APSIPA Transactions on Signal and Information Processing, 2025, 14, e25

Original Paper

Stabilizing and Enhancing Remixing-based Unsupervised Sound Source Separation

Kohei Saijo* and Tetsuji Ogawa

Waseda University, Tokyo, Japan

ABSTRACT

In this paper, we present methods to stabilize training and enhance the performance of Self-Remixing, an unsupervised source separation framework. Self-Remixing trains a model to reconstruct original mixtures by separating pseudo-mixtures, which are generated by first separating the observed mixtures and then remixing the resulting sources. Although this approach has shown promising results, it suffers from two notable limitations: *i)* reliance on pre-trained models, and *ii)* suboptimal performance on certain metrics, particularly word error rate (WER). To address these issues, we propose techniques that *i)* stabilize the training process, enabling end-to-end training from scratch without pre-training, and *ii)* identify the causes of WER degradation, introducing a tailored loss function to mitigate them. Our results demonstrate that, with improved remixing strategies and a carefully designed loss function, Self-Remixing achieves competitive performance even when trained entirely from scratch.

Keywords: Unsupervised source separation, remixing, universal sound separation

*Corresponding author: Kohei Saijo, saijo@pcl.cs.waseda.ac.jp. This work was partially supported by JSPS KAKENHI Grant Number JP24KJ2096.

Received 14 June 2025; revised 05 August 2025; accepted 26 August 2025
ISSN 2048-7703; DOI 10.1561/116.20250044

1 Introduction

Over the past decade, neural network-based approaches to source separation have seen remarkable progress [12, 59, 23, 40, 49, 35]. These models are typically trained in a supervised manner using paired data consisting of mixtures and corresponding clean source signals. However, collecting such annotated data in real-world environments is challenging, and thus, synthetic mixtures generated by simulation toolkits [37] are commonly used for training. A major limitation of this approach is the domain mismatch between synthetic and real mixtures, which often leads to performance degradation due to differences in factors such as reverberation, noise characteristics, and recording conditions.

To overcome this limitation, unsupervised source separation methods have been proposed. These methods train models directly on real-recorded mixtures without access to ground-truth sources [46, 9, 43, 1, 34, 24, 50, 36]. In monaural settings, one of the pioneering approaches is Mixture Invariant Training (MixIT) [55, 54, 7, 39], which trains models to separate a mixture of mixtures (MoMs), created by summing multiple mixtures. Despite its success, MixIT often suffers from the *over-separation* problem, where the target sources are overly suppressed or distorted. This issue arises from the mismatch between MoMs and natural mixtures, as the former tends to contain more sources.

Several approaches have been proposed to address the over-separation issue. One such method, Teacher-Student MixIT [61], trains a student model using outputs from a teacher model pre-trained with MixIT. RemixIT [44] similarly adopts a teacher-student framework but trains the student to separate *pseudo-mixtures* generated by shuffling and remixing the teacher's outputs. By exposing the student model to a diverse range of pseudo-mixtures in a similar spirit to the dynamic mixing strategy used in [60], RemixIT achieves improved performance. Further gains can be obtained by updating the teacher's weights using the student's ones by, e.g., exponential moving average (EMA) update.

However, these self-training methods are prone to instability, especially when the model begins to *under-separates* sources. A common strategy in scenarios where the number of sources is unknown is to assume the maximum number of sources and always produce that number of outputs [52]. In such cases, when the teacher model produces near-zero outputs, the student learns to reproduce this behavior, as it relies on the teachers outputs for supervision. As training progresses and the teacher is updated using the student's outputs, both models may converge toward a degenerate solution that reproduces the input mixture along with near-zero signals, thus failing to perform any meaningful separation.

To address this issue, Self-Remixing [32] was recently proposed. Like RemixIT, it trains models to separate pseudo-mixtures but crucially differs by using the original mixtures as static supervision targets. This design mitigates the instability commonly observed in self-training frameworks, where the teacher model progressively under-separates sources. Nevertheless, Self-Remixing (as well as RemixIT) still suffers from two key limitations: (*i*) In [32], the model is assumed to be pre-trained with MixIT to ensure that the resulting pseudo-mixtures remain acoustically similar to natural mixtures. However, it remains unclear how well the pre-trained model needs to perform for Self-Remixing to be effective. Moreover, the requirement of this two-stage training procedure, pre-training followed by fine-tuning with Self-Remixing, complicates the training pipeline. (*ii*) The loss functions commonly used in prior work, negative signal-to-noise ratio (SNR) [55, 44], tend to produce suboptimal results for metrics sensitive to magnitude errors, such as word error rate (WER).

This work addresses these limitations by (*i*) enabling Self-Remixing to work without pre-training, and (*ii*) introducing a loss function designed to more strongly penalize the magnitude estimation errors. Through detailed analysis, we found that outputs from a randomly initialized separation model closely resemble the input mixture, and that pseudo-mixtures constructed from these outputs can effectively act as MoMs for training. Nonetheless, training from scratch remains unstable due to the presence of trivial solutions, such as copying the input mixture, that minimize the loss without any separation. To address this, we introduce novel remixing algorithms that stabilize training and remove the need for pre-training. Additionally, to reduce the severe distortion often observed in remixing-based methods, we introduce a loss function that more effectively penalizes magnitude estimation errors. Our code[1] and the audio examples[2] are publicly available to facilitate reproducibility and further research.

This paper extends our previous work [31] by contributing: (i) a deeper analysis of Self-Remixing that clarifies how separation ability emerges from scratch; (ii) a detailed description and empirical validation of the proposed remixing algorithm; (iii) a study of loss functions effective for remixing-based methods, supported by qualitative analysis and experiments; and (iv) an extension of Self-Remixing on mapping-based separation models, which are more challenging to train in an unsupervised manner due to their higher degrees of freedom..

The remainder of this paper is organized as follows. Section 2 provides an overview of prior work on single-channel unsupervised source separation. Section 3 presents an analysis of the behavior of Self-Remixing and introduces remixing algorithms to improve training stability. Section 4 investigates loss functions suitable for remixing-based methods. Section 5 describes the experimental setup, and Section 6 reports and discusses the experimental results.

[1]https://github.com/kohei0209/self-remixing
[2]https://kohei0209.github.io/selfremixing-demo

2 Prior Work on Single-channel Unsupervised Sound Separation

Let us denote a mini-batch of B mixtures as $\boldsymbol{x} \in \mathbb{R}^{B \times T}$, where T denotes the number of samples in the time domain. Each mixture in the batch, denoted as x_b $(b = 1, \ldots, B)$, contains up to K sources, where K is assumed to be known in this work. A separation model with $N_{\mathcal{S}}$ output channels is denoted as $f_{\mathcal{S}}$ with its parameters $\boldsymbol{\theta}_{\mathcal{S}}$.

2.1 Mixture Invariant Training

MixIT trains a model in an unsupervised manner using a mixture of mixtures (MoMs). Let $B'(\leq B)$ mixtures in a mini-batch be $x_1, \ldots, x_{B'}$. MixIT adds them together to make an MoM $\bar{x} \in \mathbb{R}^T$. In total, $\bar{B} \triangleq \lfloor B/B' \rfloor$ MoMs are generated from a mini-batch of B mixtures. MixIT trains the model to separate each MoM into each source:

$$\hat{\boldsymbol{s}} = f_{\mathcal{S}}(\bar{x}; \boldsymbol{\theta}_{\mathcal{S}}) \in \mathbb{R}^{N_{\mathcal{S}} \times T}. \tag{1}$$

The MixIT loss is computed between the separated signals $\hat{\boldsymbol{s}}$ and the individual mixtures, as described in [55]:

$$\mathcal{L}_{\text{MixIT}} = \min_{\boldsymbol{A}} \sum_{b'=1}^{B'} \mathcal{L}(x_{b'}, [\boldsymbol{A}\hat{\boldsymbol{s}}]_{b'}), \tag{2}$$

where $\mathcal{L}$ is a loss function. The mixing matrix $\boldsymbol{A} \in \mathbb{B}^{B' \times N_{\mathcal{S}}}$ assigns each $\hat{s}_n$ to the original mixtures. The number of mixtures in an MoM B' is typically set to two, and $N_{\mathcal{S}}$ is set to satisfy $N_{\mathcal{S}} \geq B'K$.

While MixIT has enabled unsupervised learning, models often produce more sources than are actually present in inference. Such an *over-separation* problem happens because *i*) MoMs contain more sources than individual mixtures do and *ii*) the model has more output channels than the actual number of sources (i.e., $N_{\mathcal{S}} \geq B'K > K$). To mitigate the first problem, an auxiliary *source sparsity loss* that encourages sparsity of the separation outputs has been proposed [54]:

$$\mathcal{L}_{\text{sparsity}} = \frac{1}{N} \frac{||\boldsymbol{r}||_1}{||\boldsymbol{r}||_2}, \quad \boldsymbol{r} = \sum_n \sqrt{\frac{1}{T} \sum_t |\hat{s}_{n,t}|^2} \tag{3}$$

where $|| \cdot ||_p$ is ℓ_p norm. The resulting sparsity-induced MixIT loss is

$$\mathcal{L}_{\text{MixIT+sparsity}} = \mathcal{L}_{\text{MixIT}} + \gamma \mathcal{L}_{\text{sparsity}}, \tag{4}$$

where the weight γ is tuned based on the number of output channels $N_{\mathcal{S}}$. The sparsity loss is effective when the number of output channels is often more than the number of sources in MoMs ($N_{\mathcal{S}} > B'K$).

2.2 Mixture Permutation Invariant Training

Mixture permutation invariant training (MixPIT) aims to address the over-separation [15] by reducing the number of output channels $N_{\mathcal{S}}$. Specifically, while MixIT typically sets $N_{\mathcal{S}} = B'K$, MixPIT reduces $N_{\mathcal{S}}$ to K, the maximum number of sources in individual mixtures, where the model tries to estimate the individual mixtures directly (i.e., $B' = N_{\mathcal{S}} = K$). However, since the model does not know which source originally belonged to which mixture, separating mixtures from MoMs directly is essentially challenging. Indeed, as reported in [15], MixPIT underperforms MixIT even when $K = 2$.

2.3 RemixIT

RemixIT is a self-training framework designed to refine a pre-trained separation model [45]. Although originally developed for speech enhancement [44], we extend RemixIT to the general sound separation task and introduce several techniques to stabilize the training, one of the main contributions of this work. In addition to the student model $f_{\mathcal{S}}$, RemixIT introduces a teacher model $f_{\mathcal{T}}$ with $N_{\mathcal{T}}$ output channels and parameters $\boldsymbol{\theta}_{\mathcal{T}}$. An overview of the RemixIT framework is illustrated in Figure 1.

In RemixIT, the teacher model $f_{\mathcal{T}}$ first separates the input mixtures $\boldsymbol{x}$ into estimated sources: $\tilde{\boldsymbol{s}} = f_{\mathcal{T}}(\boldsymbol{x}; \boldsymbol{\theta}_{\mathcal{T}}) \in \mathbb{R}^{B \times N_{\mathcal{T}} \times T}$. Here, the number of output channels $N_{\mathcal{T}}$ may exceed the expected maximum number of sources K. For instance, when $f_{\mathcal{T}}$ is pre-trained using MixIT, it typically satisfies $N_{\mathcal{T}} \geq 2K$. In such cases, we select the K sources with the highest powers. We then enforce mixture consistency [53] to ensure that the selected sources sum to the original mixtures, $\sum_{n=1}^{K} \tilde{s}_{b,n} = x_b$. Next, we apply a source shuffling procedure within the batch to construct pseudo-mixtures $\tilde{\boldsymbol{x}}$. Specifically, each source $\tilde{s}_{b,n}$ is permuted across batches using an $B \times B$ permutation matrix $\boldsymbol{\Pi}_n$ for each n, and the pseudo-mixtures are obtained as:

$$\tilde{x}_b = \sum\nolimits_{n=1}^{K} \tilde{s}_{b,n}^{(\boldsymbol{\Pi})}, \qquad \tilde{s}_{b,n}^{(\boldsymbol{\Pi})} = [\boldsymbol{\Pi}_n \tilde{\boldsymbol{s}}_{:,n}]_b, \tag{5}$$

where $\tilde{\boldsymbol{s}}_{:,n} \triangleq [\tilde{s}_{1,n}, \dots, \tilde{s}_{B,n}] \in \mathbb{R}^{B \times T}$ denotes B separated sources from each output channel n. We refer to the operation in Equation (5) as *batch shuffle*, as it permutes data along the batch dimension. The student model $f_{\mathcal{S}}$ is then used to separate the pseudo-mixtures: $\hat{\boldsymbol{s}} = f_{\mathcal{S}}(\tilde{\boldsymbol{x}}; \boldsymbol{\theta}_{\mathcal{S}})$. As with the teacher model, we select the K most energetic sources when $K < N_{\mathcal{S}}$. The RemixIT loss is computed in a permutation-invariant manner using a brute-force search with $O(K!)$ complexity, between the sources generated by the teacher and those generated by the student [59].

$$\mathcal{L}_{\text{RemixIT}}^{(b)} = \min_{\boldsymbol{P}_b} \frac{1}{K} \sum\nolimits_{n=1}^{K} \mathcal{L}(\tilde{s}_{b,n}^{(\boldsymbol{\Pi})}, [\boldsymbol{P}_b \hat{\boldsymbol{s}}_b]_n), \tag{6}$$

where $\boldsymbol{P}_b$ is a $K \times K$ permutation matrix.

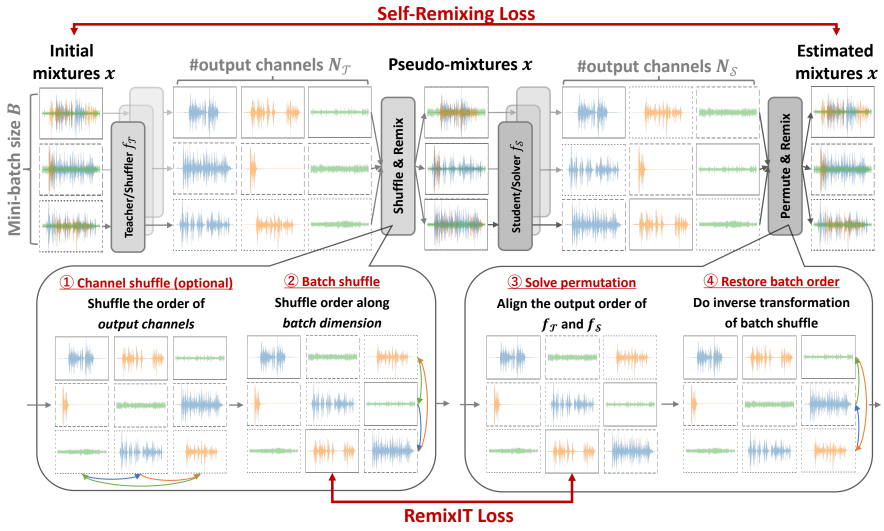

Self-Remixing Loss
Initial mixtures x
Mini-batch size B
Teacher/Shuffler $f_\mathcal{T}$
#output channels $N_\mathcal{T}$
Shuffle & Remix
Pseudo-mixtures x
Student/Solver $f_\mathcal{S}$
#output channels $N_\mathcal{S}$
Permute & Remix
Estimated mixtures x
① Channel shuffle (optional)
Shuffle the order of output channels
② Batch shuffle
Shuffle order along batch dimension
③ Solve permutation
Align the output order of $f_\mathcal{T}$ and $f_\mathcal{S}$
④ Restore batch order
Do inverse transformation of batch shuffle
RemixIT Loss

Figure 1: Overview of RemixIT and Self-Remixing. Teacher/shuffler model $f_\mathcal{T}$ first separates input mixtures. Separated signals are then shuffled along output-channel and batch dimensions (① channel shuffle and ② batch shuffle) and summed up to make pseudo-mixtures. Student/solver model $f_\mathcal{S}$ separates pseudo-mixtures. Output-channel order of $f_\mathcal{S}$ is aligned with that of $f_\mathcal{T}$ by minimizing the loss between them (i.e., RemixIT loss, ③). Next, reverse transformation of batch shuffle (④) is done to restore the order of batch dimension (i.e., signals after ④ have the same order as those in ①). Finally, outputs of $f_\mathcal{S}$ are summed up to reconstruct input mixtures to compute Self-Remixing loss.

The student model parameters $\boldsymbol{\theta}_\mathcal{S}$ are updated using gradient descent. Meanwhile, the teacher model parameters $\boldsymbol{\theta}_\mathcal{T}$ can also be updated using the student parameters. While various update strategies have been explored [42, 57] in the original RemixIT work [44], we adopt the exponential moving average (EMA) update due to its proven effectiveness and efficiency [42, 13]. At the end of each epoch, the teacher model is updated as:

$$\boldsymbol{\theta}_\mathcal{T}^{(j+1)} = \alpha \boldsymbol{\theta}_\mathcal{T}^{(j)} + (1-\alpha)\boldsymbol{\theta}_\mathcal{S}^{(j)}, \tag{7}$$

where $\alpha \in [0, 1]$ is the EMA coefficient and j denotes the epoch index.

2.4 Potential Instability of Teacher-student Learning in General Sound Separation

Although RemixIT has demonstrated strong performance in speech enhancement [45], its teacher-student learning framework can become unstable when the number of sources in mixtures is often smaller than the number of output channels. This condition is expected to arise frequently in realistic general sound separation scenarios [52, 4].

In unsupervised sound separation, where the actual number of sources in each mixture, denoted as $\bar{K}$ ($\bar{K} \leq K$), is unknown even during training, a common strategy is to assume a maximum number of sources K and design the separation model to have K output channels. Ideally, the model should output $\bar{K}$ meaningful sources and $K - \bar{K}$ silent (zero-valued) signals [52]. However, when $\bar{K} < K$ frequently holds, the student model tends to learn *under-separation* behavior because it is trained using the teacher outputs that often include low-energy or near-zero signals. As the teacher model is updated using the student model parameters (Equation (7)), this under-separation behavior propagates to the teacher, leading to unstable training. Indeed, in our experiments, RemixIT failed to achieve stable training when $\bar{K} < K$ occurred frequently, as further discussed in Section 6.3.

2.5 Self-remixing

Self-Remixing is designed to address the instability issue associated with self-training, as discussed in Section 2.4, by using the original mixture as supervision rather than relying on the teacher's outputs, as illustrated in Figure 1. Unlike RemixIT, Self-Remixing does not follow a teacher-student learning paradigm. Instead, $f_{\mathcal{T}}$ and $f_{\mathcal{S}}$ are referred to as the *shuffler* and *solver*, respectively. The shuffler model decomposes the initial mixtures and *shuffles* the outputs to create pseudo-mixtures, while the solver attempts to *solve* the task by recovering the original mixtures from these pseudo-mixtures.

In Self-Remixing, the outputs of $f_{\mathcal{T}}$ and $f_{\mathcal{S}}$ are first aligned using the optimal permutation matrix $\bar{\boldsymbol{P}}_b$ obtained in Equation (6). The original order of sources, prior to the batch shuffle in Equation (5), is then restored:

$$\hat{s}_{b,n}^{(\boldsymbol{\Pi}^{-1})} = [\boldsymbol{\Pi}_n^{-1}\hat{\boldsymbol{s}}_{:,n}^{(\bar{\boldsymbol{P}})}]_b, \qquad \hat{\boldsymbol{s}}_b^{(\bar{\boldsymbol{P}})} = \bar{\boldsymbol{P}}_b\hat{\boldsymbol{s}}_b, \tag{8}$$

where $\hat{\boldsymbol{s}}_{:,n}^{(\bar{\boldsymbol{P}})} \triangleq [\hat{s}_{1,n}^{(\bar{\boldsymbol{P}})}, \ldots, \hat{s}_{B,n}^{(\bar{\boldsymbol{P}})}] \in \mathbb{R}^{B\times T}$ represents B separated sources from each output channel n. After aligning the outputs from $f_{\mathcal{T}}$ and $f_{\mathcal{S}}$ along with the output channel and batch dimensions, the reconstructed mixture $\hat{x}_b$ is obtained by summing the solver outputs $\hat{s}_{b,n}^{(\boldsymbol{\Pi}^{-1})}$:

$$\hat{x}_b = \sum\nolimits_{n=1}^{K} \hat{s}_{b,n}^{(\boldsymbol{\Pi}^{-1})}. \tag{9}$$

The loss is computed using the original mixtures as supervision:

$$\mathcal{L}_{\mathrm{Self-Remixing}}^{(b)} = \mathcal{L}(x_b, \hat{x}_b). \tag{10}$$

Note that while Self-Remixing utilizes the RemixIT loss to efficiently resolve channel and batch permutations, this loss is not used to update model parameters.[3] Since the computations involved in Equations (8)-(10) are lightweight, the overall training cost of Self-Remixing remains comparable to that of RemixIT.

3 Training with Self-Remixing and RemixIT from Scratch

3.1 Analysis of Self-Remixing and RemixIT

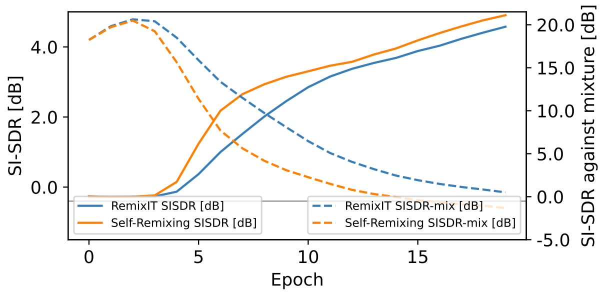
In this section, we analyze the behavior of RemixIT and Self-Remixing when the teacher model is randomly initialized, demonstrating that both models can be trained from scratch without pre-training.

To investigate the output trends of a randomly initialized model, we trained two architectures, a TF-masking-based Conformer [11] and a complex spectral mapping-based TF-GridNet [49], using RemixIT or Self-Remixing, both randomly initialized.[4] The input mixtures consisted of noisy, reverberant two-speaker mixtures from the WSJ-mix dataset (see Section 5.1). Both models had three output channels ($N_{\mathcal{T}} = N_{\mathcal{S}} = 3$) to estimate two speech and one noise signal, and the mixture consistency [53] was ensured in $f_{\mathcal{T}}$, as described in Section 2.3. Figure 2 shows the scale-invariant source-to-distortion ratio (SI-SDR) [17] of the teacher (shuffler) model's outputs relative to the clean reference signals (solid lines, denoted as SI-SDR) and the input mixtures (dotted lines, denoted as SI-SDR-mix). Note that higher SI-SDR-mix values indicate outputs that are acoustically similar to the input mixture. The upper and lower figures show the results for the Conformer and TF-GridNet models, respectively, with RemixIT and Self-Remixing denoted by blue and orange lines.

For the Conformer (upper figure), the SI-SDR-mix at epoch 0 exceeds 18 dB, indicating that the randomly initialized outputs are already close to the input mixtures. In contrast, for TF-GridNet (lower figure), the SI-SDR-mix starts around 5 dB but gradually increases to about 15 dB by epoch 5, indicating that its outputs become mixture-like after several training iterations. These results suggest that the outputs from randomly initialized models, regardless of whether they are based on masking or mapping, tend to converge toward the input mixtures in early training. This behavior allows the pseudo-mixtures created by remixing these outputs to be treated as

[3]Since the batch and channel orders of $\tilde{s}$ and $\hat{s}$ differ, directly aligning them via brute-force permutation search would be computationally expensive. Instead, permutations along the channel and batch dimensions are resolved independently. Although the RemixIT loss aids in channel permutation alignment, it does not contribute to the gradient in the Self-Remixing framework. Preliminary experiments showed that combining the RemixIT and Self-Remixing losses did not yield performance improvements.

[4]We used the default initialization strategy of Pytorch 1.12.1 [26]. We utilized the improved remixing algorithm described in Section 3.2.



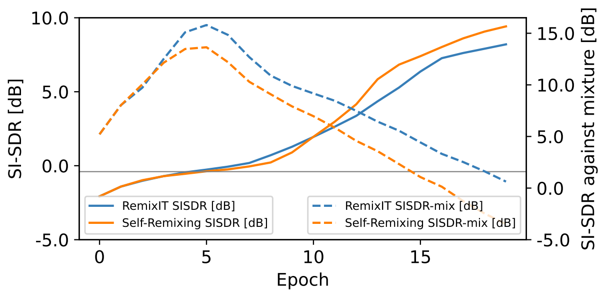


Figure 2: SI-SDR against clean signals (solid line) and input mixture (dotted line) at each epoch end. Upper and lower figures show results of Conformer and TF-GridNet, respectively. Blue and Orange lines correspond to RemixIT and Self-Remixing.

MoMs. Consequently, the RemixIT loss becomes equivalent to the MixPIT loss, where the model is trained to separate an MoM into an individual mixture. Self-Remixing similarly promotes the separation of MoMs because the model must recover the original mixtures to minimize the loss. Given that MixPIT has proven effective for source separation [15], these results support the efficacy of RemixIT and Self-Remixing even when training from scratch.

Our previous work [31] only analyzed the masking-based Conformer model, which already produced high SI-SDR-mix scores at initialization. In this work, we additionally analyze the complex spectral mapping-based TF-GridNet model, whose initial SI-SDR-mix is lower (around 5 dB), and show that it also converges toward the input mixture before learning to separate sources. This reveals a general pattern: RemixIT and Self-Remixing first produce outputs close to the mixture, then gradually improve separation. Their effectiveness across different architectures and estimation methods (masking and mapping) suggests that these approaches are broadly applicable. Audio examples from this analysis are available on our demo page.[5]

[5] https://kohei0209.github.io/selfremixing-demo/analysis.html

3.2 Remixing Algorithms for Stabilizing Training

The analysis in the previous section suggests that RemixIT and Self-Remixing can operate effectively without any pre-training. However, our preliminary experiments reveal that both algorithms can suffer from instability when trained from scratch. This section analyzes the cause of this instability and introduces methods to improve training stability.

3.2.1 Channel Shuffle (CS)

A key limitation of Self-Remixing is the presence of a trivial solution, where no actual separation occurs. Since Self-Remixing utilizes the input mixture as supervision, the model can minimize the loss by outputting the input mixture on one channel and zeros on the others (e.g., $\tilde{s}_{b,1} = x_b$ and $\tilde{s}_{b,n\neq 1} = 0$).

To mitigate this issue, we introduce *channel shuffle* (CS), a simple yet effective technique that randomly permutes the order of output channels before batch shuffling (Equation (5)). This approach is motivated by our observation that, when the model goes towards the trivial solution, the channel that outputs the mixture and the ones that output zero signals remain consistent regardless of the input. By applying channel shuffle, such channel-wise bias is mitigated, making it less likely for the model to converge to a trivial solution. Formally, we apply a random permutation matrix $\mathbf{\Lambda}_b \in \mathbb{R}^{K\times K}$ to each sample b:

$$\tilde{\boldsymbol{s}}_b \leftarrow \mathbf{\Lambda}_b \tilde{\boldsymbol{s}}_b \in \mathbb{R}^{K\times T}. \tag{11}$$

We demonstrate that this simple modification substantially improves the stability of Self-Remixing and enables successful training from scratch.

3.2.2 Constrained Batch Shuffle (CBS)

In standard RemixIT, the permutation matrix $\mathbf{\Pi}_n$ used in batch shuffling (Equation (5)) is selected randomly, which can result in pseudo-mixtures that contain sources originating from the same original mixture. This is particularly problematic during training from scratch. Consider a teacher model $f_{\mathcal{T}}$ with three output channels ($N_{\mathcal{T}} = 3$), randomly initialized. Based on our earlier findings (Section 3.1), the outputs initially resemble scaled versions of the input mixtures. Consider the case where sources to be remixied are $\tilde{s}_1^{(\mathbf{\Pi})} = \frac{1}{3}x_1$, $\tilde{s}_2^{(\mathbf{\Pi})} = \frac{1}{3}x_2$, and $\tilde{s}_3^{(\mathbf{\Pi})} = \frac{1}{3}x_2$. In this example, the resulting pseudo-mixture, $\tilde{x} = \frac{1}{3}x_1 + \frac{2}{3}x_2$, contains two sources from the same original mixture x_2. The RemixIT loss would then encourage $f_{\mathcal{S}}$ to separate $\frac{2}{3}x_2$ into two $\frac{1}{3}x_2$, which does not lead to meaningful separation and may hinder training.

To address this, we introduce *constrained batch shuffle* (CBS), which performs the batch shuffle while enforcing a constraint: sources from the same original mixture must not be remixed together. Empirically, we found CBS essential for stabilizing RemixIT training from scratch. To satisfy the constraint, the batch size B needs to be at least the number of remixed sources K (i.e., $B \geq K$), but this is a reasonable assumption in practice, as K is typically small (e.g., three or four).

Although Self-Remixing does not strictly require this constraint to function, CBS still improves its training stability. Since Self-Remixing aims to reconstruct the original mixtures, it encourages the model to separate $\tilde{x}$ into $\frac{1}{3}x_1$, $\frac{2}{3}x_2$, and zero, thereby improving separation performance. However, with random batch shuffle, there is a non-negligible chance that the pseudo-mixture is identical to the original input mixture (e.g., all $\tilde{s}_n^{(\mathbf{\Pi})} = \frac{1}{3}x_1$, yielding $\tilde{x} = x_1$). Such cases are undesirable for both Self-Remixing (as well as RemixIT), as the model is trained to replicate the input. These degenerate cases become more frequent with smaller batch sizes. In our experiments, stable training with random batch shuffling required a relatively large batch size of at least $B = 32$. In contrast, CBS requires only $B \geq K$, which is much more practical.

4 Appropriate Loss Function for Remixing-based Methods

We empirically found that RemixIT and Self-Remixing underperform in certain evaluation metrics, such as perceptual evaluation of speech quality (PESQ) [29] and WER, compared to supervised learning, despite achieving comparable performance in SI-SDR. In this section, we investigate the root cause of this performance degradation and propose a solution to mitigate it.

In unsupervised sound separation, the thresholded negative SNR [55] between the reference signal y and its estimate $\hat{y}$ has typically been adopted as the loss function $\mathcal{L}$:

$$\mathcal{L}_{\mathrm{SNR}}(y, \hat{y}) = -10 \log_{10} \frac{||y||^2}{||y - \hat{y}||^2 + \tau ||y||^2}. \tag{12}$$

By design, SI-SDR is sensitive to phase errors, while metrics such as PESQ and WER are more influenced by the accuracy of the magnitude spectrum. While the SNR loss penalizes both magnitude and phase discrepancies, inaccurate phase estimation can lead to degraded magnitude estimation, as discussed in Section III-A of [51]. Since phase estimation is generally more challenging than magnitude estimation, reliance on SNR loss can result in suboptimal PESQ and WER scores.

Although this issue has been discussed in the context of supervised learning, we argue that remixing-based methods such as RemixIT and Self-Remixing are even more susceptible to this problem. In RemixIT, the student model must predict the distorted outputs of the teacher model from distorted pseudo-mixtures, making accurate phase estimation more difficult than in supervised settings. In Self-Remixing, the model is tasked with reconstructing the original mixtures from distorted pseudo-mixtures. Because pseudo-mixtures in a mini-batch are processed independently and cannot leverage cross-sample information, phase estimation becomes especially difficult. Based on this consideration, we attribute the degradation in PESQ and WER in RemixIT and Self-Remixing to the inherent difficulty of accurate phase estimation.

To address this issue, we explore two potential strategies. The first is to disable CBS, thereby allowing sources separated from the same mixture to be remixed. This can lead to partial cancellation of distortions when such sources are combined, resulting in cleaner pseudo-mixtures that facilitate more accurate phase estimation and, in turn, improve magnitude estimation. However, while disabling CBS simplifies phase recovery, it introduces instability during training.

The second strategy is to supplement the SNR loss with a magnitude-based loss function. Prior work has shown that such loss functions are effective in improving PESQ and WER [51], and we hypothesize that they are particularly beneficial in remixing-based methods where phase estimation is inherently more difficult. Specifically, we adopt a composite loss that combines a time-domain L1 loss with multi-resolution STFT-domain magnitude L1 losses, referred to as the multi-resolution L1 (MRL1) loss[6] [6, 25, 5]:

$$\mathcal{L}_{\mathrm{MRL1}}(y, \hat{y}) = \sum_{m=1}^{M} ||\ |\mathcal{G}_m(y)| - |\mathcal{G}_m(\hat{y})|\ || + ||y - \hat{y}||, \tag{13}$$

where $\mathcal{G}$ is STFT with varying resolutions, and M is the number of resolutions [58]. Given that RemixIT has a strong convergence property when trained with L2-based loss functions (detailed in Section II-C of [45]), we adopt a combined loss function that integrates both SNR and MRL1 losses:

$$\mathcal{L}_{\mathrm{SNR+MRL1}}(y, \hat{y}) = \mathcal{L}_{\mathrm{SNR}}(y, \hat{y}) + \mathcal{L}_{\mathrm{MRL1}}(y, \hat{y}). \tag{14}$$

In our experiments, we demonstrate that this combined loss function significantly improves PESQ and WER, while only slightly reducing SI-SDR, thereby providing a better balance across evaluation metrics in remixing-based unsupervised learning.

[6]The computational complexity of the MRL1 loss is approximately $O(M \times W \log W \times \frac{T}{S})$, where W and S represent the window and shift sizes used in the STFT operation, respectively. Although this is more computationally demanding than the SNR loss, which has a complexity of $O(T)$, the cost of loss computation is much smaller than other operations of the training pipeline, such as model forward pass or gradient computation.

5 Experimental Setup

5.1 Tasks and Datasets

We conducted experiments on both speech separation and universal sound separation (USS).

Speech separation: The model was trained to separate noisy, reverberant two-speaker mixtures. We synthesized these mixtures by combining speech signals from WSJ0 [10] and WSJ1 [20] with noise samples from CHiME3 [2], all at a sampling rate of 8 kHz. This dataset, which we refer to as the **WSJ-mix** dataset, closely follows the configuration of the SMS-WSJ dataset [8], with modifications in reverberation times, noise types, and noise levels. The acoustic field was simulated using Pyroomacoustics [37], with reverberation times randomly selected between 0.2 to 1.0 seconds. The SNR of the noise ranged between 10 and 20 dB. The dataset consisted of 33561 mixtures ($\sim$87.4 hours) for training, 982 mixtures ($\sim$2.5 hours) for validation, and 1332 mixtures ($\sim$3.6 hours) for testing.

USS: USS aims to develop a model capable of separating mixtures containing arbitrary sound classes [16]. For this purpose, we utilized the free universal sound separation (FUSS) dataset [52]. This dataset comprises mixtures containing between one and four sources, drawn from 357 different audio classes. Each mixture is ten seconds long and sampled at 16 kHz. The training, validation, and test sets consist of 20000, 1000, and 1000 mixtures, respectively.

5.2 Separation Models

We employed two separation models in our experiments: Conformer [11] and TF-GridNet [49].

Conformer: The Conformer [11] model, implemented based on the configuration in [3], consisted of approximately 21.6M parameters. It was composed of 16 Conformer encoder layers, each with four attention heads, 256 attention dimensions, and 1024 feed-forward network dimensions. We replaced the batch normalization [14] with group normalization [56] with eight groups. The model took as inputs log-magnitude (or magnitude) spectrograms in the STFT domain and output real-valued TF masks in speech separation (or USS). The STFT used a 512-point FFT with a window size of 400 and a hop size of 160.

TF-GridNet: To evaluate the effectiveness of the proposed method for mapping-based models, we utilized TF-GridNet [49], as implemented in ESPnet-SE [18, 22]. A smaller variant of the model, with approximately 2.3M parameters, was used in our experiments. The model configuration followed the notation in Table 1 of [49], with parameters set to $B = 4$, $D = 48$,

$I = 4$, $J = 4$, and $H = 96$. The input to the model consisted of the real and imaginary parts of the STFT spectrograms, and the model estimated the corresponding real and imaginary components for each separated source. The STFT settings were identical to those used in the Conformer model.

5.3 Compared Methods

We compared the following methods to evaluate the effectiveness of the improvements introduced in this study.

MixIT: MixIT training was performed using Equation (2). The number of output channels, $N_{\mathcal{S}}$, was set to $2K$ (six for WSJ-mix and eight in FUSS). Mixture consistency [53] was enforced when using the SNR loss, but not when using the MRL1 loss. To reduce training costs, we employed the efficient MixIT loss [54] for experiments using the MRL1 loss. In preliminary experiments, we confirmed that the original and the efficient MixIT losses yielded comparable performance.

MixIT+Sparsity: The MixIT training was augmented with the sparsity loss as defined in Equation (4). The sparsity loss weight γ was set to four when $N_{\mathcal{S}} = 6$ (WSJ-mix) and 23 when $N_{\mathcal{S}} = 8$ (FUSS), following [54]. Consistent with [54], the model was first pre-trained without the sparsity loss before applying it.

RemixIT: The RemixIT training employed Equation (6). We evaluated the performance both by fine-tuning a MixIT pre-trained model and by training from scratch. For fine-tuning, the teacher and the student models were initialized with the same pre-trained weights. In the exponential moving average (EMA) update defined in Equation (7), the decay factor α was set to 0.8.

Self-Remixing: The Self-Remixing training followed Equation (10) under the same configuration as RemixIT.

Supervised RemixIT/Self-Remixing: Supervised version of RemixIT or Self-Remixing serve as an upper bound for each method. Ground-truth signals were used instead of the outputs of $f_{\mathcal{T}}$ to create pseudo-mixtures. This approach is similar to dynamic mixing methods [47, 60], but mixes reverberant signals with different room acoustics. Note that the pseudo-mixture occasionally became zero signals when using datasets containing zero signals as ground truth (e.g., FUSS); such mixtures were discarded. Moreover, we applied a loss function capable of handling zero references (Equation (2) of [52]), instead of Equation (12).

MixPIT was excluded from the baselines because previous studies have shown MixIT to outperform MixPIT [15]. Additionally, we did not consider remix-cycle-consistent learning [30, 33], which is related to Self-Remixing but shares parameters between $\boldsymbol{\theta}_{\mathcal{T}}$ and $\boldsymbol{\theta}_{\mathcal{S}}$, and computes gradients through two sequential separation and remixing processes. This method requires additional techniques (e.g., loss thresholding as in [32]) to function effectively in single-channel setups, making stable training from scratch challenging. Furthermore, recent RemixIT variants [19, 48] were excluded from comparison, as they are specifically tailored for speech enhancement and less straightforward to generalize to universal sound separation. Therefore, this study focuses on comparing MixIT, RemixIT, and Self-Remixing.

5.4 Training and Evaluation Details

We used the AdamW optimizer [21] with a weight decay of 1e-2, and applied gradient clipping with a maximum norm of 5. Each input mixture was normalized by subtracting its mean and dividing by its standard deviation. Only the original mixtures x were normalized; pseudo-mixtures $\tilde{x}$, generated in RemixIT and Self-Remixing, were left unnormalized, as normalization of $\tilde{x}$ was found to degrade performance in our experiments. The same normalization procedure was applied when generating MoMs in MixIT. For experiments using the MRL1 loss, we computed the loss across $M = 3$ resolutions with the following configurations for (FFTsize, stride, window length) = (512, 160, 400), (1024, 320, 800), and (2048, 640, 1600). In inference, we averaged the parameters of the five checkpoints that gave the best validation performance.

In speech separation experiments, the Conformer and TF-GridNet models were trained for 600 and 400 epochs, respectively, with batch sizes of 32 (Conformer) and 8 (TF-GridNet).[7] Each input segment was 7 seconds in duration. The learning rate was linearly increased from 0 to 2e-4 (Conformer) or 1e-3 (TF-GridNet) over the first 5000 training steps, followed by decay by a factor of 0.98 every three (Conformer) or two (TF-GridNet) epochs. For evaluation, we selected the model checkpoint that achieved the highest SI-SDR on the validation set.[8] Evaluation metrics included SI-SDR [38], short-time objective intelligibility (STOI) [41], PESQ, and WER. WER was computed using both the ASR backend provided in SMS-WSJ [8] and the Whisper Large v2 model [28]. Before Whisper-based evaluation, separated signals were upsampled to 16 kHz.

In unsupervised USS experiments, we trained the Conformer model with a batch size of 16 on 10-second input segments, for 400 epochs. The learning rate schedule matched that used in the unsupervised speech separation experiment. Evaluation metrics followed prior work [52, 54] and included: **1S**, representing the SI-SDR for single-source inputs; ***k*Si**, representing the SI-SDR improvement (SI-SDRi) for k-source mixtures ($k = 2, 3, 4$); **MSi**, representing

[7]For MixIT, we define one *epoch* as the number of training steps to process the entire dataset twice because each MoM requires $2B$ examples to create B mixtures. This definition ensures a consistent number of training steps across all methods.

[8]Although using in-domain validation data with ground-truth signals is not feasible in an unsupervised setup, we employed them for fair comparison across methods.

the average SI-SDRi across multi-source mixtures; and **TRF**, representing the total reconstruction fidelity with average performance over all test mixtures.

6 Experimental Results

6.1 Results of Unsupervised Speech Separation with Conformer

Table 1 presents the evaluation results of the Conformer model. We trained the models using both the SNR loss (Equation 12) and the SNR+MRL1 loss (Equation (14)). For RemixIT and Self-Remixing, we conducted experiments both with and without the proposed CBS and CS mechanisms to assess the effectiveness of the improved remixing strategy.

Table 1: Average SISDR [dB], STOI, PESQ, and WER [%] of **Conformer** model on WSJ-mix test set. Two WERs were obtained with ASR backend of SMS-WSJ and Whisper Large v2, respectively. CBS and CS stand for constrained batch shuffle and channel shuffle (Section 3.2), respectively.

			SNR				SNR + Multi-resolution L1			
Method	CBS	CS	SISDR$^\uparrow$	STOI$^\uparrow$	PESQ$^\uparrow$	WER$^\downarrow$	SISDR$^\uparrow$	STOI$^\uparrow$	PESQ$^\uparrow$	WER$^\downarrow$
Unprocessed	-	-	-0.4	68.4	1.82	82.9 / 76.2	-0.4	68.4	1.82	82.9 / 76.2
`A1` MixIT	-	-	8.8	84.7	2.54	42.3 / 22.2	9.0	84.7	2.61	40.8 / 24.3
`A2` +Sparsity	-	-	8.6	84.3	2.51	44.8 / 22.8	8.3	83.0	2.56	43.6 / 26.1
`A3` RemixIT	No	No	Training failed				Training failed			
`A4` RemixIT	No	Yes	Training failed				Training failed			
`A5` RemixIT	Yes	No	**10.8**	**89.0**	**2.84**	47.4 / **16.7**	**10.6**	**89.2**	**2.98**	**31.2** / **16.2**
`A6` RemixIT	Yes	Yes	10.3	87.8	2.75	43.3 / 20.9	10.1	87.8	2.81	36.8 / 19.8
`A7` Self-Remixing	No	No	Training failed				Training failed			
`A8` Self-Remixing	No	Yes	10.3	87.8	2.74	**39.7** / 19.7	10.3	87.9	2.77	34.5 / 19.5
`A9` Self-Remixing	Yes	No	Training failed				Training failed			
`A10` Self-Remixing	Yes	Yes	10.3	87.7	2.69	50.1 / 22.4	10.1	87.6	2.67	38.9 / 20.0
`A11` +Self-Remixing	Yes	No	10.5	88.3	2.75	50.9 / 19.0	10.4	88.4	2.77	34.6 / 17.5
Sup. RemixIT	Yes	No	10.9	89.6	3.01	30.9 / 16.1	10.8	89.6	3.00	26.6 / 14.6
Sup. RemixIT	Yes	Yes	10.6	88.9	2.93	33.4 / 17.6	10.5	88.8	2.89	30.3 / 16.9
Sup. Self-Remixing	Yes	No	10.9	89.7	3.02	31.0 / 15.7	10.9	89.7	2.96	27.5 / 14.6
Sup. Self-Remixing	Yes	Yes	10.6	88.9	2.93	33.7 / 17.5	10.6	88.9	2.87	30.7 / 16.6

We first examine the impact of the improved remixing algorithm. Figure 3 illustrates the training curves for RemixIT and Self-Remixing under different configurations of CBS and CS. Figure 3 and Table 1 confirm that RemixIT fails to converge without CBS (`A3` and `A4`). In contrast, Self-Remixing is able to work successfully without CBS (`A8`), but fails when CS is not applied (`A7` and `A9`). These results highlight the crucial role of CBS and CS in ensuring successful training from scratch.

Interestingly, comparing performance with and without CS reveals that excluding CS leads to better overall performance. This may be attributed to how well-trained separation models tend to assign speech and noise to distinct output channels (e.g., channels 1 and 2 for speech, and channel 3 for noise). When CS is applied, pseudo-mixtures can end up containing either only speech

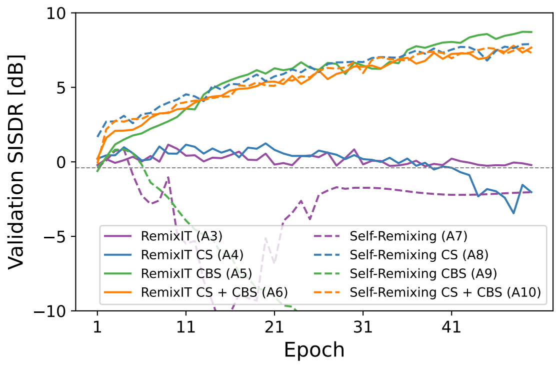


Figure 3: Validation SISDR of f_S of RemixIT and Self-Remixing in early stage of training. Conformer model and MRL1 loss are used.

or only noise sources, introducing a mismatch relative to the original mixture distributions. In contrast, without CS, pseudo-mixtures are more likely to contain a realistic combination of speech and noise sources, reducing the discrepancy between the original and pseudo-mixtures and thereby improving separation performance. Although training Self-Remixing from scratch without CS is unstable, a two-stage approach, training for 200 epochs with CS (`A10`) followed by 400 epochs of fine-tuning without CS (`A11`), yields better results than training for 600 epochs with CS alone (`A10`).

As discussed in Section 4, while CBS improves training stability, it degrades PESQ and WER performance (e.g., compare `A8` vs. `A10` under SNR loss). However, this issue can be mitigated by incorporating the MRL1 loss. As shown in Table 1, the use of the combined SNR+MRL1 loss consistently improves PESQ or WER across most methods. Although similar improvements are observed in supervised settings, the effect is particularly pronounced in unsupervised RemixIT and Self-Remixing, demonstrating the effectiveness of incorporating an amplitude-based loss in remixing-based frameworks.

6.2 Results of Unsupervised Speech Separation with TF-GridNet

Table 2 presents the evaluation results of the TF-GridNet on the WSJ-mix test set. In all experiments, we employed the SNR+MRL1 loss, as its effectiveness was confirmed in the previous section. CBS was also always applied due to the small batch size (eight) used in this setting.

The effect of CS follows a similar trend to the Conformer experiments. While overall performance is consistently better when CS is not applied, it facilitates the successful training of Self-Remixing. This pattern holds across all scenarios: pre-training with MixIT followed by fine-tuning with RemixIT

Table 2: Average SISDR [dB], STOI, PESQ, and WER [%] of **TF-GridNet** model on WSJ-mix test set. Two WERs were obtained with ASR backend of SMS-WSJ and Whisper Large v2, respectively. SNR + MRL1 loss was used, and constrained batch shuffle (CBS) was always applied. CS stands for channel shuffle (Section 3.2).

Method	epochs	CS	SISDR$^{\uparrow}$	STOI$^{\uparrow}$	PESQ$^{\uparrow}$	WER$^{\downarrow}$
Unprocessed	-	-	-0.4	68.4	1.82	82.9 / 76.2
`B1` MixIT	400	-	11.9	91.4	3.16	22.3 / 7.7
`B2` +RemixIT	150+250	No	12.9	92.6	3.27	**21.1** / 8.1
`B3` +RemixIT	150+250	Yes	12.4	92.4	3.32	25.7 / 8.3
`B4` +Self-Remixing	150+250	No	13.1	93.0	3.41	22.7 / 7.6
`B5` +Self-Remixing	150+250	Yes	12.7	92.6	3.38	22.6 / 7.8
`C1` RemixIT	400	No	13.1	92.5	3.29	22.2 / 7.7
`C2` RemixIT	400	Yes	12.3	92.0	3.28	25.4 / 8.5
`C3` Self-Remixing	400	No		Training failed		
`C4` Self-Remixing	400	Yes	12.7	92.6	3.40	21.6 / 7.9
`C5` +Self-Remixing	150+250	No	**13.4**	**93.3**	**3.47**	22.1 / **7.4**
Sup. RemixIT	400	No	13.5	93.7	3.55	19.7 / 6.8
Sup. RemixIT	400	Yes	12.7	93.0	3.45	18.4 / 7.4
Sup. Self-Remixing	400	No	13.6	93.8	3.55	17.8 / 6.7
Sup. Self-Remixing	400	Yes	12.8	93.1	3.45	18.6 / 7.3

or Self-Remixing (`B2`-`B5`), training from scratch (`C1`-`C5`), and in supervised setups.

In terms of separation performance, both RemixIT and Self-Remixing outperform MixIT. These methods are capable of separating not only speech but also noise, which remains a limitation in MixIT (see the demo page for examples). Furthermore, RemixIT and Self-Remixing models trained from scratch slightly outperform those fine-tuned from MixIT-pretrained models. We attribute this to the fact that scratch-trained models avoid inheriting redundant output channels that are often present in MixIT-pretrained models.

In contrast to the Conformer results, Self-Remixing with TF-GridNet achieves better performance than RemixIT. This can be explained by the nature of TF-GridNet, which is based on complex spectral mapping and can jointly estimate both magnitude and phase, enabling more accurate reconstruction of the original mixtures. Moreover, RemixIT relies on supervision from a teacher model, which may introduce distortions due to its own separation process. In contrast, Self-Remixing constructs pseudo-mixtures without relying on external separation outputs, avoiding such distortions and contributing to its superior performance.

6.3 Results of Unsupervised Universal Sound Separation

Table 3 presents the evaluation results on the FUSS dataset. Considering that TF-masking-based methods continue to demonstrate strong performance on the USS task [27], we employed the Conformer model. In addition, CS was

Table 3: Evaluation results of conformer models trained with SNR loss on FUSS test set. CBS stands for constrained batch shuffle (Section 3.2). Channel shuffle was always applied in RemixIT and Self-Remixing.

			Best MSi						Best TRF					
Method	epochs	CBS	1S	2Si	3Si	4Si	MSi	TRF	1S	2Si	3Si	4Si	MSi	TRF
D1 MixIT	400	-	9.5	9.8	14.4	15.9	13.2	12.2	9.5	9.9	14.3	15.8	13.2	12.2
D2 +Sparsity	300+100	-	15.2	12.3	15.5	16.6	14.7	14.8	18.0	12.3	15.5	16.5	14.7	15.6
D3 +RemixIT	300+50+50	No	29.2	14.9	16.1	15.8	15.6	19.2	32.9	15.2	16.1	15.3	15.5	20.0
D4 +RemixIT	300+50+50	Yes	29.2	15.0	16.5	16.0	15.8	19.4	31.5	15.1	16.1	15.6	15.6	19.9
D5 +Self-Remixing	300+50+50	No	27.2	14.9	**16.9**	**17.0**	16.2	19.1	32.1	15.2	**16.9**	**16.7**	**16.3**	**20.5**
D6 +Self-Remixing	300+50+50	Yes	29.1	**15.2**	**16.9**	16.8	**16.3**	**19.7**	30.8	**15.3**	16.8	**16.7**	16.2	20.1
E1 RemixIT	400	Yes	**42.1**	12.3	11.7	8.2	10.8	19.1	**63.6**	4.6	2.1	0.1	2.4	18.6
E2 Self-Remixing	400	Yes	29.4	14.7	16.2	13.6	14.8	18.7	34.4	14.8	15.9	13.1	14.6	19.8
E3 Self-Remixing	400	No	31.9	14.9	15.9	13.7	14.8	19.4	34.9	15.0	15.6	13.4	14.7	20.0
Sup. Self-Remixing	400	No	32.9	14.1	15.8	13.5	14.5	19.4	49.2	13.7	14.7	12.0	13.5	23.0

consistently applied throughout the USS experiments, as the gap between the initial and pseudo-mixtures caused by CS is not a concern in this setting (see Section 6.1 for discussion).

When training RemixIT from scratch, we observed an initial improvement in separation performance, followed by degradation. This behavior can be attributed to the fact that a large proportion of the mixtures (approximately 75%) contain fewer sources than the number of output channels. As discussed in Section 2.4, RemixIT becomes unstable under such conditions.

The performance of MixIT was substantially improved by incorporating the sparsity loss, likely due to the presence of additional output channels. Compared to MixIT with sparsity (D2), Self-Remixing trained from scratch (E2, E3) achieved comparable MSi but substantially higher 1S. Since MoMs in MixIT always include mixtures of at least two sources, gains in 1S are limited, even with the sparsity loss. Although the average MSi is similar across methods, the composition of improvements varies: Self-Remixing excels on mixtures containing two or three sources, whereas MixIT shows better performance on mixtures with four sources.

In contrast to the results on WSJ-mix, the highest performance on FUSS was achieved by fine-tuning models pre-trained with MixIT+sparsity by RemixIT or Self-Remixing. These findings suggest that, in cases where MixIT already achieves competitive or superior performance to RemixIT and Self-Remixing, fine-tuning the MixIT pre-trained model is a highly effective strategy.

7 Conclusion

To enhance the training stability and performance of Self-Remixing, we introduced new remixing algorithms and appropriate loss functions. While Self-Remixing originally relied on a pre-trained model, our detailed analysis revealed that the pseudo-mixtures generated in the early stages of training resemble mixtures of mixtures (MoMs), suggesting that the method can, in principle, be trained from scratch. Although we observed that training from scratch often leads to trivial solutions, we successfully removed the need for pre-training by introducing two novel remixing algorithms, channel shuffle and constrained batch shuffle. Although we observe that Self-Remixing performs poorly on metrics sensitive to amplitude estimation, through qualitative analysis, we identify the cause and propose a loss function that emphasizes amplitude reconstruction. This results in significant improvements in some metrics such as PESQ and WER.

References

[1] Y. Bando, K. Sekiguchi, Y. Masuyama, A. A. Nugraha, M. Fontaine, and K. Yoshii, "Neural Full-Rank Spatial Covariance Analysis for Blind Source Separation", *IEEE Signal Processing Letters*, 28, 2021, 1670–4.

[2] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third 'CHiME' speech separation and recognition challenge: Dataset, task and baselines", in *Proc. ASRU*, 2015, 504–11.

[3] S. Chen, Y. Wu, Z. Chen, J. Wu, J. Li, T. Yoshioka, C. Wang, S. Liu, and M. Zhou, "Continuous Speech Separation with Conformer", in *Proc. ICASSP*, 2021, 5749–53.

[4] Z. Chen, T. Yoshioka, L. Lu, T. Zhou, Z. Meng, Y. Luo, J. Wu, X. Xiao, and J. Li, "Continuous Speech Separation: Dataset and Analysis", in *Proc. ICASSP*, 2020, 7284–8.

[5] S. Cornell, Z.-Q. Wang, Y. Masuyama, S. Watanabe, M. Pariente, and N. Ono, "Multi-channel target speaker extraction with refinement: The wavlab submission to the second clarity enhancement challenge", *arXiv preprint arXiv:2302.07928*, 2023.

[6] A. Défossez, G. Synnaeve, and Y. Adi, "Real Time Speech Enhancement in the Waveform Domain", in *Proc. Interspeech*, 2020, 3291–5.

[7] T. Denton, S. Wisdom, and J. R. Hershey, "Improving Bird Classification with Unsupervised Sound Separation", in *Proc. ICASSP*, 2022, 636–40.

[8] L. Drude, J. Heitkaemper, C. Boeddeker, and R. Haeb-Umbach, "SMS-WSJ: Database, performance measures, and baseline recipe for multi-channel source separation and recognition", in *arXiv preprint arXiv:1910.13934*, 2019.

[9] L. Drude, D. Hasenklever, and R. Haeb-Umbach, "Unsupervised Training of a Deep Clustering Model for Multichannel Blind Source Separation", in *Proc. ICASSP*, 2019, 695–9.

[10] J. S. Garofolo *et al.*, *CSR-I (WSJ0) Complete LDC93S6A*, Web Download, Linguistic Data Consortium, Philadelphia, 1993.

[11] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented Transformer for Speech Recognition", in *Proc. Interspeech*, 2020, 5036–40.

[12] J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation", in *Proc. ICASSP*, 2016, 31–5.

[13] Y. Higuchi, N. Moritz, J. Le Roux, and T. Hori, "Momentum pseudo-labeling: Semi-supervised asr with continuously improving pseudo-labels", *IEEE Journal of Selected Topics in Signal Processing*, 16(6), 2022, 1424–38.

[14] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", in *Proc. ICML*, 2015, 448–56.

[15] E. Karamatl and S. Krbz, "MixCycle: Unsupervised Speech Separation via Cyclic Mixture Permutation Invariant Training", *IEEE Signal Processing Letters*, 2022.

[16] I. Kavalerov, S. Wisdom, H. Erdogan, B. Patton, K. Wilson, J. Le Roux, and J. R. Hershey, "Universal sound separation", in *Proc. WASPAA*, 2019, 175–9.

[17] J. Le Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, "SDR–half-baked or well done?", in *Proc. ICASSP*, 2019, 626–30.

[18] C. Li, J. Shi, W. Zhang, A. S. Subramanian, X. Chang, N. Kamo, M. Hira, T. Hayashi, C. Boeddeker, Z. Chen, *et al.*, "ESPnet-SE: End-to-end speech enhancement and separation toolkit designed for ASR integration", in *Proc. SLT*, 2021, 785–92.

[19] L. Li and S. Seki, "Remixed2Remixed: Domain adaptation for speech enhancement by Noise2Noise learning with Remixing", in *Proc. ICASSP*, 2024, 806–10.

[20] Linguistic Data Consortium, and NIST Multimodal Information Group, *CSR-II (WSJ1) Complete LDC94S13A*, Web Download, Linguistic Data Consortium, Philadelphia, 1994.

[21] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization", in *Proc. ICLR*, 2018.

[22] Y.-J. Lu, X. Chang, C. Li, W. Zhang, S. Cornell, Z. Ni, Y. Masuyama, B. Yan, R. Scheibler, Z.-Q. Wang, Y. Tsao, Y. Qian, and S. Watanabe, "ESPnet-SE++: Speech Enhancement for Robust Speech Recognition, Translation, and Understanding", in *Proc. Interspeech*, 2022, 5458–62.

[23] Y. Luo and N. Mesgarani, "Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation", *IEEE/ACM transactions on audio, speech, and language processing*, 27(8), 2019, 1256–66.

[24] H. Munakata, R. Takeda, and K. Komatani, “Training Data Generation with DOA-based Selecting and Remixing for Unsupervised Training of Deep Separation Models”, in *Proc. Interspeech*, 2022, 861–5.

[25] Z. Pan, M. Ge, and H. Li, “A Hybrid Continuity Loss to Reduce Over-Suppression for Time-domain Target Speaker Extraction”, in *Proc. Interspeech*, 2022, 1786–90.

[26] A. Paszke *et al.*, “Pytorch: An imperative style, high-performance deep learning library”, *Advances in neural information processing systems*, 32, 2019.

[27] J. Pons, X. Liu, S. Pascual, and J. Serrà, “GASS: Generalizing Audio Source Separation with Large-scale Data”, *Proc. ICASSP*, 2024.

[28] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, “Robust speech recognition via large-scale weak supervision”, *Proc. ICML*, 2023.

[29] A. Rix, J. Beerends, M. Hollier, and A. Hekstra, “Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs”, in *Proc. ICASSP*, Vol. 2, 2001, 749–52.

[30] K. Saijo and T. Ogawa, “Remix-cycle-consistent learning on adversarially learned separator for accurate and stable unsupervised speech separation”, in *Proc. ICASSP*, 2022.

[31] K. Saijo and T. Ogawa, “Remixing-based Unsupervised Source Separation from Scratch”, in *Proc. Interspeech*, 2023, 1678–82.

[32] K. Saijo and T. Ogawa, “Self-Remixing: Unsupervised Speech Separation VIA Separation and Remixing”, in *Proc. ICASSP*, 2023, 1–5.

[33] K. Saijo and T. Ogawa, “Unsupervised Training of Sequential Neural Beamformer Using Coarsely-separated and Non-separated Signals”, in *Proc. Interspeech*, 2022, 251–5.

[34] K. Saijo and R. Scheibler, “Spatial Loss for Unsupervised Multi-channel Source Separation”, in *Proc. Interspeech*, 2022, 241–5.

[35] K. Saijo, G. Wichern, F. G. Germain, Z. Pan, and J. Le Roux, “TF-Locoformer: Transformer with local modeling by convolution for speech separation and enhancement”, in *Proc. IWAENC*, 2024, 205–9.

[36] K. Saijo, G. Wichern, F. G. Germain, Z. Pan, and J. Le Roux, “Enhanced Reverberation as Supervision for Unsupervised Speech Separation”, in *Proc. Interspeech*, 2024, 607–11.

[37] R. Scheibler, E. Bezzam, and I. Dokmani, “Pyroomacoustics: A Python Package for Audio Room Simulation and Array Processing Algorithms”, in *Proc. ICASSP*, 2018, 351–5.

[38] R. Scheibler, “SDR — Medium rare with fast computations”, in *Proc. ICASSP*, May 2022.

[39] A. Sivaraman, S. Wisdom, H. Erdogan, and J. R. Hershey, “Adapting speech separation to real-world meetings using mixture invariant training”, in *Proc. ICASSP*, 2022, 686–90.

[40] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong, “Attention is all you need in speech separation”, in *Proc. ICASSP*, 2021, 21–5.

[41] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, “An Algorithm for Intelligibility Prediction of TimeFrequency Weighted Noisy Speech”, *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7), 2011, 2125–36.

[42] A. Tarvainen and H. Valpola, “Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results”, *Proc. NeurIPS*, 30, 2017.

[43] M. Togami, Y. Masuyama, T. Komatsu, and Y. Nakagome, “Unsupervised Training for Deep Speech Source Separation with Kullback-Leibler Divergence Based Probabilistic Loss Function”, in *Proc. ICASSP*, 2020, 56–60.

[44] E. Tzinis, Y. Adi, V. K. Ithapu, B. Xu, and A. Kumar, “Continual self-training with bootstrapped remixing for speech enhancement”, in *Proc. ICASSP*, 2022, 6947–51.

[45] E. Tzinis, Y. Adi, V. K. Ithapu, B. Xu, P. Smaragdis, and A. Kumar, “RemixIT: Continual self-training of speech enhancement models via bootstrapped remixing”, *IEEE Journal of Selected Topics in Signal Processing*, 2022.

[46] E. Tzinis, S. Venkataramani, and P. Smaragdis, “Unsupervised deep clustering for source separation: Direct learning from mixtures using spatial information”, in *Proc. ICASSP*, 2019, 81–5.

[47] E. Tzinis, S. Venkataramani, Z. Wang, C. Subakan, and P. Smaragdis, “Two-Step Sound Source Separation: Training On Learned Latent Targets”, in *Proc. ICASSP*, 2020, 31–5.

[48] S. Wang, H. Guan, and Y. Long, “QMixCAT: Unsupervised Speech Enhancement Using Quality-guided Signal Mixing and Competitive Alternating Model Training”, in *Proc. Interspeech*, 2024, 642–6.

[49] Z.-Q. Wang, S. Cornell, S. Choi, Y. Lee, B.-Y. Kim, and S. Watanabe, “TF-GridNet: Integrating full-and sub-band modeling for speech separation”, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

[50] Z.-Q. Wang and S. Watanabe, “UNSSOR: Unsupervised Neural Speech Separation by Leveraging Over-determined Training Mixtures”, *Proc. NeurIPS*, 36, 2024.

[51] Z.-Q. Wang, G. Wichern, and J. Le Roux, “On the compensation between magnitude and phase in speech separation”, *IEEE Signal Processing Letters*, 28, 2021, 2018–22.

[52] S. Wisdom, H. Erdogan, D. P. Ellis, R. Serizel, N. Turpault, E. Fonseca, J. Salamon, P. Seetharaman, and J. R. Hershey, “Whats all the fuss about free universal sound separation data?”, in *Proc. ICASSP*, 2021, 186–90.

[53] S. Wisdom, J. R. Hershey, K. Wilson, J. Thorpe, M. Chinen, B. Patton, and R. A. Saurous, “Differentiable consistency constraints for improved deep speech enhancement”, in *Proc. ICASSP*, 2019, 900–4.

[54] S. Wisdom, A. Jansen, R. J. Weiss, H. Erdogan, and J. R. Hershey, “Sparse, Efficient, and Semantic Mixture Invariant Training: Taming In-the-Wild Unsupervised Sound Separation”, in *Proc. WASPAA*, 2021, 51–5.

[55] S. Wisdom, E. Tzinis, H. Erdogan, R. Weiss, K. Wilson, and J. Hershey, “Unsupervised sound separation using mixture invariant training”, *Proc. NeurIPS*, 33, 2020, 3846–57.

[56] Y. Wu and K. He, “Group normalization”, in *Proc. ECCV*, 2018, 3–19.

[57] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, “Self-training with noisy student improves imagenet classification”, in *Proc. IEEE/CVF CVPR*, 2020, 10687–98.

[58] R. Yamamoto, E. Song, and J.-M. Kim, “Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram”, in *Proc. ICASSP*, 2020, 6199–203.

[59] D. Yu, M. Kolbæk, Z. H. Tan, and J. Jensen, “Permutation invariant training of deep models for speaker-independent multi-talker speech separation”, in *Proc. ICASSP*, 2017, 241–5.

[60] N. Zeghidour and D. Grangier, “Wavesplit: End-to-end speech separation by speaker clustering”, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29, 2021, 2840–9.

[61] J. Zhang, C. Zoril, R. Doddipatla, and J. Barker, “Teacher-Student MixIT for Unsupervised and Semi-Supervised Speech Separation”, in *Proc. Interspeech*, 2021, 3495–9.