Advances in Speaker Recognition: Pruning, Deepfake Detection, and Learning without Temporal Labels

Themos Stafylakis
tstafylakis@aueb.gr
tstafylakis@omilia.com

Esperanto
Exchanges for SPEech ReseArch aNd TechnOlogies
Horizon 2020 project

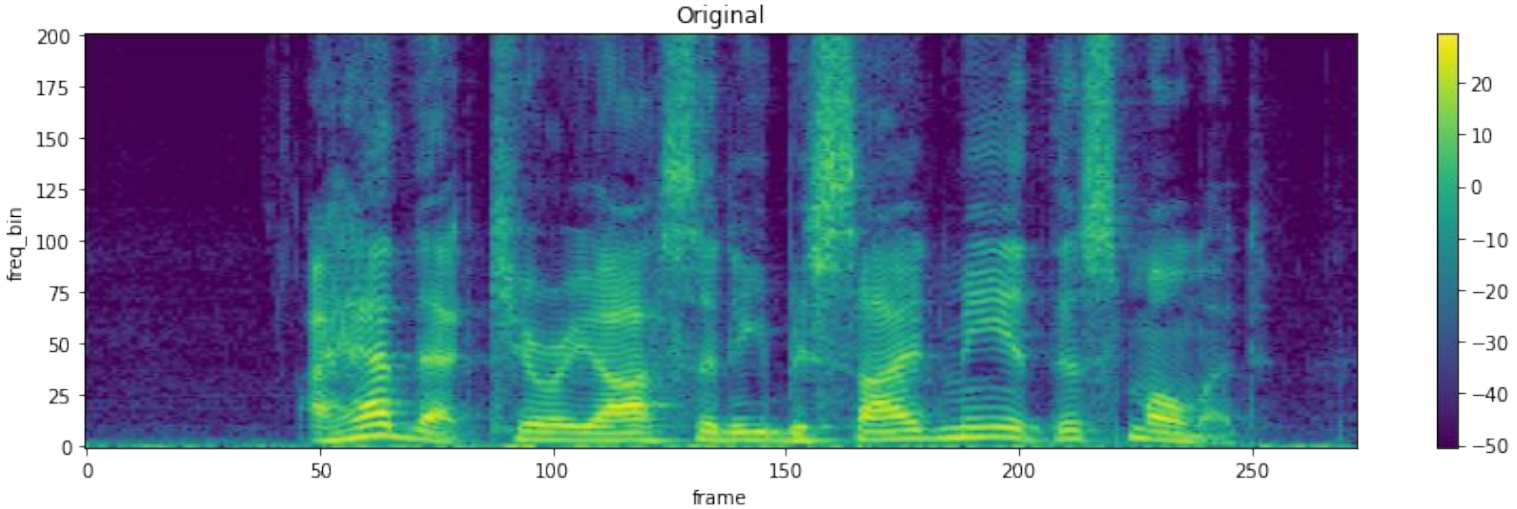
Content

1. Fundamental concepts of speaker recognition.
2. Methods for self-supervised learning (SSL) models for speaker recognition (and some anti-spoofing, aka deepfake detection).
3. Pruning SSL models while finetuning them for speaker recognition and anti-spoofing.
4. Training Speaker Embedding Extractors directly from the original uncut VoxCeleb recordings.

About speaker embeddings

- Speaker embeddings are low-dimensional vectors extracted from deep neural networks, aiming at capturing the unique voice characteristics of a person.
- As such, they should be invariant to
  - intrinsic variability (e.g. phonetic content, emotion, health conditions)
  - extrinsic variability (e.g. background noise, microphone, distance/angle from the mic)
- Speaker embeddings are ubiquitous in speech technology:
  - recognition tasks (e.g. speaker verification, diarization, and target speaker extraction)
  - generative tasks (e.g. text-to-speech synthesis, voice conversion, and speaker anonymization).
- In this talk, the focus will be on speaker verification:
  - Given two single-speaker recordings, estimate whether or not the speaker is the same in both of them.

Convolutional Neural Nets for speaker modeling



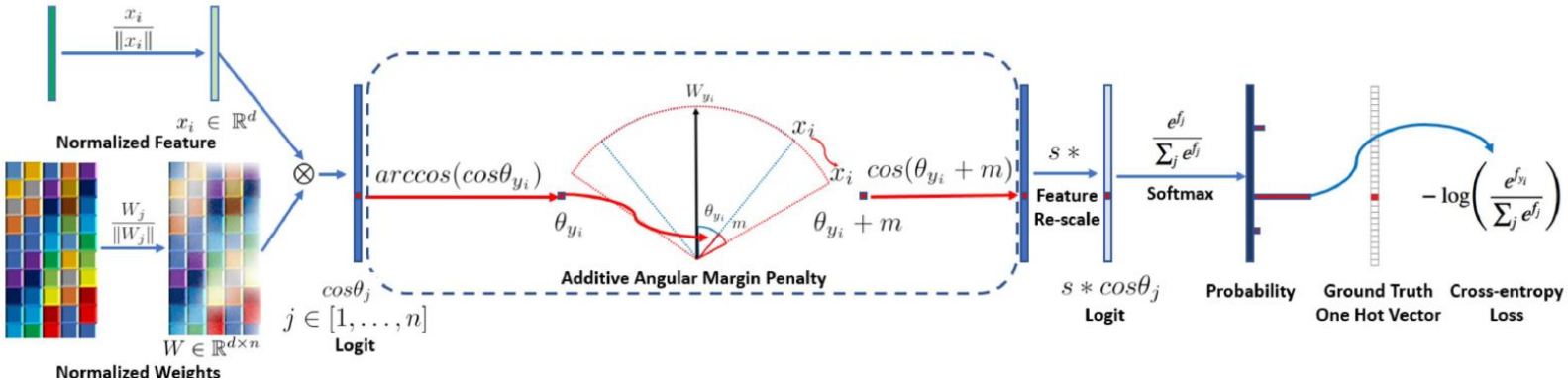
- In 1-D CNNs (TDNNs), the kernel is shifted only along the time axis.
  - The statistics pooling returns stats (e.g. mean and std) per channel.
- In 2-D CNNs (e.g. ResNets), the kernel is shifted along both the time and frequency axes.
  - Consider the log-spectrum as an image.
  - The statistics pooling returns stats per channel-frequency pair.
  - The number of frequency bins is progressively reduced via strides
- The speaker embedding is a compressed version of the statistics vector (e.g. 256-dim).

Angular-margin losses

$$L_{\text{softmax}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{\mathbf{W}_{y_i}^T\mathbf{x}_i+\mathbf{b}_{y_i}}}{\sum_{j=1}^{c}e^{\mathbf{W}_j^T\mathbf{x}_i+\mathbf{b}_j}}$$

$$L_{\text{AAM-Softmax}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i,i}+m))}}{Z}$$

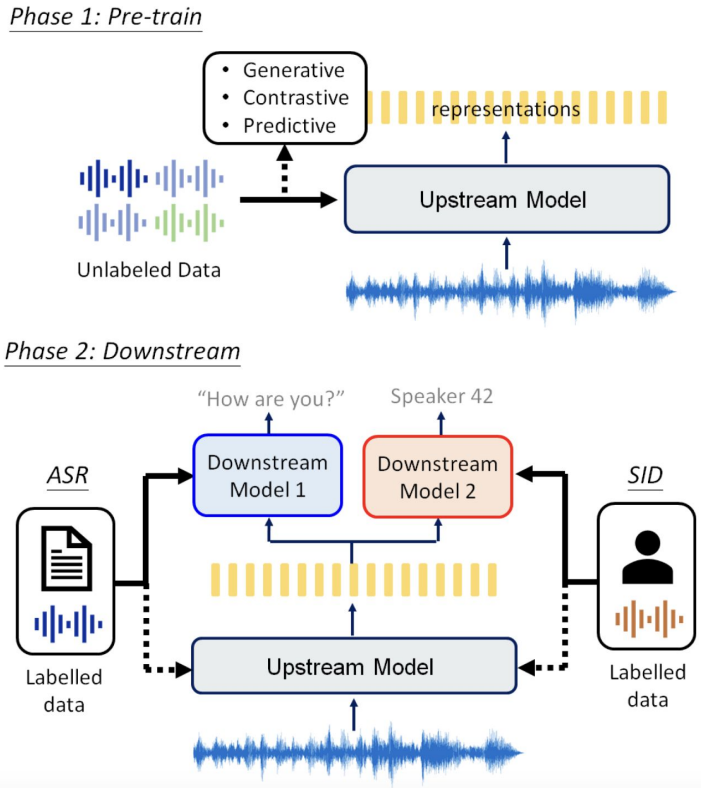
$$Z = e^{s(\cos(\theta_{y_i,i}+m))} + \sum_{j=1,j\neq i}^{c} e^{s(\cos(\theta_{j,i}))}$$



With AAM loss, the cosine similarity is typically enough, at least in certain datasets

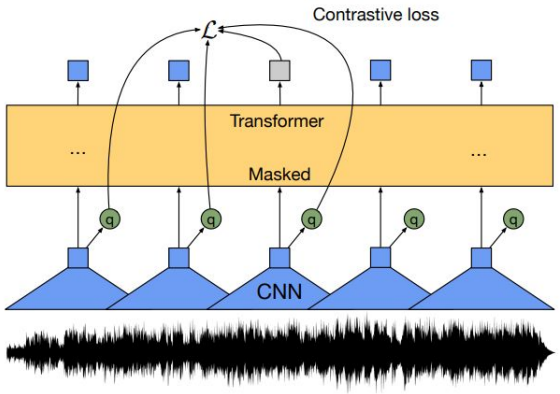
Deng, Jiankang, et al. "Arcface: Additive angular margin loss for deep face recognition." CVPR 2019.

SSL models in speech applications

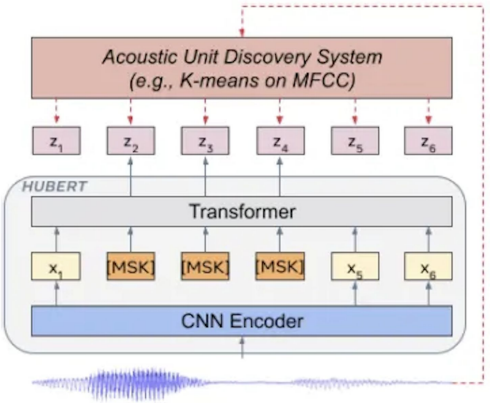


Mohamed, Abdelrahman, et al. "Self-supervised speech representation learning: A review." IEEE Journal of Selected Topics in Signal Processing 16.6 (2022): 1179-1210.

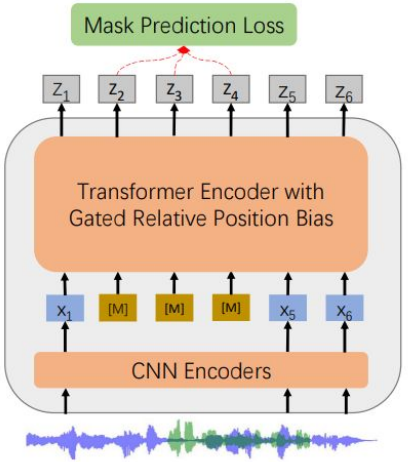
SSL models in speech applications



Wav2Vec 2.0



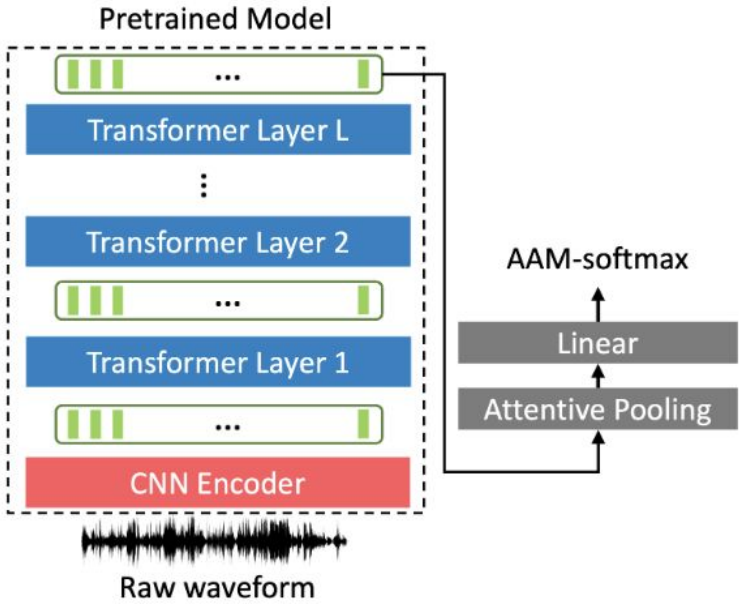
HuBERT



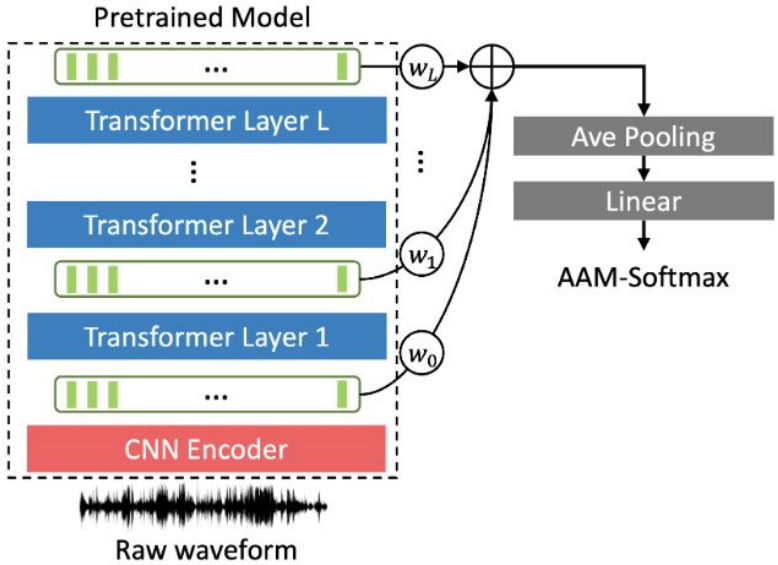
WavLM

Mohamed, Abdelrahman, et al. "Self-supervised speech representation learning: A review." IEEE Journal of Selected Topics in Signal Processing 16.6 (2022): 1179-1210.

Standard ways for pooling information from transformers



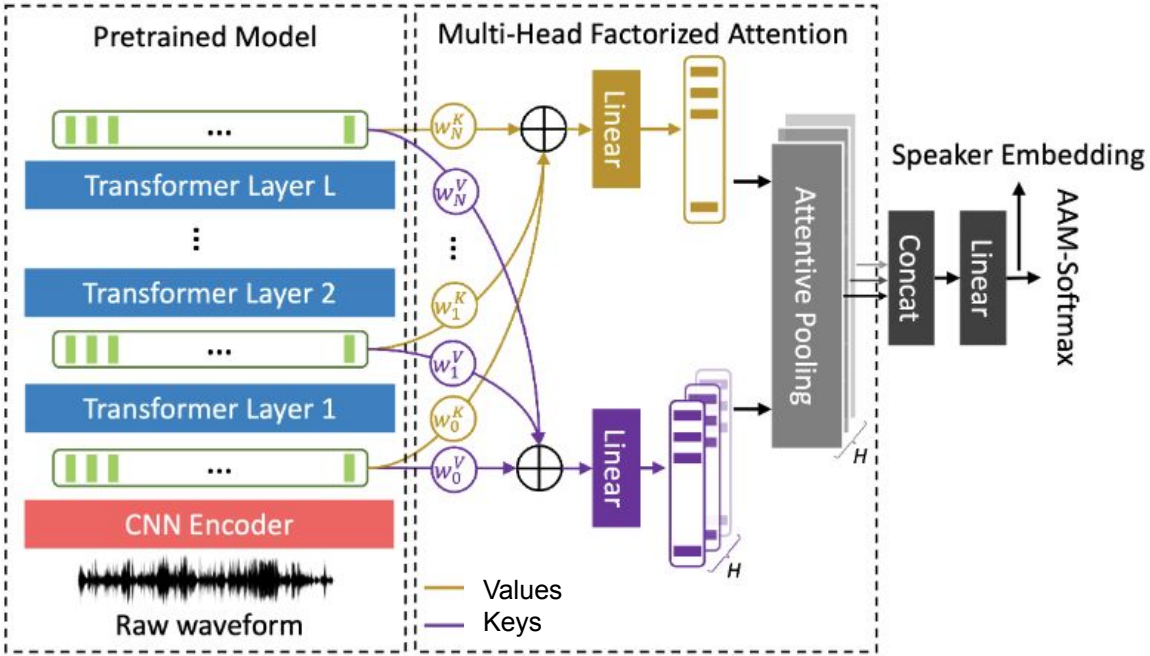
(a) Top Layer Attentive Pooling



(b) Layer-wise Weighted Average Pooling

Peng, Junyi, et al. "An attention-based backend allowing efficient fine-tuning of transformer models for speaker verification." SLT 2022

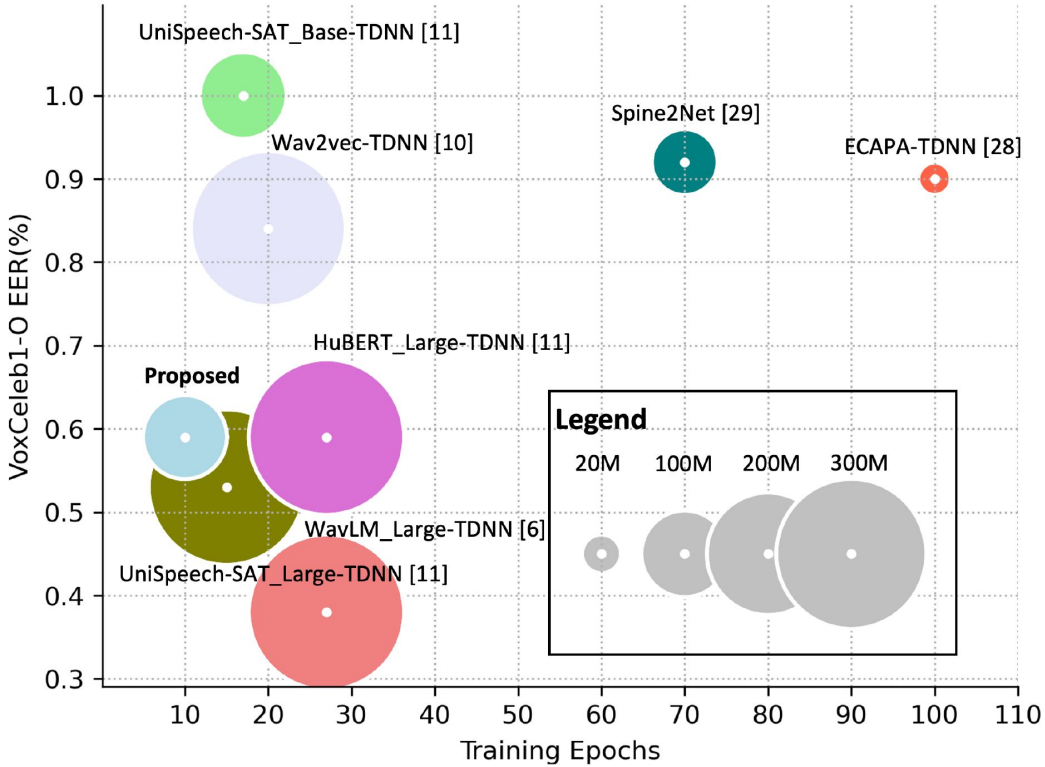
Pooling via multi-head factor attention (MHFA)



- Simple backend with multihead attention (64 heads).
- Each head models an acoustic area via a trainable query vector.
- Two different layer-pooling weights (keys and values).
  - Values should contain speaker information.
  - Keys should not.

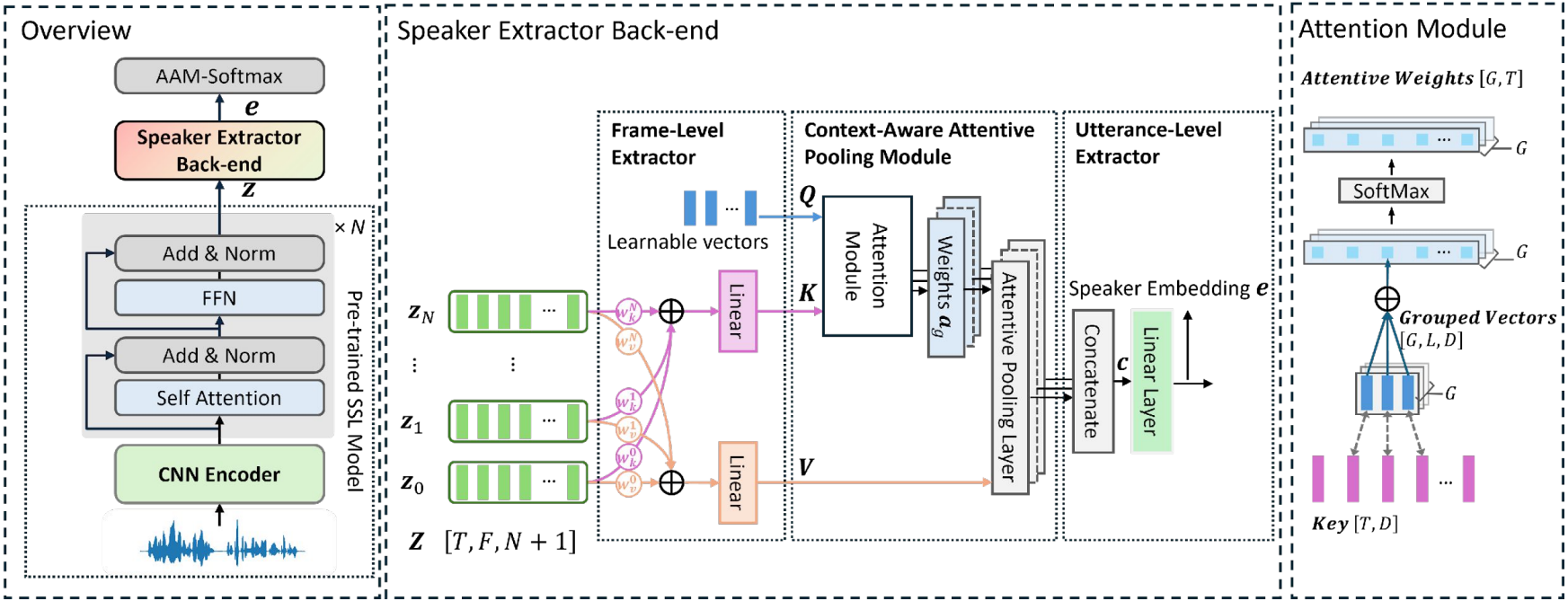
Peng, Junyi, et al. "An attention-based backend allowing efficient fine-tuning of transformer models for speaker verification." SLT 2022

Pooling via multi-head factor attention (MHFA)



Peng, Junyi, et al. "An attention-based backend allowing efficient fine-tuning of transformer models for speaker verification." SLT 2022

Context-aware MHFA - Adding 1D conv to the keys…



Peng, Junyi & Mošner, Ladislav & Zhang, Lin & Plchot, Oldřich & Stafylakis, Themos & Burget, Lukas & Černocký, Jan. (2025). CA-MHFA: A Context-Aware Multi-Head Factorized Attentive Pooling for SSL-Based Speaker Verification. 1-5. 10.1109/ICASSP 2025

Results-SUPERB Evaluation across different SSL Models

TABLE III

COMPARISON OF DIFFERENT FROZEN UPSTREAM MODELS ACROSS THREE DOWNSTREAM TASKS: SV, ER, AND DEEPFAKE DETECTION WITH VARYING BACK-END MODELS FOLLOWING SUPERB PRINCIPLE.

Upstream	SV EER(%)↓				Emotion Recognition ACC(%)↑			Deepfake Detection EER(%)↓		
	x-vector	ECAPA-TDNN	MHFA	CA-MHFA	MeanPooling [12]	MHFA	CA-MHFA	LLGF	MHFA	CA-MHFA
Wav2vec2.0 BASE	5.43	3.56	3.11	3.07	60.55	62.90	63.13	2.03	0.73	0.52
HuBERT BASE	5.65	3.29	2.78	2.82	60.73	62.35	63.96	2.39	1.19	1.22
Data2vec BASE	6.39	3.97	3.38	3.26	65.43	65.93	65.52	3.83	1.53	1.37
WavLM BASE	4.84	3.04	2.45	2.41	62.48	65.06	67.64	2.13	0.93	0.73
WavLM BASE Plus	4.10	2.67	1.87	1.79	66.72	66.58	66.45	3.64	0.20	0.30
Wav2vec2.0 Large	6.06	2.86	2.72	2.64	62.76	63.59	64.42	0.83	0.68	2.39
HuBERT Large	6.02	2.71	2.43	2.34	62.48	63.72	64.97	2.30	0.75	0.67
Data2vec Large	7.62	3.11	2.59	2.62	64.97	64.88	64.79	4.26	1.41	1.32
WavLM Large	4.87	2.17	1.78	1.77	67.92	69.72	71.52	1.54	2.23	1.21

- **VoxCeleb1:** CA-MHFA outperforms ECAPA-TDNN while using only ≈20% of its parameters.
- **IEMOCAP**: +2–3% absolute accuracy vs. baseline.
- **ASVspoof19**: consistently lower EER.
- Works across multiple upstream SSL models (HuBERT, Data2vec, WavLM).

Peng, Junyi & Mošner, Ladislav & Zhang, Lin & Plchot, Oldřich & Stafylakis, Themos & Burget, Lukas & Černocký, Jan. (2025). CA-MHFA: A Context-Aware Multi-Head Factorized Attentive Pooling for SSL-Based Speaker Verification. 1-5. 10.1109/ICASSP 2025

Results after SSL finetuning

Methods	FLOPs	#Param	Vox1-O EER	Vox1-E EER	Vox1-H EER
ECAPA-TDNN (ET.) [20]	1.04G	6.19M	0.90	1.11	2.32
ResNet221 [21]	21.29G	23.79M	0.50	0.67	1.21
ResNet293 [21]	28.10G	28.62M	0.44	0.65	1.18
WavLM_BASE_Plus + ET. [3]	-	94M + 6M	0.84	0.92	1.75
NEMO Large + MFA [22]	-	130M	0.48	0.71	1.54
Conformer + MHFA [10]	-	172M + 2.3 M	0.65	0.93	1.86
UniSpeech-SAT_Large + ET. [8]	-	316M + 6M	0.53	0.56	1.18
WavLM_Large + ET. [3]	-	316M + 6M	0.38	0.48	0.98
WavLM_Large + ET. [3]†	-	316M + 6M	0.41	0.55	1.11
WavLM_Large + MHFA [15]‡	25.79G	316M + 2.3M	0.55	0.59	1.24
WavLM_Base_Plus + CA-MHFA	11.05G	94M + 2.3M	0.70	0.72	1.45
+ LMF/QMF	11.05G	94M + 2.3M	0.59	0.65	1.30
WavLM_Large + CA-MHFA	25.79G	316M + 2.3M	0.55	0.62	1.18
+ LMF/QMF	25.79G	316M + 2.3M	0.42	0.48	0.96

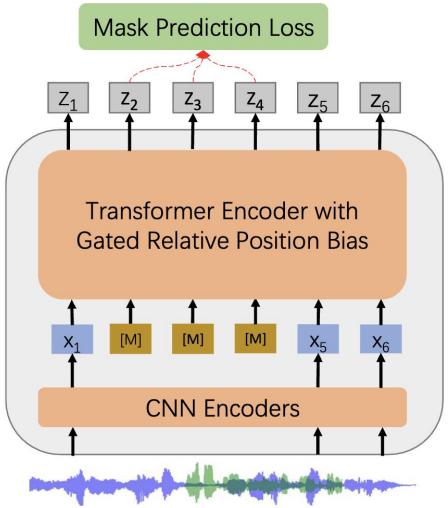
Peng, Junyi & Mošner, Ladislav & Zhang, Lin & Plchot, Oldřich & Stafylakis, Themos & Burget, Lukas & Černocký, Jan. (2025). CA-MHFA: A Context-Aware Multi-Head Factorized Attentive Pooling for SSL-Based Speaker Verification. 1-5. 10.1109/ICASSP 2025

Hybrid Pruning: In-situ Compression of Self-Supervised Speech Models for Speaker Verification and Anti-Spoofing

Self-Supervised Learning (SSL) models like **WavLM** are SOTA in speech processing tasks.

- They excel in speaker verification (SV) and anti-spoofing.
- However, their size is problematic
  - WavLM-Base has ~94 million parameters.
  - WavLM-Large has >300 million parameters.

This large size makes them difficult to deploy on **resource-constrained devices** like mobile phones or edge systems. Also, hard to do inference on CPU-nodes (high latency).



Peng, Junyi, et al. "Hybrid Pruning: In-Situ Compression of Self-Supervised Speech Models for Speaker Verification and Anti-Spoofing." arXiv preprint arXiv:2508.16232 (2025). Submitted to ICASSP 2026.

Existing Solutions & Their Limitations

Current model compression (pruning) methods are often complex and suboptimal. They typically follow a multi-stage pipeline:

1. Task-Agnostic Pruning: The model is pruned *during pre-training*, without knowledge of the final task (e.g., speaker verification).
2. Post-hoc Pruning: The model is first fully fine-tuned for a task, and then pruned as a separate, later step.

Key Drawback**:** This separation prevents the model from learning the best possible compressed architecture specifically for the end task.

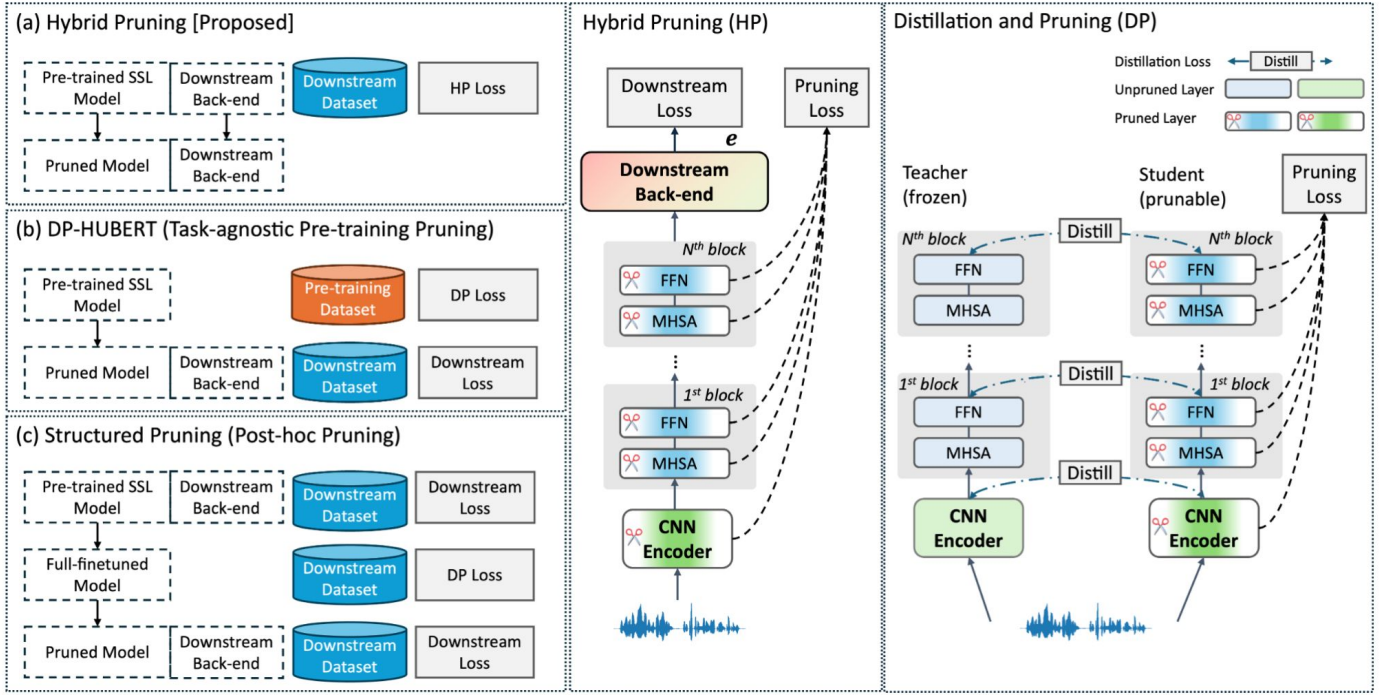
Peng, Junyi, et al. "Hybrid Pruning: In-Situ Compression of Self-Supervised Speech Models for Speaker Verification and Anti-Spoofing." arXiv preprint arXiv:2508.16232 (2025). Submitted to ICASSP 2026.

Existing Solutions & Their Limitations

Our single-stage approach jointly optimizes for the downstream task with the downstream back-end.

This avoids multi-stage pipelines like (b) task-agnostic pre-training pruning and (c) post-hoc pruning.

Notably, our method does not require the teacher-student knowledge distillation, common in other pruning techniques.



Peng, Junyi, et al. "Hybrid Pruning: In-Situ Compression of Self-Supervised Speech Models for Speaker Verification and Anti-Spoofing." arXiv preprint arXiv:2508.16232 (2025). Submitted to ICASSP 2026.

How HP Works: Joint Optimization

The key is in the loss function, which guides the model's learning.

$$\max_{\lambda_1,\lambda_2} \min_{\theta,\alpha} \left(\mathcal{L}_{\text{task}}(\theta) + \mathcal{R}_{\text{prune}}(\theta, \alpha, \lambda_1, \lambda_2)\right)$$

$$\mathcal{R}_{\text{prune}} = \mathbb{E}_{q(\theta|\bar{\theta},\alpha)}[\lambda_1(||\theta||_0 - t) + \lambda_2(||\theta||_0 - t)^2]$$

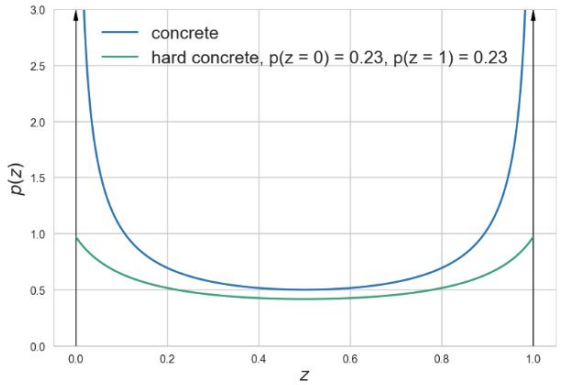
- L_{task} (Task Loss)**:** This is the standard loss for the downstream task (e.g., Binary Cross-Entropy for anti-spoofing). It pushes the model to be accurate.
- R_{prune} (Pruning Regularizer)**:** This term uses learnable "stochastic gates" to encourage parts of the model to be turned off. It pushes the model to be small and simple.

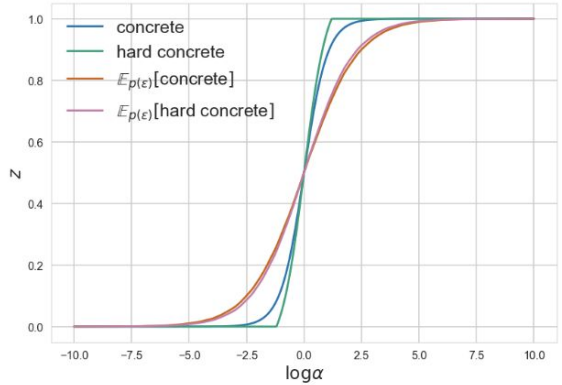
By optimizing both simultaneously, the model finds the best trade-off between accuracy and size.

Peng, Junyi, et al. "Hybrid Pruning: In-Situ Compression of Self-Supervised Speech Models for Speaker Verification and Anti-Spoofing." arXiv preprint arXiv:2508.16232 (2025). Submitted to ICASSP 2026.

Pruning based on the hard Concrete distribution

- We augment this model by inserting learnable stochastic gates at each of its prunable structural components (in CNN, MHSA and FFN).
- Each multiplicative gate is a continuous random variable $z_j \in [0, 1]$ following a distribution having delta peaks at 0 and 1 (hard concrete).
- z_j is the output of a nonlinear monotonic function (Sigmoid, affine and clamp(s,0,1)) of a trainable parameter α_j and a noise variable u (plus 3 other trainable global parameters).





C. Louizos, M. Welling, and D. P. Kingma, “Learning sparse neural networks through L0 regularization,” in ICLR 2018.

Pruning based on the hard Concrete distribution

- So the Hard Concrete is a continuous relaxation of the discrete Bernoulli distribution.
- The α_j's receive non-zero gradients also from the L_{task}, when z_j is not 0 or 1.
- This allows the network to learn, via gradient descent, to "turn off" (z_j=0) certain components.
- The CDF of the Hard Concrete is used to calculate R_{prune}, which pushes α_j towards smaller values.
- After training, z_j is determined by α_j and the other 3 trainable model parameters.

Key papers to dive deeper into the method

- Maddison, Chris J., Andriy Mnih, and Yee Whye Teh. "The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables," in ICLR 2017.
- C. Louizos, M. Welling, and D. P. Kingma, "Learning sparse neural networks through L0 regularization," in ICLR 2018.
- Z. Wang, J. Wohlwend, and T. Lei, "Structured pruning of large language models," in EMNLP 2020.

Compression & Performance in Speaker Verification

Model	Sparsity	#Param	Speedup		VoxCeleb		
			CPU	GPU	O	E	H
ResNet50 [30]		11.2 M	1.1×	2.6×	0.80	0.88	1.52
WavLM Base+	0%	95.6 M	-	-	0.70	0.69	1.40
	60%	38.9 M	2.2×	2.0×	0.70	0.78	1.50
	70%	29.5 M	2.9×	2.6×	0.73	0.84	1.61
	80%	19.9 M	3.8×	3.4×	0.92	1.02	1.91

- At 70% sparsity the model achieves EER = 1.61% on Vox1-H, close to the unpruned model's 1.40%.
- This results in a 2.6x speedup on a CPU and 2.9x on a GPU.

Peng, Junyi, et al. "Hybrid Pruning: In-Situ Compression of Self-Supervised Speech Models for Speaker Verification and Anti-Spoofing." arXiv preprint arXiv:2508.16232 (2025). Submitted to ICASSP 2026.

Compression & Performance in Deepfake Detection

Sparsity	Methods	#Param	FLOPs	Eval	
				EER (%)	minDCF
WavLM-SLIM (Best single system in ASVspoof5) [29]				5.16	0.149
0%	WavLM Base	95.6 M	57.4 G	4.56	0.116
10%	DP-HUBERT	86.2 M	48.0 G	5.13	0.139
	Structured Pruning	86.7 M	51.7 G	5.57	0.154
	Ours	86.0 M	51.9 G	**3.75**	**0.103**
30%	DP-HUBERT	67.3 M	36.4 G	7.23	0.200
	Structured Pruning	67.3 M	36.7 G	5.62	0.149
	Ours	67.3 M	39.1 G	**5.14**	**0.143**
50%	DP-HUBERT	48.5 M	25.8 G	11.73	0.321
	Structured Pruning	48.4 M	25.2 G	10.22	0.269
	Ours	48.4 M	26.9 G	**8.74**	**0.233**

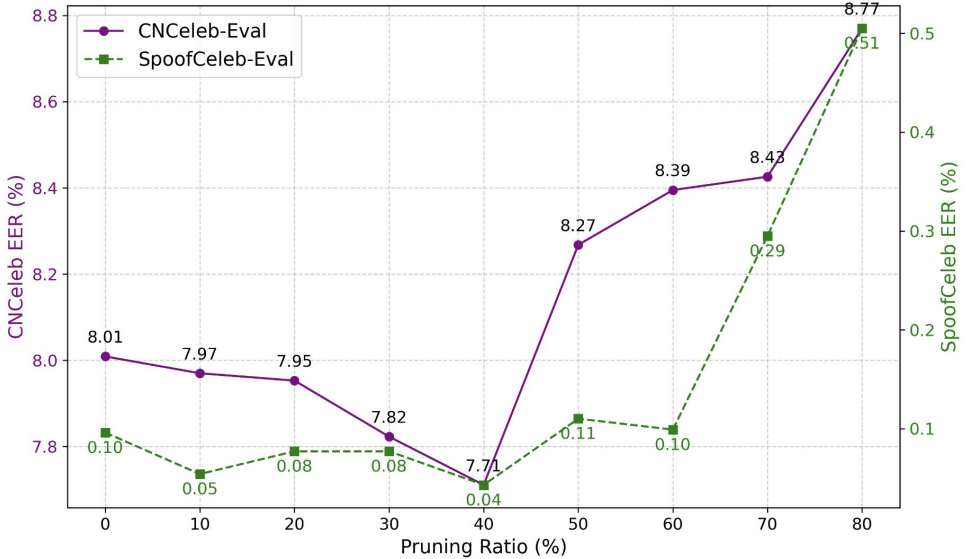
- The EER results in the ASVSpoof5 eval sets varied across different sparsity targets.
- FLOPs are calculated based on a 4-second audio input.

Peng, Junyi, et al. "Hybrid Pruning: In-Situ Compression of Self-Supervised Speech Models for Speaker Verification and Anti-Spoofing." arXiv preprint arXiv:2508.16232 (2025). Submitted to ICASSP 2026.

Pruning as a Regularizer

A interesting finding is that moderate pruning can improve generalization, especially on smaller datasets.

- On the CNCeleb (SV) and SpoofCeleb (anti-spoofing) datasets, the EER follows a "U-shaped" curve.



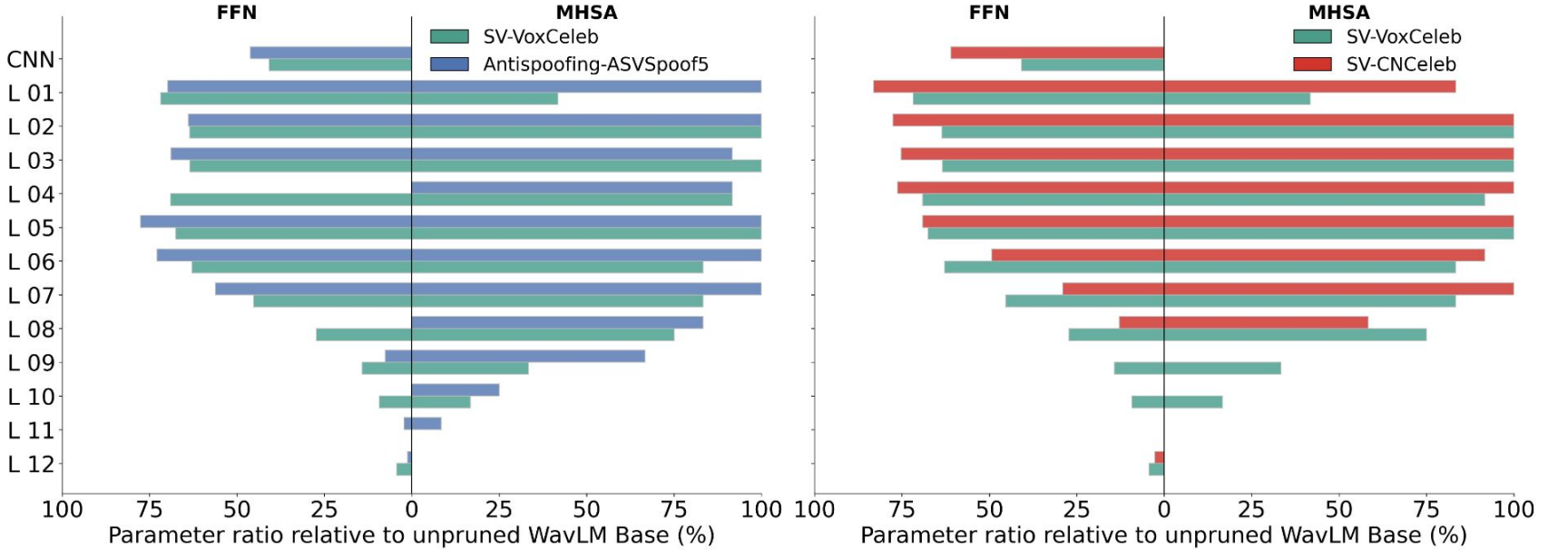
Why? Large SSL models can easily overfit to smaller downstream datasets.

By removing redundant parameters, HP reduces the model's capacity in a controlled way, forcing it to learn more robust and essential features.

Peng, Junyi, et al. "Hybrid Pruning: In-Situ Compression of Self-Supervised Speech Models for Speaker Verification and Anti-Spoofing." arXiv preprint arXiv:2508.16232 (2025). Submitted to ICASSP 2026.

Learning Task-Specific Architectures

HP learns specialized sub-architectures tailored to the task and domain.



(a) Task-Specific Pruning Patterns (SV vs. Antispoofing) (b) Domain-Specific Pruning Patterns (VoxCeleb vs. CNCeleb)

Peng, Junyi, et al. "Hybrid Pruning: In-Situ Compression of Self-Supervised Speech Models for Speaker Verification and Anti-Spoofing." arXiv preprint arXiv:2508.16232 (2025). Submitted to ICASSP 2026.

State-of-the-art Embeddings with Video-free Segmentation of the Source VoxCeleb Data

Sara Barahona, Ladislav Mošner, Themos Stafylakis, Oldřich Plchot, Junyi Peng, Lukáš Burget, Jan Černocký

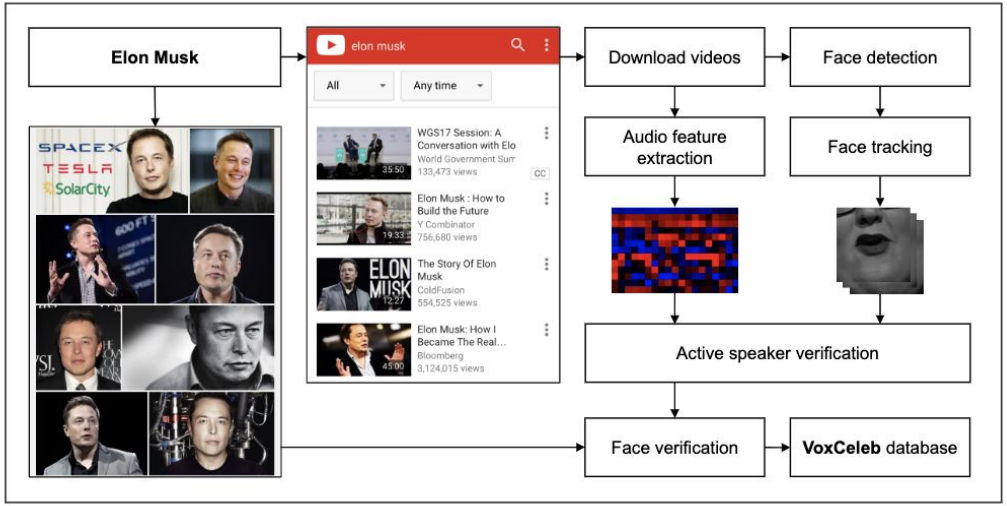
Follow-up work of:

Stafylakis, T., Mosner, L., Plchot, O., Rohdin, J., Silnova, A., Burget, L., & Černocký, J. (2022). Training speaker embedding extractors using multi-speaker audio with unknown speaker boundaries. In Proc. Interspeech 2022.

Motivation

How the VoxCeleb dataset was created:

- YouTube: search for celebrities
- ResNet50-based face recognizer
- SyncNet (active speaker verification)
- Minor human verification
- Precision maximized at the expense of recall



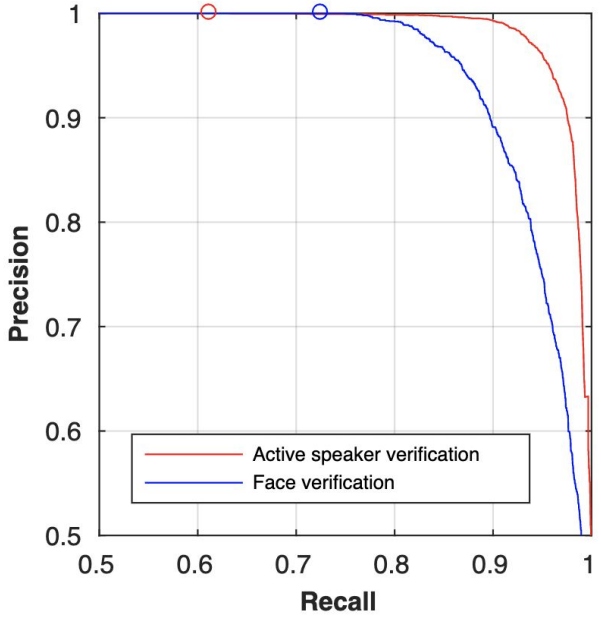
A. Nagrani, J. S. Chung, and A. Zisserman, “Voxceleb: A large-scale speaker identification dataset,” Interspeech 2017.

Weaknesses of the VoxCeleb pipeline

- Inevitable rejection of speech segments
  - When the celebrity's face does not appear
- Not applicable to speech-only databases
  - Telephone conversations
  - Radio broadcast

Task	Dataset	Precision	Recall
Active speaker verification	[31]	1.000	0.613
Face verification	[30]	1.000	0.726

Table 3: *Precision-recall values at the chosen operating points.*



A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: A large-scale speaker identification dataset," Interspeech 2017.

Weakly labeled data

- Can we detect the speech chunks where the celebrities are talking without the visual component or pretrained extractors?
- We assume we know who of the celebrities appear in each recording.
    - But we do not know when they are talking.
- Moreover, there are other people appearing in the recording (e.g. interviewers).
    - We assume no info about them.
- If we are able to detect them, then we can train a new extractor from scratch using these speech chunks.
- Maybe, the other speakers appearing in the recordings can be used to improve the embedding extractor.

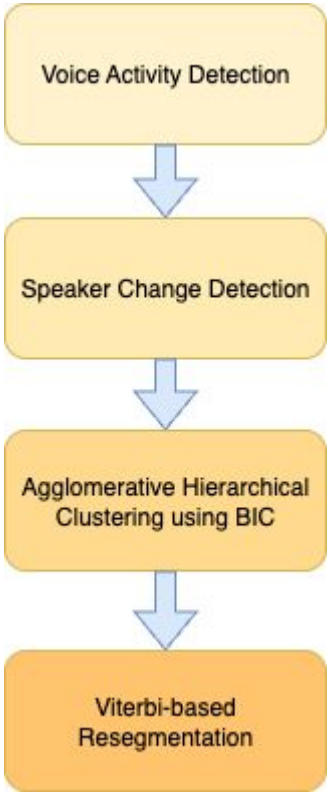
Method outline

Two stages

- Detect speech chunks of the target speakers (celebrities)
    - Basic speaker diarization algorithm (no pre-trained models used)
        - Assumption: clusters of high purity, low speaker coverage
    - Train a network with an aggregation function over clusters
    - Select chunks corresponding to the celebrity of interest
- Supervised training with the detected speech chunks
    - Train a new extractor from scratch using standard supervised training
        - Possibly, add chunks from other unknown speakers (as unknown class)

Basic speaker diarization

- Minimization of the need for annotation:
  - non-trainable diarization, no tuning
  - speaker change detection
  - BIC-based HAC with Gaussians on MFCCs
  - Viterbi-based boundary refinement (using GMMs)
- Over-estimation of the number of speakers
- SIDEKIT: https://projets-lium.univ-lemans.fr/s4d/



Steps of the baseline diarization algorithm

Training the embedding extractor with weak labels

- Denote the clusters or the recording by $X = [x_1, x_2, \ldots, x_C]$
- We want to maximize the probability of the target speaker $P(s^*|X)$
- We introduce a discrete latent variable $c \in 1, \ldots, C$ and marginalize it.

$$P(s^*|X) = \frac{P(s^*, X)}{P(X)} = \frac{\sum_c P(s^*, X, c)}{\sum_s \sum_c P(s, X, c)} = \frac{\frac{1}{C}\sum_c P(s^*, X|c)}{\frac{1}{C}\sum_s \sum_c P(s, X|c)} = \frac{\frac{1}{C}\sum_c P(s^*, x_c)}{\frac{1}{C}\sum_s \sum_c P(s, x_c)}$$

- where

$$P(s, X|c) = P(s, x_c) \quad P(c|X) = \frac{1}{C}$$

- We approximate the (log) joint $P(s, x_c)$ with the logit, i.e. the dot product between embedding of x_c and s-th speaker prototype:

$$l_s^c = h_s^T z_c \approx \log P(s, x_c)$$

Aggregating logits with log-sum-exp

- The posterior becomes:

$$P(s^*|X) = \frac{\frac{1}{C}\sum_c P(s^*, x_c)}{\frac{1}{C}\sum_s \sum_c P(s, x_c)} = \left[\text{softmax}\left(\log \sum_c P(s, x_c)\right)\right]_{s=s^*}$$

$$= \left[\text{softmax}\left(\log \sum_c \exp l_s^c\right)\right]_{s=s^*}$$

- As usual, we are training by minimizing the cross-entropy:

$$\mathcal{J}_{CE}(s^*, X; W) = -\log P(s^*|X)$$

- We call this aggregation (over clusters) method log-sum-exp (LSE) pooling.

Aggregating logits with max pooling

- We may also do temperature scaling

$$P(s^*|X,\tau) = \left[\text{softmax}\left(\tau \log \sum_c \exp \tau^{-1} l_s^c\right)\right]_{s=s^*}$$

- The limit for $\tau \to 0$ results in max pooling over clusters-specific logits:

$$P(s^*|X,\tau=0) = \left[\text{softmax}\left(\max_c l_s^c\right)\right]_{s=s^*}$$

- Max pooling is hard assignment, while LSE is soft assignment.
- In our latest set of experiments, max pooling outperformed LSE pooling.
- We didn't try max pooling followed by LSE (similar to Viterbi followed by Baum Welch).
- Note: in practice we are using a random segment xc of 4 sec instead of the whole cluster. A different augmentation configuration (noisy, reverberation) is sampled for each cluster.

What error signal are we backpropagating?

- Let's derive the gradients:

$$\frac{\partial \mathcal{J}}{\partial l_s^c} = P(c|s, X, \tau)\left(P(s|X, \tau) - \delta_{s,s^*}\right)$$

- where

$$P(c|s, X, \tau) = \left[\text{softmax}_c(\tau^{-1} l_s^c)\right]_c \xrightarrow{\tau \to 0} \left[\text{argmax}_c(l_s^c)\right]_c$$

Therefore,

- we estimate the posterior over speakers $P(s|X, \tau)$ using the chosen aggregation method.
- we backpropagate the error signal to clusters using $P(c|s, X, \tau)$ as weight
  - $P(c|s, X, \tau)$: posterior of cluster *c* given *s* and *X*.
- with max pooling, for each training speaker *s* (target and non-target) we are backpropagating only through the x_c that corresponds to the argmax cluster for *s*.

Chunk attribution to training speakers

- Once the embedding extractor is trained we need to attribute segments to target speakers.
- For each recording, we attribute a cluster to the target speaker *s** iff $\mathrm{argmax}_s l_s^c = s^*$
- In fact, we are doing it with chunks instead of clusters, to compensate for errors in diarization.

Dev. set	Celebrities	Recordings	Hours
Original	5,994	145,569	2,369.0
Restricted	5,987	110,940	1,884.3
Uncut	5,987	110,940	10,211.6
Self-labeled	5,987	110,453	6,026.9

Using non-target speaker segments

- How can we ensure that those remaining chunks are not from the target speaker?
  - We keep only chunks for which the target speaker logit in not within the top-5 logits.
- Defining a speaker prototype for the unknown class (containing numerous speakers) seems weird.
  - Especially when AAM loss is used.

Using non-target speaker segments

- We work as if there was a speaker prototype for this specific unknown speaker:
  - We do not create any additional speaker prototype.
  - In each minibatch, we calculate the average target-speaker logit from the chunks having a known target speaker (how well embeddings and corresponding prototypes match on average).
  - For chunks of the unknown class, we set the “target” logit equal to the average target-speaker logit.
- This should result in:
  - increasing the distance between prototypes and unknown speaker embeddings
  - strengthening the speaker discriminability of the extractor.

Experimental setup

- **Embedding extractor**
  - ResNet34: (64, 128, 256, 256) channels in ResNet stages
  - Instance (instead of batch) normalization
  - 400 frames, 80-dim fbanks
  - 256-dim speaker embeddings
- **Aggregation**
  - Max pooling or LSE
- **Training on estimated speech chunks**
  - The same architecture is used (without the aggregation)
  - Higher margin in AAM loss (m=0.3)

Strong vs weak supervision - ResNet34

Supervision	Data	Diarization	*m*	Agg./τ	Vox1-O EER	Vox1-E EER	Vox1-H EER
Strong	Restr.		↑0.3	–	1.11	1.13	2.07
Weak	Uncut	S4D	0.0	Max / –	3.00	2.86	4.60
Weak	Uncut	S4D	0.1	LSE / 0.5	4.33	6.39	9.99
Weak	Uncut	S4D	0.1	LSE / ↓0.1	4.53	4.86	7.31
Weak	Uncut	Pyannote	0.0	Max / –	2.55	2.37	3.93

- The weakly supervised networks appear not competitive
  - However, their purpose is not to generalize well to new speakers
  - Their purpose is to find the speech chunks of the target speakers in the training set
- Training without margin at this stage improves max-pooling performance
- SOTA diarization (Pyannote) provides a moderate improvement

Strong vs weak supervision - WavLM + MHFA

Supervision	Data	*m*	Agg./τ	Vox1-O EER	Vox1-E EER	Vox1-H EER
Strong	Restricted	↑0.3	–	0.86	0.86	1.78
Weak	Uncut	0.0	Max / –	1.51	1.76	3.36
Weak	Uncut	0.0	LSE / ↓0.1	3.59	3.10	5.60
Weak	Uncut	0.1	Max / –	**1.25**	**1.36**	**2.55**
Weak	Uncut	0.1	LSE / ↓0.1	3.82	3.71	5.86

- Diarization with S4D
- Performance much closer to strong-supervision training
- Max-pooling achieves the best results

Training with self-labeled data

Model	Data	Vox1-O		Vox1-E		Vox1-H	
		EER	M.DCF	EER	M.DCF	EER	M.DCF
ResNet34	Restricted	1.11	0.064	1.13	0.073	2.07	0.121
	Self-labeled	**0.99**	**0.054**	**1.02**	**0.065**	**1.83**	**0.107**
WavLM + MHFA	Restricted	0.86	0.065	**0.86**	**0.057**	**1.78**	**0.111**
	Self-labeled	0.83	0.062	0.92	0.062	1.91	0.121
	+ Unknown class	**0.83**	**0.061**	0.92	0.058	1.86	0.117

- ResNet34: training with self-labels outperforms training on VoxCeleb cuts
- WavLM + MHFA: top accuracy with VoxCeleb cuts, but self-labeled very close
- Adding the “unknown” class with unlabeled chunks further improves performance

Conclusions & Future Directions

- **Conclusions**
  - We proposed an approach to training an embedding extractor using weakly labeled data
  - We examined two aggregation methods
  - We achieved results close or better to supervised training with original VoxCeleb cuts
- **Future Directions**
  - Attention-based aggregation
  - Include recordings with >1 celebrity (straightforward)
  - Alternative chunk selection policies
  - Apply one more iteration:
    - redo everything using the trained extractor as starting point

Thanks!
Happy to hear your thoughts!

Themos Stafylakis
tstafylakis@aueb.gr
tstafylakis@omilia.com

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Esperanto
Exchanges for SPEech ReseArch aNd TechnOlogies
Horizon 2020 project

Omilia
Conversational Intelligence

ARCHIMEDES

ATHENA
Research & Innovation Information Technologies

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY