2024 IEEE Spoken Language Technology Workshop (SLT) | 979-8-3503-9225-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/SLT61566.2024.10832160

# INVESTIGATION OF SPEAKER REPRESENTATION FOR TARGET-SPEAKER SPEECH PROCESSING

*Takanori Ashihara*[1], *Takafumi Moriya*[1], *Shota Horiguchi*[1], *Junyi Peng*[2], *Tsubasa Ochiai*[1], *Marc Delcroix*[1], *Kohei Matsuura*[1], *Hiroshi Sato*[1]

[1]NTT Corporation, Japan
[2]Brno University of Technology, Czechia

## ABSTRACT

Target-speaker speech processing (TS) tasks, such as target-speaker automatic speech recognition (TS-ASR), target speech extraction (TSE), and personal voice activity detection (p-VAD), are important for extracting information about a desired speaker's speech even when it is corrupted by interfering speakers. While most studies have focused on training schemes or system architectures for each specific task, the auxiliary network for embedding target-speaker cues has not been investigated comprehensively in a unified cross-task evaluation. Therefore, this paper aims to address a fundamental question: what is the preferred speaker embedding for TS tasks? To this end, for the TS-ASR, TSE, and p-VAD tasks, we compare pre-trained speaker encoders (i.e., self-supervised or speaker recognition models) that compute speaker embeddings from pre-recorded enrollment speech of the target speaker with ideal speaker embeddings derived directly from the target speaker's identity in the form of a one-hot vector. To further understand the properties of ideal speaker embedding, we optimize it using a gradient-based approach to improve performance on the TS task. Our analysis reveals that speaker verification performance is somewhat unrelated to TS task performances, the one-hot vector outperforms enrollment-based ones, and the optimal embedding depends on the input mixture.

***Index Terms***— Speaker representation, target-speaker automatic speech recognition, target speech extraction, personal voice activity detection, self-supervised learning

## 1. INTRODUCTION

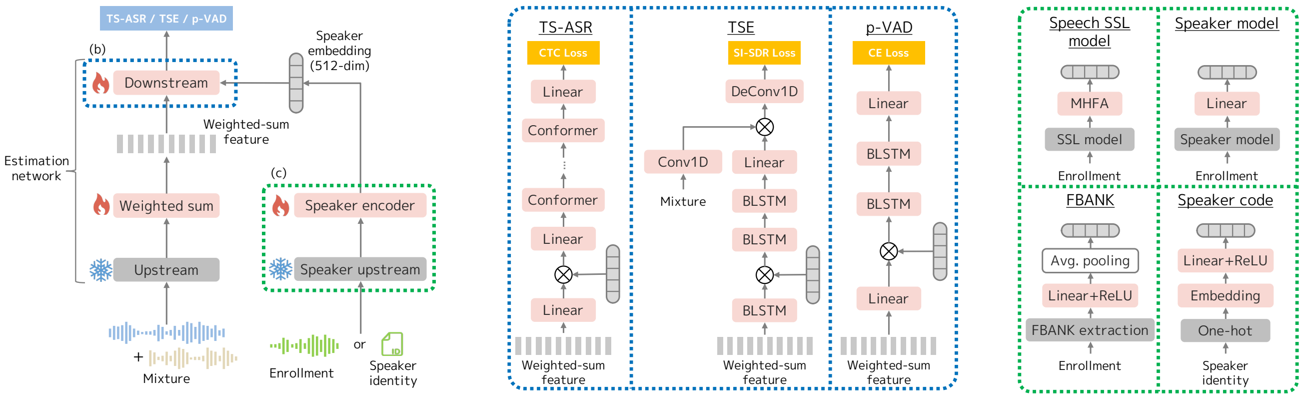
In daily conversational scenarios, a desired (or target) speaker's speech is often degraded by the speech of other (interfering) speakers. To tackle such realistic situations, several speech processing models have been extended to enable conditioning with cues from a target speaker [1–14], referred to as target-speaker speech processing (TS) models. TS models aim to extract information about a target speaker from a speech recording of multiple speakers. While TS tasks are more challenging than conventional single-speaker tasks due to the need to handle two processes—identifying the speaker of interest and extracting desired information—in a single model, this speaker-conditioned speech processing has attracted much attention, achieving significant improvements in multi-talker speech processing challenges [15, 16].

TS tasks cover various areas depending on the nature of the information to be extracted, such as content, speech, and segments. For *content*, target-speaker automatic speech recognition (TS-ASR) [1–5] focuses on transcribing utterances of a target speaker from a speech mixture and has shown to be more effective than standard ASR models without any speaker cues. Regarding *speech*, target speech extraction (TSE) is designed to estimate a clean signal of a target speaker from a mixture of talkers [1, 6–10]. With respect to *segments*, personal voice activity detection (p-VAD) focuses on determining whether a frame of audio contains a target speaker's speech in a mixture sequence [11–14].

As shown in Fig. 1(a), standard TS systems developed with a neural network (NN) consisting of a recognizing/enhancing network (hereinafter referred to as an estimation network) with a speech mixture as input, which is conditioned on a speaker embedding. This embedding helps to identify the target speaker in a mixture. It is computed with an auxiliary network that accepts a pre-recorded enrollment of a target speaker as input. Several options for obtaining speaker embeddings have been proposed [1, 3, 6–8], such as using signal processing acoustic features or a one-hot vector corresponding to a speaker identity (hereinafter called the *speaker code*) [7] passed through multilayer NNs as an auxiliary network. Other studies employed pre-trained speaker representations obtained from off-the-shelf models such as a Gaussian mixture model (i.e., i-vector [17]) [2,6,12], NN-based supervised speaker models (such as d-vector [18] and x-vector [19]) [4,9,11,13,14], and self-supervised learning (SSL) models [9, 10].

Despite the progress of TS systems, studies on the properties of auxiliary networks and trained speaker representations are relatively limited. Therefore, the present study addresses a fundamental question: what auxiliary networks or speaker representations are suitable for TS tasks? Moreover, by conducting a cross-task evaluation, we aim to understand whether the optimal speaker representation is consistent or varies across different TS tasks. Answering this question may indicate whether the synergy between the tasks should be explored more thoroughly.

In this paper, we explore speaker embeddings from an auxiliary network by comprehensively comparing the performance of diverse SSL models, supervised speaker recognition models, and a speaker code on TS-ASR, TSE, and p-VAD under a unified experimental environment. Intuitively, a speaker code applicable only to *speaker-closed* conditions, where the test speaker is fully included in the training set, would surpass enrollment-based models, in which embeddings are derived from the enrollment samples. This is because a speaker code-based TS system can directly learn embeddings that optimally capture speaker characteristics for each TS task, unlike an enrollment-based system. However, it is unclear if there are even more optimal speaker embeddings that could further improve TS tasks. Therefore, inspired by successes in the computer vision community [20–23], we additionally perform gradient-based optimization on the speaker embedding so that the score for the true class is maximized. Our major findings are as follows: (1) While a different auxiliary network is suitable for each TS task, SSL models, includ-




(a) System overview used in each TS task

(b) Downstream models for each TS task

(c) Auxiliary network

**Fig. 1**: Schematic diagrams of the SUPERB-based TS evaluation system.

ing both typical Transformer-based and ECAPA-TDNN-based models, are generally effective as auxiliary networks for all three tasks. (2) Although the representation of speaker embeddings tends to form clusters by speaker, especially for the TS-ASR and TSE tasks, the performance of the automatic speaker verification (ASV) task is irrelevant to the performance of the TS tasks. (3) While a speaker code outperforms other auxiliary networks, gradient-based optimization reveals that the embedding can be further refined. (4) Gradient-based optimization also shows that the optimal embedding shifts to make one speaker more distinguishable from others and varies according to the input mixture. We hope that the results of this exploration will provide insight into developing more effective speaker representations for TS tasks.

## 2. METHOD

### 2.1. SUPERB-based target-speaker speech processing system

To evaluate speaker representations from an auxiliary network, we developed a unified experimental environment across TS tasks. As illustrated in Fig. 1(a), the design of the estimation network follows the SUPERB framework [24]. The input to the downstream model consists of a weighted-sum sequence of the outputs from each layer of the upstream model, where only the weights and downstream model are trainable, while the upstream model parameters remain frozen. By adopting the SUPERB framework, speaker representation can be evaluated efficiently using downstream models with relatively small parameters compared to training an entire system from scratch. This efficiency is made possible by speech SSL upstream models pre-trained on large amounts of unlabeled data. To condition the estimation network on target-speaker cues, the SUPERB framework is extended to accept speaker embeddings. Here, 512-dimensional speaker embeddings are used for all tasks in this paper.

#### *2.1.1. Downstream model in estimation network*

Figure 1(b) illustrates the downstream model for each TS task. Note that we always use the Hadamard product for speaker conditioning in the estimation network regardless of the downstream model and task.[1] Furthermore, we empirically determined which layer to condition with speaker embeddings.

[1]We experimented with different conditioning methods, such as addition and concatenation, but the Hadamard product proved to be the most effective across all three tasks.

**TS-ASR:** The left panel of Fig.1(b) shows the downstream model architecture for TS-ASR. Although the architecture of the ASR task in SUPERB utilizes two layers of bidirectional long short-term memory (BLSTM) with 1024 dimensions, the TS-ASR system using this architecture exhibited instability of training and unreasonably low performance in preliminary experiments. Therefore, we employ multiple blocks of Conformer [25], as utilized in previous ASR/TS-ASR studies [3, 4, 25]. Since one of the purposes of this research is to compare auxiliary networks in a unified system rather than achieving state-of-the-art performance in TS-ASR, we empirically chose the Conformer model with the minimum number of parameters (approximately 7 million) while ensuring reasonable performance. The speaker embedding is applied to the output from the first linear layer, which aligns the dimension of the weighted-sum feature with the embedding (i.e., 512 dimensions). This conditioned feature is then fed into a subsequent linear layer to match the dimensions of the Conformer blocks. Finally, an additional linear layer is applied on top of the Conformer blocks to predict a token sequence of the target speaker. The tokens and training loss adhere to the ASR task in SUPERB, specifically character tokens and connectionist temporal classification (CTC) loss [26].

**TSE:** The middle section of Fig.1(b) depicts the architecture of the downstream model used for TSE. The downstream architecture is based on the speech enhancement (SE) task in SUPERB-SG [27] with some modifications. Specifically, similar to the original SE task in SUPERB-SG, the system also consists of three BLSTM layers to predict the mask for the target speaker's signal. To mask the mixture signal, the SE recipe in SUPERB-SG utilizes short-time Fourier transform (STFT) and inverse STFT (iSTFT) to encode and decode the mixture signal and masked spectral signal, respectively. Meanwhile, a recent SSL-based TSE system [10] demonstrated higher extraction performance compared to the STFT/iSTFT pipeline by utilizing 1D-convolution and deconvolution layers. This improvement is attributed to the joint optimization of the encoder/decoder, which led to task-suitable frequency bands. Therefore, we also incorporate 1D-convolution and deconvolution layers for the encoder/decoder. To inject the target speaker cue, the speaker embedding is multiplied by the output from the first BLSTM layer, which is determined empirically. For the training objective, we minimize the scale-invariant signal-to-distortion ratio (SI-SDR) [28] between the inferred masked signal and the target speaker's clean signal. According to a previous study [10], this objective achieves better extraction performance than the mean square error used in the SE recipe of SUPERB-SG.

**p-VAD:** In Fig.1(b), the right panel illustrates the downstream model architecture employed for p-VAD. We use the *embedding conditioned training* from the original p-VAD paper [11] with certain adjustments to align with the other TS tasks. Specifically, the weighted-sum feature is fed into a linear layer to match the dimension of the speaker embedding, and the features are multiplied by the embedding using the Hadamard product. The speaker-conditioned feature is then processed by two BLSTM layers, followed by a linear layer that predicts three classes: non-speech (`ns`), target speaker speech (`tss`), and non-target speaker speech (`ntss`) using the BLSTM output. To train the model, we minimize the cross-entropy (CE) loss as described in the original p-VAD paper [11].

### 2.1.2. Speaker upstream and encoder models in auxiliary network

Figure 1(c) illustrates the auxiliary networks consisting of a speaker upstream and a speaker encoder. For the speaker upstream, similar to the upstream and downstream models of SUPERB, the parameters are always fixed to ensure a fair comparison of pre-trained models. In contrast, the parameters of the speaker encoder are updated during training in each TS task so that task-optimized embeddings can be established.

**Speech SSL model:** To encode an enrollment waveform, our study applies speech SSL models as a speaker upstream, as displayed in the top left of Fig. 1(c). Regarding the speaker encoder, since the speech SSL model's output is a sequential feature, the sequence needs to be accumulated over time to form a single embedding vector. We used multi-head factorized attention (MHFA) [29] to aggregate layer- and frame-wise features, which has been shown to be more effective than temporal averaging of the weighted-sum features.

**Speaker model:** We can also use speaker embedding extractors trained for speaker recognition tasks (e.g., speaker identification (SID) and ASV) [17–19, 30–38] as an auxiliary network. For example, the network architecture is time-delay neural networks (TDNNs) used by x-vector [19] or ECAPA-TDNN [30]. Our study employs the speaker models as one of the variations of an auxiliary network, as displayed in the top right of Fig. 1(c). The output vector is further fed into a linear layer to align the dimensions to 512.

**FBANK:** Several TS models have used hand-crafted acoustic features as input instead of outputs from pre-trained models [1, 3, 6–8]. The FBANK auxiliary network used in our study is depicted in the bottom left of Fig. 1(c). Specifically, log mel-filterbank features extracted from an enrollment speech are fed into a linear layer with a rectified linear unit (ReLU), followed by temporal average pooling to aggregate information in the time domain.

**Speaker code:** In a *speaker-closed* task setting, where the target speaker of a test set is completely seen during training, a one-hot vector corresponding to the speaker's identity (called speaker code) could be a suitable speaker embedding, as it can optimally represent speaker properties for TS tasks. Therefore, we apply a speaker code to the TS tasks as illustrated in the bottom right of Fig. 1(c). For the architecture, a one-hot vector corresponding to a speaker identity is transformed by an embedding layer and input into a linear layer with ReLU. While auxiliary networks other than a speaker code can work with a *speaker-open* condition where test speakers are not seen during training, the speaker code is only applicable in a *speaker-closed* condition, where test speakers are fully duplicated in the training data. Although this limitation is somewhat impractical, we verify its performance and representation as one of the most effective auxiliary networks.

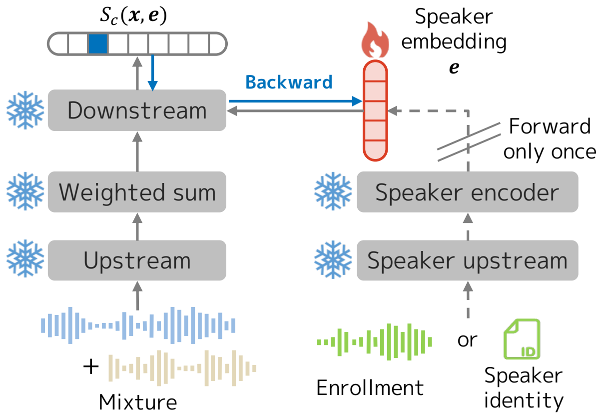



**Fig. 2**: Schematic diagram of gradient-based speaker embedding optimization. The forwarding process of an auxiliary network is executed only once to obtain an initial speaker embedding to be optimized.

## 2.2. Gradient-based speaker embedding optimization

To explore optimal speaker embeddings, we propose directly optimizing a speaker embedding using a gradient calculated from a mixture and the corresponding ground truth of a target speaker. This approach is summarized in Fig.2 and is inspired by the successes of prior computer vision studies [20–23].

In the procedure, using a mixture and enrollment speech or speaker identity, weighted-sum feature $\mathbf{x}_t$ and speaker embedding $\mathbf{e}_0$ are calculated from an upstream model followed by weighted-sum computation with trained weights and a trained auxiliary network, respectively. These features are then fed into a trained downstream model to obtain a score $S_{c_t}(\mathbf{x}_t, \mathbf{e}_0)$ (i.e., an output of a final prediction head before a softmax function) of a true class $c_t$. Here, a true class $c_t = [c_1, \dots, c_T]$ is, for example, a token sequence corresponding to a ground-truth transcription for the TS-ASR task, where $t$ and $T$ are a temporal frame index and the total number of frames, respectively. To optimize the embedding directly, we use the unnormalized class score to discover an embedding where $S_{c_t}$ achieves a high value (i.e., $\arg\max_{\mathbf{e}} S_c(\mathbf{x}, \mathbf{e})$). This has been shown to be more suitable than using the posteriors for analyzing the network behavior, as the class posterior can be maximized by minimizing the scores of other classes [20, 21]. Note that while we attempted to minimize the standard CE loss in preliminary TS-ASR experiments, we also verified that optimizing the embedding by maximizing the score led to a more significant reduction in error rates compared to the slight reduction achieved through CE loss minimization. During optimization, we iteratively subtract a gradient multiplied by a step size $\alpha$ from an input embedding to adjust it in a direction that increases the score. The update equation at the $n$-th iteration is

$$\mathbf{e}_{n+1} = \mathbf{e}_n - \alpha \nabla_{\mathrm{e}} \sum\nolimits_t S_{c_t}(\mathbf{x}_t, \mathbf{e}_n), \quad (1)$$

where $n \in \{0, \dots, N-1\}$. Here, $\nabla_{\mathrm{e}} \sum_t S_{c_t}(\mathbf{x}_t, \mathbf{e}_n)$ and $N$ denotes the gradient of a score with respect to an embedding and the number of iterations for optimization, respectively. During optimization, the speaker embedding is trainable, while the other parameters remain frozen. In this paper, we apply this approach to the trained TS-ASR models, as TSE is not a classification problem, making it difficult to straightforwardly apply score maximization. Additionally, the performance of p-VAD has almost reached saturation, leaving seemingly little room for further optimization.

## 3. EXPERIMENTAL SETUP

### 3.1. Target-speaker speech processing system

The system for the TS tasks was based on S3PRL[2] developed for SUPERB [24]. While SUPERB mainly focuses on comparing the performance of upstream models, our objective is to compare auxiliary networks. Therefore, we utilized WavLM BASE+ [39] as the upstream model, since it is reported to be the optimal upstream model in TSE [10]. Note that we observed a similar trend with the HuBERT BASE, although the overall performance was generally lower.

**Speech SSL model:** For the speaker upstream model, we compare wav2vec2.0 BASE [40], HuBERT BASE [41], data2vec BASE [42], WavLM BASE, WavLM BASE+, and WavLM LARGE [39], as summarized in Table 1. All models are publicly available (or directly callable via S3PRL). After extracting features through speaker upstream models, MHFA, configured with eight attention heads and a compression layer with a dimension of 128, was applied to yield an embedding with 512 dimensions.

**Speaker model:** We employed x-vector [19] and ECAPA-TDNN [30], which were trained using the WeSpeaker toolkit [43]. These models were pre-trained on supervised speaker identification tasks with angular additive margin softmax loss, followed by large margin fine-tuning, a technique widely used in speaker verification challenges [44]. Other training details followed the WeSpeaker's VoxCeleb2 [45] recipes. The architectures were identical to that of the originals [19, 30], with the ECAPA-TDNN models having different numbers of channels (512 and 1024, denoted as ECAPA-TDNN-c512 and ECAPA-TDNN-c1024, respectively). Additionally, considering recent developments in utterance-wise speech SSL models for speaker representation [31–38, 46], we employed the *self-DIstillation with NO labels* (DINO) [47] model with the ECAPA-TDNN architecture and 1024 channels. The DINO model was trained in an SSL fashion using a self-distillation framework that transfers knowledge from a teacher model to a student model, resulting in higher ASV performance than other utterance-wise speech SSL models (for more details, see the previous papers [34–38, 46, 47]). The training configuration for the DINO model is identical to the WeSpeaker setting.[3] Note that the amount of pre-training data from VoxCeleb2 differs from that of the off-the-shelf speech SSL models as described above. Nevertheless, VoxCeleb2 is widely used in ASV tasks and competitions [44] and contains a relatively large amount of data with supervisory speaker labels. Therefore, we believe it is a reasonable choice for comparing the most promising and widely accepted speaker models in recent years, similar to the speech SSL models, offering valuable insights into aspects such as model efficiency and effectiveness relative to data and model size.

**FBANK:** We extracted 80-dimensional log mel-filterbank outputs using a 25-ms window and a 10-ms shift. The linear layer that processes the acoustic features has 512 dimensions.

**Speaker code:** The input speaker identity depends on the number of speakers present in the training set. As described in Section 3.2, we utilized the training and evaluation dataset from Libri2Mix, which includes 251 speakers. The one-hot vector was transformed into a 512-dimensional embedding through an embedding layer, followed by a linear layer with 512 dimensions and a ReLU activation. Note that although we conducted comprehensive preliminary experiments without the linear layer and ReLU (i.e., using only the embedding layer), the TS models did not converge.

[2]https://github.com/s3prl/s3prl

[3]https://github.com/wenet-e2e/wespeaker/tree/master/examples/voxceleb/v3/dino

### 3.2. Downstream tasks

**Dataset:** We utilized LibriMix [52] for the training and evaluation dataset of all TS tasks. Specifically, the training set was the `train-100` subsets in Libri2Mix (i.e., two speakers in a mixture), which included `clean` and `both` subsets (without any noise and with WHAM! noise [53], respectively). The corresponding enrollment speech was prepared according to SpeakerBeam's configuration[4] [6, 8]. Following this configuration, an enrollment speech utterance was assigned on a one-to-one basis for each mixture in the test and development subsets, unlike in the training set, where it was selected randomly from multiple enrollment utterances of the target speaker. The test subsets prepared by the above procedure were called the *speaker-open* evaluation set and were used unless otherwise mentioned. For the speaker code experiment, as described in Section 2.1.2, the speakers in the test data must be included in the training data. Therefore, we created an additional test subset called the *speaker-closed* set, consisting of 6000 mixtures, which is the same number as the *speaker-open* set. The speakers of this subset were fully contained in the training set, while the utterances were not duplicated. All datasets were sampled at 16kHz. For the true label of the p-VAD task, we created labels from the diarization labels[5] used in the speaker diarization task of SUPERB.

**Task configuration:** The training hyperparameters were based on the ASR[6], SE[7], and speaker diarization[8] recipes, for the TS-ASR, TSE, and p-VAD tasks, respectively. Only changes from the original configuration are described below.

For **TS-ASR**, the learning rate was warmed up to 0.001 in the first 15k steps and then linearly decayed for the remaining steps. Eight Conformer blocks were used, each consisting of four attention heads with 144 dimensions, a 1D convolution with a kernel size of 15, and a feedforward module with 1024 dimensions. Performance was reported in terms of word error rates (WERs).

For **TSE**, the total training and validation steps were set to 150k and 4k steps, respectively. The Adam optimizer with 3k warm-up steps followed by linear decay was used. The evaluation metrics were SI-SDR [28], short-time objective intelligibility (STOI) [54], and perceptual evaluation of speech quality (PESQ) [55].

For **p-VAD**, two BLSTM layers with 128 cells were used as a downstream model. The trained models were evaluated by calculating the mean average precision (mAP) over all classes.

In addition to the TS tasks described above, the performance of **ASV** was measured to evaluate the quality of speaker representation. While we followed the ASV setup of SUPERB using VoxCeleb1 [45], the downstream models were substituted to align with the auxiliary network. Specifically, we utilized the MHFA for speech SSL models and identity processing for speaker models. Identity processing is a parameter-less module that treats the output of speaker models as the resultant embedding and has been reported to outperform the original downstream models (i.e., multi-layer TDNN) [46]. The models are evaluated in terms of equal error rates (EERs).

### 3.3. Gradient-based speaker embedding optimization

The number of iterations $N$ explained in Section 2 was set to 100, which was large enough to saturate performance. We utilized the Adam optimizer to determine the direction of the gradient, i.e.,

[4]https://github.com/BUTSpeechFIT/speakerbeam

[5]https://github.com/s3prl/LibriMix

[6]https://github.com/s3prl/s3prl/tree/main/s3prl/downstream/asr

[7]https://github.com/s3prl/s3prl/tree/main/s3prl/downstream/separation_stft2

[8]https://github.com/s3prl/s3prl/tree/main/s3prl/downstream/diarization

**Table 1**: Evaluation results of each task with various network combinations. Each slash-separated cell of the TS tasks shows the results on the `clean` subset (left) and the `both` subset (right) in Libri2Mix. The ASV task follows SUPERB [24] except for downstream model architectures. `LS`, `LL`, `GS`, `VP`, and `VC` denote LibriSpeech [48], Libri-Light [49], GigaSpeech [50], VoxPopuli [51], and VoxCeleb2 [45], respectively.

| | | | | TS-ASR | TSE | | | p-VAD | ASV |
|---|---|---|---|---|---|---|---|---|---|
| Auxiliary network | Speaker upstream | Model size | Pre-training data | WER (%)↓ | SI-SDR (dB)↑ | STOI (%)↑ | PESQ↑ | mAP (%)↑ | EER (%)↓ |
| FBANK | FBANK | - | - | 27.62 / 50.34 | 8.62 / 7.16 | 84.60 / 76.77 | 1.782 / 1.377 | **99.00** / 97.31 | 25.72 |
| Speech SSL model | wav2vec2.0 BASE [40] | 94M | `LS` | 19.25 / 39.53 | 10.50 / 8.40 | 88.68 / 79.58 | **1.920** / 1.418 | 98.83 / 97.44 | 3.37 |
| | HuBERT BASE [41] | 94M | `LS` | 19.08 / 38.74 | 10.57 / 8.39 | 88.66 / 79.64 | 1.912 / 1.415 | 98.93 / 97.42 | 3.08 |
| | data2vec BASE [42] | 94M | `LS` | 19.25 / 38.66 | 10.60 / 8.51 | 88.79 / 79.75 | 1.915 / 1.418 | 98.53 / 96.96 | 3.94 |
| | WavLM BASE [39] | 94M | `LS` | 19.86 / 39.19 | **10.61** / 8.70 | **88.85** / 80.30 | 1.917 / **1.425** | 98.77 / 97.43 | 2.86 |
| | WavLM BASE+ [39] | 94M | `LL` + `GS` + `VP` | 19.21 / 38.70 | 10.58 / **8.78** | 88.78 / **80.50** | 1.907 / 1.419 | 98.37 / 97.15 | 2.10 |
| | WavLM LARGE [39] | 315M | `LL` + `GS` + `VP` | 19.00 / 37.93 | 10.37 / 8.53 | 88.23 / 79.84 | 1.881 / 1.414 | 98.76 / 97.52 | 1.79 |
| Speaker model | x-vector [19] | 5M | `VC` | 19.01 / 40.33 | 10.01 / 7.80 | 87.54 / 78.26 | 1.856 / 1.384 | 98.76 / 96.95 | 2.50 |
| | ECAPA-TDNN-c512 [30] | 6M | `VC` | 19.68 / 39.74 | 9.89 / 7.81 | 87.42 / 78.33 | 1.851 / 1.392 | 98.50 / 96.59 | 1.73 |
| | ECAPA-TDNN-c1024 [30] | 15M | `VC` | 20.64 / 40.76 | 9.64 / 7.65 | 86.99 / 77.94 | 1.856 / 1.383 | 98.28 / 96.70 | **1.32** |
| | ECAPA-TDNN-DINO | 15M | `VC` | **18.84** / **37.88** | 10.59 / 8.40 | **88.85** / 79.63 | 1.904 / 1.401 | 98.82 / **97.54** | 2.67 |

$\nabla_{\mathrm{e}} \sum_t S_{c_t}(\mathbf{x}_t, \mathbf{e}_n)$ in Equation 1. Since the original dataset does not have frame-level transcriptions, we used outputs from the standard single-speaker ASR model as the pseudo-ground-truth labels. The ASR model was trained using Libri2Mix without interfering speakers and demonstrated a 3.78% WER on the *speaker-closed* subset. The learning rate was empirically optimized for each auxiliary network and was set to 4 for WavLM BASE+ and 1 for FBANK and the speaker code.

# 4. RESULTS

## 4.1. Task performances

Table 1 shows the results for each TS task using the WavLM BASE+ upstream model on the `clean` and `both` subsets, i.e., without any noise and with WHAM! noise, respectively. FBANK demonstrated the lowest performance for all tasks except for p-VAD, suggesting that the pre-training scheme is promising for the auxiliary network of all TS tasks. However, the FBANK result in p-VAD suggested that simple signal processing methods may be sufficient for relatively easier tasks. Notably, while experimenting with a deeper speaker encoder, FBANK performance improved up to two linear layers but remained poor.[9] As for pre-trained models, although the supervised ECAPA-TDNN models (ECAPA-TDNN-c512 and ECAPA-TDNN-c1024) achieved higher ASV performance than the speech SSL models, their performance on the TS tasks was comparable or lower. This suggests that *the performance between TS and ASV tasks is uncorrelated*, indicating the need for other metrics or tasks to identify a suitable pre-trained speaker upstream for an auxiliary network rather than relying on ASV, which is a basic task evaluating the quality of speaker representation. This tendency extends the findings of previous research [9], which only validated the TSE task, to the TS-ASR and p-VAD tasks. Moreover, comparing the optimal configuration across the tasks, we observed that *ECAPA-TDNN-DINO was the most effective for all three tasks*. This result suggests that the synergy between the TS tasks could be exploited to obtain more suitable speaker representations for speaker-conditioned speech processing.

Table 2 demonstrates the results of the speaker code on the *speaker-closed* evaluation subset under two conditions, `clean` and `both`. For all three tasks, the speaker code demonstrates higher performance in both conditions compared to the other models. Therefore, we can conclude that *the speaker code provides one of the most effective speaker representations for all three tasks*, although the *speaker-closed* condition is not practical as described in Section 2.1.2. Future work includes investigating methods to minimize the distance between the representations of pre-trained models and the speaker code, for example, through knowledge distillation.

[9]For example, the FBANK model with two linear layers achieved 23.92% and nearly reached saturation on the `clean` set for the TS-ASR task.

**Table 2**: Evaluation results for speaker code on the *speaker-closed* evaluation sets. WavLM BASE+ is used as the upstream model.

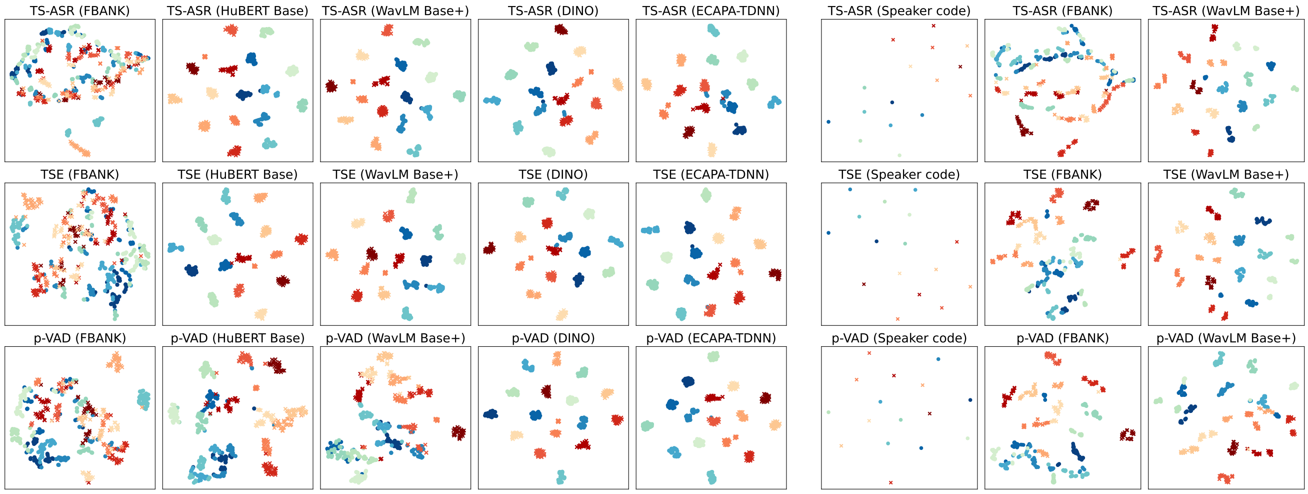
| | | TS-ASR | TSE | p-VAD |
|---|---|---|---|---|
| Mixture | Speaker upstream | WER (%)↓ | SI-SDR (dB)↑ | mAP (%)↑ |
| `clean` | FBANK | 32.81 | 9.90 | 98.96 |
| | WavLM BASE+ [39] | 21.79 | 11.31 | 98.87 |
| | Speaker code | **18.56** | **11.70** | **99.82** |
| `both` | FBANK | 55.29 | 8.02 | 97.57 |
| | WavLM BASE+ [39] | 43.18 | 9.21 | 97.49 |
| | Speaker code | **37.50** | **9.60** | **98.95** |

## 4.2. Embedding of each model and task

Figure 3 visualizes 512-dimensional speaker embeddings reduced to two dimensions using t-distributed stochastic neighbor embedding (t-SNE) [56]. The panels in each column represent each auxiliary network, and the panels in each row represent each TS task. As shown in Fig. 3(a), which shows the embeddings on the *speaker-open* set, *the pre-trained models (i.e., the right four columns) form speaker clusters, especially for the TS-ASR and TSE tasks*. Note that the speech SSL models were not explicitly trained to group by speaker, indicating the importance of representing the same speaker similarly for TS tasks. However, the panels in the last row exhibit high intra-speaker variability, suggesting that p-VAD does not require strict speaker representation that provides similar embeddings for the same speaker. This is supported by the fact that, while clear speaker clusters appear even in p-VAD, the performance of ECAPA-TDNN models is comparable to or lower than that of other models. Figure 3(b), which illustrates the embeddings on the *speaker-closed* subset, shows a similar tendency for FBANK and WavLM BASE+. The embeddings of the speaker code tend to be grouped by gender, except for the p-VAD task, which further supports the notion that strictly grouped embeddings are not necessary for the p-VAD task.

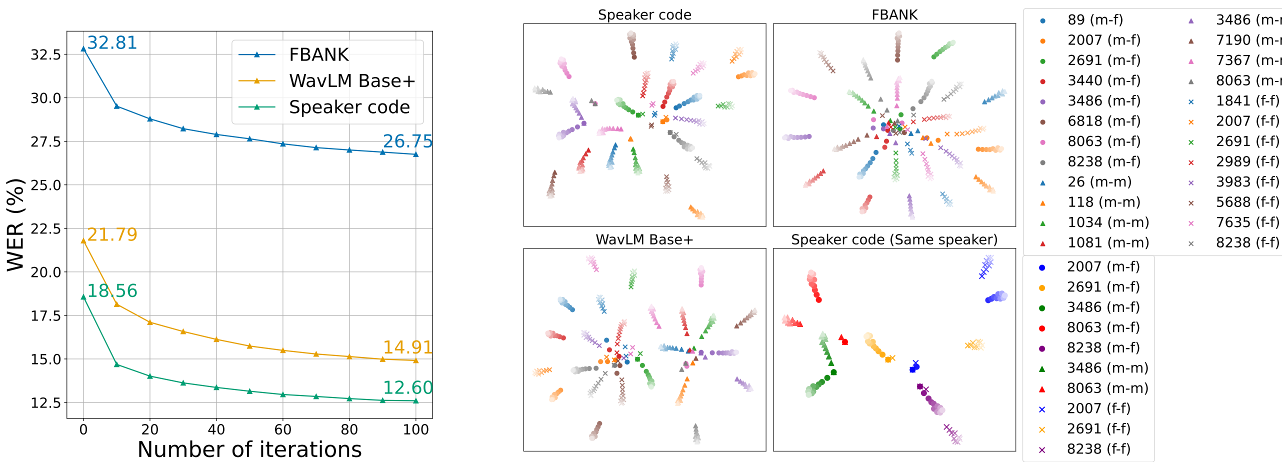
## 4.3. Embedding of gradient-based optimization

The speaker code is one of the optimal auxiliary networks, as described in Section 4.1. To explore further improvement, we apply gradient-based optimization to the speaker embedding. Figure 4(a) shows the WERs for each iteration when only the speaker embedding is iteratively optimized to maximize the score of the true class. Note that since this experiment was conducted on the `clean` mixture, the performance at the 0-th iteration is the same as in Table 2. Performance improves with each iteration, achieving a relative WER reduction of 32.11% for the speaker code and 31.57% for WavLM




(a) Speaker-open evaluation set (b) Speaker-closed evaluation set

**Fig. 3**: Visualization results of speaker embedding. Each color represents each speaker. Blue square and red cross markers indicate male and female speakers, respectively. DINO and ECAPA-TDNN denote ECAPA-TDNN-DINO and ECAPA-TDNN-c1024 models.




(a) Evaluation performances (b) Visualization

**Fig. 4**: Gradient-based optimization results. (a)Evaluation results using optimized embeddings on TS-ASR at each number of iterations. (b)Visualization results of speaker embeddings at each iteration for each speaker on the TS-ASR task. Each color represents each speaker. The colors are depicted to fade from darker to lighter as the number of iterations increases. Different markers represent different mixture conditions (`m-f`, `m-m`, and `f-f` indicating mixtures of male and female, male and male, and female and female, respectively).

BASE+. This suggests that *refining speaker representation can significantly enhance performance further, even for the speaker code.*

Figure 4(b) visualizes the embedding at iterations from 0 to 100 in increments of 10. Each panel represents an auxiliary network (speaker code, FBANK, and WavLM BASE+). To highlight the representation of the same speaker in two different mixtures, the bottom right panel also visualizes the embeddings of the speaker code for the selected speakers. *The optimized embedding generally tends to increase the inter-speaker difference.* Interestingly, *the optimal speaker embedding seems to vary for different mixtures, even for the same target speaker.* Therefore, future work should develop a system architecture that adjusts speaker embedding according to mixtures. Note that although we fix all network parameters and only optimize the embedding to try to capture the speaker representation as much as possible, the optimal embeddings may reflect both speaker and content information since they are optimized against the score of the true token class.

## 5. CONCLUSION

This paper investigates the speaker representation for TS tasks, i.e., TS-ASR, TSE, and p-VAD, through performance comparisons and visualization of embeddings. Our results suggest that SSL models with Transformer-based and ECAPA-TDNN-based architectures are more effective for all three TS tasks than supervised speaker models which demonstrate the highest performance in the ASV task. While the speaker code-based auxiliary network outperforms other enrollment-based auxiliary networks, gradient-based embedding optimization demonstrates that there is room to further optimize speaker representation. We hope that these results will contribute to the development of more effective TS systems and speaker embeddings.

## 6. REFERENCES


[1] Marc Delcroix, Katerina Zmolikova, Keisuke Kinoshita, Atsunori Ogawa, and Tomohiro Nakatani, "Single channel target speaker extraction and recognition with speaker beam," in *ICASSP*, 2018.

[2] Naoyuki Kanda, Shota Horiguchi, Ryoichi Takashima, Yusuke Fujita, Kenji Nagamatsu, and Shinji Watanabe, "Auxiliary interference speaker loss for target-speaker speech recognition," in *Interspeech*, 2019.

[3] Takafumi Moriya, Hiroshi Sato, Tsubasa Ochiai, Marc Delcroix, Takanori Ashihara, Kohei Matsuura, Tomohiro Tanaka, Ryo Masumura, Atsunori Ogawa, and Taichi Asami, "Knowledge distillation for neural transducer-based target-speaker ASR: Exploiting parallel mixture/single-talker speech data," in *Interspeech*, 2023.

[4] Yang Zhang, Krishna C. Puvvada, Vitaly Lavrukhin, and Boris Ginsburg, "Conformer-based target-speaker automatic speech recognition for single-channel audio," in *ICASSP*, 2023.

[5] Hao Ma, Zhiyuan Peng, Mingjie Shao, Jing Li, and Ju Liu, "Extending Whisper with prompt tuning to target-speaker ASR," in *ICASSP*, 2024.

[6] Kateřina Žmolíková, Marc Delcroix, Keisuke Kinoshita, Tsubasa Ochiai, Tomohiro Nakatani, Lukáš Burget, and Jan Černocký, "SpeakerBeam: Speaker aware neural network for target speaker extraction in speech mixtures," *JSTSP*, 2019.

[7] Kateřina Žmolíková, Marc Delcroix, Keisuke Kinoshita, Takuya Higuchi, Atsunori Ogawa, and Tomohiro Nakatani, "Speaker-aware neural network based beamformer for speaker extraction in speech mixtures," in *Interspeech*, 2017.

[8] Marc Delcroix, Tsubasa Ochiai, Katerina Zmolikova, Keisuke Kinoshita, Naohiro Tawara, Tomohiro Nakatani, and Shoko Araki, "Improving speaker discrimination of target speech extraction with time-domain speakerbeam," in *ICASSP*, 2020.

[9] Xiaoyu Liu, Xu Li, and Joan Serrà, "Quantitative evidence on overlooked aspects of enrollment speaker embeddings for target speaker separation," in *ICASSP*, 2023.

[10] Junyi Peng, Marc Delcroix, Tsubasa Ochiai, Oldrich Plchot, Takanori Ashihara, Shoko Araki, and Jan Cernocky, "Probing self-supervised learning models with target speech extraction," in *ICASSP Self-supervision in Audio, Speech and Beyond (SASB) workshop*, 2024.

[11] Shaojin Ding, Quan Wang, Shuo-Yiin Chang, Li Wan, and Ignacio Lopez Moreno, "Personal VAD: Speaker-conditioned voice activity detection," in *Odyssey*, 2020.

[12] Ivan Medennikov, Maxim Korenevsky, Tatiana Prisyach, Yuri Khokhlov, Mariya Korenevskaya, Ivan Sorokin, Tatiana Timofeeva, Anton Mitrofanov, Andrei Andrusenko, Ivan Podluzhny, Aleksandr Laptev, and Aleksei Romanenko, "Target-speaker voice activity detection: A novel approach for multi-speaker diarization in a dinner party scenario," in *Interspeech*, 2020.

[13] Shaojin Ding, Rajeev Rikhye, Qiao Liang, Yanzhang He, Quan Wang, Arun Narayanan, Tom O'Malley, and Ian McGraw, "Personal VAD 2.0: Optimizing personal voice activity detection for on-device speech recognition," in *Interspeech*, 2022.

[14] Holger Severin Bovbjerg, Jesper Jensen, Jan Østergaard, and Zheng-Hua Tan, "Self-supervised pretraining for robust personalized voice activity detection in adverse conditions," in *ICASSP*, 2024.

[15] Ivan Medennikov, Maxim Korenevsky, Tatiana Prisyach, Yuri Khokhlov, Mariya Korenevskaya, Ivan Sorokin, Tatiana Timofeeva, Anton Mitrofanov, Andrei Andrusenko, Ivan Podluzhny, Aleksandr Laptev, and Aleksei Romanenko, "The STC system for the CHiME-6 challenge," in *CHiME*, 2020.

[16] Yuhao Liang, Mohan Shi, Fan Yu, Yangze Li, Shiliang Zhang, Zhihao Du, Qian Chen, Lei Xie, Yanmin Qian, Jian Wu, Zhuo Chen, Kong Aik Lee, Zhijie Yan, and Hui Bu, "The second multi-channel multi-party meeting transcription challenge (M2MeT 2.0): A benchmark for speaker-attributed ASR," in *ASRU*, 2023.

[17] Najim Dehak, Patrick J. Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet, "Front-end factor analysis for speaker verification," *TASLP*, 2011.

[18] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno, "Generalized end-to-end loss for speaker verification," in *ICASSP*, 2018.

[19] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *ICASSP*, 2018.

[20] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *ICLR workshop*, 2014.

[21] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson, "Understanding neural networks through deep visualization," in *ICML deep learning workshop*, 2015.

[22] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy, "Explaining and harnessing adversarial examples," in *ICLR*, 2015.

[23] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio, "Adversarial examples in the physical world," in *ICLR workshop*, 2016.

[24] Shu wen Yang, Po-Han Chi, Yung-Sung Chuang, Cheng-I Jeff Lai, Kushal Lakhotia, Yist Y. Lin, Andy T. Liu, Jiatong Shi, Xuankai Chang, Guan-Ting Lin, Tzu-Hsien Huang, Wei-Cheng Tseng, Ko tik Lee, Da-Rong Liu, Zili Huang, Shuyan Dong, Shang-Wen Li, Shinji Watanabe, Abdelrahman Mohamed, and Hung yi Lee, "SUPERB: Speech Processing Universal PERformance Benchmark," in *Interspeech*, 2021.

[25] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang, "Conformer: Convolution-augmented Transformer for speech recognition," in *Interspeech*, 2020.

[26] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.

[27] Hsiang-Sheng Tsai, Heng-Jui Chang, Wen-Chin Huang, Zili Huang, Kushal Lakhotia, Shu-wen Yang, Shuyan Dong, Andy Liu, Cheng-I Lai, Jiatong Shi, Xuankai Chang, Phil Hall, Hsuan-Jui Chen, Shang-Wen Li, Shinji Watanabe, Abdelrahman Mohamed, and Hung-yi Lee, "SUPERB-SG: Enhanced speech processing universal PERformance benchmark for semantic and generative capabilities," in *ACL*, 2022.

[28] Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R. Hershey, "SDR – half-baked or well done?," in *ICASSP*, 2019.

[29] Junyi Peng, Oldřich Plchot, Themos Stafylakis, Ladislav Mošner, Lukáš Burget, and Jan Černocký, "An attention-based backend allowing efficient fine-tuning of Transformer models for speaker verification," in *SLT*, 2022.

[30] Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck, "ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification," in *Interspeech*, 2020.

[31] Themos Stafylakis, Johan Rohdin, Oldřich Plchot, Petr Mizera, and Lukáš Burget, "Self-supervised speaker embeddings," in *Interspeech*, 2019.

[32] Wei Xia, Chunlei Zhang, Chao Weng, Meng Yu, and Dong Yu, "Self-supervised text-independent speaker verification using prototypical momentum contrastive learning," in *ICASSP*, 2021.

[33] Theo Lepage and Reda Dehak, "Label-efficient self-supervised speaker verification with information maximization and contrastive learning," in *Interspeech*, 2022.

[34] Jaejin Cho, Raghavendra Pappagari, Piotr Żelasko, Laureano Moro Velazquez, Jesus Villalba, and Najim Dehak, "Non-contrastive self-supervised learning of utterance-level speech representations," in *Interspeech*, 2022.

[35] Chunlei Zhang and Dong Yu, "C3-DINO: Joint contrastive and non-contrastive self-supervised learning for speaker verification," *JSTSP*, 2022.

[36] Zhengyang Chen, Yao Qian, Bing Han, Yanmin Qian, and Michael Zeng, "A comprehensive study on self-supervised distillation for speaker representation learning," in *SLT*, 2023.

[37] Bing Han, Wen Huang, Zhengyang Chen, and Yanmin Qian, "Improving DINO-based self-supervised speaker verification with progressive cluster-aware training," in *ICASSP Workshops*, 2023.

[38] Yafeng Chen, Siqi Zheng, Hui Wang, Luyao Cheng, and Qian Chen, "Pushing the limits of self-supervised speaker verification using regularized distillation framework," in *ICASSP*, 2023.

[39] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, Xiangzhan Yu, and Furu Wei, "WavLM: Large-scale self-supervised pre-training for full stack speech processing," *JSTSP*, 2022.

[40] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *NeurIPS*, 2020.

[41] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed, "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *TASLP*, 2021.

[42] Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli, "data2vec: A general framework for self-supervised learning in speech, vision and language," in *ICML*, 2022.

[43] Hongji Wang, Chengdong Liang, Shuai Wang, Zhengyang Chen, Binbin Zhang, Xu Xiang, Yanlei Deng, and Yanmin Qian, "WeSpeaker: A research and production oriented speaker embedding learning toolkit," in *ICASSP*, 2023.

[44] Jaesung Huh, Andrew Brown, Jee-weon Jung, Joon Son Chung, Arsha Nagrani, Daniel Garcia-Romero, and Andrew Zisserman, "VoxSRC 2022: The fourth VoxCeleb speaker recognition challenge," *arXiv preprint arXiv:2302.10248*, 2023.

[45] Arsha Nagrani, Joon Son Chung, Weidi Xie, and Andrew Zisserman, "VoxCeleb: Large-scale speaker verification in the wild," *Computer Speech & Language*, 2019.

[46] Takanori Ashihara, Marc Delcroix, Takafumi Moriya, Kohei Matsuura, Taichi Asami, and Yusuke Ijima, "What do self-supervised speech and speaker models learn? New findings from a cross model layer-wise analysis," in *ICASSP*, 2024.

[47] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin, "Emerging properties in self-supervised vision transformers," in *ICCV*, 2021.

[48] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "LibriSpeech: An ASR corpus based on public domain audio books," in *ICASSP*, 2015.

[49] Jacob Kahn, Morgane Riviere, Weiyi Zheng, Evgeny Kharitonov, Qiantong Xu, Pierre-Emmanuel Mazaré, Julien Karadayi, Vitaliy Liptchinsky, Ronan Collobert, Christian Fuegen, Tatiana Likhomanenko, Synnaeve Gabriel, Joulin Armand, Mohamed Abdelrahman, and Emmanuel Dupoux, "Libri-Light: A benchmark for ASR with limited or no supervision," in *ICASSP*, 2020.

[50] Guoguo Chen, Shuzhou Chai, Guan-Bo Wang, Jiayu Du, Wei-Qiang Zhang, Chao Weng, Dan Su, Daniel Povey, Jan Trmal, Junbo Zhang, Mingjie Jin, Sanjeev Khudanpur, Shinji Watanabe, Shuaijiang Zhao, Wei Zou, Xiangang Li, Xuchen Yao, Yongqing Wang, Zhao You, and Zhiyong Yan, "GigaSpeech: An evolving, multi-domain ASR corpus with 10,000 hours of transcribed audio," in *Interspeech*, 2021.

[51] Changhan Wang, Morgane Riviere, Ann Lee, Anne Wu, Chaitanya Talnikar, Daniel Haziza, Mary Williamson, Juan Pino, and Emmanuel Dupoux, "VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation," in *IJCNLP*, 2021.

[52] Joris Cosentino, Manuel Pariente, Samuele Cornell, Antoine Deleforge, and Emmanuel Vincent, "LibriMix: An open-source dataset for generalizable speech separation," *arXiv preprint arXiv:2005.11262*, 2020.

[53] Gordon Wichern, Joe Antognini, Michael Flynn, Licheng Richard Zhu, Emmett McQuinn, Dwight Crow, Ethan Manilow, and Jonathan Le Roux, "WHAM!: Extending speech separation to noisy environments," in *Interspeech*, 2019.

[54] Cees H. Taal, Richard C. Hendriks, Richard Heusdens, and Jesper Jensen, "A short-time objective intelligibility measure for time-frequency weighted noisy speech," in *ICASSP*, 2010.

[55] Antony W. Rix, John G. Beerends, Michael P. Hollier, and Andries P. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in *ICASSP*, 2001.

[56] Laurens van der Maaten and Geoffrey Hinton, "Visualizing data using t-SNE," *JMLR*, 2008.