# SELF-REFINING OF PSEUDO LABELS
# FOR MUSIC SOURCE SEPARATION WITH NOISY LABELED DATA

**\*Junghyun Koo**[1]     **\*Yunkee Chae**[2]     **Chang-Bin Jeon**[1]     **Kyogu Lee**[1,2,3]

[1]Department of Intelligence and Information, [2]Interdisciplinary Program in Artificial Intelligence,
[3]Artificial Intelligence Institute, Seoul National University

`{dg22302, yunkimo95, vinyne, kglee}@snu.ac.kr`

## ABSTRACT

Music source separation (MSS) faces challenges due to the limited availability of correctly-labeled individual instrument tracks. With the push to acquire larger datasets to improve MSS performance, the inevitability of encountering mislabeled individual instrument tracks becomes a significant challenge to address. This paper introduces an automated technique for refining the labels in a partially mislabeled dataset. Our proposed self-refining technique, employed with a noisy-labeled dataset, results in only a 1% accuracy degradation in multi-label instrument recognition compared to a classifier trained on a clean-labeled dataset. The study demonstrates the importance of refining noisy-labeled data in MSS model training and shows that utilizing the refined dataset leads to comparable results derived from a clean-labeled dataset. Notably, upon only access to a noisy dataset, MSS models trained on a self-refined dataset even outperform those trained on a dataset refined with a classifier trained on clean labels.

## 1. INTRODUCTION

Music source separation (MSS) is a critical task in the field of music information retrieval (MIR), with applications ranging from remixing [1–3] to transcription [4–6] and music education [7, 8]. To train high-performing MSS models, it is essential to have clean single-stem music recordings for guidance, which serve as the ground truth for model training. However, obtaining clean, large-scale datasets of single instrument tracks remains a challenging task.

With the increasing availability of music data on the internet, platforms such as YouTube provide a vast pool of potential single-instrument tracks. Although these sources offer an opportunity for performance gains through larger training datasets, collecting single instrument tracks from such platforms inevitably leads to encountering tracks with
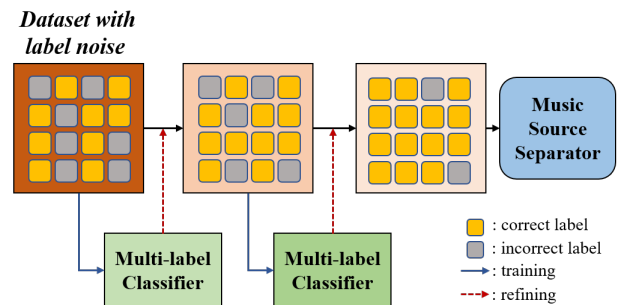
---

**Figure 1**. Overview of self-refining procedure on a noisy-labeled dataset for music source separation.

incorrect labels. For example, a query aimed at obtaining drum recordings might yield results that contain other types of instruments or noise, causing discrepancies between the expected and actual content of the collected recordings.

**Label noise** in datasets can arise from various factors, such as bleeding between instrument tracks, mislabeling due to human error, or the ambiguous timbre of instruments that resemble other instrument categories [9]. These factors make it challenging to assign a single definitive instrument label to a given recording. Such label noise is detrimental to the performance of MSS models, and there is a pressing need for an approach that can effectively train MSS models using partially corrupted datasets.

In response to this challenge, we propose an automated approach for refining mislabeled instrument tracks in a partially noisy-labeled dataset. Our *self-refining* technique, which leverages noisy-labeled data, results in only a 1% accuracy degradation for multi-label instrument recognition compared to a classifier trained with a clean-labeled dataset. The study highlights the importance of refining noisy-labeled data for training MSS models and demonstrates that utilizing the refined dataset for MSS yields results comparable to those obtained using a clean-labeled dataset. Notably, when only a noisy dataset is available, MSS models trained on self-refined datasets even outperform those trained on datasets refined with a classifier trained on clean labels. This paper presents a comprehensive analysis of our proposed method and its impact on the performance of MSS models.
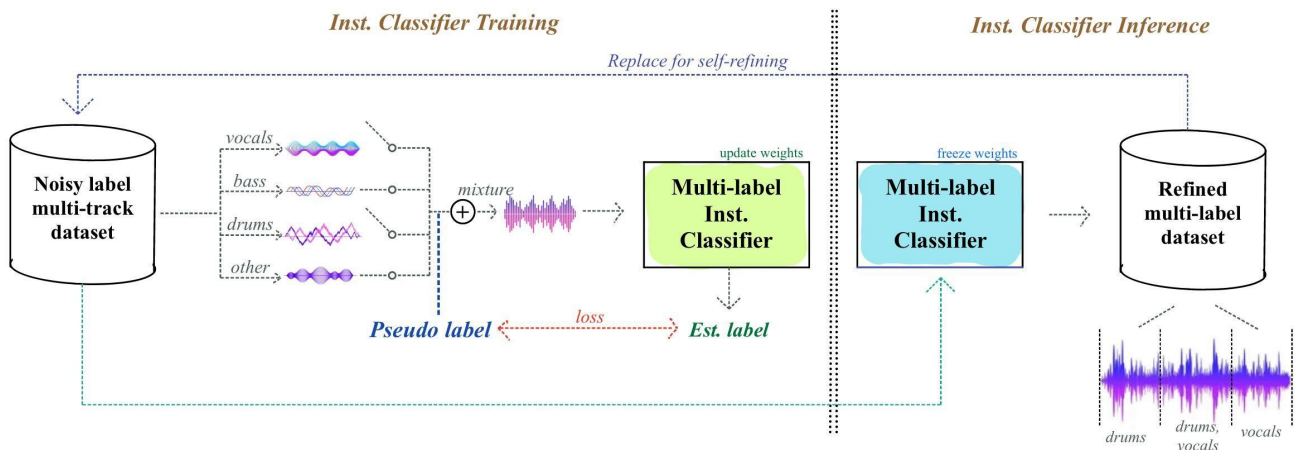
**Figure 2**. Overall training procedure of the Instrument Classifier $\Psi$. The classifier is trained to perform instrument recognition with mixtures that are synthesized by randomly selecting each stem from the noisy labeled dataset. After this training procedure, we refine the original noisy dataset and then use this new dataset to train the final $\Psi$.

## 2. RELATED WORKS

**Self-training** of machine learning models has been studied in various literatures, where a teacher model is first trained with clean labeled data and is used as a label predictor of unlabeled data, then a student model is trained with clean and pseudo-labeled data [10, 11]. Recently, Xie et al. proposed a noisy student method for self-training [12], which uses an iterative training of teacher-student models and noise injection methods for training student models. Thanks to their usefulness, these self-training methods have been used in diverse MIR tasks, such as singing voice detection [13, 14] and vocal melody extraction [15].

**Instrument recognition** or classification has been researched in various literatures, both in single-instrument [16–19] or multi-source settings [20–27]. Although such research has been focused on single or predominant-label prediction, Zhong, et al. [28] recently proposed the hierarchical approach for multi-label music instrument classification.

Our self-refining method for training of instrument classifier shares similar attributes with noisy student training [12] and the previous multi-label instrument classification [28] but differs from some perspectives. *i)* We train all our models only with partially noisy-labeled data, without access to clean-labeled data. *ii)* We train the classifiers for direct prediction of labels used in standard music source separation, e.g., vocals, bass, drums, and others, instead of the hierarchical approach. *iii)* We train multi-label classifiers with mixtures of randomly selected instruments, which are based on the characteristic of musical audio. If there exist two different instruments in one audio signal, that can be classified into two instruments. This random mixing of different instrumental tracks has been used in music source separation as well [29]. Note that the mixup method [30], which is also a mixing method of two different images, also shares a similar attribute with our method but is used for regularization of training single-label classifiers, not like our multi-label classifiers.

## 3. METHODOLOGY

Given a real-world scenario where the available multi-track dataset for MSS is partially incorrect with its instrument labels, a possible naive approach is first to rectify mislabeled tracks and then train an MSS model using stems with the revised labels. In this section, we introduce an effective training technique that first performs instrument recognition by only utilizing data with noisy labels and then leverages the refined dataset inferred with the trained multi-label instrument classifier to train the MSS model. With this two-stage approach, we explore the impact of the refined noisy datasets on the performance of MSS models.

### 3.1 Multi-label Instrument Recognition

Figure 2 summarizes the proposed training procedure of the Instrument Classifier $\Psi$. Similar yet different from self-training, our approach learns directly from noisy labeled data and re-labels the training data to train the final $\Psi$ using this refined dataset. We call this training procedure *self-refining*, and this is possible by *random mixing*, a method to synthesize a mixture of multiple instruments with pseudo labels. The *random mixing* technique takes advantage of the acoustic music domain in that mixing sources of different instrument tracks still leads to natural output mixture, whereas naively combining different images in the image domain is likely to produce unrealistic results. We further discuss about the benefits and the detailed process of *random mixing* at 3.1.1.

The network architecture of $\Psi$ is that of the ConvNeXt model [31], where it has shown great performance on a multi-instrument retrieval in [27]. The input of the network is a stereo-channeled magnitude linear spectrogram. Followed by a sigmoid layer, the model outputs four labels indicating the presence of each stem. The objective function for instrument recognition $\mathcal{L}_\Psi$ is a mean absolute loss between the estimated and synthesized pseudo labels. Preliminary experiments showed no significant difference in performance when employing mean absolute loss as compared to binary cross-entropy loss. This is likely due to
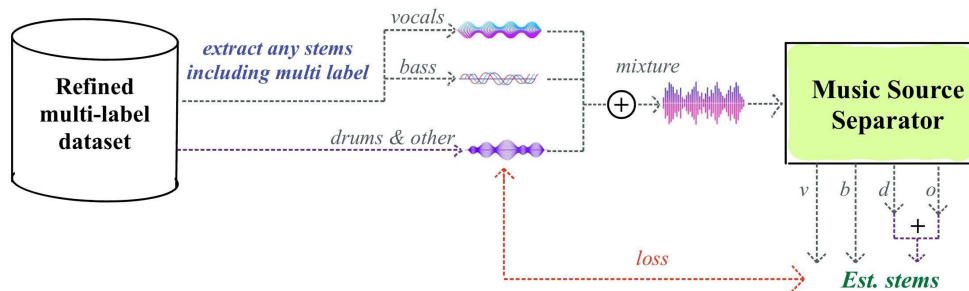
**Figure 3**. Music source separation training. Similar to the training procedure of the instrument classifier, we randomly mix each stem from the refined dataset to synthesize a mixture and use it as a network input. When a multi-labeled segment is selected for synthesis, the corresponding estimated stems are summed for loss computation.

the random mixing sampling that ensures similar occurrences of positive and negative labels of each instrument class during the training procedure.

### 3.1.1 Random Mixing

Randomly mixing stems with label noise not only creates various combinations of multi-labeled mixtures for training the instrument classifier but also brings the chance to generate a correct pseudo label from mislabeled stems. For instance, if we randomly select one correctly labeled drum track and a track that contains both sources of drums and vocals but is mislabeled as vocals, the mixing process synthesizes a correctly labeled mixture. Thanks to these fortunate chances, the random mixing technique assists the instrument classifier's accuracy in refining the label noise dataset by utilizing mislabeled stems.

To synthesize a random mixture and its pseudo label, each stem is first selected with a chance rate from the noisy dataset. The audio effects manipulation is then applied to each chosen track by simulating the music mixing process for data augmentation [3]. The order of applying audio effects with random parameters is 1. *dynamic range compression*, 2. *algorithmic reverberation*, 3. *stereo imaging*, and 4. *loudness manipulation*. Labels corresponding to randomly selected stems are used as a multi-label objective for the instrument classifier.

### 3.2 Music Source Separation

In this section, we describe the training procedure of MSS model employing a multi-labeled refined dataset curated by the classifier trained in Section 3.1. The majority of MSS research has focused on estimating each of the four instrument groups (*vocals*, *bass*, *drums*, and *other*) [32–35]. However, our refined dataset contains sources labeled with multiple stems, which are unsuitable for use as distinct target instruments. To utilize multi-labeled sources, we propose an appropriate MSS training framework tailored to our refined dataset.

First, we determine whether to include the multi-stem source for each input mixture sample by considering the probability $p$. If we decide not to include the multi-labeled source, we can train the MSS model in a conventional manner, computing the losses for each stem. Otherwise, we select a multi-labeled source from the refined dataset. Sub-

sequently, we choose the remaining stems that do not correspond to the selected multi-labeled source from a pool of single-labeled sources and combine them to simulate a mixture. For example, when selecting a multi-labeled source *bass+drums*, we opt for single sources labeled as *vocals* and *others* to synthesize the mixture. After conducting inference with the MSS model, we add the estimated stems corresponding to the multi-stem source of the input mixture and assess the loss between them. Figure 3 illustrates our training procedure when a multi-labeled source is selected. We compute the losses for each stem, treating the multi-labeled source as an individual stem, and subsequently sum these losses to derive the final loss value.

## 4. EXPERIMENTS

### 4.1 Dataset

We use the label noise dataset provided by the Music Demixing Challenge 2023 (MDX2023) [36], which consists of 203 songs, licensed by *Moises.AI* [1]. Similar to MUSDB18 [37], the provided dataset contains mixtures of music recordings segregated into four different instrumental stems: *vocals*, *bass*, *drums*, and *other*. Each stem and its corresponding label are intentionally altered to produce a corrupted dataset to simulate mislabeling such as bleeding or human mistakes. That is, for instance, `drums.wav` may contain drum sounds and singing voices simultaneously, which is likely to be caused by bleeding. For another example, a kick-drum sound might be mislabeled as `bass.wav` when the pitch of the kick drum is melodic enough to trick a human labeler. Due to the nature of the MDX2023 challenge, the dataset does not contain the actual ground truth labels. Hence, we use all 203 songs of the MDX2023 dataset only as training data.

To validate our system trained with noisy labeled data, we employed the MUSDB18 [37] as the clean dataset for comparison and evaluation. MUSDB18 comprises 150 songs, with 100 songs for the training and 50 songs for the test set. We adopt the test subset for evaluating all systems, while the training subset is used to train the upper bound system for observation.

**Data preprocessing.** To prevent models from mislabels caused by silence, we remove all silent sections through-

---

[1] https://moises.ai/

| Label Type | Training Data | Accuracy / F1 Score Precision / Recall | | | | |
| | | vocals | bass | drums | other | avg |
|---|---|---|---|---|---|---|
| Single-Label | clean | 97.8% / 0.947 0.91 / 0.98 | 94.4% / 0.891 0.84 / 0.94 | 95.1% / 0.914 0.85 / 0.98 | 93.2% / 0.880 0.90 / 0.85 | 95.1% / 0.906 0.87 / 0.93 |
| | noisy | 93.6% / 0.860 0.76 / 0.97 | **90.0% / 0.821** **0.73** / 0.93 | **93.7% / 0.893** **0.81 / 0.98** | **92.6% / 0.865** **0.92** / 0.81 | 92.5% / 0.860 **0.80** / 0.92 |
| | refined | **96.1% / 0.911** **0.84 / 0.98** | 89.6% / 0.818 0.71 / **0.96** | 93.1% / 0.884 0.79 / **0.98** | 92.3% / 0.862 0.90 / **0.82** | **92.8% / 0.866** 0.80 / 0.93 |
| Multi-Label | clean | 92.4% / 0.929 0.92 / 0.93 | 89.6% / 0.905 0.89 / 0.92 | 90.5% / 0.913 0.87 / 0.95 | 88.1% / 0.878 0.90 / 0.85 | 90.2% / 0.907 0.90 / 0.91 |
| | noisy | 87.9% / 0.895 0.83 / 0.96 | 87.5% / 0.888 **0.86** / 0.93 | 87.7% / 0.891 0.82 / **0.96** | 87.3% / 0.872 **0.88** / 0.87 | 87.6% / 0.887 0.85 / 0.93 |
| | refined | **91.9% / 0.928** **0.88 / 0.97** | **87.8% / 0.894** 0.84 / **0.95** | **89.6% / 0.906** **0.85 / 0.96** | **87.4% / 0.874** **0.88 / 0.87** | **89.2% / 0.901** **0.86 / 0.94** |

**Table 1**. Instrument recognition performance on single and multi-label instrument classifiers trained with different datasets. The training data of *clean*, *noisy*, and *refined* each represents the training subset of MUSDB18, MDX2023, and MDX2023 refined with the instrument classifier trained with MDX2023 $\Psi_{noisy}$, respectively.

out both datasets. The preprocessing procedure for silence removal is as follows:

1. For each song, detect silent areas that are below 30 dB relative to the maximum peak amplitude.

2. Remove all detected areas then merge them into one single long audio track.

3. Repeat 1. (with the threshold of 60 dB) and 2. based on the merged audio track, in case of stems that are almost silent.

After trimming silent regions, the total durations for each stem in the respective order of *vocals*, *bass*, *drums*, and *other* are [7.2, 7.8, 9.2, 10.3] hours for the MDX2023 dataset, and [2.2, 2.7, 2.9, 3.3] hours for the test subset of MUSDB18. Note that for evaluating MSS performance, we instead follow the original convention of processing entire songs from the test subset without any silence removal. We use the original audio specifications of both datasets where all audio tracks are stereo-channeled and have a sampling rate of 44.1 KHz.

### 4.2 Experimental Setups

For multi-label instrument recognition, the network architecture of $\Psi$ is ConvNeXt's tiny version [31], which consists of 27.8M parameters. We feed the network with stereo-channeled mixtures of instruments that are of 2.97 seconds, which are transformed into a time-frequency domain linear magnitude spectrogram with an FFT size of 2048 and a hop size of 512. We train all $\Psi$ for 100 epochs. During inference, $\Psi$ performs classification by processing the entire input audio in windows of a size equivalent to the network input size, with a hop size of one-fourth of this window size. The output labels from these windows are then averaged to yield the final decision, based on a threshold value of 0.9. We utilized this inference procedure to refine the noisy dataset, which was then used to train our MSS models. Our final version of the instrument

classifier trained on the refined dataset $\Psi_{refined}$ only uses stems inferred as a single-labeled for better performance based on our preliminary experiments.

We employed two MSS models, Hybrid Demucs (Demucs v3) [38] and CrossNet-Open-Unmix (X-UMX) [39], to evaluate their performance when trained on the processed datasets. Multi-labeled sources were selected with a probability of 0.4, and input loudness normalization (-14 LUFS) was applied for both training and inference in accordance with [40]. `pyloudnorm` [41] was used for loudness calculation [42].

For Demucs, the input duration was set to 3 seconds, and optimization was performed using Adam optimizer [43] and L1 loss on the time domain. The model was trained for 21,000 iterations with a batch size of 160.

For X-UMX, the input duration was set to 6 seconds, and optimization was performed using AdamW optimizer [44] and mean squared error loss on the time-frequency domain. For the sake of simplicity, we omit the multi-domain and combination loss proposed in [39]. The model was trained for 56,400 iterations with a batch size of 32. For the + *finetune w/ multi-labeled* model in Table 3, we first train the model with only single-labeled data for 20,680 iterations, then finetune it with multi-labeled data for another 35,720 iterations.

## 5. RESULTS

### 5.1 Instrument Recognition

Table 1 presents the instrument recognition performance of the multi-instrument classifier on single-labeled and multi-labeled data. As ground-truth labels are not available for the MDX2023 dataset, we validate the classification performance according to single and multi-labeled data with the MUSDB18 test set for evaluation. For the multi-label evaluation, we synthesized 3,941 mixtures from the test set with the random mixing technique described in 3.1.1. We observe the performance of $\Psi$ trained with MUSDB18 (*clean*), MDX2023 (*noisy*), and MDX2023 once refined

| Network | Training Data | SDR [dB] | | | | |
|---|---|---|---|---|---|---|
| | | vocals | bass | drums | other | avg |
| Demucs [38] | clean | 5.92 | 6.16 | 5.58 | 4.43 | 5.52 |
| | noisy | 3.37 | 1.92 | 0.70 | 0.86 | 1.71 |
| | w/ $\Psi_{clean}$ | 5.31 | **5.12** | 1.32 | 2.16 | 3.48 |
| | w/ $\Psi_{noisy}$ | 4.15 | 4.58 | 1.62 | 2.85 | 3.30 |
| | w/ $\Psi_{refined}$ | **5.36** | 5.04 | **3.09** | **3.13** | **4.16** |
| X-UMX [39] | clean | 5.76 | 4.44 | 5.47 | 3.65 | 4.83 |
| | noisy | 3.39 | 1.78 | 1.52 | 0.96 | 1.91 |
| | w/ $\Psi_{clean}$ | 4.50 | 3.22 | 3.66 | 2.73 | 3.53 |
| | w/ $\Psi_{noisy}$ | 4.72 | **4.11** | 3.22 | 2.89 | 3.74 |
| | w/ $\Psi_{refined}$ | **4.99** | 3.93 | **5.00** | **3.18** | **4.28** |

**Table 2**. Source separation performance of Demucs v3 [38] and CrossNet-Open-Unmix [39] trained on different training datasets. Sub-items below *noisy* dataset indicate data refined with the respective instrument classifiers, denoted as $\Psi_\bullet$.

with $\Psi_{noisy}$ (*refined*). The evaluation metrics used are accuracy, F1 score, precision, and recall for each instrument class and the overall averaged result.

For single-labeled data, the classifier achieves the highest average performance on the *clean* dataset, with an accuracy of 95.1% and an F1 score of 0.906. As *clean* dataset does not contain any noisy labels, the obtained results can be considered an upper bound for the performances of the classifiers. The $\Psi$ trained on *refined* dataset results in slightly lower performance, with an accuracy of 92.8% and F1 score of 0.866, while the *noisy* dataset shows an accuracy of 92.5% and F1 score of 0.860. Although the accuracy, F1 score, and precision are higher for the *noisy* dataset in the *bass, drums*, and *other* stems, the performance metrics for *vocals* and recall values across all stems exhibit superior results when trained with the *refined* dataset.

For instrument recognition on multi-labeled data, $\Psi$ trained on *clean* dataset yields an average accuracy of 90.2% and F1 score of 0.907. The *noisy* dataset results in an accuracy of 87.6% and an F1 score of 0.887. The *refined* dataset achieves superior performance, with an accuracy of 89.2% and an F1 score of 0.901, which is comparable to the results obtained from the *clean* dataset. Contrary to the evaluation with single-labeled data, the *refined* dataset generally demonstrates superior performance across all metrics in comparison to the *noisy* dataset. Notably, the recall values are observed to be even higher than those of the *clean* dataset. An in-depth analysis of the multi-instrument classifier results, alongside the performance outcomes of the MSS models, is discussed in Section 5.2.

## 5.2 Source Separation

The results of MSS models trained on different training datasets are presented in Table 2. In our evaluation, we used Signal-to-Distortion Ratio (SDR) [45], which is calculated using the `museval` toolkit [46]. For all MSS experiments, we report the SDR median of frames and the median of tracks. The Demucs and X-UMX models are

| Method | SDR [dB] | | | | |
|---|---|---|---|---|---|
| | vocals | bass | drums | other | avg |
| *proposed* | 4.99 | 3.93 | 5.00 | 3.18 | 4.28 |
| *threshold = 0.5* | 5.06 | 4.13 | 4.77 | 3.06 | 4.25 |
| *adaptive thresholds* | 4.70 | 3.72 | 3.70 | 2.62 | 3.68 |
| *train only w/ single-labeled* | 4.90 | 3.73 | 4.54 | 3.18 | 4.09 |
| *+ finetune w/ multi-labeled* | 4.33 | 4.33 | 4.19 | 3.14 | 4.00 |
| *self-refining ×5* | 4.65 | 3.87 | 5.07 | 2.89 | 4.12 |

**Table 3**. Ablation studies on MSS performances with CrossNet-Open-Unmix.

trained on *clean*, *noisy*, and data processed with multi-instrument classifiers, denoted by $\Psi_\bullet$. In this context, $\Psi_\bullet$ represents the classifier trained on each respective dataset, as described in Section 5.1.

The baseline for this experiment is established using MSS models trained on the *noisy* dataset. It is noteworthy that all the results presented in the table exceed the baseline performance. For the dataset processed with the multi-instrument classifier $\Psi_{refined}$, average SDR improvements of 2.45 and 2.31 are observed for Demucs and X-UMX models, respectively, in comparison to the *noisy* dataset. Specifically, in $\Psi_{refined}$ case, both Demucs and X-UMX models demonstrate substantial improvements in SDR values across all stems compared to those of $\Psi_{noisy}$, with the exception of *bass* in the X-UMX model.

### 5.2.1 Analysis in relation to instrument recognition

In Table 2, it is noteworthy that the performance of $\Psi_{refined}$ exceeds the performance of $\Psi_{clean}$, even though $\Psi_{clean}$ is trained with a noise-free labeled dataset. This implies the classification performance of $\Psi_{clean}$ is inferior to the classification performance of $\Psi_{refined}$. This discrepancy could be attributed to differences in the data distribution between the MUSDB18 and MDX2023 datasets. Moreover, the number of training samples varies, with 100 samples in the MUSDB18 dataset and 203 samples in the MDX2023 dataset. When refining a partially noisy dataset, employing the same partially noisy dataset can yield advantageous outcomes than using the smaller clean dataset. This observation might be aligned with the findings in [12], which report an improvement in performance when a larger quantity of unlabeled data is present.

An additional factor to consider is the distinctive nature of the MSS model training framework in our approach. MSS models utilize the output of the classifier as input. The performance of the MSS model can be affected differently depending on the type of error in the classifier's output. For example, assume that the MSS model receives a sample misclassified as a vocal stem when no vocals are actually present (i.e. a false-positive sample for vocals). In this case, the MSS model simply needs to predict silence for the vocals stem and produce it as output, resulting in no significant confusion. Conversely, consider a scenario in which the MSS model receives a sample misclassified as a non-vocal stem (e.g. drums + bass), despite the presence of vocals, resulting in a false-negative sample for vocals. In such a case, the model will attempt to allocate the vocals
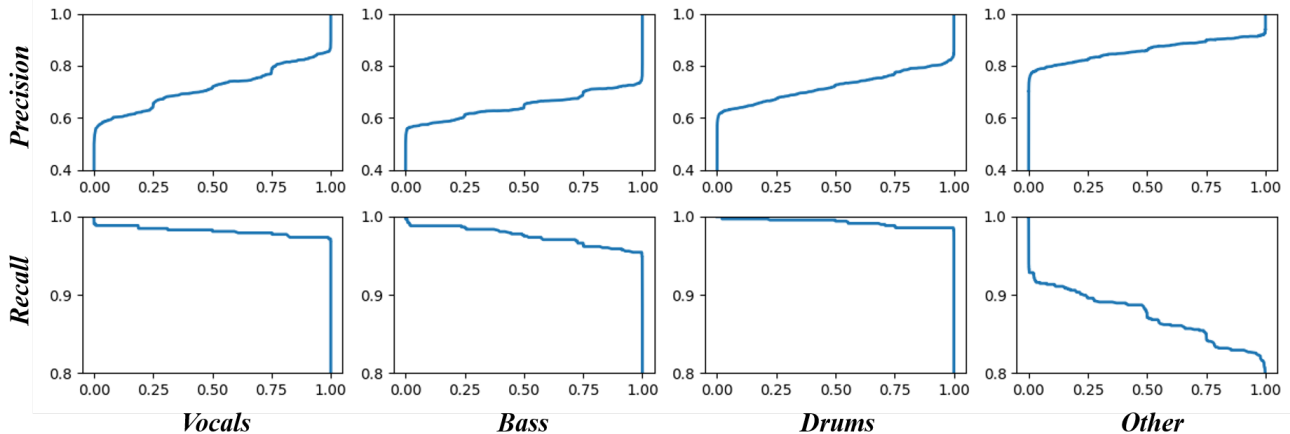
**Figure 4**. Precision and recall curves of the proposed classifier across different thresholds (x-axis) on each instrument. The curves are generated using the MUSDB18 test set (*clean*).

present in the input data to the drum and bass stems. Furthermore, our model differs from traditional MSS training methods as it also accepts multi-stem data as input. In this context, the vocals are present as the correct answer for multiple mislabeled non-vocal stems, which confuses the model. This not only negatively affects the performance of the mislabeled stems but also the vocal stem itself.

As a consequence of the unique characteristics of our training process, false-negative samples have a more significant impact on MSS compared to false-positive samples, highlighting the increased significance of the recall metric. Considering this perspective, the results presented in Table 1 imply the possibility of the sub-optimal performance of MSS trained on outputs of $\Psi_{clean}$, where the recall values are lower for all stems compared to $\Psi_{refined}$.

### 5.2.2 Ablation studies

As shown in Table 3, we evaluate the performance of X-UMX under various conditions to better understand the significance of distinct aspects of our proposed method.

**Threshold.** We conduct experiments to examine the impact of threshold determination for the classifier during the training of MSS models using a classified dataset. The evaluation is performed on the MUSDB18 test set. We observe that reducing the threshold to 0.5 only exhibits an SDR of 0.03 degradation compared to the original threshold value of 0.9. This outcome can be attributed to the fact that only 8% of $\Psi_{refined}$ outputs fall within the range of [0.1, 0.9] upon inference on the MUSDB18 test set. In Figure 4, we present the precision and recall curves for each threshold on individual instruments. It is evident from the curves that the variations within that range for both precision and recall are not substantial. Consequently, the choice between thresholds of 0.9 or 0.5 does not yield any noticeable disparity. Furthermore, we conduct an experiment involving adaptive thresholds for each instrument, where the threshold for each instrument was set to maximize the F1 score of the classification performance. However, we observe a significant degradation in performance across all instruments when employing adaptive thresholds. Maximizing the F1 score necessitates a trade-off between recall and precision, often leading to a decline in recall to

enhance precision. Consequently, the performance of the MSS model experience degradation, aligning with the discussion presented in Section 5.2.1.

**Training with multi-labeled data.** When training solely with the data estimated as single-labeled, the performance is not as good as that of the proposed method. Incorporating both single- and multi-labeled data for fine-tuning after the initial training on single-labeled data leads to a slightly diminished performance, despite utilizing both types of labeled data during the training process.

**Iterative self-refining.** Finally, we examine the influence of the iterative self-refining technique on MSS performance. The results indicate that an MSS model trained with a noisy-labeled dataset refined five times through our method does not yield superior performance compared to the proposed model, trained on a dataset refined twice, and the performance difference is insignificant. This observation suggests that excessive refinement iterations do not necessarily lead to improved performance and that refining the dataset twice may be sufficient for optimal results.

## 6. CONCLUSION

In conclusion, this paper presented a self-refining approach to address the challenges of noisy-labeled data in training music source separation (MSS) models. Our proposed method refines mislabeled instrument tracks in partially noisy-labeled datasets, resulting in only a 1% accuracy degradation for multi-label instrument recognition compared to a classifier trained on a clean-labeled dataset. This study highlights the importance of refining noisy-labeled data for training MSS models effectively and demonstrates that utilizing the refined dataset for MSS yields results comparable to those obtained using a clean-labeled dataset. Considering the real-world scenario of accessibility only to a noisy dataset, MSS models trained on self-refined datasets outperformed those trained on datasets refined with a classifier trained on clean labels. The self-refining approach we introduced offers a promising direction for future research in the field of music information retrieval and has the potential to be extended to other applications requiring robust training on noisy-labeled datasets.

## 7. ACKNOWLEDGEMENTS

## 8. REFERENCES

[1] J. F. Woodruff, B. Pardo, and R. B. Dannenberg, "Remixing stereo music with score-informed source separation." in *ISMIR*, 2006, pp. 314–319.

[2] J. Pons, J. Janer, T. Rode, and W. Nogueira, "Remixing music using source separation algorithms to improve the musical experience of cochlear implant users," *The Journal of the Acoustical Society of America*, vol. 140, no. 6, pp. 4338–4349, 2016.

[3] J. Koo, M. A. Martínez-Ramírez, W.-H. Liao, S. Uhlich, K. Lee, and Y. Mitsufuji, "Music mixing style transfer: A contrastive learning approach to disentangle audio effects," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[4] M. D. Plumbley, S. A. Abdallah, J. P. Bello, M. E. Davies, G. Monti, and M. B. Sandler, "Automatic music transcription and audio source separation," *Cybernetics &Systems*, vol. 33, no. 6, pp. 603–627, 2002.

[5] L. Lin, Q. Kong, J. Jiang, and G. Xia, "A unified model for zero-shot music source separation, transcription and synthesis," in *The 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[6] K. W. Cheuk, K. Choi, Q. Kong, B. Li, M. Won, J.-C. Wang, and Y.-N. H. D. Herremans, "Jointist: Simultaneous improvement of multi-instrument transcription and music source separation via joint training," *arXiv preprint arXiv:2302.00286*, 2023.

[7] C. Dittmar, E. Cano, J. Abeßer, and S. Grollmisch, "Music information retrieval meets music education," in *Dagstuhl Follow-Ups*, vol. 3. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2012.

[8] E. Cano, G. Schuller, and C. Dittmar, "Pitch-informed solo and accompaniment separation towards its use in music education applications," *EURASIP Journal on Advances in Signal Processing*, vol. 2014, pp. 1–19, 2014.

[9] C. T. Hoopen, "Issues in timbre and perception," *Contemporary Music Review*, vol. 10, no. 2, pp. 61–71, 1994.

[10] H. Scudder, "Probability of error of some adaptive pattern-recognition machines," *IEEE Transactions on Information Theory*, vol. 11, no. 3, pp. 363–371, 1965.

[11] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *33rd annual meeting of the association for computational linguistics*, 1995, pp. 189–196.

[12] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves imagenet classification," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 687–10 698.

[13] J. Schlüter, "Learning to pinpoint singing voice from weakly labeled examples." in *ISMIR*, 2016, pp. 44–50.

[14] G. Meseguer-Brocal, A. Cohen-Hadria, and G. Peeters, "DALI: A large dataset of synchronized audio, lyrics and notes, automatically created using teacher-student machine learning paradigm," in *The 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[15] S. Keum, J.-H. Lin, L. Su, and J. Nam, "Semi-supervised learning using teacher-student models for vocal melody extraction," in *The 21th International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[16] E. Benetos, M. Kotti, and C. Kotropoulos, "Musical instrument classification using non-negative matrix factorization algorithms and subset feature selection," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 5. IEEE, 2006, pp. V–V.

[17] A. Eronen and A. Klapuri, "Musical instrument recognition using cepstral coefficients and temporal features," in *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100)*, vol. 2. IEEE, 2000, pp. II753–II756.

[18] V. Lostanlen and C.-E. Cella, "Deep convolutional networks on the pitch spiral for musical instrument recognition," in *The 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[19] S. Essid, G. Richard, and B. David, "Hierarchical classification of musical instruments on solo recordings," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 5. IEEE, 2006, pp. V–V.

[20] Y. Han, J. Kim, and K. Lee, "Deep convolutional neural networks for predominant instrument recognition in polyphonic music," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 1, pp. 208–221, 2016.

[21] Y.-N. Hung and Y.-H. Yang, "Frame-level instrument recognition by timbre and pitch," in *The 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[22] S. Gururani, C. Summers, and A. Lerch, "Instrument activity detection in polyphonic music using deep neural networks." in *The 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018, pp. 569–576.

[23] S. Gururani, M. Sharma, and A. Lerch, "An attention mechanism for musical instrument recognition," in *The 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

[24] Y.-N. Hung, Y.-A. Chen, and Y.-H. Yang, "Multi-task learning for frame-level instrument recognition," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 381–385.

[25] H. Flores Garcia, A. Aguilar, E. Manilow, and B. Pardo, "Leveraging hierarchical structures for few-shot musical instrument recognition," in *The 22nd International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[26] L. C. Reghunath and R. Rajan, "Transformer-based ensemble method for multiple predominant instruments recognition in polyphonic music," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2022, no. 1, p. 11, 2022.

[27] K. Kim, M. Park, H. Joung, Y. Chae, Y. Hong, S. Go, and K. Lee, "Show me the instruments: Musical instrument retrieval from mixture audio," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[28] Z. Zhong, M. Hirano, K. Shimada, K. Tateishi, S. Takahashi, and Y. Mitsufuji, "An attention-based approach to hierarchical multi-label music instrument classification," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[29] S. Uhlich, M. Porcu, F. Giron, M. Enenkl, T. Kemp, N. Takahashi, and Y. Mitsufuji, "Improving music source separation based on deep neural networks through data augmentation and network blending," in *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 261–265.

[30] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, 2018.

[31] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 976–11 986.

[32] F.-R. Stöter, S. Uhlich, A. Liutkus, and Y. Mitsufuji, "Open-unmix-a reference implementation for music source separation," *Journal of Open Source Software*, vol. 4, no. 41, p. 1667, 2019.

[33] M. Kim, W. Choi, J. Chung, D. Lee, and S. Jung, "Kuielab-mdx-net: A two-stream neural network for music demixing," *arXiv preprint arXiv:2111.12203*, 2021.

[34] S. Rouard, F. Massa, and A. Défossez, "Hybrid transformers for music source separation," *arXiv preprint arXiv:2211.08553*, 2022.

[35] Y. Luo and J. Yu, "Music source separation with band-split rnn," *arXiv preprint arXiv:2209.15174*, 2022.

[36] Y. Mitsufuji, G. Fabbro, S. Uhlich, F.-R. Stöter, A. Défossez, M. Kim, W. Choi, C.-Y. Yu, and K.-W. Cheuk, "Music demixing challenge 2021," *Frontiers in Signal Processing*, vol. 1, p. 18, 2022.

[37] Z. Rafii, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, and R. Bittner, "Musdb18-hq - an uncompressed version of musdb18," Aug. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3338373

[38] A. Défossez, "Hybrid spectrogram and waveform source separation," *arXiv preprint arXiv:2111.03600*, 2021.

[39] R. Sawata, S. Uhlich, S. Takahashi, and Y. Mitsufuji, "All for one and one for all: Improving music separation by bridging networks," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 51–55.

[40] C.-B. Jeon and K. Lee, "Towards robust music source separation on loud commercial music," in *Proc. of the 23rd Int. Society for Music Information Retrieval Conference*, 2022.

[41] C. J. Steinmetz and J. D. Reiss, "pyloudnorm: A simple yet flexible loudness meter in python," in *150th AES Convention*, 2021.

[42] R. ITU-R, "Itu-r bs. 1770-2, algorithms to measure audio programme loudness and true-peak audio level," *International Telecommunications Union, Geneva*, 2011.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[44] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019.

[45] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE transactions on audio, speech, and language processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[46] F. Stöter, A. Liutkus, and N. Ito, "The 2018 signal separation evaluation campaign," in *International Conference on Latent Variable Analysis and Signal Separation*. Springer, 2018, pp. 293–305.