1	Correct	Correct	✓	✓	✓	1	1
2	Correct	Incorrect	✓	✓	×	1	$0 \leq \omega_{noise} < 1$
3	Incorrect	Correct	×	✓	✓	$Clamp(1/\hat{y}_t, 1, upper\_bound)$	1
4	Incorrect	Incorrect	✓	×	✓	1	$Clamp(1/\hat{y}_t, 1, upper\_bound)$

lack the target sources and pitch labels, respectively. The splitting way for MIR_ST500 and MUSDB18 datasets is introduced in [58] and [50], respectively. During experiments, we only consider the target source of vocals due to the lack of fully-labeled data from other sources such as bass and drums.

B.4 Comparison Systems

We compare MAJL with several existing methods, including end-to-end, pipeline, and joint learning methods. For end-to-end methods, we chose CNN-Raw [11] and JDC [37], as they achieved the best performance in the pitch estimation task among all end-to-end methods. For pipeline methods, we consider U-Net [29] and ResUNetDecouple+ [36] as music source separation models because U-Net is the base model for many music source separation models, and ResUNetDecouple+ is the state-of-the-art model for the music source separation task. For the pitch estimation task, we use CREPE [32] and HARMOF0 [61], as they are the most common and state-of-the-art models. Finally, for joint learning methods, we select HS-W$_p$[44] and S→P→S→P[28] as the baselines since they are the latest joint learning methods for music source separation and pitch estimation tasks.

C ADDITION METHODS

As illustrated in Section 4.2.2, the most direct method involves setting different weights for different cases as outlined in Table 6. We refer to this method as naive DWHS, which utilizes two hyper-parameters (ω_{noise} and $upper_bound$) to assign weights to hard samples and noisy samples, as specified in Table 6.

Starting with default weights set at 1, for *Case 1*, where there are no issues with the MSS Module, PE Module, or the music data, the default weights remain unchanged. In *Case 2*, involving music data with noisy pitch labels, we decrease the weight of such samples by adjusting the weight (ω_{noise}) within the range of 0 to 1. This adjustment aims to mitigate the impact of noisy labels. In *Case 3*, where the music data is hard for the MSS task, we increase the weight of such samples in the MSS task by setting the weight to $1/\hat{y}_t$. The $\hat{y}_t$ is defined as $max(y) \times max(\hat{y}) + (1 - max(y)) \times (1 - max(\hat{y}))$, where y is the ground truth of pitch results and $\hat{y}$ is the predicted value. For *Case 4*, where the music data is hard for the PE task, we increase the weight of such samples in the PE task by setting the weight to $1/\hat{y}_t$. Besides, we constrain $1/\hat{y}_t$ to fall within the range of 1 to $upper\_bound$ to avoid invalid weights that are too large. The weights (ω_{mss} and ω_{pe}) for different cases are shown in Table 1. With the naive DWHS, the loss function of stage I is written as:

$$\mathcal{L}_{total} = \omega_{mss} \times \mathcal{L}_{mss} + \omega_{pe} \times \mathcal{L}_{pe} \tag{19}$$

And the loss function of stage II is written as:

$$\mathcal{L}_{total} = \text{confi}_{mss} \times \omega_{mss} \times \mathcal{L}_{mss} + \text{confi}_{pe} \times \omega_{pe} \times \mathcal{L}_{pe} \tag{20}$$

where confi_{mss} and confi_{pe} are the same as those in Eq. 8.

D ADDITIONAL EXPERIMENTAL RESULTS

D.1 Comparison of naive DWHS and DWHS

To compare the effectiveness of naive DWHS and DWHS, we conduct an ablation study on the MIR-1K dataset using ResUNetDecouple+ as the MSS Module and CREPE as the PE Module, consistent with the previous experiment in Section 6.1.

Table 7: Comparison results of the naive DWHS and the DWHS on MIR-1K dataset.

Methods	$upper\_bound$	ω_{noise}	MSS		PE (%)	
			SDR	GNSDR	RPA	RCA
Pipeline			12.06	9.13	91.40	92.07
Naive Joint Learning			11.91	8.92	91.88	92.15
MAJL-Stage I with naive DWHS	4	0	11.63	8.68	91.67	92.14
	5	0	12.09	9.16	92.46	92.74
	6	0	12.02	9.04	91.92	92.34
	5	0.2	12.13	9.18	92.62	93.03
	5	0.4	11.97	9.03	91.98	92.49
MAJL-Stage I with DWHS			**12.33**	**9.36**	**93.17**	**93.65**

The results in Table 7 demonstrate that the DWHS significantly enhances the joint learning of MSS and PE tasks, with MAJL-Stage I with DWHS achieving the best performance among all methods for both tasks. Specifically, the naive joint learning method leads to a decrease of 0.15 in SDR for the MSS task when compared to the pipeline method, due to the problem of misalignment between different objectives. In contrast, MAJL-Stage I with DWHS outperforms both the pipeline and naive joint learning methods, concurrently enhancing both tasks. For instance, it achieves a SDR improvement of 0.42 and a RPA improvement of 1.29% compared to the naive joint learning method. This is because the DWHS implemented in MAJL can effectively handle hard samples for both tasks and assign appropriate weights to different samples. Moreover, MAJL-Stage I with naive DWHS outperforms the naive joint learning method by 0.22 in SDR and 0.74% in RPA when hyper-parameters $upper\_bound$ is set to 5 and ω_{noise} to 0.2. Besides, the DWHS outperforms the naive DWHS and achieves better performance on both tasks without these hyper-parameters. These results indicate that the DWHS not only offers improved performance but also has a lower training cost, making it effective for the joint learning of both tasks.

D.2 Visualization and Analysis of Dynamic Weights

To provide an intuitive representation of the weights set by both the naive DWHS and the DWHS, we visualize the changes in these weights over iterations. In this experiment, we employ ResUNetDecouple+ as the MSS Module and CREPE as the PE Module, consistent

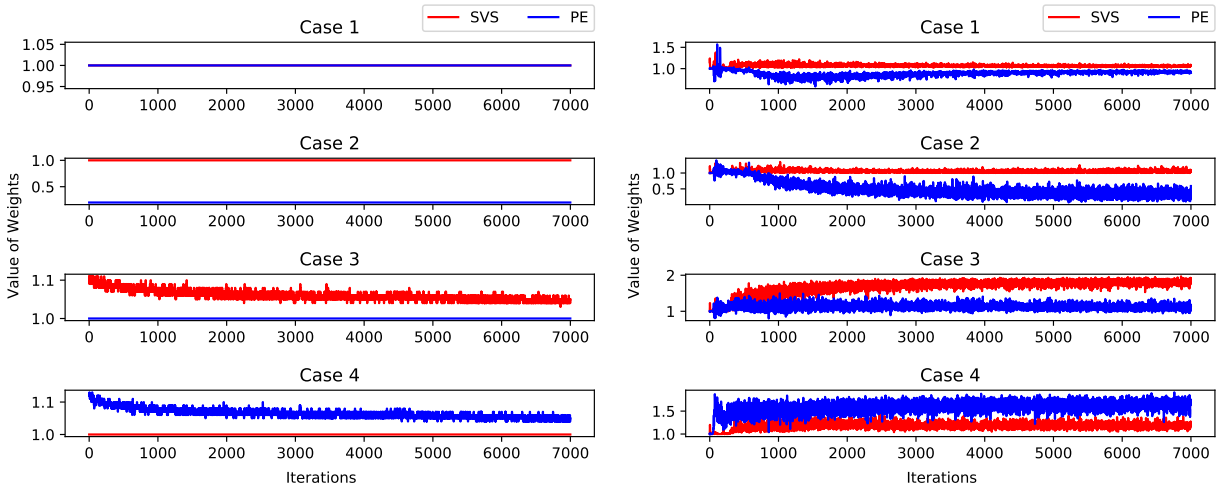


Figure 7: (a) Dynamic weights extracted by the naive DWHS. (b) Dynamic weights extracted by the DWHS.

with previous experiment detailed in Section 6.1. In addition, the dynamic weights extracted by Dynamic Weights on Hard Samples (DWHS) are obtained from our framework, specifically MAJL-Stage I, to exclusively investigate the weight results of DWHS and eliminate potential interference, such as single-labeled music data.

As shown in Figure 7, Figure 7(a) illustrates the dynamic weights set by the naive DWHS, while Figure 7(b) displays the dynamic weights extracted by the learned DWHS. For the naive DWHS method, the weight assigned to the pitch estimation task for noisy music data is set to 0.2, thereby mitigating the negative impact on the pitch estimation task. In contrast, for Case 3 and Case 4, the weights assigned to music source separation and pitch estimation exceed 1, emphasizing the importance of hard samples. Regarding the learned DWHS method, the weights for music source separation and pitch estimation tasks correspond to those of the naive DWHS method, as shown in Figure 7(b). When considering the results presented in Table 7, the DWHS method outperforms the naive DWHS method. These results indicate that the DWHS method can adaptively determine appropriate weights for both noisy and hard samples, resulting in enhanced performance for both music source separation and pitch estimation tasks.

D.3 Significance Test

In this section, we validate the significance of the improvements achieved by our framework through statistical significance tests. Thus, we perform multiple experimental runs (6 times) from training to testing and calculate p-values for the SDR and the RPA. The summarized results of these experiments are provided in Table 8.

In this experiment, the pipeline method, ResUNetDecouple+ with CREPE is used as the baseline, since it has displayed the best performance among all baselines, as shown in Table 2. While our framework use both single-labeled datasets (MUSDB18 and MIR_ST500) in the Stage II, and the DWHS is used in this experiment. It is important to note that due to the stochastic nature of gradient-based optimization techniques like the Adam optimizer [34] employed in this paper, there may be slight variations in results even under identical experimental conditions. This variability arises from random factors during training within our framework: the random initialization of model parameters and the random ordering of training samples in each epoch. For each experiment in Table 8, these random initialization use the default initialization method in PyTorch [46]. The results reported in Table 2 are based on the best performance achieved from these repeated experiments. Based on the results presented in Table 8, we calculate the p-values for both the SDR and the RPA. The obtained p-value for SDR is 1.80e-4, and for RPA it is 1.78e-6. These results indicate that our proposed framework achieves a significant improvement when compared to the best-performing baseline.

Table 8: Performance results with significance test on the MIR-1K dataset. Baseline is the pipeline method using ResUNetDecouple+ [36] with CREPE [32]. MAJL here uses the DWHS method, and both MIR_ST500 and MUSDB18 datasets are used as the single-labeled music data.

Methods	Index	MSS		PE(%)	
		SDR	GNSDR	RPA	RCA
Baseline	0	12.06	9.13	91.40	92.07
	1	11.91	8.92	90.87	91.56
	2	11.86	8.92	90.79	91.40
	3	11.97	9.03	91.03	91.63
	4	12.04	9.11	91.21	91.85
	5	12.06	9.12	91.33	91.77
MAJL	0	12.98	9.99	94.11	94.38
	1	12.31	9.36	92.31	92.70
	2	12.41	9.47	92.70	93.15
	3	12.55	9.59	92.88	93.27
	4	12.74	9.77	93.16	93.63
	5	12.90	9.94	93.38	93.96