On Temporal Guidance and Iterative Refinement in Audio Source Separation

Tobias Morocutti[2*], *Jonathan Greif*[2*], *Paul Primus*[1*], *Florian Schmid*[1], *Gerhard Widmer*[1,2]

[1]Institute of Computational Perception (CP-JKU),[2]LIT Artificial Intelligence Lab,
Johannes Kepler University Linz, Austria
{tobias.morocutti, jonathan.greif, paul.primus}@jku.at

***Abstract*—Spatial semantic segmentation of sound scenes (S5) involves the accurate identification of active sound classes and the precise separation of their sources from complex acoustic mixtures. Conventional systems rely on a two-stage pipeline—audio tagging followed by label-conditioned source separation—but are often constrained by the absence of fine-grained temporal information critical for effective separation. In this work, we address this limitation by introducing a novel approach for S5 that enhances the synergy between the event detection and source separation stages. Our key contributions are threefold. First, we fine-tune a pre-trained Transformer to detect active sound classes. Second, we utilize a separate instance of this fine-tuned Transformer to perform sound event detection (SED), providing the separation module with detailed, time-varying guidance. Third, we implement an iterative refinement mechanism that progressively enhances separation quality by recursively reusing the separator's output from previous iterations. These advancements lead to significant improvements in both audio tagging and source separation performance, as demonstrated by our system's second-place finish in Task 4 of the DCASE Challenge 2025. Our implementation and model checkpoints are available in our GitHub repository[1].**

***Index Terms*—DCASE Challenge, Audio Source Separation, M2D, ResUNet, AudioSep, Time-FiLM, Iterative Refinement, DPRNN**

1. INTRODUCTION

Spatial semantic segmentation of sound scenes (S5) aims to identify and isolate individual sound sources from complex acoustic mixtures. To achieve this, S5 systems typically operate in two stages. In the first stage, a clip-level event detector identifies the set of sound events present in the mixture. In the second stage, a separator model takes the mixture and one of the predicted class labels as input, and estimates the corresponding isolated source for that class.

For Task 4 of the DCASE 2025 Challenge [1], participants were tasked with training such a system for 18 predefined sound categories, including speech and various domestic sound events. The challenge baseline system [2] employs an M2D-based audio tagger [3] to identify the active sound classes in the mixture (Stage 1), and subsequently separates them using a model based on the ResUNet architecture [4], conditioned on the target class via feature-wise linear modulation (FiLM) [5] (Stage 2).

In our submission to Task 4 of the DCASE 2025 Challenge [6], we enhance the baseline system with several key improvements. This paper presents our system (Figure 1) and evaluates the impact of our design choices, focusing on the following core areas:

- **Sound Event Detection (Stage 1):** We fine-tune a pre-trained SED Transformer [7], based on the M2D architecture [3], [8], using both clip-level and frame-level annotations for the 18 target classes. At inference time, we use attention-pooled frame-wise predictions to identify active acoustic events in a given mixture.
- **Temporal Guidance for Source Separation (Stage 2):** A trainable copy of the Stage 1 SED model is integrated into the separator to provide temporal guidance. It aids separation by: (1) supplying frame-wise class predictions for enhanced FiLM conditioning (*Time-FiLM*); and (2) injecting a weighted sum of its hidden features into the ResUNet, adding temporally aligned semantics (*Embedding Injection*).
- **Dual-Path RNN Integration:** To better capture long-range temporal dependencies, we augment the ResUNet's embedding space with a Dual-Path RNN [9], enabling more effective context modeling across time.
- **Iterative Refinement:** Inspired by recent iterative separation strategies [10], we apply a simple yet effective refinement technique: the separator's outputs are recursively fed back into the model, progressively improving the separation quality.

Section 2 provides a brief formalization of the problem. In Sections 3 and 4, we describe our modifications to the event detection and separation stages, respectively. Section 5 outlines the experimental setup, and Section 6 presents the results of our investigation into the effectiveness of the proposed enhancements. Finally, Section 7 concludes this paper.

2. PROBLEM FORMALIZATION

The goal of the *S5* task is to detect and separate individual sound events from multi-channel time-domain mixtures recorded in realistic environments. Let

$$\boldsymbol{X} = [\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(M)}]^\top \in \mathbb{R}^{M \times T}$$

be the multi-channel mixture of length T, recorded with M microphones. Each channel $\boldsymbol{x}^{(m)} \in \mathbb{R}^T$ is modeled as:

$$\boldsymbol{x}^{(m)} = \sum_{k=1}^{K} \boldsymbol{h}_k^{(m)} * \boldsymbol{s}_k + \sum_{j=1}^{J} \boldsymbol{h}_j^{(m)} * \boldsymbol{s}_j + \boldsymbol{n}^{(m)} \tag{1}$$

where $*$ denotes the convolution operator, K is the number of active target sound sources, and J is the number of interference events. The terms $\boldsymbol{s}_k, \boldsymbol{s}_j \in \mathbb{R}^T$ represent the single-channel dry source signals for a target event of class c_k and an interference event of class c_j, respectively. Similarly, $\boldsymbol{h}_k^{(m)}, \boldsymbol{h}_j^{(m)} \in \mathbb{R}^H$ are the room impulse responses (RIRs) from the target source k and interfering source j to microphone m, respectively, where H denotes the RIR length. Finally, $\boldsymbol{n}^{(m)} \in \mathbb{R}^T$ represents additive noise on the m-th microphone channel.

The objective is to estimate the active target classes c_k and recover their direct-path waveforms $\{\boldsymbol{s}_1^{(d,m)}, \ldots, \boldsymbol{s}_K^{(d,m)}\}$ for a chosen reference microphone channel m, where

$$\boldsymbol{s}_k^{(d,m)} = \boldsymbol{h}_k^{(d,m)} * \boldsymbol{s}_k \tag{2}$$

is the anechoic source $\boldsymbol{s}_k$ convolved with the direct-path RIR $\boldsymbol{h}_k^{(d,m)}$, corresponding to the direct arrival sound without reverberation.

*These authors contributed equally to this work.
[1]https://github.com/theMoro/dcase25task4

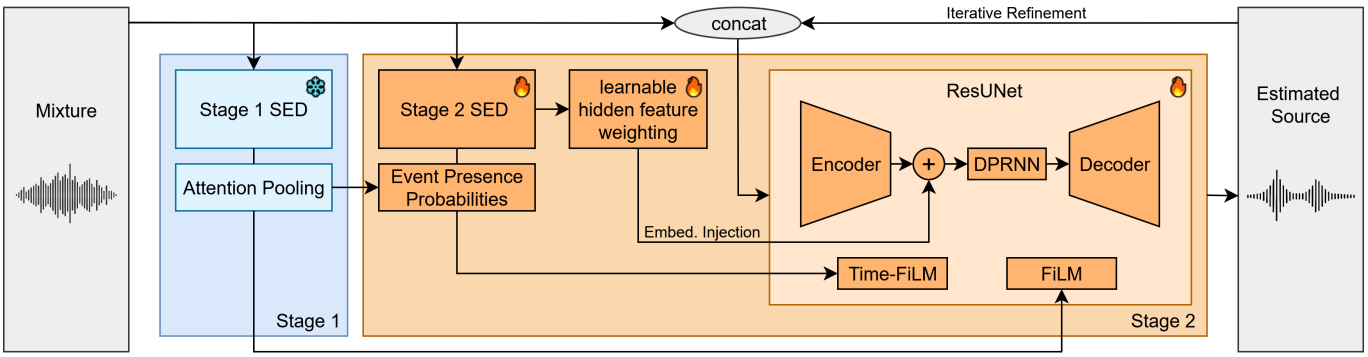

Fig. 1: Overview of the proposed two-stage audio source separation system. Stage 1 uses a sound event detection (SED) model to identify active sound classes in the mixture. Stage 2 performs class-conditioned source separation using a ResUNet architecture enhanced with temporal conditioning (Time-FiLM), Embedding Injection from the dedicated Stage 2 SED model, and an optional Dual-Path RNN. An iterative refinement mechanism allows the separator to progressively improve output quality by reusing previous predictions as an additional input.

For our purposes, $\boldsymbol{h}_k^{(d,m)}$ is approximated from the full RIR $\boldsymbol{h}_k^{(m)}$ by applying a time-domain window around the first significant energy peak, using a window spanning 6 ms before and 50 ms after the peak.

3. ACTIVE EVENT DETECTION

Stage 1 relies on an audio classifier to estimate the presence or absence of the C target acoustic events at the clip level. Unlike the baseline system, which employs a standard audio tagger, we use a sound event detection (SED) model that predicts class activity at the frame level. These predictions are then aggregated into clip-level class scores via attention pooling. This design choice is motivated by two key factors: (1) we hypothesize that frame-level supervision offers more informative training signals, potentially improving tagging accuracy, and (2) the resulting SED model can be integrated into the separator in Stage 2 to provide temporal conditioning.

For training the SED model, we use only the first channel of the multi-channel mixture, denoted $\boldsymbol{x}^{(1)}$, and convert it into a mel-spectrogram:

$$\{\mathbf{y}_t\}_{t=1}^{T} = \mathrm{MEL}(\boldsymbol{x}^{(1)}), \quad \mathbf{y}_t \in \mathbb{R}^F,$$

where T is the number of time frames and F is the number of mel frequency bins. The model, denoted by a function g, processes this input and produces a sequence of embedding vectors:

$$\{\hat{\mathbf{e}}_s\}_{s=1}^{S} = g\big(\{\mathbf{y}_t\}_{t=1}^{T}\big), \quad \hat{\mathbf{e}}_s \in \mathbb{R}^D, \tag{3}$$

where D is the embedding dimension, and S is the output sequence length. Note that S may differ from T due to subsampling within the model architecture.

Next, frame-level (strong) class prediction probabilities are computed using a linear layer followed by a sigmoid activation:

$$\{\hat{\mathbf{o}}_s^{(\mathrm{strong})}\}_{s=1}^{S} = \sigma\big(\mathbf{W_h}\{\hat{\mathbf{e}}\}_{s=1}^{S} + \mathbf{b_h}\big), \tag{4}$$

where $\mathbf{W_h} \in \mathbb{R}^{C\times D}$ and $\mathbf{b_h} \in \mathbb{R}^C$ are the learnable weight matrix and bias vector of the output layer, and σ is the element-wise sigmoid function. To obtain clip-level (weak) predictions $\hat{\mathbf{o}}^{(\mathrm{weak})}$ from the frame-level outputs $\{\hat{\mathbf{o}}_s^{(\mathrm{strong})}\}_{s=1}^{S}$, we compute class-specific attention weights over time. These are generated using a separate attention head:

$$\{\boldsymbol{\alpha}_s\}_{s=1}^{S} = \mathrm{softmax}\big(\mathbf{W_a}\{\hat{\mathbf{e}}_s\}_{s=1}^{S} + \mathbf{b_a}\big), \tag{5}$$

where $\mathbf{W_a} \in \mathbb{R}^{C\times D}$ and $\mathbf{b_a} \in \mathbb{R}^C$ are the weight matrix and the bias vector of the attention head, and the softmax function is applied independently for each class across time steps. The final clip-level prediction is then obtained as the attention-weighted average of the strong outputs:

$$\hat{\mathbf{o}}^{(\mathrm{weak})} = \sum_{s=1}^{S} \boldsymbol{\alpha}_s \cdot \hat{\mathbf{o}}_s^{(\mathrm{strong})}.$$

We train the model using a loss function that combines binary cross-entropy (BCE) losses for both strong and weak labels, balanced by a weighting factor λ:

$$\begin{aligned}\mathcal{L} = \lambda \cdot \frac{1}{SC} \sum_{s=1}^{S}\sum_{c=1}^{C} \mathrm{BCE}\big(\hat{o}_{s,c}^{(\mathrm{strong})},\ y_{s,c}^{(\mathrm{strong})}\big) \\ + (1-\lambda)\cdot \frac{1}{C}\sum_{c=1}^{C} \mathrm{BCE}\big(\hat{o}_{c}^{(\mathrm{weak})},\ y_{c}^{(\mathrm{weak})}\big).\end{aligned} \tag{6}$$

In all experiments, we set $\lambda = 0.5$.

4. SOURCE SEPARATION

The second stage employs a modified ResUNet architecture [2], [4], which takes the multi-channel mixture waveform and a target class label as input to produce an estimated single-channel source. The ResUNet operates on magnitude spectrograms and predicts magnitude and phase masks to reconstruct the target's complex spectrogram, which is then inverted to the time domain via the inverse Short-Time Fourier Transform.

4.1. Time-FiLM Conditioning

In addition to standard clip-level conditioning via Feature-wise Linear Modulation (FiLM) [5], we provide finer-grained temporal guidance by integrating a dedicated, trainable SED model—referred to as the Stage 2 SED model—into the separator architecture. We refer to this approach as *Time-FiLM* as it generalizes FiLM [5] by incorporating a temporal dimension. The Stage 2 SED model is initialized with the weights of the fine-tuned model described in Section 3, but is subsequently optimized jointly with the separator.

For a class predicted by the Stage 1 SED model, the corresponding class activity probability map obtained by the Stage 2 SED model is passed through a feedforward network to generate a sequence of embedding vectors. These embeddings are transformed into a sequence of channel-wise scale and shift parameters for each conditioned layer in the ResUNet. To align those sequences with the feature maps, the sequences are linearly interpolated in time. Finally, Time-FiLM

modulation is applied by scaling and shifting the feature maps with their time-aligned scale and shift parameters.

4.2. Embedding Injection

To further enhance separation quality, we inject intermediate representations from the Stage 2 SED model into the ResUNet's latent space. Inspired by [11], we extract hidden representations after each of the N blocks in the Stage 2 SED model, denoted by $Z_{\text{TF}}^{(i)} \in \mathbb{R}^{C_{\text{TF}} \times F'_{\text{TF}} \times T'_{\text{TF}}}$, and compute a weighted combination using learnable weights:

$$\alpha_i = \frac{\exp(w_i)}{\sum_{j=1}^{N} \exp(w_j)}, \quad Z_{\text{TF}} = \sum_{i=1}^{N} \alpha_i \cdot Z_{\text{TF}}^{(i)}, \tag{7}$$

where w_i are learned scalar parameters. The resulting fused representation Z_{TF} is projected through a linear transformation and then interpolated along both the frequency and time dimensions to match the shape of the ResUNet's latent features $Z_{\text{RN}} \in \mathbb{R}^{C_{\text{RN}} \times F'_{\text{RN}} \times T'_{\text{RN}}}$. After normalizing both the fused representation and the ResUNet's latent features, we add them element-wise.

4.3. Dual-Path RNN

To capture long-range dependencies, we integrate a Dual-Path Recurrent Neural Network (DPRNN) [9] into the ResUNet's embedding space. The DPRNN consists of two blocks, each with two stacked bidirectional GRUs [12]: the first processes along the temporal dimension, the second along the frequency dimension.

4.4. Iterative Refinement

Instead of generating the target source in a single forward pass, we treat separation as an iterative process. At each step, the estimated single-channel output is stacked with the original M-channel mixture to form a new $(M+1)$-channel input. The ResUNet's first convolution is adapted for $M+1$ channels, with the extra channel initialized to zeros during the first iteration.

This iterative mechanism allows the separator to refine its predictions over multiple steps. During training, the number of iterations is randomly sampled between 1 and a predefined maximum. To reduce the memory footprint during training, gradients are detached between iterations and only the final step contributes to parameter updates.

5. EXPERIMENTAL SETUP

This section details the experimental setup, including the dataset, evaluation metric, model configurations, and the design used to assess our key contributions.

5.1. Dataset

Training and validation use the DCASE 2025 Task 4 (S5) development set, with 4-channel 10-second mixtures synthesized on-the-fly at 32kHz using a modified SpatialScaper library [13]. Training data is randomly sampled, while validation mixtures are generated deterministically. Each mixture combines the following four components:

- **Target sound events** $\boldsymbol{s}_k$: Anechoic recordings from 18 target classes (e.g., *AlarmClock*, *FootStep*, *Speech*) are used, mainly from FSD50K [14] and EARS [15], with additional data from NTT. The training set includes 7,336 samples (33–2,063 per class), and the validation set has 317 samples (1–37 per class), showing significant class imbalance.
- **Interference events** $\boldsymbol{s}_j$: We utilize isolated recordings of 94 non-target classes from the Semantic Hearing dataset [16]. This set includes 6,184 training and 845 validation samples, displaying class distribution inbalance similar to the target sound events.
- **Room impulse responses** $\boldsymbol{h}$: For acoustic environment simulations we employ multichannel RIRs captured from various real indoor locations. Our setup incorporates 5 RIRs for training and 3 for validation. The recordings are partially sourced from the FOA-MEIR dataset [17], with additional samples newly recorded by NTT.
- **Ambient noise** $\boldsymbol{n}$: We include real-world background noise recordings obtained from the FOA-MEIR dataset [17]. The noise dataset contains 59 training and 21 validation samples.

This synthetic setup also provides precise event onsets and offsets, which are required for training the SED models (see Section 3). For testing, we use the pre-synthesized mixtures provided by the DCASE challenge organizers. For all sets, each mixture contains between one and three target events and up to two interfering events. The target events have per-event Signal-to-Noise Ratios (SNRs) ranging from 5 to 10 dB, while interfering events have SNRs between 0 and 15 dB.

5.2. Evaluation Metric

We evaluate our system using Class-Aware Signal-to-Distortion Ratio improvement (CA-SDRi) [2], a metric that jointly measures source separation quality and classification accuracy. Let C be the set of ground truth classes and $\hat{C}$ the set of predicted classes. For correctly predicted classes $c_k \in C \cap \hat{C}$, we compute the SDR improvement between the estimated signal $\hat{\boldsymbol{s}}_k$ and the corresponding target source relative to the mixture's first channel $x^{(1)}$. For simplicity, we denote the target direct-path signal $\boldsymbol{s}_k^{(d,m)}$ as $\boldsymbol{s}_k$ in the following equations.

$$\text{SDRi}(\hat{\boldsymbol{s}}_k, \boldsymbol{s}_k, \boldsymbol{x}^{(1)}) = \text{SDR}(\hat{\boldsymbol{s}}_k, \boldsymbol{s}_k) - \text{SDR}(\boldsymbol{x}^{(1)}, \boldsymbol{s}_k) \tag{8}$$

with SDR defined as:

$$\text{SDR}(\hat{\boldsymbol{x}}, \boldsymbol{x}) = 10 \log_{10} \left(\frac{|\boldsymbol{x}|^2}{|\boldsymbol{x} - \hat{\boldsymbol{x}}|^2} \right) \tag{9}$$

The final CA-SDRi score is computed by averaging over all unique classes in $C \cup \hat{C}$:

$$\text{CA-SDRi} = \frac{1}{|C \cup \hat{C}|} \sum_{c_k \in C \cup \hat{C}} P_{c_k}, \tag{10}$$

where $P_{c_k} = \text{SDRi}(\hat{\boldsymbol{s}}_k, \boldsymbol{s}_k, \boldsymbol{x}^{(1)})$ if $c_k \in C \cap \hat{C}$, and $P_{c_k} = 0$ otherwise.

5.3. Class Detection Model

The model used for detecting active acoustic events is based on the M2D architecture [3], [8], pre-trained for audio tagging on AudioSet [18]. We use a checkpoint further fine-tuned for sound event detection (SED) on temporally strong labels from AudioSet Strong [19], following the procedure in [7]. This model is chosen for its state-of-the-art SED performance, as shown in [7]. The model is trained as outlined in Section 3 for 22,000 steps with a batch size of 32. We employ a cosine learning rate schedule with 4,000 warm-up steps and utilize the ADAM optimizer [20] with no weight decay. The initial learning rate is set to 1×10^{-3}, and a layer-wise learning rate decay factor of 0.8 is applied to adjust the learning rates across different layers of the model.

5.4. Source Separation Model

We use a ResUNet architecture [4], [21] for sound source separation, operating at 32kHz with a window size of 2048 and a hop size of 160 samples, matching the baseline. The model is initialized with pre-trained weights from AudioSep [22], a language-conditioned separation framework. While AudioSep was originally trained with a 320-sample hop size, we found a hop size of 160 samples to perform better in our setup. For fine-tuning the separator, we use separate

Config	M2D [2]	M2D (Ours)	Oracle
DCASE Baseline [2]	11.03	–	–
AudioSep-SED	**12.71±0.03**	**13.42±0.11**	**15.29±0.13**
- w/o Time-FiLM	12.41±0.03	13.07±0.06	14.82±0.06
- w/o Embedding Injection	12.46±0.01	13.12±0.07	14.91±0.09
- w/o Trainable S2-SED	11.95±0.04	12.53±0.03	14.04±0.04
AudioSep-SED + DPRNN	12.55±0.07	13.31±0.07	15.20±0.12

Table 1: CA-SDRi scores (↑) for different system variants on the development test split, where *AudioSep-SED w/o Trainable S2-SED* represents the AudioSep-SED configuration, where the Stage 2 SED model is not trainable. Columns correspond to class detection using the baseline M2D, our fine-tuned M2D, and oracle targets. Bold indicates the best per column.

learning rates: 2×10^{-4} for pre-trained components, 5×10^{-4} for DPRNN parts and 1×10^{-3} for newly introduced parts such as the Time-FiLM embedding layers. Additionally, we train the Stage 2 SED model using a learning rate of 4×10^{-4} and a learning rate decay factor of 0.9. Training runs for 450,000 steps with an batch size of 8, using the Adam optimizer [20] (no weight decay) and a cosine learning rate schedule with 12,000 warm-up steps.

5.5. Experimental Design

To assess the efficacy of the proposed techniques for enhancing source separation, we conduct a series of experiments structured as follows. Initially, we evaluate the impact of Time-FiLM and Embedding Injection, both of which utilize the Stage 2 SED model. We also investigate the benefits of allowing this model to be trainable during the training phase of the separation model. We refer to the configuration incorporating Time-FiLM, Embedding Injection, and a trainable Stage 2 SED model as *AudioSep-SED*. This model serves as the foundation for subsequent experiments. Building upon it, we integrate a Dual-Path RNN (DPRNN) into the latent space to potentially enhance the model's ability to capture long-range dependencies. Subsequently, we explore the iterative refinement approach by training models with varying numbers of iterations (2, 3, and 4) and assessing their performance when employing up to 10 iterations during inference.

6. RESULTS

The Stage 1 sound event detection model proposed in Section 3 increases test set accuracy by 8 points over the DCASE baseline, from 59.8% to 67.8%.

Table 1 compares the performance of the DCASE baseline, AudioSep-SED, and AudioSep-SED + DPRNN, highlighting the impact of each technique on the test split of the DCASE 2025 S5 development set. Performance is measured using CA-SDRi under three Stage 1 models: (1) baseline M2D tagger, (2) our M2D model, and (3) oracle targets. Results are averaged over two runs.

The second section in Table 1 shows that AudioSep-SED outperforms the DCASE baseline system. Integrating Time-FiLM, Embedding Injection, and a trainable Stage 2 SED model improves performance regardless of Stage 1 predictions, with the trainable Stage 2 SED yielding the largest gain. While Time-FiLM contributes more than Embedding Injection, their combination with a trainable SED achieves the best results. With baseline M2D predictions, this setup reaches a CA-SDRi of 12.71 vs. 11.03 for the DCASE baseline. Using our M2D model, the score further increases to 13.42.

Adding a DPRNN to AudioSep-SED improves performance on the *validation split*, increasing CA-SDRi from 13.79 to 14.35 using baseline M2D predictions. However, it underperforms on the *test split* (third section in Table 1), suggesting a distribution mismatch.

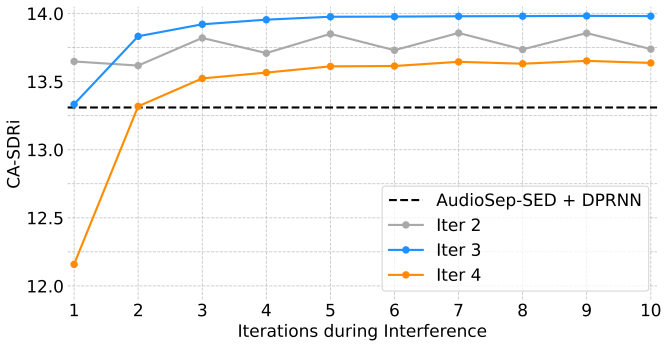

Fig. 2: Performance of the iterative refinement experiments compared to the AudioSep-SED + DPRNN configuration evaluated on the test set. *Iter* {*2,3,4*} denotes experiments with the maximum training iterations set to 2, 3, or 4, respectively. Class detection is performed using our fine-tuned M2D model.

Following standard practice, we select the best validation setup for further experiments; thus, DPRNN is included in our iterative refinement runs.

Figure 2 shows the performance of models trained with up to 2, 3, or 4 iterations and evaluated using 1–10 iterations at inference. Iterative refinement clearly outperforms the AudioSep-SED + DPRNN setup, with most gains occurring in early iterations and diminishing improvements thereafter—indicating convergence after a few cycles. Notably, using more inference iterations than used during training can still boost performance; e.g., a model trained with 3 iterations reached 13.92 after 3 inference steps and 13.98 after 10.

Our iterative refinement models generally enhance performance as the number of inference iterations increases, although the performance changes between consecutive iterations can be inconsistent and occasionally negative. A notable pattern emerges where odd-numbered iterations frequently outperform their even-numbered counterparts, a trend most pronounced in models trained with two iterations (Iter 2). However, this odd-even performance gap narrows in models trained with three or four iterations (Iter 3 and Iter 4), which suggests that extending the training iterations helps stabilize the refinement process.

Furthermore, Iter 3 outperforms both Iter 2 and Iter 4, indicating it strikes a good balance between training complexity and output quality. That said, because these experiments relied on a single run per configuration, the observed results should be interpreted with caution, as they may be influenced by random variability.

7. CONCLUSION

We introduced a two-stage audio source separation system that leverages sound event detection (SED) models to (1) identify active events and (2) guide the separator via temporal conditioning (Time-FiLM) and Embedding Injection. Evaluated on the DCASE 2025 S5 dataset, our results show that SED improves event detection in Stage 1 and provides effective guidance for Stage 2 separation. We also proposed an iterative refinement strategy, where outputs are recursively fed back into the model, which progressively enhanced the separation quality.

8. ACKNOWLEDGMENT

The computational results presented were obtained in part using the Austrian Scientific Cluster (ASC) and the Linz Institute of Technology (LIT) AI Lab Cluster. The LIT AI Lab is supported by the Federal State of Upper Austria. Gerhard Widmer's work is supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement No 101019375 (Whither Music?).

REFERENCES

[1] M. Yasuda, B. T. Nguyen, N. Harada, R. Serizel, M. Mishra, M. Delcroix, S. Araki, D. Takeuchi, D. Niizumi, Y. Ohishi *et al.*, "Description and discussion on dcase 2025 challenge task 4: Spatial semantic segmentation of sound scenes," *arXiv preprint arXiv:2506.10676*, 2025.

[2] B. T. Nguyen, M. Yasuda, D. Takeuchi, D. Niizumi, Y. Ohishi, and N. Harada, "Baseline systems and evaluation metrics for spatial semantic segmentation of sound scenes," *arXiv preprint arXiv:2503.22088*, 2025.

[3] D. Niizumi, D. Takeuchi, Y. Ohishi, N. Harada, and K. Kashino, "Masked modeling duo: Towards a universal audio pre-training framework," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2024.

[4] Q. Kong, K. Chen, H. Liu, X. Du, T. Berg-Kirkpatrick, S. Dubnov, and M. D. Plumbley, "Universal source separation with weakly labelled data," *CoRR*, vol. abs/2305.07447, 2023.

[5] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. C. Courville, "Film: Visual reasoning with a general conditioning layer," in *Proc. of the AAAI conference on artificial intelligence*, 2018.

[6] T. Morocutti, F. Schmid, J. Greif, P. Primus, and G. Widmer, "Transformer-aided audio source separation with temporal guidance and iterative refinement," DCASE2025 Challenge, Tech. Rep., 2025.

[7] F. Schmid, T. Morocutti, F. Foscarin, J. Schlüter, P. Primus, and G. Widmer, "Effective pre-training of audio transformers for sound event detection," in *Proc. of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2025.

[8] D. Niizumi, D. Takeuchi, Y. Ohishi, N. Harada, M. Yasuda, S. Tsubaki, and K. Imoto, "M2D-CLAP: masked modeling duo meets CLAP for learning general-purpose audio-language representation," in *Proc. of the Interspeech Conference*, 2024.

[9] Y. Luo, Z. Chen, and T. Yoshioka, "Dual-path RNN: efficient long sequence modeling for time-domain single-channel speech separation," in *Proc. of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[10] Y. Yuan, X. Liu, H. Liu, M. D. Plumbley, and W. Wang, "Flowsep: Language-queried sound separation with rectified flow matching," in *Proc. ICASSP*, 2025.

[11] C. Hernandez-Olivan, M. Delcroix, T. Ochiai, D. Niizumi, N. Tawara, T. Nakatani, and S. Araki, "Soundbeam meets m2d: Target sound extraction with audio foundation model," in *Proc. of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2025.

[12] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734.

[13] I. R. Román, C. Ick, S. Ding, A. S. Roman, B. McFee, and J. P. Bello, "Spatial scaper: A library to simulate and augment soundscapes for sound event localization and detection in realistic rooms," in *Proc. of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024.

[14] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "FSD50K: an open dataset of human-labeled sound events," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 2022.

[15] J. Richter, Y. Wu, S. Krenn, S. Welker, B. Lay, S. Watanabe, A. Richard, and T. Gerkmann, "EARS: an anechoic fullband speech dataset benchmarked for speech enhancement and dereverberation," in *Proc. of the Interspeech Conference*, 2024.

[16] B. Veluri, M. Itani, J. Chan, T. Yoshioka, and S. Gollakota, "Semantic hearing: Programming acoustic scenes with binaural hearables," in *Proc. of the ACM Symposium on User Interface Software and Technology, UIST*, 2023.

[17] M. Yasuda, Y. Ohishi, and S. Saito, "Echo-aware adaptation of sound event localization and detection in unknown environments," in *Proc. of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.

[18] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[19] S. Hershey, D. P. W. Ellis, E. Fonseca, A. Jansen, C. Liu, R. C. Moore, and M. Plakal, "The benefit of temporally-strong labels in audio event classification," in *Proc. of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.

[21] Q. Kong, K. Chen, H. Liu, X. Du, T. Berg-Kirkpatrick, S. Dubnov, and M. D. Plumbley, "Universal source separation with weakly labelled data," *arXiv preprint arXiv:2305.07447*, 2023.

[22] X. Liu, Q. Kong, Y. Zhao, H. Liu, Y. Yuan, Y. Liu, R. Xia, Y. Wang, M. D. Plumbley, and W. Wang, "Separate anything you describe," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2024.