

SEPARATE BUT TOGETHER: UNSUPERVISED FEDERATED LEARNING FOR SPEECH ENHANCEMENT FROM NON-IID DATA

Efthymios Tzinis¹, Jonah Casebeer¹, Zhepei Wang¹, Paris Smaragdis^{1,2}

¹ University of Illinois at Urbana-Champaign, Department of Computer Science, Urbana, IL, USA

² Adobe Research, San Jose, CA, USA

ABSTRACT

We propose FEDENHANCE, an unsupervised federated learning (FL) approach for speech enhancement and separation with non-IID distributed data across multiple clients. We simulate a real-world scenario where each client only has access to a few noisy recordings from a limited and disjoint number of speakers (hence non-IID). Each client trains their model in isolation using mixture invariant training while periodically providing updates to a central server. Our experiments show that our approach achieves competitive enhancement performance compared to IID training on a single device and that we can further facilitate the convergence speed and the overall performance using transfer learning on the server-side. Moreover, we show that we can effectively combine updates from clients trained locally with supervised and unsupervised losses. We also release a new dataset *LibriFSD50K* and its creation recipe in order to facilitate FL research for source separation problems.

Index Terms— Speech enhancement, federated learning, unsupervised learning, source separation, non-IID learning

1. INTRODUCTION

Recent advances in deep learning have enabled the development of neural network architectures capable of separating individual sound sources from mixtures of sounds with high fidelity. Discriminative separation models with supervised training have obtained state-of-the-art performance on multiple tasks such as music separation [1], speech separation [2, 3] and speech enhancement [4, 5]. However, gathering clean source waveforms to perform supervised training under various domains can be cumbersome or even impossible. Other works have focused on less supervised approaches by leveraging contrastive learning [6] or using weak labels containing sound-class information [7, 8]. Moreover, unsupervised approaches using spatial information from multiple microphones [9, 10, 11] and visual cues [12] have also shown promising results. Mixture invariant training (MixIT) has shown great potential for single-channel sound separation and speech enhancement by training on synthetic mixtures of mixtures [13]. Despite that these works lessen the reliance on supervised data, they still require huge consolidated audio data collections being available for IID training on a single device.

Federated learning (FL) [14] has provided a distributed and privacy preserving framework where each client trains on their local data and communicates updates to a central server. The central server aggregates those updates and distributes a new model at each communication round. Several studies have shown that by repeating this process, one can aptly train a global model without violating client privacy [15] even under cases where data are not distributed

IID to the clients [16]. FL has also been applied to train audio models for keyword spotting [17], automatic speech recognition [18] and sound event detection [19]. Nevertheless, the aforementioned approaches, as well as most FL setups, require supervised data to be available on the client side and are even less successful when multiple clients are present or the IID assumption is violated [20]. The aforementioned problem led to the development of recent FL algorithms which require less supervision [21, 22].

In this work, we tackle the real world problem of learning a speech enhancement model where a central server (e.g. a company) aggregates updates across numerous clients. The local client datasets contain noisy mixtures and may greatly vary across clients. Thus, we present FEDENHANCE, an FL system which is capable of learning a separation model for speech enhancement without relying on several common assumptions such as: a) requiring supervised data and b) assuming IID distribution of the data across the clients. In essence, each client has access to a limited number of noisy speech mixtures and isolated noise recordings. All clients perform local unsupervised training using MixIT by synthesizing a mixture from a noisy speech recording and a noise recording. To evaluate our approach, we introduce a realistic and challenging speech enhancement dataset, namely, *LibriFSD50K* containing around 50 hours of training data and with 280 speaker IDs. We use the speaker variability for simulating real-world situations that clients only have access to noisy speech recordings from one speaker (e.g. a smartphone client can easily record themselves). We analyze the convergence behavior of FEDENHANCE and show that our approach can scale to multiple clients. Moreover, we show that we can expedite the convergence and boost the overall performance of our FL method by transferring knowledge from another medium-size speech enhancement dataset. Finally, our experimental results show that we can effectively combine updates from clients with supervised and unsupervised data using different loss functions.

2. UNSUPERVISED FEDERATED LEARNING FOR SPEECH ENHANCEMENT

FEDENHANCE follows a conventional FL setup with one central server orchestrating the overall communication and C clients which perform local training on their private data $\{\mathcal{D}_c\}_{c=1,\dots,C}$.

2.1. Server-clients communication

The server owns a global copy of the model weights $\theta_g^{(r)}$ which are distributed to the decentralized clients at each round r . Formally, we assume that the server shares the same separation network architecture with the clients $f(\cdot; \theta_g^{(r)})$ which is parameterized through the global weights $\theta_g^{(r)}$ at the communication round r . For an input

mixture with T samples in the time-domain $\mathbf{x} \in \mathbb{R}^T$, each separation model outputs M sources, namely, $\hat{\mathbf{s}} = f(\mathbf{x}; \boldsymbol{\theta}) \in \mathbb{R}^{M \times T}$. At the update step, the c -th client from the set of available clients $\mathcal{A}^{(r)}$ provides the updated weights $\boldsymbol{\theta}_c^{(r)}$ after training independently on its private data \mathcal{D}_c . Finally, the server aggregates those weights in order to produce the updated model $\boldsymbol{\theta}_g^{(r+1)}$. The new model is going to be distributed again across all clients and the process is repeated for R communication rounds as described in Algorithm 1.

Algorithm 1: FEDAVG [15] with a server and C clients.

Input: $\boldsymbol{\theta}_g^{(0)}$ // Initial server model weights
Output: $\boldsymbol{\theta}_g^{(R)}$
for $r = 1; r++;$ **while** $r \leq R$ **do**
 // Available clients $\mathcal{A}^{(r)} \subseteq \{1, \dots, C\}$
 for $n \in \mathcal{A}^{(r)}$ **do**
 // Server distributes the model
 $\boldsymbol{\theta}_c^{(r)} \leftarrow \boldsymbol{\theta}_g^{(r-1)}$
 // Local training on private \mathcal{D}_c
 $\boldsymbol{\theta}_c^{(r)} \leftarrow \text{CLIENTUPDATE}(\boldsymbol{\theta}_c^{(r)}, \mathcal{D}_c)$
 end
 // Server update for round r
 $\boldsymbol{\theta}_g^{(r)} \leftarrow \frac{1}{|\mathcal{A}^{(r)}|} \sum_{n \in \mathcal{A}^{(r)}} \boldsymbol{\theta}_n^{(r)}$
end

2.2. Local training on the client-side

We present how each client performs local training on its private data for the task of speech enhancement. We assume that the c -th client has access only to its private data \mathcal{D}_c which consist of two portions $\mathcal{D}_c = (\mathcal{D}_c^m, \mathcal{D}_c^n)$. Specifically, \mathcal{D}_c^m is the part of the data that contains mixtures of speech and noise while \mathcal{D}_c^n contains only clean noise. Following the training procedure presented in [13], each client generates artificial mixtures of mixtures (MoMs) $\mathbf{x} = \mathbf{s} + \mathbf{n}_1 + \mathbf{n}_2$ using a noisy speech example $\mathbf{m} = \mathbf{s} + \mathbf{n}_1 \sim \mathcal{D}_c^m$ and a clean noise recording $\mathbf{n}_2 \sim \mathcal{D}_c^n$. The separation model always estimates $M = 3$ sources. We distinguish two different cases for each client's private data \mathcal{D}_c^m while we always assume that each client has access to clean noise recordings $\mathbf{n}_2 \sim \mathcal{D}_c^n$.

Supervised data: A supervised client would have to noisy speech files $\mathbf{m} = \mathbf{s} + \mathbf{n}_1$ as well as the isolated speech \mathbf{s} and the noise \mathbf{n}_1 waveforms, where $(\mathbf{m}, \mathbf{s}, \mathbf{n}_1) \sim \mathcal{D}_c^m$ (e.g. research institutes with many hours of clean speech recordings might fall in this category). The loss function which is minimized is shown next:

$$\mathcal{L}_{\text{sup}} = \mathcal{L}(\hat{\mathbf{s}}_1, \mathbf{s}) + \frac{1}{2} \min_{\pi \in \Pi_{2,3}} [\mathcal{L}(\hat{\mathbf{s}}_{\pi_1}, \mathbf{n}_1) + \mathcal{L}(\hat{\mathbf{s}}_{\pi_2}, \mathbf{n}_2)], \quad (1)$$

where \mathcal{L} could be any signal-level loss which measures the reconstruction fidelity of each separated signal $\hat{\mathbf{s}}_i$ w.r.t. the targets \mathbf{s} , \mathbf{n}_1 , and \mathbf{n}_2 . $\Pi_{2,3} = \{(2, 3), (3, 2)\}$ symbolizes the set of permutations used for the latter two slots $\hat{\mathbf{s}}_2$ and $\hat{\mathbf{s}}_3$. Thus, we train with permutation invariance w.r.t. the noise sources \mathbf{n}_1 and \mathbf{n}_2 while also forcing the model to produce the reconstructed speaker in the first slot $\hat{\mathbf{s}}_1$.

Unsupervised data: This client has access only to mixtures of noisy speech $(\mathbf{m}, -) \sim \mathcal{D}_c^m$. This is the most interesting case since most of the clients would have a small data collection with noisy recordings that do not want to share directly but are willing

to contribute to the FL framework by sending updated weights to the server. Now the unsupervised loss follows the mixture invariant training setup [13] and can be described as shown next:

$$\mathcal{L}_{\text{unsup}} = \min_{\pi \in \Pi_{2,3}} [\mathcal{L}(\hat{\mathbf{s}}_1 + \hat{\mathbf{s}}_{\pi_1}, \mathbf{m}) + \mathcal{L}(\hat{\mathbf{s}}_{\pi_2}, \mathbf{n}_2)]. \quad (2)$$

We still force the model to produce the reconstructed speaker waveform at the first slot and assume that the sources in the input MoM $\mathbf{x} = \mathbf{s} + \mathbf{n}_1 + \mathbf{n}_2$ are independent. The assumption about clients (even unsupervised ones) having access to noise recordings $\mathbf{n}_2 \sim \mathcal{D}_c^n$ is fairly reasonable since distributing such data from the server and using them locally does not raise any privacy issues. Moreover, noise data can also be gathered independently on the client-side by running a simple voice activity detection mechanism even real-time on a modern mobile device [23].

Finally, the c -th client belonging to the set of the available nodes at the r -th communication round $c \in \mathcal{A}^r$ performs K_c local mini-batch updates to its local weights $\boldsymbol{\theta}_c^{(r)}$ by minimizing one of the aforementioned loss functions in Equations 1, 2.

3. EXPERIMENTAL FRAMEWORK

3.1. LibriFSD50K Dataset

To construct a realistic FL scenario, we combine speech data from the LibriSpeech dataset [24], and noise data from FSD50k [25]. We first split the combined 100h and 360h LibriSpeech training set into 280 folds where each fold contains data from a unique speaker ID. Then, we remove all clips with speech related tags from both train and test portions of FSD50K. The remaining noise files are split evenly across the 280 speaker-folds by matching four second noise segments to four second chunks of unique speech utterances. We leave the test sets from both datasets intact. *LibriFSD50K* consists of approximately 50, 3 and 3 hours of data for training, validation and test, respectively. To produce a training mixture with two noise sources $\mathbf{s} + \mathbf{n}_1 + \mathbf{n}_2$, we discard half of the speech recordings from a speaker and pair the singleton noise files $\mathbf{n}_2 \sim \mathcal{D}_c^n$ with another noisy speech mixture $\mathbf{m} = \mathbf{s} + \mathbf{n}_1 \sim \mathcal{D}_c^m$ from the same speaker. This procedure leads to the challenging but realistic distributions of input signal-to-noise ratio SNR shown in Figure 1.

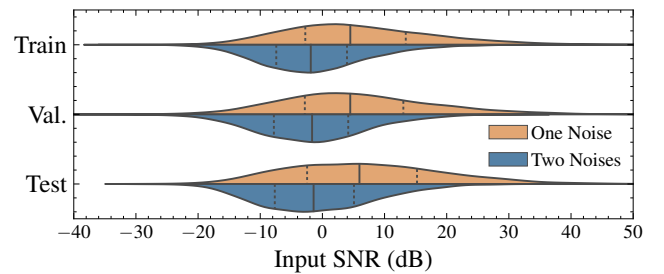


Figure 1: Input SNR distribution for the *LibriFSD50K* dataset.

3.2. Separation model

Although our method can work with any separation architecture, we use a variation of the Sudo rm -rf [26] separation architecture since it achieves a good trade-off between separation fidelity and time-memory computational requirements. This Sudo rm -rf variation,

uses 8 repetitive U-ConvBlocks as well as the group communication mechanism proposed in [27] which divides all intermediate representations in 16 groups of sub-bands and process them independently. By doing so, we obtain an efficient and light-weight model with only 794 921 trainable parameters which can easily fit and run on an edge device with 32 bit precision. We force the $M = 3$ estimated sources to add up to the input mixture using a mixture consistency layer [28]. For all the other parameters we choose the default settings provided in [29] for a sampling rate of 16k Hz.

3.3. Training details

For our simulations, a uniformly randomly sampled set with a cardinality of $|\mathcal{A}^{(r)}| = C/4$ clients is chosen to contribute to the update of r -th communication round. Each client that is available at communication round r performs training on its private data $\mathcal{D}_c = (\mathcal{D}_c^m, \mathcal{D}_c^n)$ for K optimization steps equivalent to one local epoch $K_c = \lfloor |\mathcal{D}_c|/B \rfloor$, where B is the batch size. The signal-level loss function used (see Section 2.2) is the negative permutation-invariant scale-invariant signal to distortion ratio (SI-SDR) [30]:

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = -\text{SI-SDR}(\hat{\mathbf{y}}, \mathbf{y}) = -10 \log_{10} \left(\frac{\|\alpha \mathbf{y}\|^2}{\|\alpha \mathbf{y} - \hat{\mathbf{y}}\|^2} \right), \quad (3)$$

where $\alpha = \hat{\mathbf{y}}^\top \mathbf{y} / \|\mathbf{y}\|^2$ makes the loss invariant to the scale of the estimated source $\hat{\mathbf{y}} \in \mathbb{R}^T$ and the target signal \mathbf{y} . We train all our local-client models using the Adam optimizer [31] with an initial learning rate of 10^{-3} and a batch size of $B = 6$ to obtain maximum parallelization on a single Nvidia GeForce RTX 2080 Ti.

3.4. Evaluation details

We evaluate the robustness of our learned speech-enhancement models after a communication round is complete (the server has aggregated the weights from the individual clients). We evaluate under two different noise conditions when one $\mathbf{x} = \mathbf{s} + \mathbf{n}_1$ or two noise sources $\mathbf{x} = \mathbf{s} + \mathbf{n}_1 + \mathbf{n}_2$ are active alongside the speech utterance. Specifically, we measure the SI-SDR improvement over the input mixture $\text{SI-SDR}_i = \text{SI-SDR}(\hat{\mathbf{s}}_1, \mathbf{s}) - \text{SI-SDR}(\mathbf{x}, \mathbf{s})$.

3.5. Federated learning configurations

We perform several FL simulations where the speaker data are distributed in a non-IID way across the clients (which makes the setup challenging [16] but also more realistic). Specifically, each noisy speech example $\mathbf{m} = \mathbf{s} + \mathbf{n}_1 \sim \mathcal{D}_c^m$ contains a unique speaker utterance and a unique noise snippet. Each client contains only utterances from certain speaker IDs which are not shared with any other client. In the case where we assume C clients, we split the dataset to almost equal $\lfloor 280/C \rfloor$ parts according to the speaker IDs. For each local dataset, we preserve the structure of noisy mixtures \mathcal{D}_c^m and noise recordings \mathcal{D}_c^n , as explained in Section 2.2. We discuss in detail all our setups and how they relate to real-world scenarios.

3.5.1. Unsupervised federated learning at scale

We want to provide a good baseline for the most challenging FL setup which is analyzed in this work, namely, developing a distributed and private system capable of learning to perform high fidelity speech enhancement where $C \in \{16, 64, 256\}$ clients learn directly from their limited noisy datasets. We analyze the convergence of the global model learned in a federated setup and we compare it against training on each one of the private datasets of the

C individual nodes. Moreover, we test whether our unsupervised FL setup can obtain similar accuracy to IID training of the whole dataset (all 280 speaker IDs available) on a single node.

3.5.2. Transfer learning with pre-training

Instead of distributing a random initialization for the global model in the beginning $\theta_g^{(0)}$, the server could pre-train those weights in order to provide a better initialization point for the FL system (e.g. a big company could provide its clients a pre-trained model using its big data collection to facilitate the federated learning process). We use the WHAM dataset [32] where we follow a similar training setup as previously explained but now completely supervised and on a single node. After training for 100 epochs we use those weights as the global model initialization $\theta_g^{(0)}$ and because we are fine-tuning on the client-side, we lower the learning rate of all local Adam [31] optimizers to 10^{-4} . We perform a head-to-head comparison with the same FL setups trained from scratch w.r.t. the benefits that we can get in terms of faster convergence (less communication rounds and data processed on the client-side) as well as the overall performance on the test set.

3.5.3. Combining supervised and unsupervised clients

In this experiment we fix the number of clients to $C = 64$ and focus on the capability of the FL system to harness the benefits of supervised data which are distributed across clients in a non-IID fashion. In this sense, we combine both *supervised* and *unsupervised* clients where each one performs local training, as explained in Section 2.2, with their local losses \mathcal{L}_{sup} and $\mathcal{L}_{\text{unsup}}$, respectively. We sweep the proportion of supervised nodes in $p_s \in \{0, 0.25, 0.5, 0.75, 1\}$.

4. RESULTS AND DISCUSSION

4.1. Unsupervised federated speech enhancement

In Figure 2, we depict the speech enhancement performance obtained while training for 1000 communication rounds. The convergence becomes slower as we increase the number of nodes since aggregating the updates on the weight space becomes harder [20]. However, FEDENHANCE even with $C = 256$ clients significantly outperforms individual training where the dataset is split across 16 nodes and the local models overfit. Assuming $C = 256$ clients, in each communication round r , the server needs to aggregate results from $|\mathcal{A}^{(r)}| = C/4$ available clients which accounts for a total memory of $|\theta_c^{(r)}| \cdot |\mathcal{A}^{(r)}| \approx 204$ MBs. Increasing the number of active clients per round $|\mathcal{A}^{(r)}|$ or increasing the number of local optimization steps per client K transfers the computational load from the communication side to the local audio processing side. From our empirical validation, those values do not seem to affect significantly the overall convergence in terms of communication rounds for our setups with up to $C = 256$ clients.

4.2. Transfer learning

By pre-training on the WHAM [32] dataset before distributing the first model from the server to the C clients we see that we can significantly improve the convergence speed of our federated system as shown in Figure 3. Specifically, using pre-training we close the performance gap between $C = 16$ (easy) and $C = 256$ (difficult) cases and with less than 100 hours of locally processed audio data.

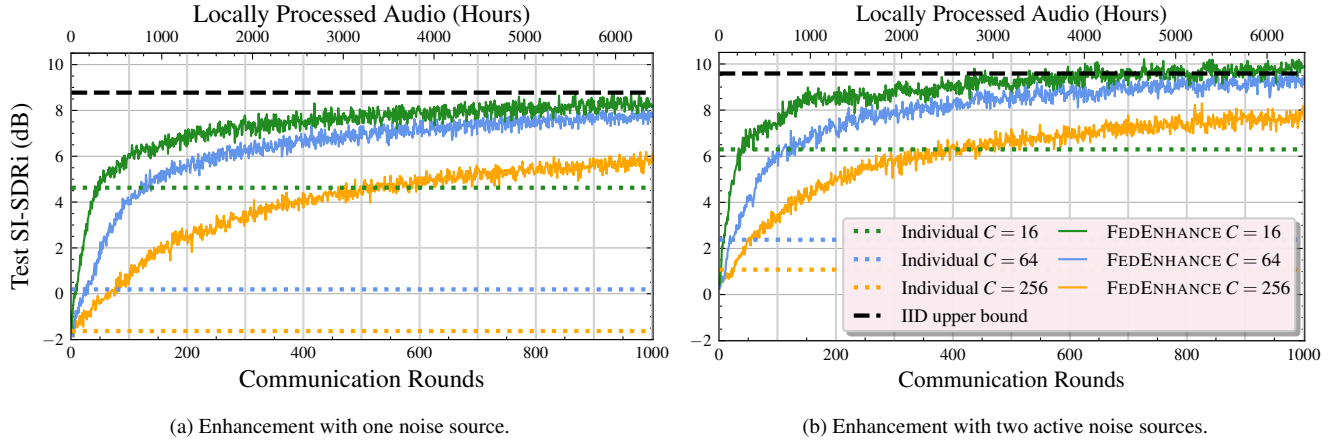


Figure 2: Speech enhancement performance obtained on *LibriFSD50K* with one noise source (left) and two noise sources (right) versus the total communication rounds between the C clients and the central server as well as the total hours of audio being processed on the client-side. The dotted straight lines symbolize the maximum performance obtained by training in isolation 5 out of C random clients for 1 000 epochs on their private data $|\mathcal{D}_c| = |\mathcal{D}|/C$ and taking their mean. The black dashed line on top is the upper bound we could expect assuming that all of the data \mathcal{D}_c , $c = 1, \dots, C$ are available for IID unsupervised training on a single node. Notice that our proposed FEDENHANCE approach is able to scale to multiple nodes when learning a speech enhancement model using only non-IID and noisy speech recordings.

The pre-trained models also achieve higher overall SI-SDRi compared to randomly initialized models on *LibriFSD50K* with one or two noise sources present (we omit the Figure for the case with one active noise source since the qualitative results are similar).

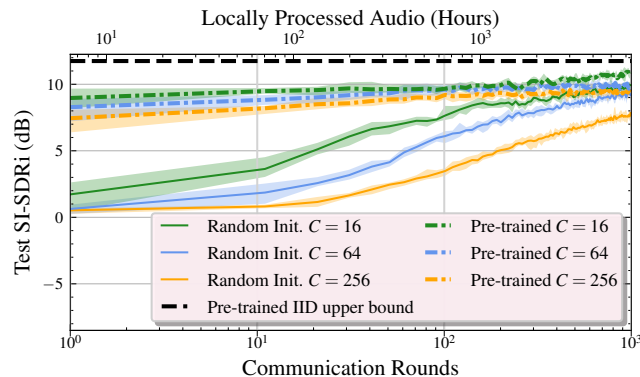


Figure 3: Convergence of FEDENHANCE on *LibriFSD50K* test set with two active noise sources when starting from the pre-trained model (dotted) and when starting from a random initialization (solid). For better visualization, we plot the mean across 10 communication rounds in log scale while the shaded regions are defined by the minimum and maximum SI-SDRi obtained in the same number of rounds. The black dashed line on top is the upper bound obtained with IID unsupervised training on a single node starting from the same pre-trained model.

4.3. Combining supervised and unsupervised clients

In Figure 4, we show the performance obtained with FEDENHANCE with $C = 64$ clients for *LibriFSD50K* validation and test sets and with the presence of one or two noise sources after 1 000 rounds. We select the model with the highest SI-SDRi on the validation set

over the last 50 communication rounds. Notice that in the case of all clients having access to supervised data (rightmost) we can obtain a Test SI-SDRi of 9.3dB and 10.9dB for one and two noise sources, respectively. By assuming 50% of supervised clients, FEDENHANCE can leverage the supervised information flow and improve upon the totally unsupervised configuration (leftmost) in terms of Test SI-SDRi (8.0 \rightarrow 8.9dB and 9.4 \rightarrow 10.5dB for one and two noise sources, respectively).

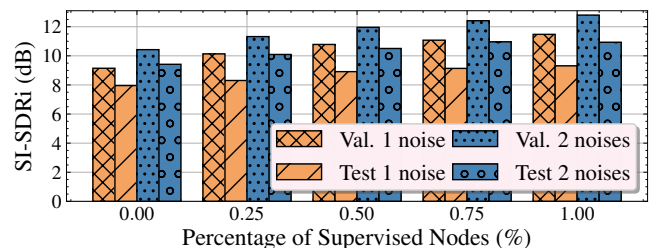


Figure 4: Speech enhancement performance on *LibriFSD50K* validation and test sets with one or two active noise sources while sweeping the number of supervised nodes from totally unsupervised FL (left) to totally supervised FL (right).

5. CONCLUSION

We have presented an unsupervised federated learning approach, namely, FEDENHANCE, which is capable of collectively training a speech enhancement model with non-IID distributed noisy speech recordings across multiple clients. Our experiments show that our approach is able to obtain competitive performance with IID training on a single node and can be further boosted by using pre-training on other datasets as well as by combining nodes with supervised data. In the future, we aim to extend our approach to multi-task problems including simultaneous sound recognition and separation.

6. REFERENCES

- [1] N. Takahashi and Y. Mitsufuji, “D3net: Densely connected multidilated densenet for music source separation,” *arXiv preprint arXiv:2010.01733*, 2020.
- [2] E. Nachmani, Y. Adi, and L. Wolf, “Voice separation with an unknown number of multiple speakers,” in *Proc. ICML*, 2020, pp. 7164–7175.
- [3] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong, “Attention is all you need in speech separation,” in *Proc. ICASSP*, 2021 (To appear).
- [4] Z.-Q. Wang, P. Wang, and D. Wang, “Complex spectral mapping for single-and multi-channel speech enhancement and robust asr,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1778–1787, 2020.
- [5] U. Isik, R. Giri, N. Phansalkar, J.-M. Valin, K. Helwani, and A. Krishnaswamy, “PoCoNet: Better Speech Enhancement with Frequency-Positional Embeddings, Semi-Supervised Conversational Data, and Biased Loss,” in *Proc. Interspeech*, 2020, pp. 2487–2491.
- [6] A. Sivaraman, S. Kim, and M. Kim, “Personalized speech enhancement through self-supervised data augmentation and purification,” *arXiv preprint arXiv:2104.02018*, 2021.
- [7] F. Pishdadian, G. Wichern, and J. Le Roux, “Finding strength in weakness: Learning to separate sounds with weak supervision,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2386–2399, 2020.
- [8] Q. Kong, H. Liu, X. Du, L. Chen, R. Xia, and Y. Wang, “Speech enhancement with weakly labelled data from audioset,” *arXiv preprint arXiv:2102.09971*, 2021.
- [9] E. Tzinis, S. Venkataramani, and P. Smaragdis, “Unsupervised deep clustering for source separation: Direct learning from mixtures using spatial information,” in *Proc. ICASSP*, 2019, pp. 81–85.
- [10] P. Seetharaman, G. Wichern, J. Le Roux, and B. Pardo, “Bootstrapping single-channel source separation via unsupervised spatial clustering on stereo mixtures,” in *Proc. ICASSP*, 2019, pp. 356–360.
- [11] L. Drude, D. Hasenklever, and R. Haeb-Umbach, “Unsupervised training of a deep clustering model for multichannel blind source separation,” in *Proc. ICASSP*, 2019, pp. 695–699.
- [12] R. Gao and K. Grauman, “Co-separating sounds of visual objects,” in *Proc. ICCV*, 2019, pp. 3879–3888.
- [13] S. Wisdom, E. Tzinis, H. Erdogan, R. Weiss, K. Wilson, and J. Hershey, “Unsupervised sound separation using mixture invariant training,” in *Advances in Neural Information Processing Systems*, vol. 33, 2020.
- [14] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, “Federated learning: Strategies for improving communication efficiency,” *arXiv preprint arXiv:1610.05492*, 2016.
- [15] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.
- [16] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, “Robust and communication-efficient federated learning from non-iid data,” *IEEE transactions on neural networks and learning systems*, vol. 31, no. 9, pp. 3400–3413, 2019.
- [17] D. Leroy, A. Coucke, T. Lavril, T. Gisselbrecht, and J. Dureau, “Federated learning for keyword spotting,” in *Proc. ICASSP*, 2019, pp. 6341–6345.
- [18] D. Guliani, F. Beaufays, and G. Motta, “Training speech recognition models with federated learning: A quality/cost framework,” *arXiv preprint arXiv:2010.15965*, 2020.
- [19] D. S. Johnson, W. Lorenz, M. Taenzer, S. Mimilakis, S. Grollmisch, J. Abeßer, and H. Lukashevich, “Desed-fl and urban-fl: Federated learning datasets for sound event detection,” *arXiv preprint arXiv:2102.08833*, 2021.
- [20] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al., “Advances and open problems in federated learning,” *arXiv preprint arXiv:1912.04977*, 2019.
- [21] B. van Berlo, A. Saeed, and T. Ozcelebi, “Towards federated unsupervised representation learning,” in *Proc. EdgeSys*, 2020, pp. 31–36.
- [22] M. Servetnyk, C. C. Fung, and Z. Han, “Unsupervised federated learning for unbalanced data,” in *Proc. GLOBECOM*, 2020, pp. 1–6.
- [23] A. Sehgal and N. Kehtarnavaz, “A convolutional neural network smartphone app for real-time voice activity detection,” *IEEE Access*, vol. 6, pp. 9017–9026, 2018.
- [24] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, “Librispeech: an asr corpus based on public domain audio books,” in *Proc. ICASSP*, 2015, pp. 5206–5210.
- [25] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, “Fsd50k: an open dataset of human-labeled sound events,” *arXiv preprint arXiv:2010.00475*, 2020.
- [26] E. Tzinis, Z. Wang, and P. Smaragdis, “Sudo rm-rf: Efficient networks for universal audio source separation,” in *Proc. MLSP*, 2020, pp. 1–6.
- [27] Y. Luo, C. Han, and N. Mesgarani, “Ultra-lightweight speech separation via group communication,” in *Proc. ICASSP*, 2021 (To appear).
- [28] 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, pp. 900–904.
- [29] E. Tzinis, Z. Wang, X. Jiang, and P. Smaragdis, “Compute and memory efficient universal sound source separation,” *arXiv preprint arXiv:2103.02644*, 2021.
- [30] J. Le Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, “Sdr-half-baked or well done?” in *Proc. ICASSP*, 2019, pp. 626–630.
- [31] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [32] G. Wichern, J. Antognini, M. Flynn, L. R. Zhu, E. McQuinn, D. Crow, E. Manilow, and J. L. Roux, “WHAM!: Extending Speech Separation to Noisy Environments,” in *Proc. Interspeech*, 2019, pp. 1368–1372.