Automatic Data Augmentation Selection and Parametrization in Contrastive Self-Supervised Speech Representation Learning

- Select a distribution on the choice of augmentations and their parametrization according to the downstream task of interest

the probability of applying an augmentation or a boundary for a uniform law from which a augmentation's internal parameter

$$\boxed{\tau^*} = \arg\min_{\tau} HSIC(f(X,\tau), Z|Y) \quad (1)$$

with (X, Y) the downstream datapoints and labels, and Z the pretext labels corresponding here for every augmented view of a speech sample to the ID of the speech sample it originates from.

generate N augmented segments per speech sample

$$\mathcal{L} = -\log \frac{e^{s(\tilde{x}_i, \tilde{x}'_i)}}{e^{s(\tilde{x}_i, \tilde{x}'_i)} + \sum_{j \neq i} e^{s(\tilde{x}_i, \tilde{x}'_j)}}.$$

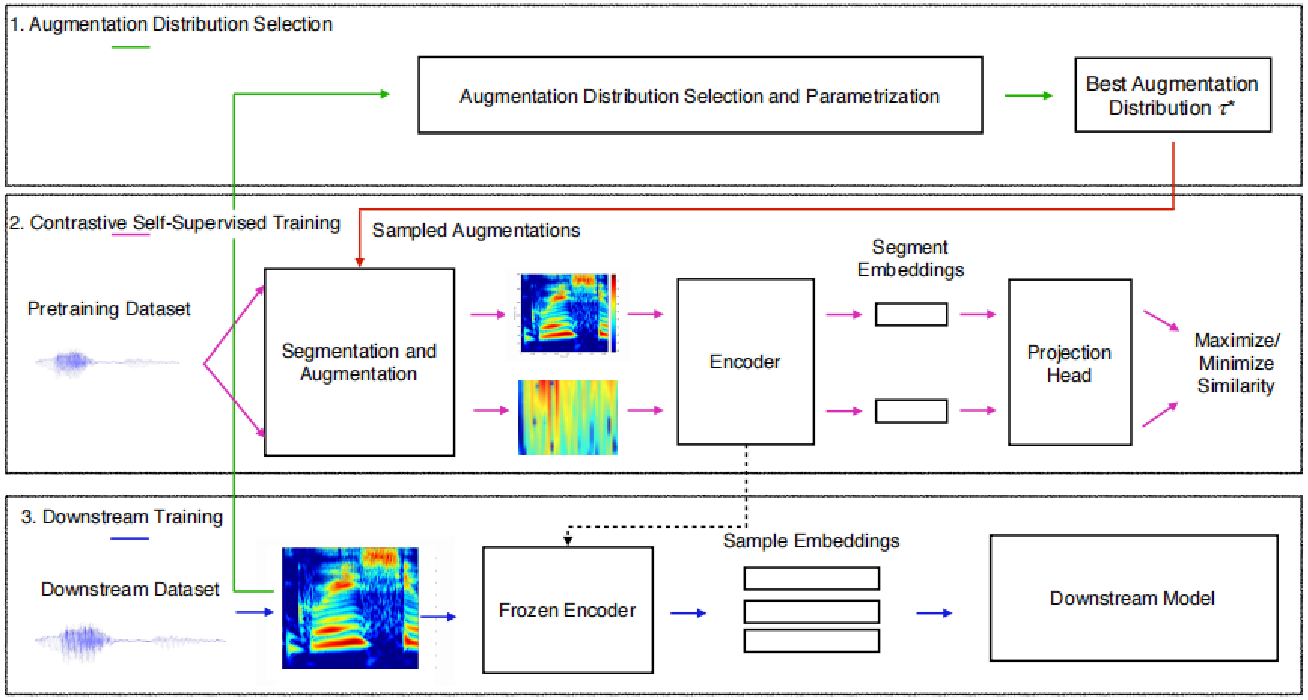


Figure 1: *The three steps of the validation process. (a) select the best augmentation distribution. (b) contrastive pretraining alterating the input points with the selected augmentation. (c) use the learned speech representations as input for downstream finetuning.*

Figure 2: *Difference of the probability of picking an augmentation between the best and worst scoring augmentations, depending on the downstream dataset. Green bars show augmentations that are more likely to get picked for the best scoring distributions for that task. For instance, the far right bars indicate that clipping is an encouraged augmentation on VoxForge, and is discouraged on VoxCeleb1.*

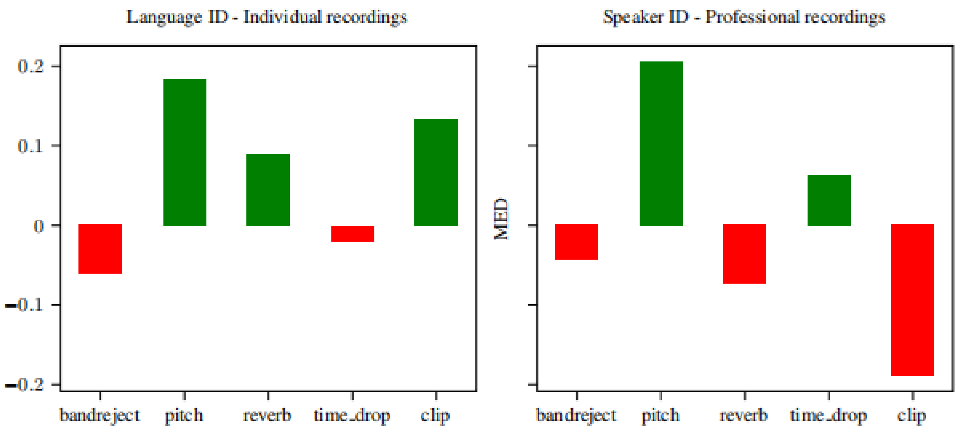
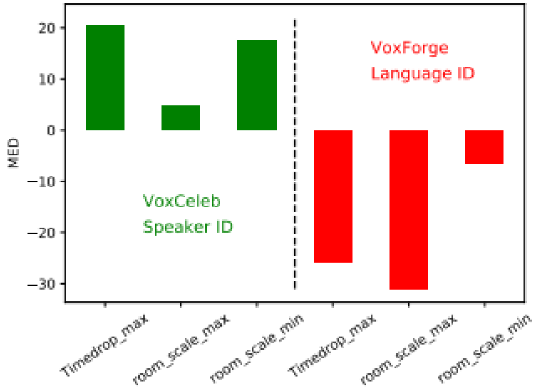


Table 1: *Parameters considered, descriptions and ranges*

Name	**Description**	**Range**
Room scale min	Min room size	[0,30]
Room scale max	Max room size	[30,100]
Band Scaler	Scales the rejected band	[0,1]
Pitch Shift Max	Amplitude of a pitch shift	[150,450]
Pitch Quick pr.	Speeds pitch shifting	[0,1]
Clip Min	Minimal clip factor	[0.3, 0.6]
Clip Max	Maximal clip factor	[0.6, 1]
Timedrop max	Size of a time dropout	[30-150] ms

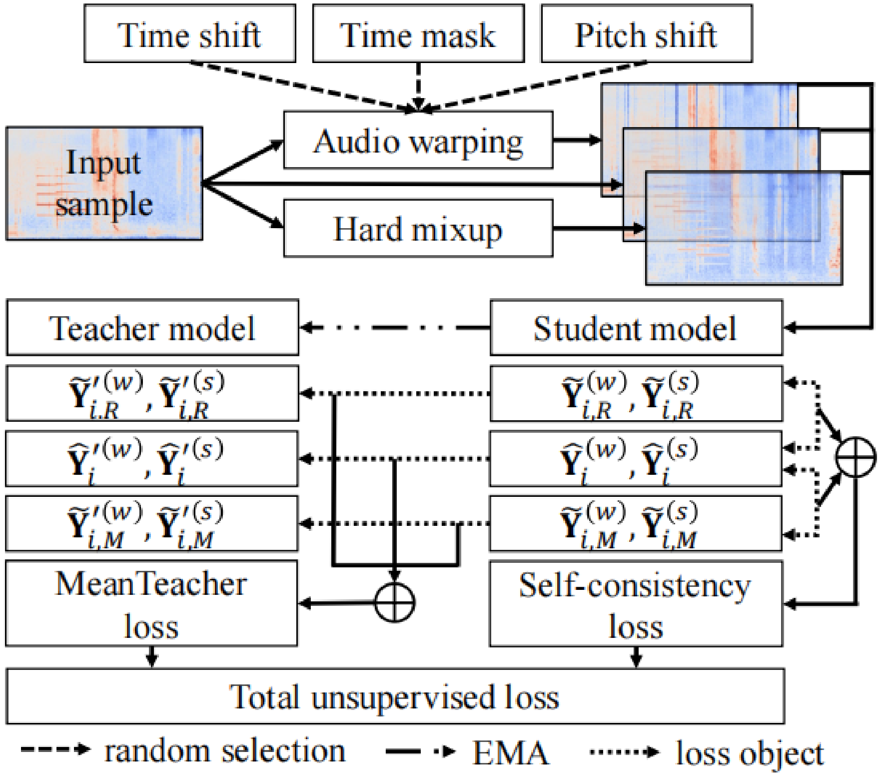


$$MED(p) = \frac{1}{k}\left(\sum_{i=0}^{k} \tau_i^{best}(p) - \tau_i^{worst}(p)\right)$$

Down. Task	**COLA**	**Our Implementations**		
		Without	Basic	Selected
Language ID	71.3	84.9	84.3	**85.2**
Speaker ID	29.9	32.0	45.1	**46.9**

RCT: RANDOM CONSISTENCY TRAINING FOR SEMI-SUPERVISED SOUND EVENT DETECTION

- a novel semi-supervised learning (SSL) strategy, for sound event detection (SED) task



- Hard mixup

  add multiple samples together, and the mixture is labelled with all the classes in all original samples

- RandomWarping

  Time shift, Time mask, Pitch shift

- Self-consistency training

$$\mathcal{D}^{(l)}_{\text{mixup}}(\hat{\mathbf{Y}}^{(l)}_i) = \vee_{i\in\mathcal{M}}\, \text{harden}(\hat{\mathbf{Y}}^{(l)}_i),$$

$$\mathcal{L}_{\text{SC}} = r(step)\,\frac{1}{N^{(w)}C}\sum_{i}^{N^{(w)}} \|\mathcal{D}^{(w)}_{\text{aug}}(\hat{\mathbf{Y}}^{(w)}_i) - \tilde{\mathbf{Y}}^{(w)}_i\|_2^2 + r(step)\,\frac{1}{N^{(s)}CT'}\sum_{i}^{N^{(s)}} \|\mathcal{D}^{(s)}_{\text{aug}}(\hat{\mathbf{Y}}^{(s)}_i) - \tilde{\mathbf{Y}}^{(s)}_i\|_2^2,$$

^: original ~: augmented

$$\mathcal{L} = \mathcal{L}_{\text{Supervised}} + \underbrace{\mathcal{L}_{\text{MeanTeacher}}}_{\text{unlabelled}} + \mathcal{L}_{\text{SC}}$$

l ∈ {w,s,u} weakly labelled, strongly labelled and unlabelled

C and Y_i(l) the number of sound event classes and data labels

$$\mathbf{Y}^{(w)}_i \in \mathbb{R}^{C} \text{ and } \mathbf{Y}^{(s)}_i \in \mathbb{R}^{T'\times C}$$

Table 1: *Ablation study for RCT. Different modules are added step by step and each score is obtained by averaging three trials.*

Model	**PSDS$_1$ (%)**	**PSDS$_2$ (%)**
Baseline	34.7	53.7
+ Vanilla mixup [5]	34.9	57.9
+ Hard mixup	36.4	57.4
+ RandomWarping	38.1	58.5
+ ICT consistency [7]	38.0	59.2
+ Self-consistency	40.1	61.4

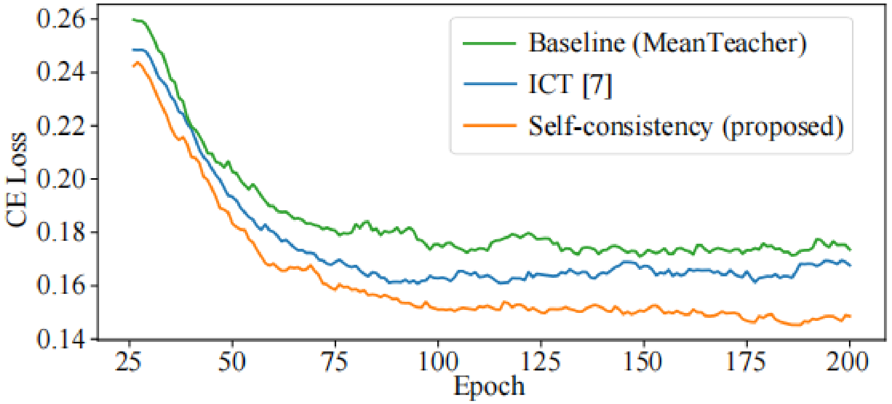


Figure 3: *Cross-entropy loss of strongly-supervised validation data when training with or without self-consistency loss, comparing with the ICT scheme [7].*

Table 2: *Comparing the proposed SSL strategy with other alternatives. Each score is obtained by averaging three trials.*

Model	**PSDS$_1$ (%)**	**PSDS$_2$ (%)**
Baseline [12]	34.7	53.7
SCT [16]	36.0	55.6
ICT [7]	37.7	57.7
ICT+SCT [16]	37.0	58.7
RCT (proposed)	40.1	61.4

Table 3: *Comparing the proposed system with DCASE2021 top-ranked submissions. All models are named in the form of network architecture plus the SSL strategy.*

Model	**PSDS$_1$ (%)**	**PSDS$_2$ (%)**
CRNN (baseline) [12]	34.7	53.7
FBCRNN+MLFL [20]	40.1	59.7
CRNN+IPL [15]	40.7	65.3
CRNN+DA [21]	41.9	63.8
CRNN+HeavyAug. [14]	43.4	63.9
RCRNN+NS [19]	45.1	67.9
SKUnit+ICT/SCT [5]	45.4	67.1
CRNN+**RCT** (proposed)	44.0	67.1

Deep versus Wide: An Analysis of Student Architectures for Task-Agnostic Knowledge Distillation of Self-Supervised Speech Models

- how varying the depth and width impacts the internal representation of the small-footprint model.

- task-agnostic distillation
- apply two simple KD approaches: prediction layer distillation and layer-to-layer (L2L) distillation methods.
- For student models, alter the depth and width of self-attention layers only while fixing the size of the CNNs.

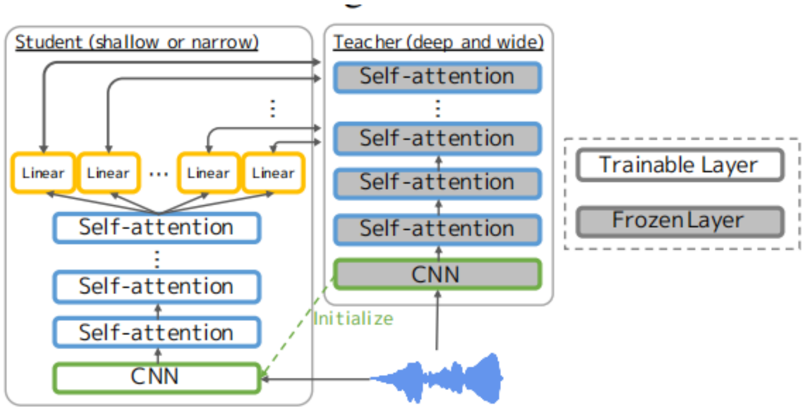


Figure 1: *Illustration of student model trained by KD between student's last and teacher's intermediate layers based on DistilHuBERT [16].*

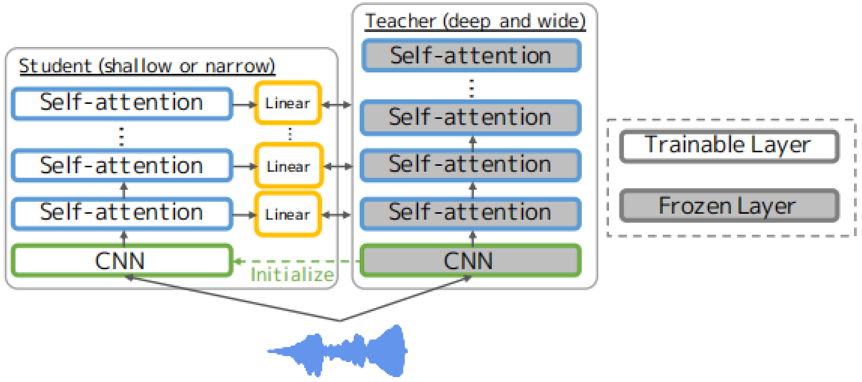


Figure 2: *Illustration of student model trained by KD between intermediate-layers such as FitNets [19].*

Table 1: *Evaluation result for each model distilled from HuBERT* BASE *and each task on SUPERB. The values in the first and second row are taken from [12] and [16], respectively. Pred. means the predication-layer distillation and L2L indicates the layer-to-layer distillation in the second column. For clarity, the KD models are indexed from (a) to (h) as shown in the second column.*

Model	KD Loss	PR	ASR (w/ LM)	KS	QbE	SID	ASV	SD	IC	SF	ER	Rank↓
		PER↓	WER↓	Acc↑	MTWV↑	Acc↑	EER↓	DER↓	Acc↑	F1↑ / CER↓	Acc↑	
HuBERT BASE	-	5.41	6.42 (4.79)	96.30	0.0736	81.42	5.11	5.88	98.34	88.53 / 25.20	64.92	1.7
DistilHuBERT	Pred.	16.27	13.34 (9.21)	95.98	0.0511	73.54	8.55	6.19	94.99	82.57 / 35.59	63.02	5.8
12-L HALF	(a) Pred.	13.09	11.87 (8.07)	96.97	0.0501	69.11	6.32	6.67	94.91	84.49 / 32.54	62.76	4.6
	(b) L2L	10.67	10.96 (7.68)	97.24	0.0604	69.52	6.13	6.81	96.97	86.11 / 30.93	63.24	2.6
12-L FOURTH	(c) Pred.	18.92	14.02 (9.25)	96.44	0.0495	49.51	6.74	7.12	87.03	81.21 / 37.27	62.82	8.1
	(d) L2L	16.96	13.84 (9.20)	96.40	0.0562	47.67	6.41	7.12	91.62	84.81 / 32.77	61.84	7.0
3-L ONE	(e) Pred.	13.34	12.23 (8.64)	96.69	0.0489	75.71	6.48	6.56	94.15	82.89 / 34.65	63.95	4.6
	(f) L2L	13.96	12.94 (9.11)	96.52	0.0568	47.76	6.18	7.17	96.02	85.99 / 32.38	62.57	5.2
3-L HALF	(g) Pred.	18.62	13.91 (9.27)	96.22	0.0482	62.59	6.86	6.69	91.88	82.78 / 35.75	61.83	8.1
	(h) L2L	18.11	14.48 (9.86)	96.48	0.0502	60.40	6.82	7.31	94.91	81.82 / 37.36	62.78	7.5

Table 2: *Model settings of teacher and student models. With respect to self-attention blocks,* HALF *and* FOURTH *means the parameter reductions to one half and one fourth, respectively, and* ONE *is the same as the HuBERT* BASE.

Models	#Params	#Layers	Embed.	FFN	#Head
HuBERT BASE [9]	94.68M	12	768	3072	12
HuBERT LARGE [9]	316.61M	24	1024	4096	16
DistilHuBERT [16]	23.49M	2	768	3072	12
12-L HALF	26.87M	12	384	1536	6
12-L FOURTH	9.93M	12	192	768	3
3-L ONE	30.58M	3	768	3072	12
3-L HALF	10.90M	3	384	1536	6
6-L HALF	16.23M	6	384	1536	6

prediction-layer loss is suitable for wider architectures such as (e) ,
whereas L2L loss is effective for deeper architectures such as (b) and (d).

deeper networks have higher performance in content-oriented tasks such as PR, ASR and QbE,
wider networks have higher performance in speaker-oriented tasks such as SID and SD.

Table 3: *Evaluation result for each model distilled from HuBERT LARGE. The values of first row are taken from [12]. The values shown from the second row are the results of the KD models trained in our experiment.*

Model	KD Loss	PR PER↓	ASR (w/ LM) WER↓	KS Acc↑	QbE MTWV↑	SID Acc↑	ASV EER↓	SD DER↓	IC Acc↑	SF F1↑ / CER↓	ER Acc↑	Rank↓
HuBERT LARGE	-	3.53	3.62 (2.94)	95.29	0.0353	90.33	5.98	5.75	98.76	89.81 / 21.76	67.62	2.7
12-L HALF	Pred.	9.67	9.59 (6.84)	95.79	0.0507	49.25	5.84	6.20	95.07	84.88 / 31.17	63.59	4.2
	L2L	7.97	9.24 (6.82)	96.24	0.0513	52.42	6.36	6.60	96.92	87.26 / 28.92	64.51	3.2
12-L FOURTH	Pred.	14.10	12.49(8.47)	96.20	0.0482	37.18	6.86	6.93	91.91	83.66/35.11	62.45	6.8
	L2L	12.86	12.91(9.11)	95.34	0.0443	47.51	7.26	7.05	92.86	83.83/34.22	62.20	7.4
3-L ONE	Pred.	12.11	11.35 (8.00)	96.50	0.0474	76.97	7.22	6.61	96.63	85.36 / 31.60	65.80	3.7
	L2L	10.24	12.23 (8.78)	96.40	0.0540	68.90	7.59	7.33	96.97	84.56 / 32.88	65.22	4.3
3-L HALF	Pred.	15.78	13.28(9.34)	96.20	0.0430	60.17	7.17	6.77	94.02	84.67 / 33.82	64.55	5.9
	L2L	15.11	14.31 (9.84)	96.01	0.0532	55.35	7.47	7.81	92.72	84.04 / 34.33	63.40	6.9

Table 1: *Evaluation result for each model distilled from HuBERT BASE and each task on SUPERB. The values in the first and second row are taken from [12] and [16], respectively. Pred. means the predication-layer distillation and L2L indicates the layer-to-layer distillation in the second column. For clarity, the KD models are indexed from (a) to (h) as shown in the second column.*

Model	KD Loss	PR PER↓	ASR (w/ LM) WER↓	KS Acc↑	QbE MTWV↑	SID Acc↑	ASV EER↓	SD DER↓	IC Acc↑	SF F1↑ / CER↓	ER Acc↑	Rank↓
HuBERT BASE	-	5.41	6.42 (4.79)	96.30	0.0736	81.42	5.11	5.88	98.34	88.53 / 25.20	64.92	1.7
DistilHuBERT	Pred.	16.27	13.34 (9.21)	95.98	0.0511	73.54	8.55	6.19	94.99	82.57 / 35.59	63.02	5.8
12-L HALF	(a) Pred.	13.09	11.87 (8.07)	96.97	0.0501	69.11	6.32	6.67	94.91	84.49 / 32.54	62.76	4.6
	(b) L2L	10.67	10.96 (7.68)	97.24	0.0604	69.52	6.13	6.81	96.97	86.11 / 30.93	63.24	2.6
12-L FOURTH	(c) Pred.	18.92	14.02 (9.25)	96.44	0.0495	49.51	6.74	7.12	87.03	81.21 / 37.27	62.82	8.1
	(d) L2L	16.96	13.84 (9.20)	96.40	0.0562	47.67	6.41	7.12	91.62	84.81 / 32.77	61.84	7.0
3-L ONE	(e) Pred.	13.34	12.23 (8.64)	96.69	0.0489	75.71	6.48	6.56	94.15	82.89 / 34.65	63.95	4.6
	(f) L2L	13.96	12.94 (9.11)	96.52	0.0568	47.76	6.18	7.17	96.02	85.99 / 32.38	62.57	5.2
3-L HALF	(g) Pred.	18.62	13.91 (9.27)	96.22	0.0482	62.59	6.86	6.69	91.88	82.78 / 35.75	61.83	8.1
	(h) L2L	18.11	14.48 (9.86)	96.48	0.0502	60.40	6.82	7.31	94.91	81.82 / 37.36	62.78	7.5

Table 2: *Model settings of teacher and student models. With respect to self-attention blocks, HALF and FOURTH means the parameter reductions to one half and one fourth, respectively, and ONE is the same as the HuBERT BASE.*

Models	#Params	#Layers	Embed.	FFN	#Head
HuBERT BASE [9]	94.68M	12	768	3072	12
HuBERT LARGE [9]	316.61M	24	1024	4096	16
DistilHuBERT [16]	23.49M	2	768	3072	12
12-L HALF	26.87M	12	384	1536	6
12-L FOURTH	9.93M	12	192	768	3
3-L ONE	30.58M	3	768	3072	12
3-L HALF	10.90M	3	384	1536	6
6-L HALF	16.23M	6	384	1536	6

students distilled from HuBERT LARGE show better performance on PR, ASR and SF tasks in particular.

Table 4: *Evaluation result for each model distilled from HuBERT BASE. The values in the fifth row represent the model trained by the linear interpolation loss (Comb.) between the prediction-layer and L2L losses.*

Model	KD Loss	PR PER↓	ASR (w/ LM) WER↓	KS Acc↑	QbE MTWV↑	SID Acc↑	ASV EER↓	SD DER↓	IC Acc↑	SF F1↑ / CER↓	ER Acc↑	Rank↓
HuBERT BASE	-	5.41	6.42 (4.79)	96.30	0.0736	81.42	5.11	5.88	98.34	88.53 / 25.20	64.92	1.3
DistilHuBERT	Pred.	16.27	13.34 (9.21)	95.98	0.0511	73.54	8.55	6.19	94.99	82.57 / 35.59	63.02	4.1
6-L HALF	Pred.	15.14	12.72 (8.68)	96.85	0.0504	67.06	6.36	6.81	93.75	83.65 / 34.35	63.72	3.4
	L2L	13.40	12.66 (8.59)	96.38	0.0545	62.90	6.85	6.95	95.86	83.80 / 33.51	63.09	3.3
	Comb.	14.68	12.43 (8.51)	96.77	0.0516	65.75	6.81	6.83	94.57	84.32 / 33.99	64.78	2.9

Pushing the limits of raw waveform speaker recognition

- propose a new raw waveform speaker recognition architecture, namely RawNet3

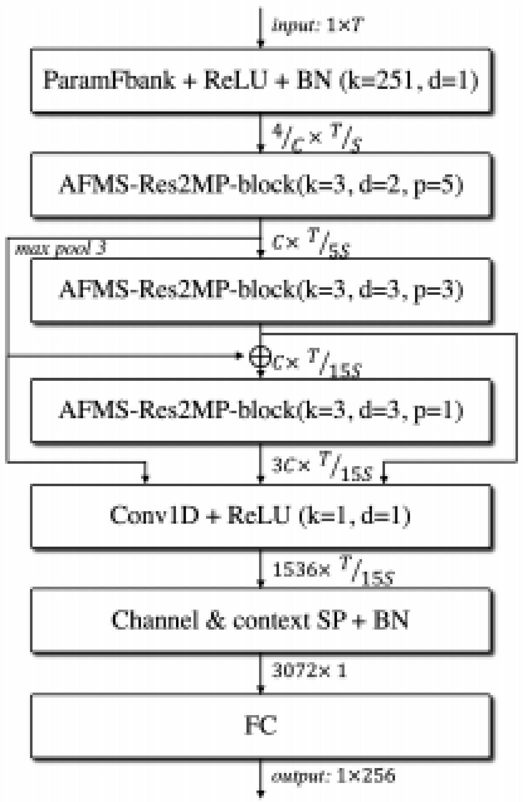


Figure 1: *The RawNet3 architecture. It is in a hybrid form of the ECAPA-TDNN [2] and the RawNet2 [32] with additional features including logarithm and normalisation. k, d, p, C, S, and ⊕ correspond to kernel length, dilation, max pooling size, number of channels, stride size of the parameterised filterbank layer, and element-wise addition.*

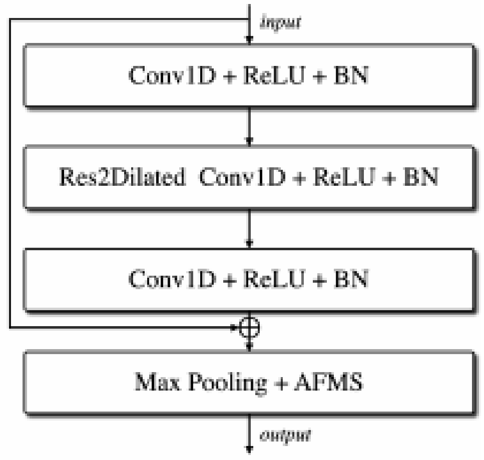


Figure 2: *The AFMS-Res2MP-block of the RawNet3 architecture. AFMS refers to the extended feature map scaling module of RawNet2.*

Pushing the limits of raw waveform speaker recognition

Table 1: *Results on supervised learning using the AAM-softmax [41] objective function. Trained on VoxCeleb1&2 development sets. The two numbers in* Hz *denote frame resolutions after the first parameterised filterbank and the last max pooling layer.*

Configurations	**EER(%)**	**minDCF**
RawNet2 [32]	2.48	N/R
RawNet3 (stride=48)	1.05	0.0763
− param fbank log	1.27	0.0852
− param fbank norm	1.22	0.0838
− param fbank log&norm	1.23	0.0927
− ch&context stat pool	1.45	0.0975
→ stride=10, 1600Hz→106Hz	**0.89**	0.0669
→ stride=16, 1000Hz→66Hz	0.90	**0.0593**
→ stride=24, 666Hz→44Hz	0.96	0.0773
→ stride=64, 250Hz→16Hz	1.11	0.0851
→ stride=96, 166Hz→11Hz	1.31	0.0937

Table 4: *Comparison with recent literature of supervised speaker verification.* †*: calculated with* $P_{target} = 0.01$.

	In Feat	**EER(%)**	**minDCF**
Desplanques et al. [2]	MFCC	0.87	0.1066†
Ravanelli et al. [17]	Fbank	0.69	N/R
Kuzmin et al. [18]	Fbank	**0.66**	**0.0640**†
Zhu et al. [12]	Waveform	2.60	0.2390
Li et al. [14]	Waveform	2.31	N/R
Lin et al. [15]	Waveform	1.95	0.2030
Kim et al. [16]	Waveform	1.29	0.1420
Ours−stride=10	Waveform	**0.89**	0.0659
Ours−stride=16	Waveform	0.90	**0.0593**

Table 2: *Results on self-supervised learning using the DINO [25] framework. Trained on VoxCeleb2 development set.*

Configurations	**EER(%)**	**minDCF**
RawNet3	**5.74**	**0.3507**
− param fbank log	10.46	0.5775
− param fbank mean norm	8.87	0.4969
− param fbank log&mean norm	9.98	0.5386
+ DINO temp warm-up	5.89	0.4004
+ DINO last layer norm	**5.40**	**0.3396**
→ DINO T momentum 0.99	6.17	0.3987
→ half batch size (400→200)	6.87	0.4513

Table 3: *Results on fine-tuning the pre-trained model. Trained on VoxCeleb1 development set.*

Configurations	**EER(%)**	**minDCF**
RawNet3 (w/ pre-train)	**2.18**	**0.1519**
RawNet3 (w/o pre-train)	2.98	0.2268

Table 5: *Comparison with self-supervised learning models.*

	Framework	**EER(%)**	**minDCF**
Huh et al. [27]	AP+AAT	8.65	0.4540
Xia et al. [28]	MOCO+Wav-Aug(ProtoNCE)	8.23	0.5900
Mun et al. [29]	CEL	8.01	N/R
Tao et al. [30]	Contrastive	7.36	N/R
Sang et al. [31]	SSReg	6.99	0.4340
Ours	DINO	**5.40**	**0.3396**

Speech Sequence Embeddings using Nearest Neighbors Contrastive Learning

query-by-example spoken term discovery

Voice Activity Detection
Time stretch

$$L_{nce}(z_i, P^+) = -log(\frac{\exp(sim(z_i, z_i^+)/\tau)}{\sum_{j \leq 2n j \neq i} \exp(sim(z_i, z_j)/\tau)})$$

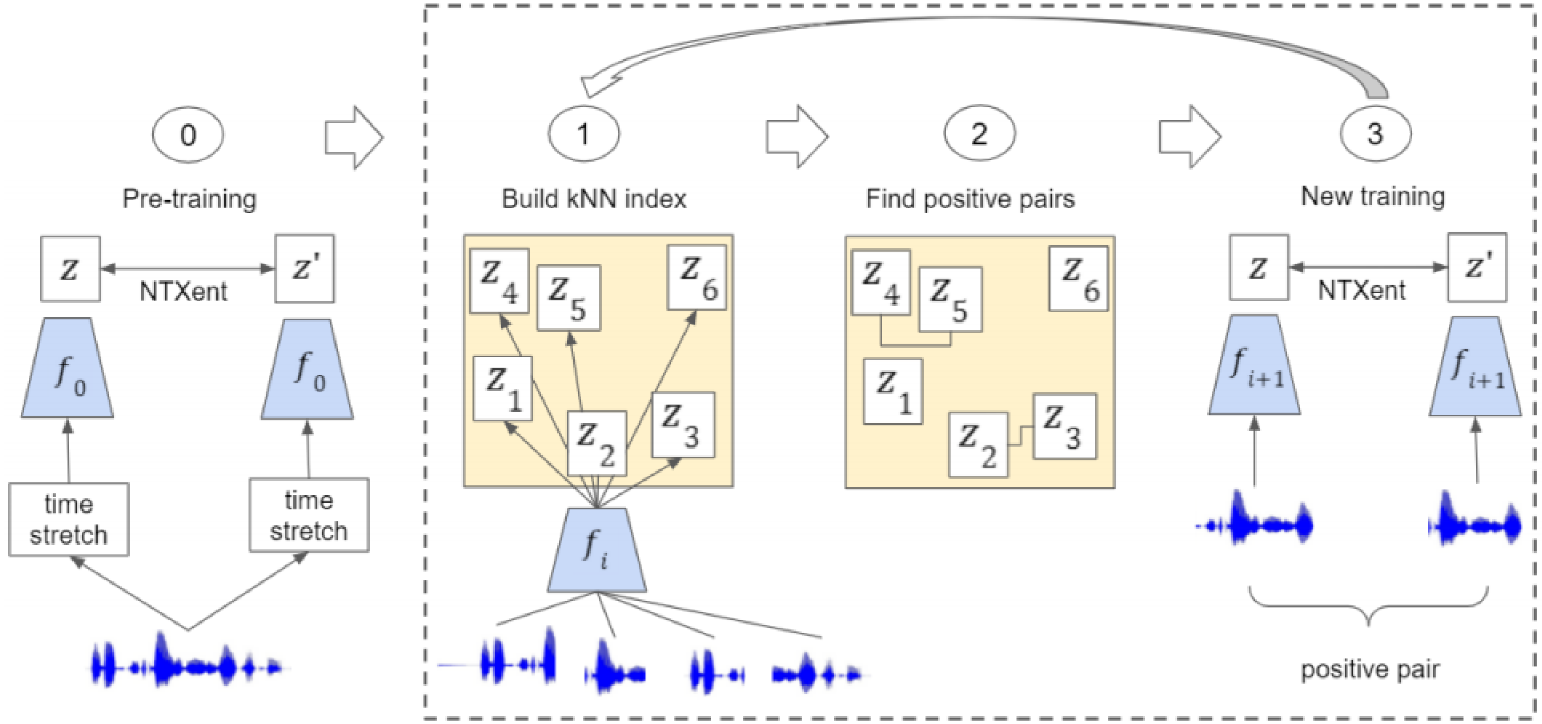


Figure 1: *The main steps of our method. 0) Train a model f_0 with the NTXent loss on time-stretched pairs of sequences of speech. 1) Use f_0 to encode many random sequences of speech from the corpus. 2) List close embeddings in the kNN as positive pairs. 3) Use the speech sequences associated to the positive pairs to train a new model f_1. Go back to step 1 using f_1 instead of f_0 and iterate.*

- all neighbors that are temporally overlapping with s are removed.
- when two neighbors are temporally overlapping with each other, keep only the one that has the smallest cosine distance with s
- apply a distance threshold above which all pairs are discarded.

retraining another SSE model

Speech Sequence Embeddings using Nearest Neighbors Contrastive Learning

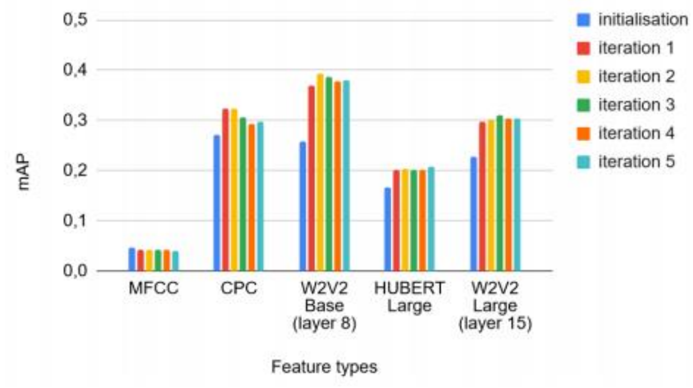


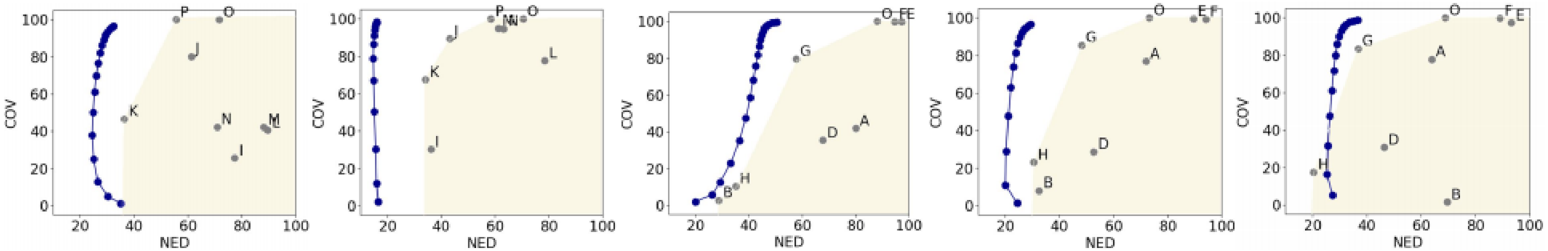
Figure 3: *Phoneme-ngrams MAP computed on LibriSpeech dev-clean for different iteration of our model and different input feature types*

Supervision	Models	dev-clean	dev-other	test-clean	test-other
unsup.	Max-pooling	0,07	0,048	0,07	0,047
self-sup.	CAE-Siamese	0,21	0,154	0,212	0,151
self-sup.	Ours (iter. 2)	**0,398**	**0,307**	**0,399**	**0,305**
weakly-sup.	Topline	0,789	0,647	0,784	0,648

Table 1: *Phoneme-ngrams MAP computed on LibriSpeech held-out sets for different SSE models. All models take as input features the Wav2vec2.0 Base at layer 8*

Models	buckeye	xitsonga	mandarin	french	english	*average*
Max-pooling	0,05	0,053	0,075	0,039	0,052	0,054
CAE-Siamese	0,16	0,23	0,26	0,2	0,19	0,208
Ours (iter. 2)	**0,235**	**0,362**	**0,277**	**0,283**	**0,346**	**0,301**
Topline	0,751	0,948	0,822	0,71	0,857	0,818

Table 2: *Phoneme-ngrams MAP computed on Zerospeech corpora for different SSE models. All models take as input features the Wav2vec2.0 Base at layer 8*

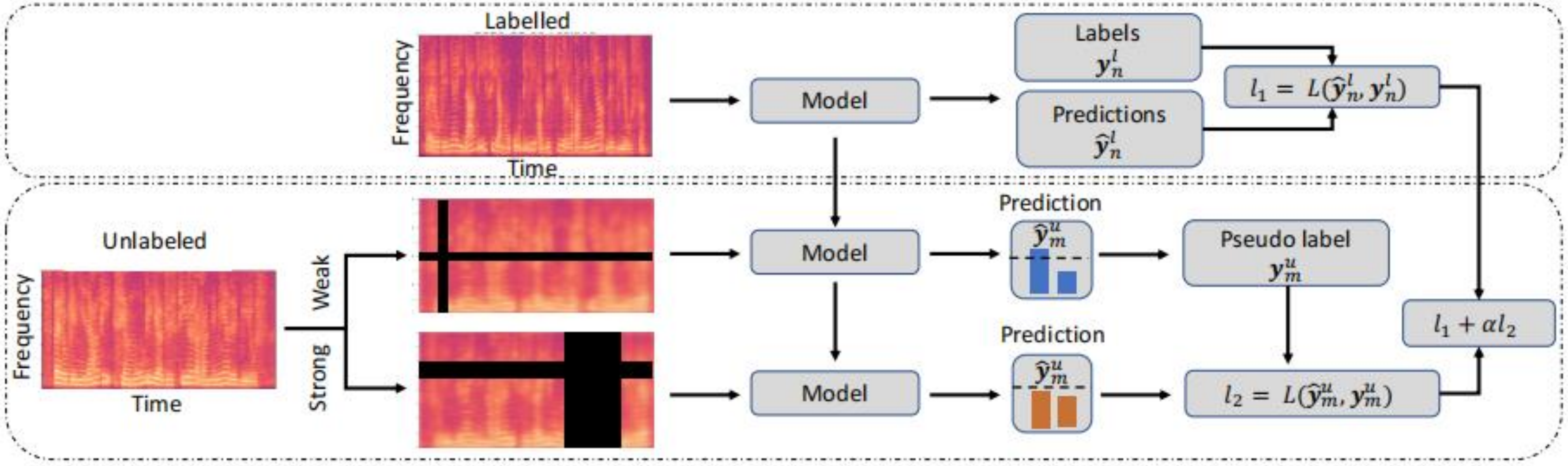


(a) Buckeye (b) Xitsonga (c) Mandarin (d) English (e) French

Figure 2: *NED/COV curves on Zerospeech corpora. Our method is represented with a line of blue dots and each other competitors as grey points: Garcia-Granada (A [33]), Jansen (B,[6]), Räsänen (L,M,N [7] and D [8] and G,H [9]) , Kamper (O [4] and P [34]), Lynsinski (I,L,J [10]), Bhati (E,F[11]).*

Exploring Semi-supervised Learning for Audio-based COVID-19 Detectionusing FixMatch

- A semi-supervised learning framework (SSL) for audio-based COVID-19 detection.
  - Labelled samples are first used to develop the supervised model, which is then adopted to gather the predictions for the weakly augmented unlabelled samples. Those with the predicted probability above a threshold for each class are selected as the confident samples. Their predictions are served as the artificial labels for the corresponding strongly augmented samples, which are combined with the labelled dataset to further optimise the model.



$$l_1 = \frac{1}{N}\sum_{n=1}^{n=N} L(\hat{\mathbf{y}}^l_n, \mathbf{y}^l_n)$$

$$\hat{\mathbf{y}}^u_m = f(\phi_w(\mathbf{x}^u_m))$$

$$l_2 = \frac{1}{M_1}\sum_{m_1=1}^{m_1=N} L(f(\phi_s(\mathbf{x}^u_m)), \mathbf{y}^u_{m_1})$$

$$l = l_1 + \alpha l_2$$

Exploring Semi-supervised Learning for Audio-based COVID-19 Detectionusing FixMatch

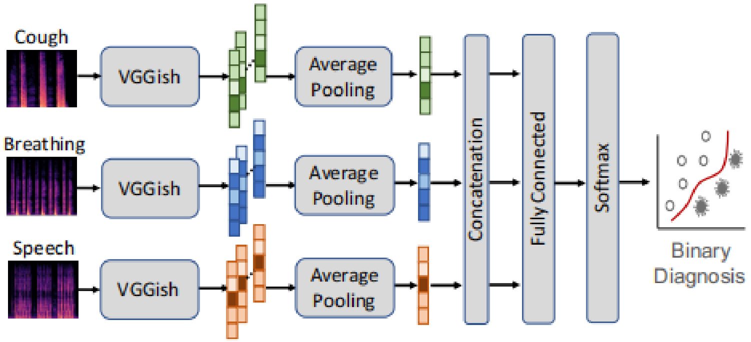


Figure 2: *Model structure. Three different modalities are used, and VGGish is used for feature extraction. These features from different modalities are concatenated and processed by fully connected layers for binary classification.*

- **Task 1**: Distinguish positive participants from negative (healthy) participants, which is the general case and referred as 'Pos-Neg'.
- **Task 2**: Distinguish symptomatic positive participants who reported at least one symptom from asymptomatic negative participants. This is expected to be a simple task as the audio sounds may show clear difference between the two subgroups. This task is referred as 'sPos-aNeg'.
- **Task 3**: Distinguish symptomatic positive participants from symptomatic negative participants, refereed as 'sPos-sNeg'.
- **Task 4**: Distinguish asymptomatic positive participants from asymptomatic negative participants, refereed as 'aPos-aNeg'.

- Size of labelled training data（Task1）

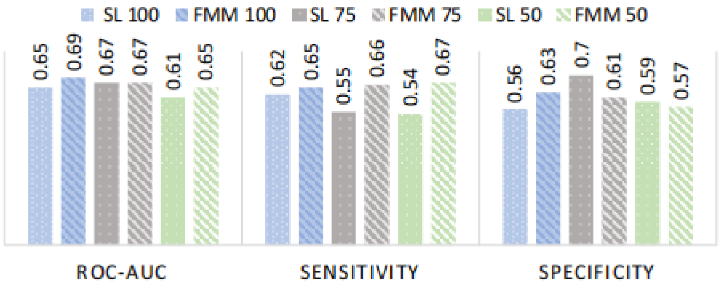


Figure 3: *System performance with different percentages of labelled data within [100% 75% 50%] for SL model and FMM^d. FMM^d outperforms or shows comparable performance as SL (ROC-AUC), and yields higher relative improvements or more balanced sensitivity and specificity with less labelled data.*

- Comparison with supervised model

Table 1: *System performance using supervised learning (SL) and SSL of pseudo labelling (PL) and Fixmatch (FMM) for COVID-19 detection. Both static ^{s} and dynamic ^{d} learning schemes are reported. FMM^d outperforms other systems.*

System	ROC-AUC	Sensitivity	Specificity
SL	0.65(0.59-0.71)	0.62(0.54-0.69)	0.56(0.49-0.64)
PL^s	0.65(0.58-0.70)	0.67(0.59-0.74)	0.51(0.43-0.58)
PL^d	0.67(0.61-0.73)	0.80(0.73-0.86)	0.41(0.34-0.48)
FMM^s	0.67(0.61-0.73)	0.68(0.61-0.75)	0.54(0.46-0.62)
FMM^d	**0.69(0.63-0.74)**	**0.65(0.58-0.73)**	**0.63(0.56-0.71)**

- Evaluation for different subtasks

Table 2: *System performance for SL and FMM^d for Tasks 2-4. Number of samples for each task is included in parenthesis (training/test). FMM^d shows great advantages in balancing sensitivity and specificity.*

Task	System	Accuracy	ROC-AUC	Sensitivity	Specificity
T2: sPos (*433/150*)-aNeg (*282/88*)	SL	0.69(0.63-0.75)	0.8(0.74-0.86)	0.55(0.45-0.65)	0.83(0.77-0.89)
	FMM^d	0.74(0.68-0.79)	0.78(0.72-0.84)	0.69(0.61-0.76)	0.78(0.69-0.86)
T3: sPos (*433/150*)- sNeg (*336/113*)	SL	0.57(0.51-0.63)	0.62(0.55-0.69)	0.73(0.66-0.8)	0.41(0.32-0.5)
	FMM^d	0.58(0.52-0.64)	0.63(0.56-0.7)	0.57(0.49-0.66)	0.59(0.5-0.68)
T4: aPos (*82/30*)-aNeg (*336/113*)	SL	0.55(0.47-0.65)	0.66(0.55-0.76)	0.27(0.12-0.43)	0.84(0.76-0.92)
	FMM^d	0.54(0.45-0.63)	0.6(0.47-0.71)	0.43(0.26-0.61)	0.72(0.62-0.81)

Speech Pre-training with Acoustic Piece

- Extract the patterns in HuBERT codes, named “acoustic piece”, and take it as the target label for

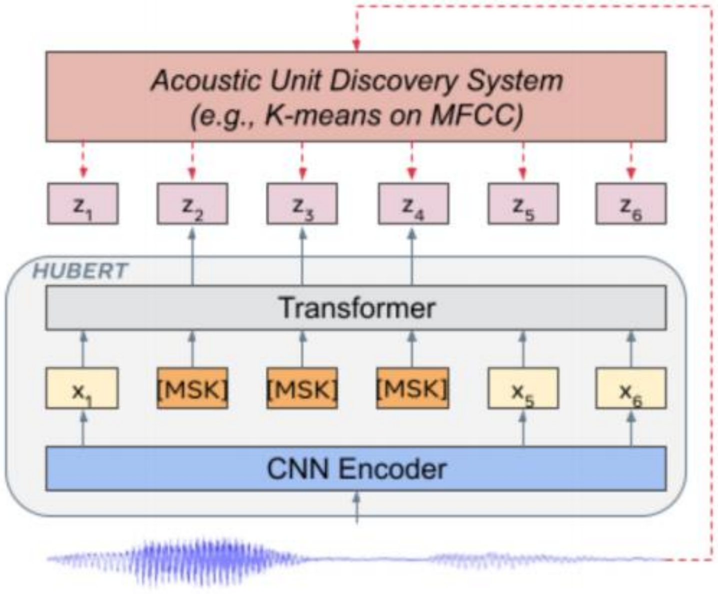


Figure 1: *The HuBERT model [4].*

- In the first iteration, the assigned labels are generated with k-means clustering (k=100) on the MFCC features extracted from the raw audio data. In the second iteration, the labels are generated with k-means clustering (k=500) based on the 6th layer hidden representations of the HuBERT model after the first iteration.

- Analysis on HuBERT Codes

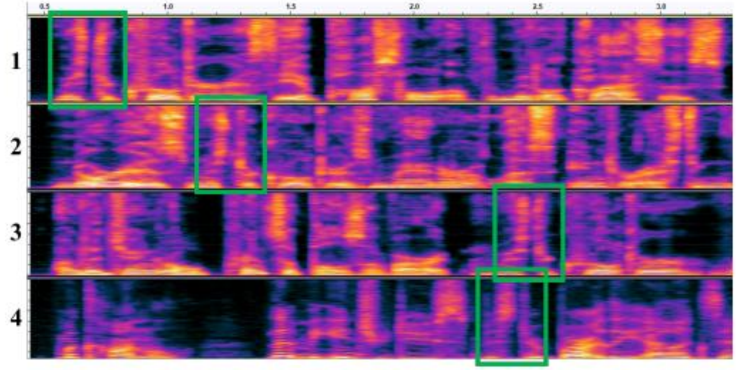


Figure 2: *The Mel-Spectrum of four sentences. The part in the green box corresponds to the word “mister”.*

	Codes corresponding to “mister” in the four samples
1	...178 **285 285 285 285 279 279 138 374 374 374** 224...
2	...309 **285 285 285 285** 378 **279 138 374 374 374** 52...
3	...178 **285 285 285 285** 378 **279 138 374 374** 52...
4	...309 **285 285 285 285** 258 378 **279 138 374 374** 52...

Table 1: *The codes of “mister” in different samples. The bold codes are shared by the four samples.*

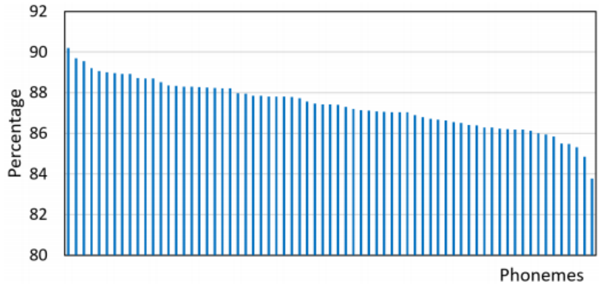


Figure 3: *The average percentage of the sharing code for each phoneme. The x-axis denote 69 phonemes sorted according to the value of the y-axis.*

Speech Pre-training with Acoustic Piece

- Merge the highly frequent code patterns into one piece
  - first use the 6th layer of the released HuBERT model to generate original labels with the offline clustering, then do sentence piece on it with different vocabulary to generate acoustic piece labels

- predefined vocabulary sizes of 1k, 2k and 3k.

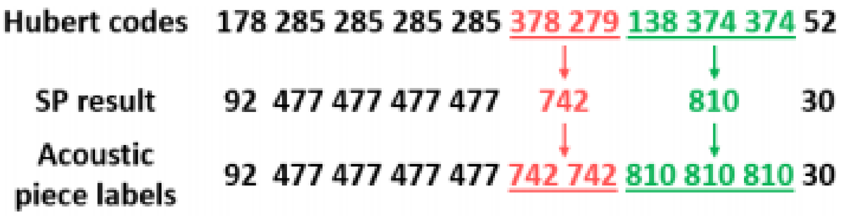


Figure 4: *Example of the acoustic piece generation.*

$$\log P_{CTC}(Y|X) + w_1 \log P_{LM}(Y) + w_2|Y| \qquad (1)$$

where Y is the predicted text, $|Y|$ is the length of the text, and w_1 and w_2 denote the language model weight and word score. The decoding hyperparameters are searched with Ax[2].

- Dataset
  - pre-training: Libri-Light , VoxPopuli , GigaSpeech
  - fine-tuning: train clean-100 subset (100 hours labeled data) of LibriSpeech

Model	LM	test	
		clean	other
1-hour labeled			
wav2vec 2.0 Base	None	24.5	29.7
WavLM Base	None	24.5	29.2
WavLM Base+	None	22.8	26.7
*HuBERT-AP Base	None	17.0	23.3
*HuBERT-AP Base+	None	16.9	22.3
DeCoAR 2.0	4-gram	13.8	29.1
DiscreteBERT	4-gram	9.0	17.6
wav2vec 2.0 Base	4-gram	5.5	11.3
HuBERT Base	4-gram	6.1	11.3
WavLM Base	4-gram	5.7	10.8
WavLM Base+	4-gram	5.4	9.8
*HuBERT-AP Base	4-gram	5.5	10.6
*HuBERT-AP Base+	4-gram	5.3	9.6
10-hour labeled			
wav2vec 2.0 Base	None	11.1	17.6
WavLM Base	None	9.8	16.0
WavLM Base+	None	9.0	14.7
*HuBERT-AP Base	None	9.1	15.2
*HuBERT-AP Base+	None	8.4	13.9
DeCoAR 2.0	4-gram	5.4	13.3
DiscreteBERT	4-gram	5.9	14.1
wav2vec 2.0 Base	4-gram	4.3	9.5
HuBERT Base	4-gram	4.3	9.4
WavLM Base	4-gram	4.3	9.2
WavLM Base+	4-gram	4.2	8.8
*HuBERT-AP Base	4-gram	4.2	9.0
*HuBERT-AP Base+	4-gram	4.1	8.4
100-hour labeled			
wav2vec 2.0 Base	None	6.1	13.3
WavLM Base	None	5.7	12.0
WavLM Base+	None	4.6	10.1
*HuBERT-AP Base	None	4.9	10.7
*HuBERT-AP Base+	None	4.6	9.5
DeCoAR 2.0	4-gram	5.0	12.1
DiscreteBERT	4-gram	4.5	12.1
wav2vec 2.0 Base	4-gram	3.4	8.0
HuBERT Base	4-gram	3.4	8.1
WavLM Base	4-gram	3.4	7.7
WavLM Base+	4-gram	2.9	6.8
*HuBERT-AP Base	4-gram	3.1	7.1
*HuBERT-AP Base+	4-gram	2.9	6.6

Table 3: *WER of ASR on the LibriSpeech and test sets, when trained on the LibriLight low-resource labeled data setups of 1 hour, 10 hours and the clean 100h subset of LibriSpeech. * means our method. + means pre-training with 1M update steps.*

Method	Precision	Recall	F1
HuBERT codes	0.387	0.672	0.421
Acoustic piece	0.579	0.712	0.628

Table 4: *Quantitative evaluation of segment boundaries of different methods wrt. golden phoneme boundaries.*

Model	test clean	test other
HuBERT	3.4	8.1
AP 1k	**3.1**	**7.1**
AP 1k + HuBERT	3.2	7.1
AP 2k	3.2	7.3
AP 3k	3.1	7.2
AP 5k	3.2	7.3
AP 10k	3.3	7.5

Table 5: *The influence of vocabulary size of the sentence piece model. "AP" means our method and "1k", "2k" ,"3k", "5k", "10k" mean different vocabulary sizes.*

Acoustic Feature Shuffling Network for Text-Independent Speaker Verification

- Propose an acoustic feature shuffling network to learn the order-insensitive speaker embeddings via a joint learning method..

- Backbone：SE-ResNet
- Dataset：Voxceleb2

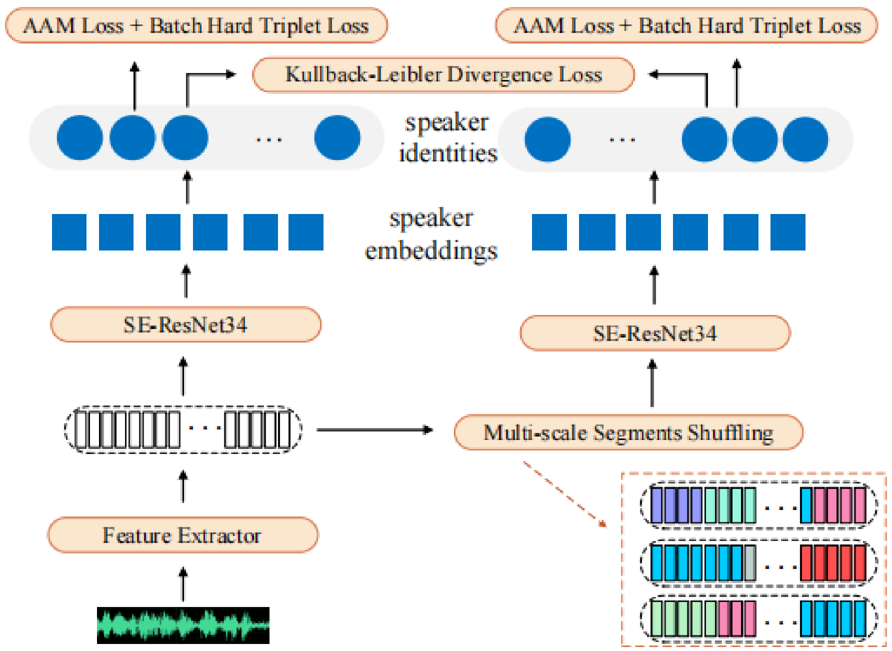


Figure 1: *Acoustic Feature Shuffling Network.*

- Multi-scale segments shuffling
  - the local sequential dependency is important for speech perception, and the frame-level shuffling would completely convert the feature sequence to noise sequence
  - different lengths of text contents needs different scales to shuffle segments

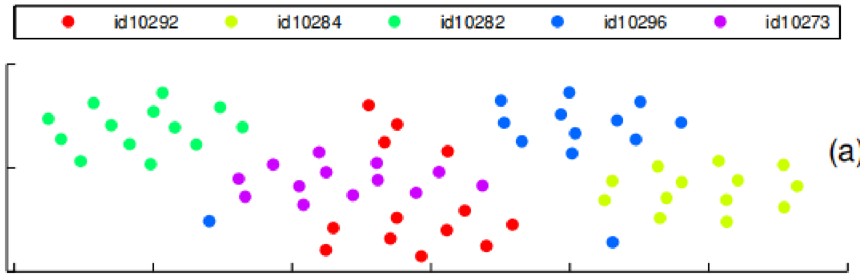
- Joint learning approach
  - the parameters are updated independently

$$Loss(\lambda) = Loss_{AAM}(\lambda) + Loss_{BHT}(\lambda) + D_{KL}(\lambda, \theta)$$

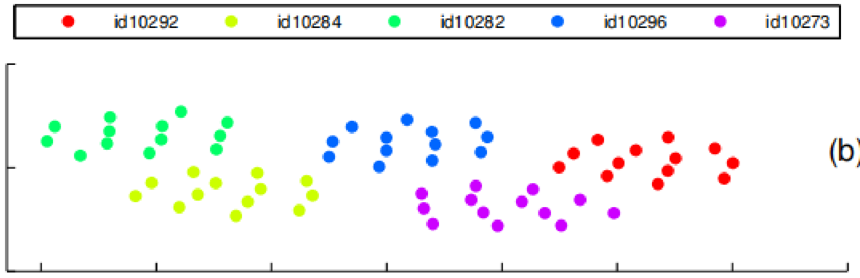
$$Loss(\theta) = Loss_{AAM}(\theta) + Loss_{BHT}(\theta) + D_{KL}(\lambda, \theta)$$

Acoustic Feature Shuffling Network for Text-Independent Speaker Verification

Randomly select 5 speakers from cleaned VoxCeleb1, every speaker only provides an utterance. Acoustic feature shuffling is carried on all utterances at many segment scales, so lots of features are generated from every utterance.



(a)



(b)

Figure 3: *2D t-SNE plot of speaker embeddings. (a) are speaker embeddings extracted from the baseline SE-ResNet34; (b) are speaker embeddings extracted from the proposed joint learning method.*

Table 1: *comparison of EER (%) performance on the cleaned testsets of different segment scales.*

Model	training method	segment scale of frames	feature sequential order of train data		EER(%)		
			normal order	shuffled order	Vox1-O*	Vox1-E*	Vox1-H*
SE-ResNet34	—	—	✓		1.412	1.353	2.419
	—	1		✓	7.596	7.660	11.094
	Pool		✓	✓	2.252	2.112	3.569
	Symmetric KLD		✓	✓	1.715	1.644	2.801
	—	10		✓	1.795	1.687	2.924
	Pool		✓	✓	1.529	1.392	2.410
	Symmetric KLD		✓	✓	1.300	1.343	2.320
	—	50		✓	1.306	1.335	2.360
	Pool		✓	✓	1.369	1.250	2.246
	Symmetric KLD		✓	✓	1.327	1.191	2.171
	—	80		✓	1.204	1.288	2.289
	Pool		✓	✓	1.279	1.255	2.335
	Symmetric KLD		✓	✓	1.215	1.231	2.157
	—	100		✓	1.311	1.266	2.294
	Pool		✓	✓	1.327	1.349	2.318
	Symmetric KLD		✓	✓	1.274	1.195	2.179

* These are just separately short for VoxCeleb1-O,VoxCeleb1-E and VoxCeleb1-H testset.

Table 2: *comparison of EER (%) performance on the cleaned testsets of different multi-scale architectures.*

Model	multi scales of frames	EER(%)		
		Vox1-O*	Vox1-E*	Vox1-H*
SE-ResNet34 + Symmetric KLD	10-50-80	1.412	1.300	2.314
	10-50-100	1.242	1.258	2.218
	10-80-100	1.316	1.221	2.203
	50-80-100	**1.183**	**1.220**	**2.152**

Table 3: *comparison of EER (%) performance on the cleaned testsets with recently reported ResNet34-based systems.*

Model	EER(%)		
	Vox1-O*	Vox1-E*	Vox1-H*
ResNet34 [18]	1.67	1.81	3.23
ResNet34+Circle-Stage [19]	1.31	1.51	2.61
ResNet34+ISKConv+MSSP [20]	1.292	1.319	2.396
SE-ResNet34	1.412	1.353	2.419
SE-ResNet34+Symmetric KLD	**1.183**	**1.220**	**2.152**

Self-Supervised Speaker Verification Using Dynamic Loss-Gate and Label Correction

- Propose dynamic loss-gate and label correction (DLG-LC) to alleviate the performance degradation caused by unreliable estimated labels.

$$L_{dino} = L_{ce} + \alpha \sum_{e \in \{e_1^l, e_2^l\}} \sum_{e' \in \{e_1^s \dots e_4^s\}} \left(1 - \frac{e \cdot e'}{\|e\| \, \|e'\|}\right) \quad (3)$$

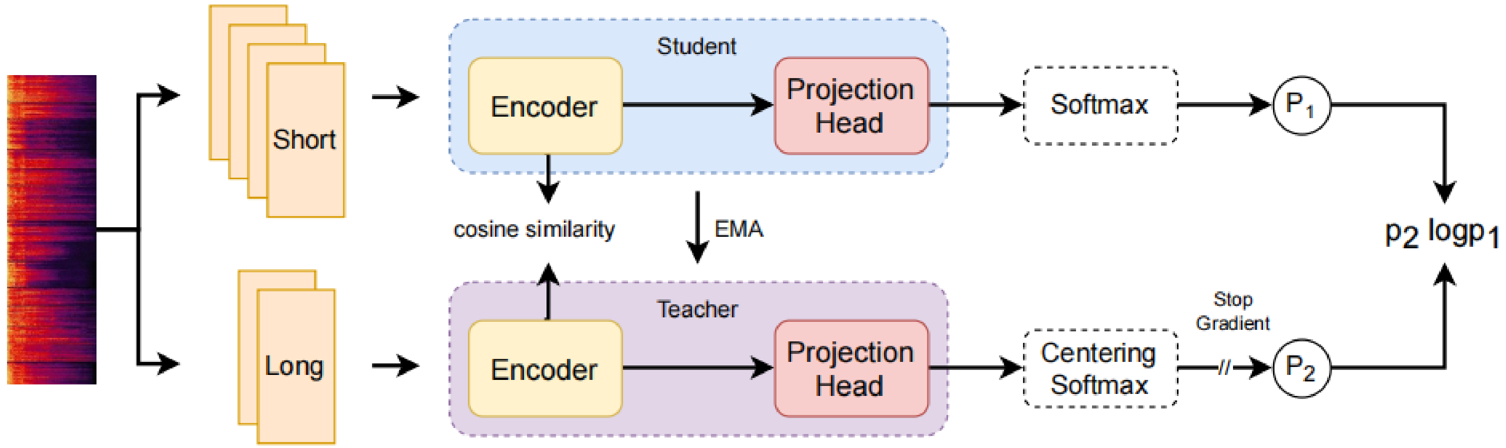
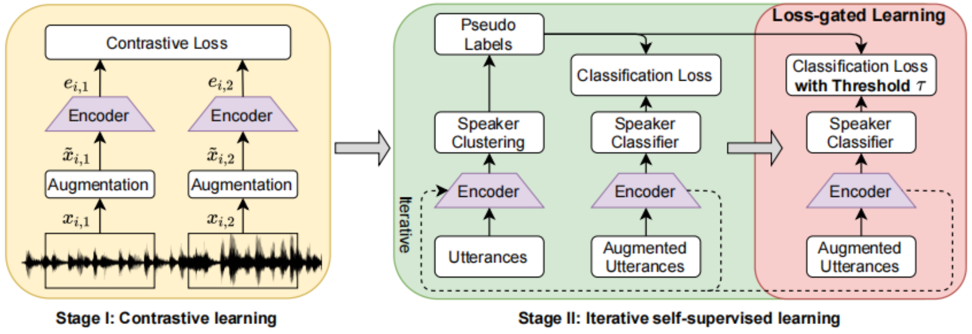


Figure 1: *Framework of Distillation with no label (DINO) for self-supervised speaker representation learning*

Table 1: *Comparison of DINO with other self-supervised speaker verification work. EER (%) and minDCF are evaluated on Vox-O test set.*

Method	EER (%)	minDCF
Disent [5]	22.09	-
CDDL [6]	17.52	-
GCL [7]	15.26	-
i-vector [8]	15.28	0.63 (p=0.05)
AP + AAT [8]	8.65	0.45 (p=0.05)
SimCLR + uniform [9]	8.28	0.610
MoCo + WavAug [10]	8.23	0.590
Unif+CEL [11]	8.01	-
DINO	31.23	0.990
+ EMA	7.02	0.579
+ + Multi-Crop	6.35	0.566
+ + + Cosine loss	**6.16**	**0.524**

- Dynamic Loss-Gate



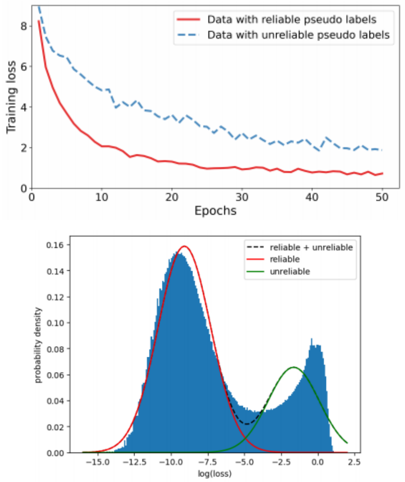


Figure 2: *Loss distribution of LGL on Voxceleb 2. Loss value is scaled by log function. And the lines are estimated by GMM.*

$$p(x) = \lambda_1 \mathcal{N}\left(\mu_1, \sigma_2^2\right) + \lambda_2 \mathcal{N}\left(\mu_2, \sigma_2^2\right)$$

$$\tau : p_1(\tau) = p_2(\tau)$$

$$L_{DLG} = \sum_{i=1}^{N} \mathbb{1}_{l_i < \tau} \log \frac{e^{s(\cos(\theta_{y_i,i} + m))}}{Z}$$

$$Z = e^{s(\cos(\theta_{y_i,i}+m))} + \sum_{j=1, j \neq i}^{c} e^{s(\cos(\theta_{y_i,i}))},$$

Self-Supervised Speaker Verification Using Dynamic Loss-Gate and Label Correction

- Label Correction
- the model's output prediction is more reliable than pseudo labels which are generated by clustering

$$L_{LC} = \sum_{i=1}^{N} \mathbb{1}_{l_i > \tau, \max(\hat{p_i}) > \tau_2} H(\hat{p_i} \mid p_i) \qquad (8)$$

where p_i and $\hat{p}_i$ represent the output probability of augmented segments and their corresponding clean version respectively.

H(·) denotes the cross-entropy between two probability distributions.

$$L = L_{DLG} + L_{LC}$$

Table 3: *EER (%) comparison on Vox-O test set for different iterations of the proposed DLG-LC with other strategies. SimCLR and DINO mean that we simply used all the estimated pseudo labels without any loss-gate during training process.*

Method	SimCLR [9]	DINO	LGL [15]	DLG-LC
Loss-Gate	×	×	✓	✓
Iter-1	6.281	4.255	3.520	**2.723**
Iter-2	5.914	3.946	2.410	**1.888**
Iter-3	5.547	3.824	2.070	**1.670**
Iter-4	4.872	3.691	1.950	**1.495**
Iter-5	4.484	3.510	1.660	**1.468**

Table 2: *EER (%) comparison on Vox-O, E, H of the proposed DLG-LC in Iteration 1. In this experiment, pseudo labels are estimated from our pre-trained DINO system. DINO means we simply used all the data with the estimated pseudo labels as the supervisory signal without any loss-gate during system training.*

Method	Vox-O	Vox-E	Vox-H
DINO	4.255	4.900	8.005
+ LGL [15]	3.590	4.373	6.935
+ DLG	3.202	3.525	5.805
++ LC	**2.723**	**3.179**	**5.442**

Table 4: *EER (%) comparison for different self-supervised speaker verification methods on Vox-O, E, H*

Method	Clustering	Iter	Vox-O	Vox-E	Vox-H
ID [35]	AHC(7500)	7	2.10	-	-
JHU [36]	AHC(7500)	5	1.89	-	-
DKU [37]	K-M(6000)	4	1.81	-	-
SNU [38]	AHC(7500)	5	1.66	-	-
LGL [15]	K-M(6000)	5	1.66	2.18	3.76
DLG-LC	K-M(7500)	5	**1.47**	**1.78**	**3.19**

Non-Contrastive Self-Supervised Learning of Utterance-Level Speech Representations

- the DINO embedding may include attributes that are consistent within the utterance, such as speaker information, accent/language, emotion, and age.

Table 1: *Comparison between DINO, momentum contrast (MoCo), and x-vector embeddings for speaker verification. The results are on the original VoxCeleb1 test with equal error rate (EER)(%) and MinDCF with P_T=0.01. The PLDA back-end was trained with VoxCeleb1 dev where its data size is 1/7 of VoxCeleb2 dev.*

	Cosine scoring		PLDA	
	EER(%)	MinDCF	EER(%)	MinDCF
DINO	4.83	0.463	2.38	0.289
MoCo [18]	7.3	-	-	-
x-vector	1.94	0.207	1.88	0.189

- iterative clustering stage
  - trained a new larger model, ResNet34 x-vector model, in a supervised way with the AAM loss based on pseudo speaker labels generated using the initial DINO model.
- robust training stage
  - used a new larger model, Res2Net50 with pseudo labels generated from the ResNet34. After the first 30 epochs of training, the post pooling layers of the model were fine-tuned with a larger margin, 0.5, in the AAM loss.

Table 2: *Speaker verification results over 3 different trial lists with progressing/different systems over the three stages. The numbers from [18] seems rounded to the nearest tenth. Pseudo labels for robust training were generated from ResNet34 (iter3).*

Stage	Algorithm/Loss	Model	EER (%) with cosine scoring		
			voxceleb1_test_o	VoxSRC-21 *val*	VoxSRC-21 *test*
Initial model training (self-supervised learning)	DINO	LResNet34	4.83	13.96	-
	MoCo	ECAPA [18]	7.3	-	-
Iterative clustering	AAM loss (margin=0.3)	ResNet34 (iter1)	2.56	8.59	-
		ResNet34 (iter2)	2.13	7.35	-
		ResNet34 (iter3)	2.13	6.97	-
		ResNet34 (iter4)	2.14	6.88	-
		ECAPA (iter7) [18]	2.1	-	-
Robust training + larg-margin fine-tuning	AAM loss (margin=0.5)	Res2Net50	1.89	6.50	6.88
			1.91	6.32	6.64

Non-Contrastive Self-Supervised Learning of Utterance-Level Speech Representations

- Emotion recognition

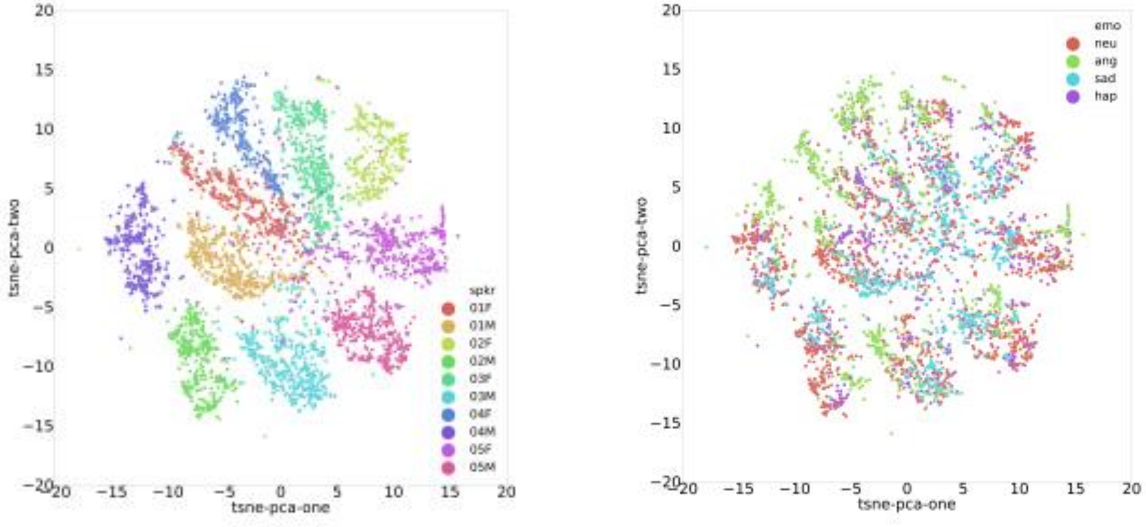


Figure 2: *Analysis of DINO embedding space for IEMOCAP using t-SNE plots. Each color represents one speaker in the top plot and one emotion in the bottom plot.*

Table 3: *Emotion classification results on three different dataset. All numbers in this table are micro-f1 (%) scores*

	IEMOCAP	Crema-D	MSP-Podcast
x-vector [25]	56.11	75.65	52.58
DINO	**60.87**	**79.21**	**56.98**

Reducing Domain mismatch in Self-supervised speech pre-training

- Propose ask2mask (ATM), a novel approach to focus on specific samples during MSM pre-training.

  - let the input speech sequence X = [x1, x2, ..., xT0], where xt is the log Mel-filterbank feature vector at time t.
  - X is sent to the feature encoder Φ to obtain the encoded representations E = Φ(X). Get E = [e1, e2, ..., eT].
  - The masking is done over sets of frames or blocks b1, b2, ..., bK and accommodates overlap between blocks. Here K is the number of masked blocks in a randomly masked encoded sequence ˜E.
  - The block bk = [ik, c], where ik is the starting index of the masked block and c is the corresponding right context size.

  - For each encoded feature frame et ∈ E, the scorer emits probabilities p(vt = l | E); l ∈ L of the frame belonging to a particular label.

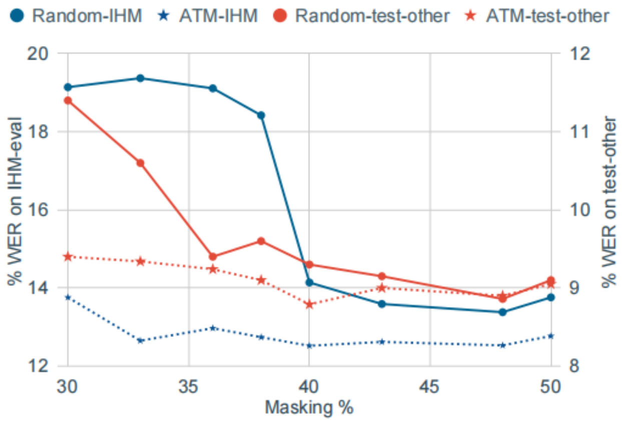
$$s_t = \max_{l} p(v_t = l \,|\, \mathbf{E})$$

sample beginning frames with probability proportional to the scores of each frame.

Reducing Domain mismatch in Self-supervised speech pre-training

Pretraining (PT): Libri-light (LL-60k) dataset
Finetuning (FT): 1) 100hrs of Librispeech (LS-100). 2) 100 hours of AMI and 3) speechstew (5k hours)
Evaluation: ATM performance on AMI using IHM-eval and SDM-eval.

- Masking percentages



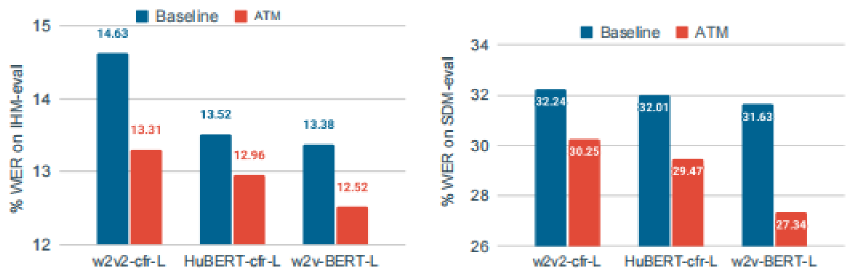


Figure 2: *Performance comparison of different MSM architectures with and without applying ATM on IHM-eval and SDM-eval in AMI. All these models are FT using AMI. Here "cfr" refers to conformer.*

Table 3: *%WER obtained by FT with AMI using w2v-BERT-XL model using baseline and ATM. Evaluation is done on AMI test sets to highlight the effect on mismatched condition.*

MSM arch.	IHM-eval	SDM-eval
w2v2-cfr-XL	10.4	25.7
+ATM	10.0	24.5
w2v-BERT-XL	10.1	25.1
+ATM	9.5	23.7

Table 2: *Performance comparison of different MSM architectures with and without applying ATM on all evaluation sets on Librispeech.*

Model	PT-LL, FT-LS100			
	dev	dev-other	test	test-other
w2v-BERT-L	3.78	8.86	3.85	9.32
+ATM	3.71	8.97	3.89	8.92
w2v2-cfr-XL	2.5	4.7	2.6	4.9
+ATM	2.4	4.6	2.5	5.0
HuBERT-cfr-XL	2.5	4.7	2.6	5.0
+ATM	2.5	4.6	2.5	5.0
w2v-BERT-XL	2.4	4.4	2.5	**4.6**
+ATM	**2.3**	**4.4**	**2.4**	4.7

Table 4: *Comparison with state-of-the-art results on SpeechStew. The FT is done on SpeechStew and the results are evaluated using Kaldi scoring to match published results. Note that the model has never seen any CHiME-6 data, and we use it as an example for* ***zero-shot*** *learning mode on how the model performs on chime-6 without seeing any of its training data.*

Model	AMI		CHiME-6
	IHM	SDM	
Speechstew [10]	9.0	21.7	57.2
w2v2-cfr-XL [10]	9.6	23.8	56.4
w2v-BERT-XL	9.2	21.5	55.5
+ ATM	9.0	21.0	54.3

Using Data Augmentation and Consistency Regularization to Improve Semi-supervised Speech Recognition

- Consistency Regularization (CR)
- the decision boundary between classes lies in low density regions
- when a realistic perturbation is applied to a model's input then its prediction should not diverge.
- The success of CR is therefore related to the quality and diversity of input perturbations.

$$\mathcal{L} = -\sum_{(X_l, Y_l) \in \mathcal{D}_L} \log P(Y_l | X_l, \theta) + w \sum_{X_u \in \mathcal{D}_U} D(F(X_u), F(\widehat{X_u}))$$

- E2E ASR model
- Conformer encoder Fe encodes at time t, each acoustic feature xt into a hidden representation ht.
- The prediction network Fp maps a output token into another hidden representation gi.
- The joint network Fj fuses information from both Fe and Fp to compute the posterior probability of next token or blank.

- prediction network tends to produce spiky posteriors and augmentation of input features, such as time warping, can cause these posteriors to spike at different positions in the posterior lattice.

- Proposed Approach

$$\tilde{Y}_u = \underset{Y_u}{\operatorname{argmax}} \log P(Y_u | \tilde{X}_u, \theta)$$ weakly augmented version $\tilde{X}_u$

$$\mathcal{L} = -\sum_{X_l, Y_l} \log P(Y_l | X_l, \theta) - w \sum_{X_u} \log P(\tilde{Y}_u | \widehat{X}_u, \theta)$$ strongly augmented version $\widehat{X}_u$

  - errors in predictions can get reinforced due to enforced consistency

$$\theta'_\tau = \alpha \theta'_{\tau-1} + (1-\alpha)\theta_\tau \qquad \tilde{Y}_u = \underset{Y_u}{\operatorname{argmax}} \log P(Y_u | \tilde{X}_u, \theta')$$

Using Data Augmentation and Consistency Regularization to Improve Semi-supervised Speech Recognition

- Data Augmentation

  Pitch shift, Background Noise, Reverberations, Time frequency masking, Input Mixup

* Pretraining Stage:

1. Sample $\mathcal{T}_W \sim \mathcal{A}_W$
2. Compute frame level pseudo labels at time t for feature $X_{u,t}$ as $\tilde{E}_{u,t} = \operatorname{argmax} F^e(\mathcal{T}_W(X_{u,t}))$
3. Sample k different augmentations $\mathcal{T}_1, ..., \mathcal{T}_k \sim \mathcal{A}_S$.
4. For $i = 1...k$, apply $\mathcal{T}_i$ such that,

$$\mathcal{T}_i(X) = \begin{cases} X, & \text{if } p_i < q_i \sim U(0,1) \\ \mathcal{T}_i(X), & \text{otherwise} \end{cases}$$

5. Compose strong augmentation $\hat{X}_u = \mathcal{T}_1, ..., \mathcal{T}_k(X_u)$
6. Use cross-entropy loss and, frame level targets $E_{l,t}$ and $\tilde{E}_{u,t}$ in (3), to pretrain the encoder by minimizing the sum of supervised and consistency loss.

* E2E Stage:

7. Apply Step 1. above to get weakly augmented feature $\tilde{X}_u$.
8. Using $\tilde{X}_u$ perform beam search at the output of F^j to find the best pseudo label sequence $\tilde{Y}_u$
9. Apply Step 3. to Step 5. above to get strongly augmented feature $\hat{X}_u$
10. Use E2E transducer loss and, sequence labels Y_l and $\tilde{Y}_u$ in (3), to minimize the sum of supervised and consistency loss.

Table 1: *Model architecture and setup*

Feature representation	3 * 64 dimensional LFBE Features
Label representation	4000 Word Pieces (Plus Blank Symbol)
Feature Embedding	CNN: Layers = 2, Kernel = 3x3, Stride Layer 1= 2 Stride Layer 2 = 1
Encoder architecture	Conformer Block : Layer = 14, Kernel = 15, Attention Heads = 8, Encoder Dim = 512, FeedForward Dim = 1024
Decoder architecture	LSTM: Unidirectional, Layers = 2, Units = 1024
Labeled data	2000 hours
Unlabeled data	$\sim$ 100000 hours

Table 2: *WER reduction for* $\boxed{\textit{CE pretrained models}}$ *on 100000 hours of unlabeled audio and 2000 hours of labeled audio. Compared to baseline negative WERR means degradation in performance and Positive WERR means improvement.*

Method	Labeled Aug.	Unlabeled Strong Aug.	Test WERR(%)
Supervised Baseline	SA	-	0.00
Vanilla CR	SA	SA	6.53
Random CR	SA	RA	**8.60**

self-labeling: first pretrained using cross-entropy training followed by end-to-end training using transducer loss on labeled data.

Table 3: *WER reduction for models E2E (transducer loss) trained on 100000 hours of unlabeled audio and 2000 hours of labeled audio. Transforms applied in training include: 1.) No Augmentation (NA); 2.) Spec Augment (SA); 3.) Randomly Combined Augmentation (RA). Were applicable only SA was applied as weak augmentation. Random-MA applies model averaging. All the models were CE pretrained, except those indicated by (NP).*

Method	Labeled Aug.	Unlabeled Strong Aug.	Test WERR(%)	Rare Words WERR(%)
Self-labeling	SA	-	0.00	0.00
Supervised (NP)	SA	-	-28.27	-46.90
Supervised	SA	SA	-7.22	-19.86
Vanilla CR	SA	SA	4.11	0.77
Random CR	SA	RA	8.53	8.26
Random-MA CR	SA	RA	**9.16**	**12.32**

Table 4: *Comparing the difference in performance due to different distance measures in CR: 1.) transducer loss computed from Pseudo Labels (PL), 2.) L2 distance and 3.) Cosine distance.*

Method	Transducer (PL)	MSE	Cosine
Test (WERR %)	0.0	-6.29	-2.89
Rare Words (WERR %)	0.0	-8.89	-5.03

SPLICEOUT: A Simple and Efficient Audio Augmentation Method

- Audio Augmentations
- Warping-based Mixing-based Masking-based Noise-based

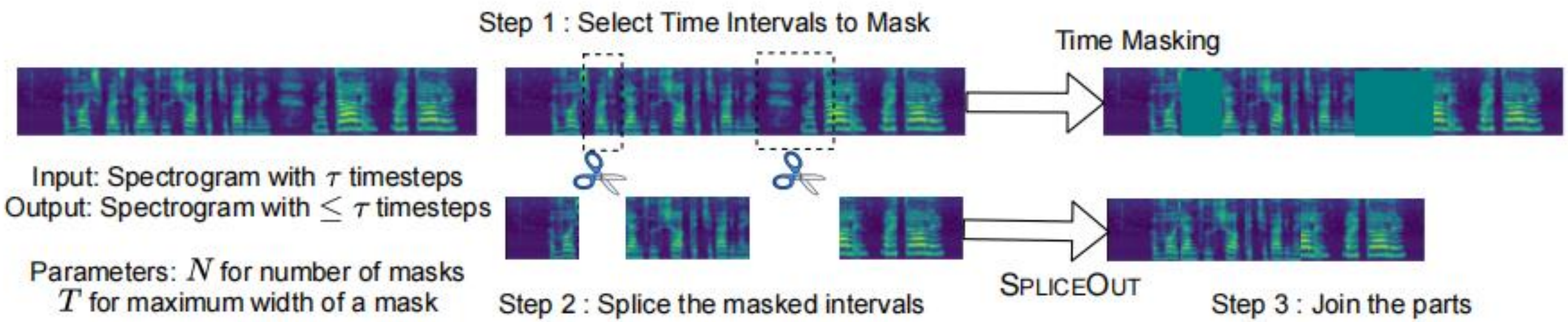


Figure 1: Illustration of SPLICEOUT and time masking.

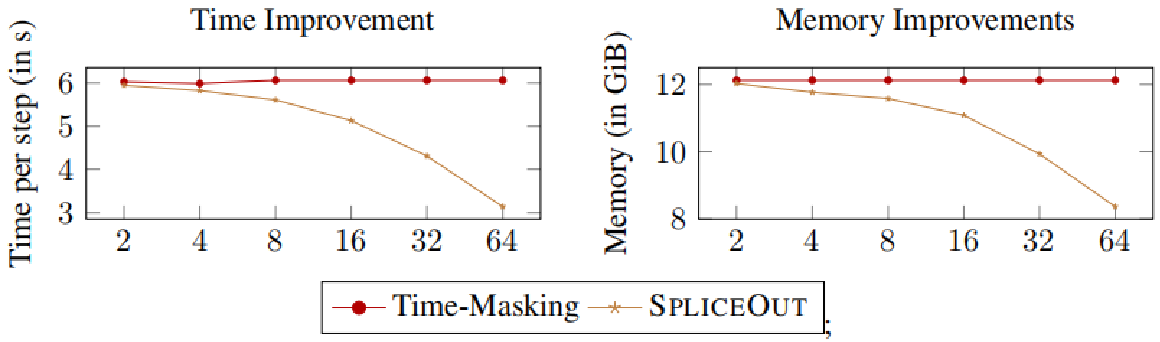


Figure 2: Comparison of running time and memory requirements during training using Time-Masking and SPLICEOUT augmentations, with varying number of masks.

SPLICEOUT: A Simple and Efficient Audio Augmentation Method

- ASR: LibriSpeech

Table 2: WERs on LibriSpeech test sets, using TM, FM and SPLICEOUT (SO), with $N = 2$.

Augmentation	test-clean	test-other
TM	7.6±0.19	21.8±0.31
SO	**7.5**±0.18	**21.4**±0.31
FM + TM	**7.2**±0.17	18.3±0.28
FM + SO	**7.2**±0.18	**18.2**±0.29
TW + FM + TM [14, 54]	**7.0**±0.16	18.1±0.28
TW + FM + SO	7.1±0.17	17.9±0.29
TW + FM + TM + SO	7.1±0.18	**17.7**±0.29

Table 3: Effect of increasing the number of masks, N, in Time-Masking and SPLICEOUT (SO) augmentations, on WERs of LibriSpeech test sets.

N	Method	test-clean	test-other
2	TM	7.6±0.19	21.8±0.31
	SO	**7.5**±0.18	**21.4**±0.31
4	TM	7.3±0.18	20.4±0.30
	SO	**7.0**±0.16	**20.3**±0.31
8	TM	**6.8**±0.17	19.2±0.29
	SO	**6.8**±0.18	**19.0**±0.30

Table 4: WERs on LibriSpeech test sets comparing Semantic-Mask and Semantic-Splice.

Method	test-clean	test-other
Semantic-Mask [55]	9.2	22
Semantic-Splice (Ours)	**8.8**	**21.5**

- ASR for Multiple Languages

Augmentation TW + FM	Swedish 10 hrs	Turkish 22 hrs	Kyrgyz 22 hrs	Ukrainian 25 hrs	Tatar 28 hrs	Welsh 96 hrs
+ TM [14, 54]	33.2	6.7	37.3	14.6	36.3	15.4
+ SPLICEOUT	**32.1**	**6.5**	**36.2**	**14.0**	**35.5**	**14.8**

- Speech Translation: Libri-Trans

Table 6: Evaluating TM and SPLICEOUT using development and test set BLEU scores for the Libri-Trans task. Higher is better.

Augmentation	Dev BLEU	Test BLEU
TW + FM + TM [14, 58]	18.43	17.18
TW + FM + SPLICEOUT	**18.57**	17.15
TW + FM + TM + SPLICEOUT	18.42	**17.35**

- Sound Classification: ESC-50 and UrbanSound8K

Table 7: Evaluating TM and SPLICEOUT on two sound classification tasks, with the standard augmentation combinations [15, 16]. Higher is better.

Augmentation	Accuracy	$F1_{micro}$	mAP
Dataset: ESC-50			
MX + FM + TM	90.40±0.02	89.42±0.02	94.98±0.01
MX + FM + SO	90.95±0.02	89.96±0.02	95.17±0.01
Dataset: UrbanSound8k			
MX + FM + TM	86.39±0.04	**86.32**±0.04	**93.04**±0.03
MX + FM + SO	**86.67**±0.04	**86.31**±0.04	**93.04**±0.03

SPLICEOUT: A Simple and Efficient Audio Augmentation Method

- Music Classification: GTZAN

Table 8: Evaluating TM and SPLICEOUT, with and without FM, on GTZAN Music Genre Classification. SPLICEOUT is complementary to TM. Higher is better.

Augmentation	Accuracy
MX + TM	90.7±0.03
MX + SO	90.7±0.03
MX + FM + TM [15]	91.3±0.03
MX + FM + SO	**91.4**±0.03
MX + FM + TM + SO	**92.8**±0.03

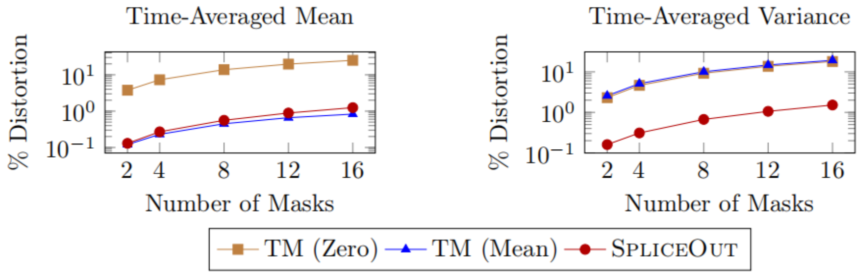


Figure 3: Comparison of % distortion in the Time-Averaged Statistics of different augmentation methods, compared to the unaltered input, with varying number of masks.

- Representation Learning

$$\mathcal{L}_{CL} = -\sum_{i,j}^{N} \log \frac{\exp\left(\mathrm{sim}\left(\mathbf{z}_i, \mathbf{z}_j\right)/\tau\right)}{\sum_{k=1}^{2N} \mathbf{1}_{[k \neq i]} \exp\left(\mathrm{sim}\left(\mathbf{z}_i, \mathbf{z}_k\right)/\tau\right)}$$

Table 9: Comparing classification accuracies using SPLICEOUT and TM in semi-supervised (with different amounts of labeled data) and self-supervised settings on the Speech Commands Dataset. Higher is better.

		Labeled Data Percentage			
Type	Method	100%	20%	10%	1%
Supervised	Cross Entropy	94.9	86.4	68.4	28.6
Semi-Supervised	SupCon	96.0	87.9	82.1	26.6
	CLAR (FD + TM) [14, 30]	97.2	94.7	91.7	**72.8**
	CLAR (FD + SPLICEOUT)	**97.4**	**95.6**	**92.6**	71.2
Self-Supervised	SimCLR (FD + TM) [14, 30]	89.0			
	SimCLR (FD + SPLICEOUT)	88.9			

Table 10: Perceptual speech metrics, both absolute and relative, comparing the quality of speech modified by TM (Zero), TM(Mean), and SPLICEOUT transformations. Higher is better.

	Absolute	Relative	
Augmentation	SRMR	Wide-Band PESQ	Narrow-Band PESQ
TM (Zero)	**9.24**	3.07	3.35
TM (Mean)	9.14	3.05	3.46
SPLICEOUT	**9.24**	**3.33**	**3.59**

Supervision-Guided Codebooks for Masked Prediction in Speech Pre-training

- SSL: HuBert
- Self-Training: a teacher model is trained on the labeled data and then the unlabeled set is labeled with this initial model (a.k.a. pseudo-labeling). Finally, a new student model is trained on the combined labeled and pseudo-labeled data.
- Combination of SSL and Self-Training: first pre-trains a model on dataset unlabeled data, fine-tunes it on dataset labeled data. Then this fine-tuned model is used as the initial teacher model for pseudo-labeling.

Algorithm 1 Pipeline of our methods

Input: Labeled dataset S, Unlabeled dataset U.

1. Train a supervised model M_0 on dataset S, set $M = M_0$.
2. Generate pseudo-labeled dataset $M(U)$ with M.
3. Generate frame-level alignments or K-means clusters $A(U)$ and $A(S)$ with M.
4. Pre-train a masked-prediction model M' on dataset $A(U) \cup A(S)$.
5. (Optional) set $M = M'$, go to 2.
6. Fine-tune the pre-trained model M' on S or $M(U) \cup S$.
7. (Optional) Set $M = M'$, generate pseudo-labeled dataset $M(U)$, go to 5.

a hybrid(PBERT) or an end-to-end one (CTC clustering)

Table 1: *Results and comparisons in base model setting. All models only utilize 100h labeled data, and 860 unlabeled data.*

Model	**LM**	test-clean	test-other
Supervised Baselines			
Transformer CTC	None	8.8	26.5
	4-gram	5.0	16.8
LF-MMI (Hybrid)	4-gram	4.6	15.0
Self-supervised Baselines			
wav2vec 2.0 [5]	None	6.1	13.3
	4-gram	3.4	8.0
HuBERT iter 1 [7]	None	7.4	16.2
	4-gram	3.9	9.5
HuBERT iter 2 [7]	None	5.9	13.0
	4-gram	3.4	8.1
+ rel bias	None	5.7	12.3
	4-gram	3.4	8.1
Random-codebook [33] + rel bias	None	6.9	14.6
	4-gram	3.7	9.0
WavLM + rel bias [34]	None	5.7	12.0
	4-gram	3.4	7.7
Self-training Baselines			
Self Training (ST) [16]	GCNN	5.8	20.1
IPL [17]	4-gram + Trans.	5.6	9.0
Noisy Student [18]	LSTM	4.2	8.6
self-training (Ours)	None	4.9	14.4
	4-gram	3.5	9.7
+ 2nd iteration	None	4.3	11.0
	4-gram	3.3	8.4
Our Methods			
PBERT	None	4.9	11.7
	4-gram	**3.1**	7.7
+ rel bias	None	4.7	11.2
	4-gram	**3.1**	7.5
+ 2nd iteration	None	4.7	10.7
	4-gram	**3.1**	7.3
+ self-training	None	**4.2**	**9.5**
	4-gram	**3.1**	**7.2**
CTC clustering + rel bias	None	5.2	11.4
	4-gram	3.2	7.4
Ground-truth phones + rel bias	None	*4.5*	*10.0*
	4-gram	*3.1*	*6.8*

Supervision-Guided Codebooks for Masked Prediction in Speech Pre-training

- Non-ASR Task Transfer

Table 2: *Equal error rates (EERs) on VoxCeleb 1 speaker verification test set.*

Model	EER (%)		
	Vox1-O	Vox1-E	Vox1-H
FBank	1.01	1.24	2.32
ASR Encoder	1.159	1.256	2.434
wav2vec 2.0	0.973	0.933	1.831
HuBERT	0.84	0.879	1.726
PBERT	0.867	0.918	1.776

“ASR encoder” means we pre-train a CTC model with labeled train-960, and feed the ASR encoder outputs to the downstream model to obtain speaker embeddings.

Impairment Representation Learning for Speech Quality Assessment

- proposed an impairment representation learning approach to pre-train the network on a large amount of simulated data without MOS annotation. Then further fine-tune the pre-trained model for the MOS prediction task on annotated data.

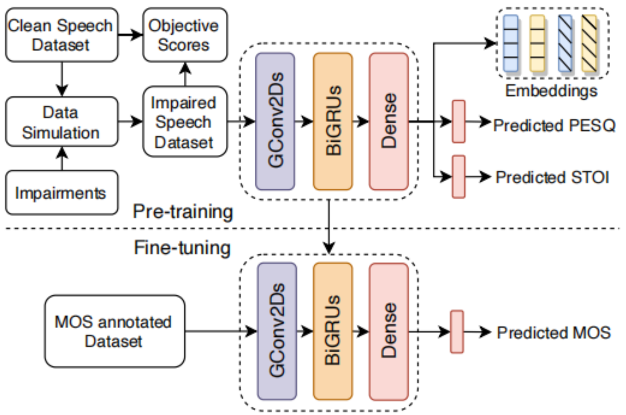


Figure 2: *The proposed two-stage system with pre-training for impairment representation learning and fine-tuning for MOS prediction.*

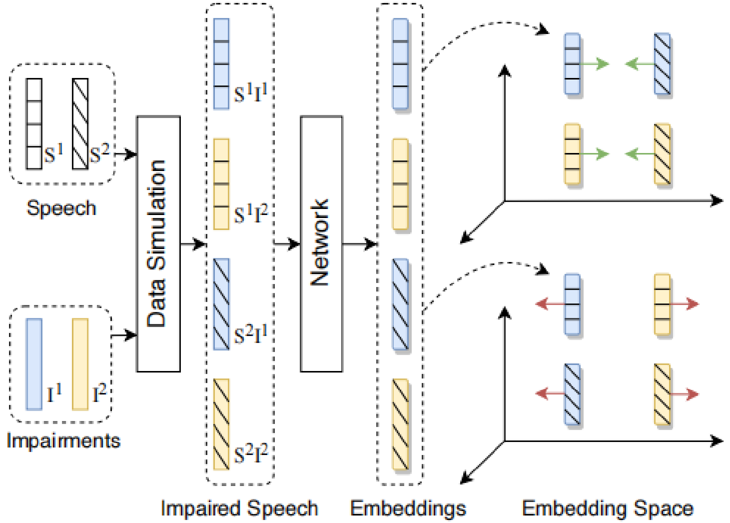


Figure 1: *The proposed impairment representation learning.*

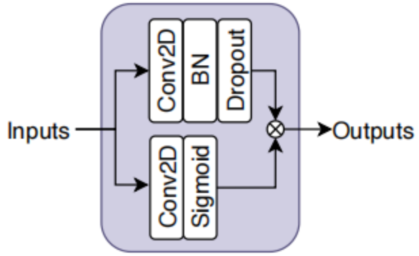


Figure 3: *The structure of GConv2D block.*

Table 1: *Hyper-parameters of the proposed neural network.*

Layer	CNN Channel	CNN Kernel	CNN Stride	RNN Units	FC Units
GConv2d^{1st}	16	(3, 3)	(1, 1)		
GConv2d^{2nd}	32	(3, 3)	(1, 2)		
GConv2d^{3rd}	64	(3, 5)	(1, 2)		
GConv2d^{4th}	128	(3, 7)	(1, 3)		
GConv2d^{5th}	256	(3, 3)	(1, 1)		
GConv2d^{6th}	512	(3, 1)	(1, 1)		
BiGRU^{1st}				128	
BiGRU^{2nd}				96	
BiGRU^{3rd}				64	
Dense^{1st}					96
Dense^{2nd}					P

- Pre-training

$$x_n = \frac{1}{2}(||F_\theta(S_n^1 I_n^1) - F_\theta(S_n^2 I_n^1)||_2 + ||F_\theta(S_n^1 I_n^2) - F_\theta(S_n^2 I_n^2)||_2)$$

$$x'_n = \frac{1}{2}(||F_\theta(S_n^1 I_n^1) - F_\theta(S_n^1 I_n^2)||_2 + ||F_\theta(S_n^2 I_n^1) - F_\theta(S_n^2 I_n^2)||_2)$$

$$\mathcal{L}_{all} = \mathcal{L}_{emb} + \mathcal{L}_{pesq} + \mathcal{L}_{stoi}$$

$$\mathcal{L}_{emb} = Y * X + (1 - Y) * maximum(1 - X, 0) \qquad X = [x_1, ..., x_N, x'_1, ..., x'_N] \qquad Y = [1, ..., 1, 0, ..., 0]$$

- Fine-tuning

$$\mathcal{L}_{MOS} = MSE(0.5 * M + 0.5 * r_1, M') \qquad M'_{fusion} = \sum_j^J \alpha_j * M'_j$$

r1 is the predicted MOS by the first trained model

Impairment Representation Learning for Speech Quality Assessment

- Pre-training dataset: LibriSpeech and ST Mandarin
- Fine-tuning dataset: Tencent Corpus, PSTN Corpus and NISQA Corpus.

- 'None' is the baseline system without pre-training.
- 'ICC' is the baseline pre-training method, a dense layer is added on the top of embedding layer to classify 4 impairment categories (noise, reverberation, device coloration and audio compression).
- 'CL' is the proposed pre-training method of Contrastive Learning (CL).
- 'CL+OSQP' is the proposed pre-training method with contrastive learning and Objective Speech Quality Prediction (OSQP)

Table 2: *RMSE of MOS prediction for different pre-training methods and different size of annotated fine-tuning dataset.*

PT-Method	FT-Size	Tencent	PSTN	All
None	1K	0.914	0.731	0.822
ICC		0.664	0.680	0.672
CL		0.614	0.644	0.629
CL+OSQP		**0.475**	**0.581**	**0.528**
None	5K	0.517	0.637	0.577
ICC		0.526	0.614	0.570
CL		0.466	0.610	0.538
CL+OSQP		**0.392**	**0.557**	**0.474**
None	120K	0.341	0.519	0.430
ICC		0.360	0.514	0.437
CL		0.334	0.514	0.424
CL+OSQP		**0.317**	**0.512**	**0.414**

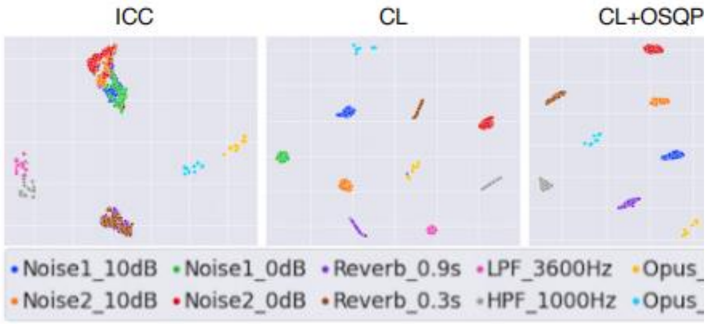


Figure 4: *A visualization of learned representations for* *pre-training methods.*

Table 3: *RMSE of MOS prediction with self-teaching loss and model fusion.*

PT-Method	FT-Size	DP	ST	Tencent	PSTN	All
CL+OSQP	120K	10s-z	0	0.317	0.512	0.414
CL+OSQP	120K	10s-z	1	0.309	0.509	0.409
CL+OSQP	120K	16s-r	0	0.326	0.518	0.422
CL+OSQP	120K	16s-r	1	0.324	0.512	0.418
Fusion				**0.293**	**0.503**	**0.398**

Table 4: *Results on ConferencingSpeech 2022 challenge.*

System	PLCC	RMSE	RMSE-Map
Baseline1	0.530	0.768	0.497
Fusion	**0.778**	**0.460**	**0.337**

Label-Efficient Self-Supervised Speaker Verification With Information Maximization and Contrastive Learning

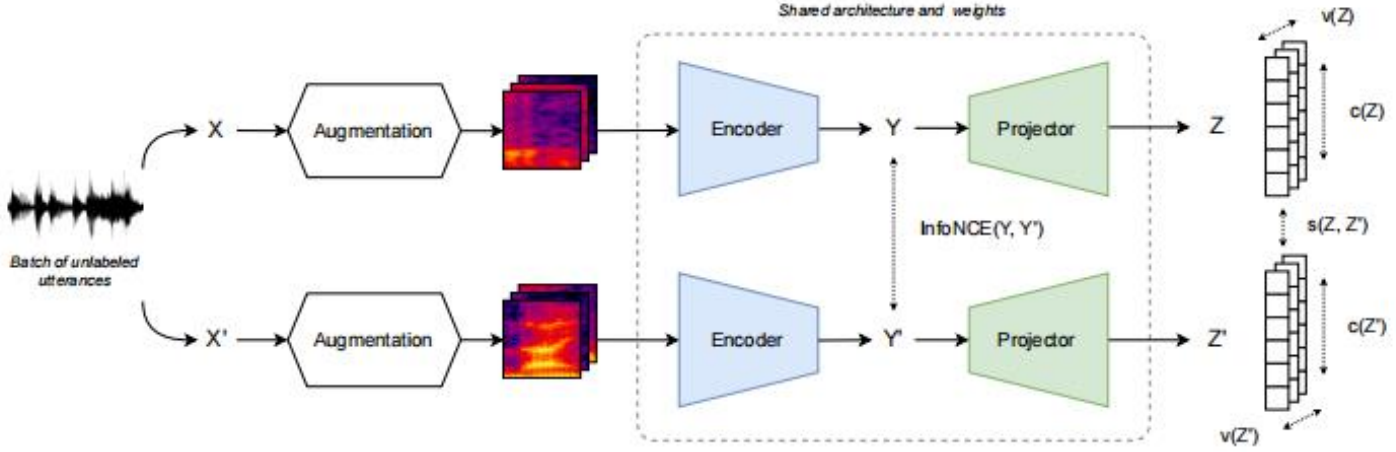


Figure 1: *Diagram of our self-supervised training framework.*

$$\mathcal{L}_{\text{InfoNCE}} = \frac{1}{N}\sum_{i=1}^{N} -\log \frac{\exp\left(\mathbf{z}_i \cdot \mathbf{z}_i'/\tau\right)}{\sum_{j=1}^{N} \exp\left(\mathbf{z}_i \cdot \mathbf{z}_j/\tau\right)} \quad (1)$$

- Barlow Twins
- The redundancy reduction term, by pushing all coefficients off-diagonal to be 0, decorrelates the different vector components and thus reduces the redundancy between them.

$$\mathcal{L}_{\text{BarlowTwins}} = \sum_{i}\left(1 - \left[\mathcal{C}\left(\mathbf{Z}, \mathbf{Z}'\right)\right]_{ii}\right)^2 + \lambda \sum_{i}\sum_{j \neq i}\left[\mathcal{C}\left(\mathbf{Z}, \mathbf{Z}'\right)\right]_{ij}^2 \quad (2)$$

- Variance-Invariance-Covariance Regularization

$$\mathcal{L}_{\text{VICReg}} = \lambda s\left(\mathbf{Z}, \mathbf{Z}'\right) + \mu\left(v(\mathbf{Z}) + v\left(\mathbf{Z}'\right)\right) + \nu\left(c(\mathbf{Z}) + c\left(\mathbf{Z}'\right)\right) \quad (3)$$

where λ, μ and ν are hyper-parameters to scale the *variance*, *invariance* and *covariance* terms. s, v and c represent the invariance, variance and covariance components, respectively.

$$s\left(\mathbf{Z}, \mathbf{Z}'\right) = \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{z}_i - \mathbf{z}_i'\right\|_2^2 \quad (5)$$

$$v(\mathbf{Z}) = \frac{1}{D}\sum_{j=1}^{D}\max\left(0, 1 - \sqrt{\operatorname{Var}(\mathbf{z}^j)}\right) \quad (4)$$

$$c(\mathbf{Z}) = \frac{1}{D}\sum_{i \neq j}[C(\mathbf{Z})]_{i,j}^2 \quad (6)$$

Label-Efficient Self-Supervised Speaker Verification With Information Maximization and Contrastive Learning

- Datasets: Voxceleb1 Encoder: Thin-ResNet34

- Exploring the complementarity of these methods

$$\mathcal{L}_{\text{comp}}^{1} = \mathcal{L}_{\text{VICReg}}\left(\mathbf{Y},\mathbf{Y}'\right) + \mathcal{L}_{\text{InfoNCE}}\left(\mathbf{Z},\mathbf{Z}'\right) \quad (7)$$

$$\mathcal{L}_{\text{comp}}^{2} = \mathcal{L}_{\text{InfoNCE}}\left(\mathbf{Y},\mathbf{Y}'\right) + \mathcal{L}_{\text{VICReg}}\left(\mathbf{Z},\mathbf{Z}'\right) \quad (8)$$

$$\mathcal{L}_{\text{reg}}^{\mathbf{Y}} = \mathcal{L}_{\text{InfoNCE}}\left(\mathbf{Y},\mathbf{Y}'\right) + \alpha\,\mathcal{L}_{\text{VICReg}}\left(\mathbf{Y},\mathbf{Y}'\right) \quad (9)$$

$$\mathcal{L}_{\text{reg}}^{\mathbf{Z}} = \mathcal{L}_{\text{InfoNCE}}\left(\mathbf{Z},\mathbf{Z}'\right) + \alpha\,\mathcal{L}_{\text{VICReg}}\left(\mathbf{Z},\mathbf{Z}'\right) \quad (10)$$

Table 1: *The performance of our self-supervised SV system when trained with different data augmentation strategies.*

Method	EER	minDCF
No augmentation	29.87	0.8833
Musan	21.22	0.8388
RIR	22.28	0.8525
Musan + RIR	**11.14**	**0.6843**

Table 2: *The impact of different scaling factors for VICReg loss components: λ (Invariance), μ (Variance) and ν (Covariance).*

λ	μ	ν	EER	minDCF
1	1	0	24.00	0.9964
1	0.5	0.1	15.71	0.8554
1	1	0.04	**11.14**	**0.6843**
1	1	0.1	11.87	0.7101

Table 3: *Effect of projector dimensionality (number of hidden and output units) on the performance of our self-supervised SV system.*

Architecture	EER	minDCF
No projector	14.96	0.9369
512	11.34	0.7826
1024	**10.77**	0.7208
2048	11.14	**0.6843**

hypothesize that the covariance mechanism benefits from a larger dimensionality to spread the information more efficiently.

Table 4: *Self-supervised SV results on VoxCeleb1 test set.*

Method	Loss	EER	minDCF
NPC [21]	Cross-entropy	15.54	0.8700
SimCLR [6]	InfoNCE	9.87	0.6760
Ours	$\mathcal{L}_{\text{InfoNCE}}$	10.42	**0.6276**
	$\mathcal{L}_{\text{BarlowTwins}}$	13.46	0.8473
	$\mathcal{L}_{\text{VICReg}}$	**9.25**	0.6432
Ours (Section 2.3)	$\mathcal{L}_{\text{comp}}^{1}$	13.14	0.6950
	$\mathcal{L}_{\text{comp}}^{2}$	**8.47**	**0.6400**
	$\mathcal{L}_{\text{reg}}^{\mathbf{Y}}$	9.09	0.6894
	$\mathcal{L}_{\text{reg}}^{\mathbf{Z}}$	10.38	0.6913

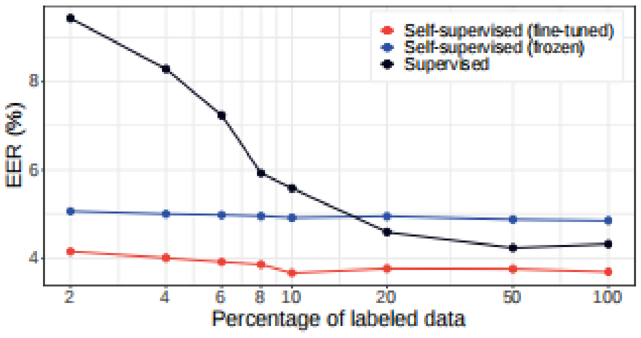


Figure 2: *Results on SV with different percentage of labeled data used during training.*