Tencent 腾讯

Tencent AI Lab

# Recent progress of self-supervised speaker representation learning at Tencent AI Lab

Chunlei Zhang

Tencent AI Lab

05/28/2022

# Outline

- **Introduction**
  - SSL in speech processing, what and why?
  - Generative/predictive learning (frame-level)
  - Contrastive learning
  - Non-contrastive learning
- **Speaker embedding training with SSL**
  - MoCo speaker embedding and class-collision corrections (C3)
  - DINO speaker embedding and C3-DINO
  - Experimental results and discussion
- **Others**
  - Disentangled speech representation learning
  - Applications

# SSL in speech processing, what and why?

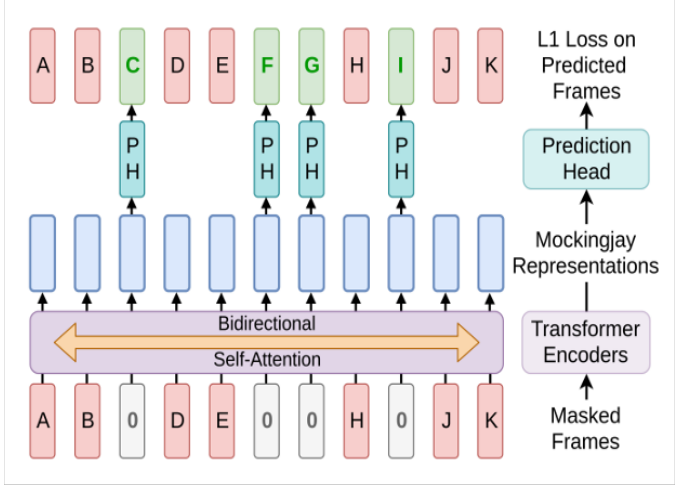
**Self-supervised learning:**

A form of unsupervised leaning where data itself provides the supervision. SSL usually defines a proxy loss, which is directly or indirectly connected with the downstream tasks.

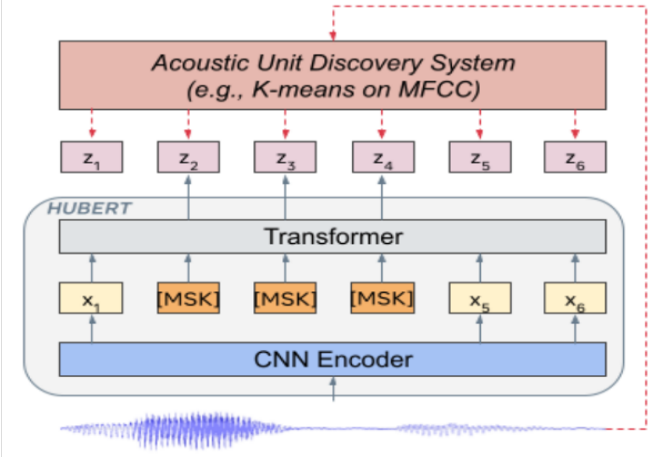
**Why SSL ?**

- Fully supervised learning frameworks oftentimes are **domain-specific** and expensive to build.
- Vast numbers of unlabeled (in-domain) data are available, a small portion of labeled data; however, it is difficult to directly apply them to the supervised frameworks.
- And both machine and human annotators introduce their systematic errors/bias into the labels, which revent the supervised models from achieving the optimal performance

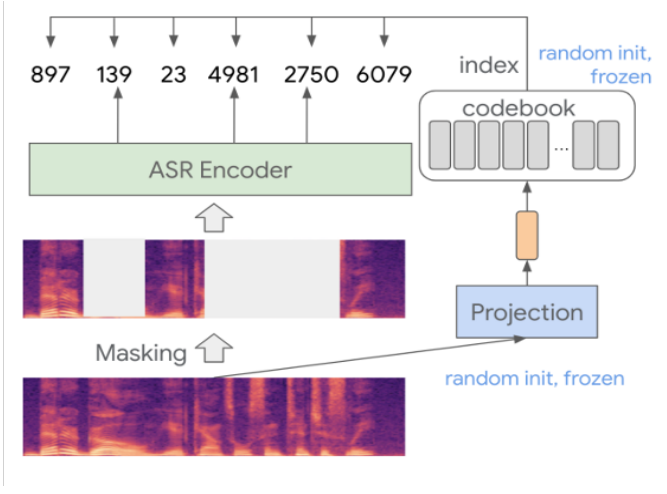
# Generative/predictive learning (frame-level)




(a) Mockingjay [Liu et.al. 2020]



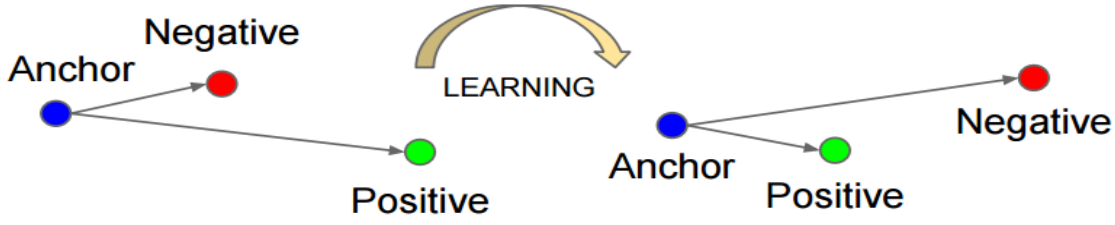

(b) HuBERT [Hsu et.al. 2021]




(c) BEST-RQ [Chiu et.al. 2022]

# Contrastive learning (instance-level)

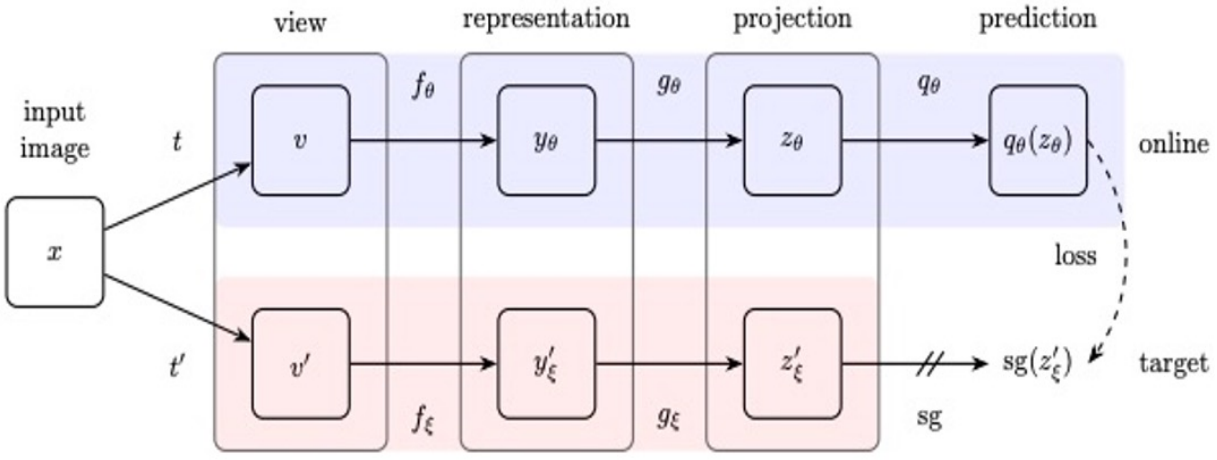
General objective is, after training:



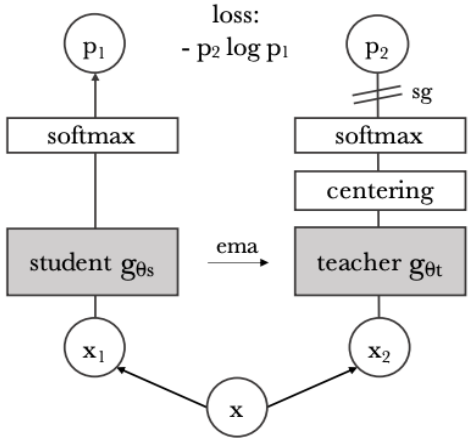

- Supervised contrastive losses: triplet loss [Zhang et.al. 2017], GE2E [Wan et.al. 2018], SupCon [Khosla et.al. 2020]
- Self-supervised contrastive loss: InfoNCE loss [Oord et.al. 2018]
- Discriminating positive and negative examples, MoCo [He et al., 2020], SimCLR [Chen et al., 2020]

$$\mathcal{L}_N = -\mathbb{E}_X \left[ \log \frac{\exp\left(f(x)^T f(x^+)\right)}{\exp\left(f(x)^T f(x^+)\right) + \sum_{j=1}^{N-1} \exp\left(f(x)^T f(x_j)\right)} \right]$$

# Non-contrastive learning (instance-level)



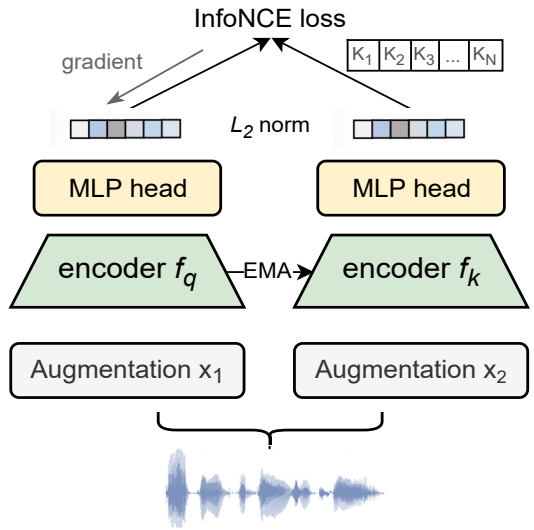

(a) BYOL [Grill et.al. 2020]




(b) DINO [Caron et.al. 2021]

- BYOL links to Mockingjay, BYOL's latent "Target" is from a momentum encoder (teacher), while Mockingjay's "Target" is the raw feature
- DINO is more like HuBERT and BEST-RQ, DINO's "Target" is from the distribution sharping of the teacher output, HuBERT and BEST-RQ is from a tokenizer (either iterative k-means index or random projection index)

# MoCo speaker embedding




a) MoCo speaker embedding system

- A data augmentation module that transforms a data example into two different views, which formulates a positive pair.
- A neural encoder module that encodes the input features into a fixed latent space.
- A MLP projection head that maps the latent representation to a $L_2$-normalized embedding (i.e., speaker embedding) where a contrastive loss is applied.

$$\mathcal{L}_{MoCo} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp\left(\mathbf{v}_{qi}\cdot\mathbf{v}'_{ki}/\tau\right)}{\exp\left(\mathbf{v}_{qi}\cdot\mathbf{v}'_{ki}/\tau\right)+\sum_{j=1}^{K}\exp\left(\mathbf{v}_{qi}\cdot\mathbf{v}_{kj}/\tau\right)}, \tag{1}$$

# Class-Collision Correction (C3) for MoCo systems

MoCo's negative samples in the memory queue are randomly sampled. It may contain wrong anchor, negative pairs – **class-collision** problem.
**Proposal 1**: Re-weighted InfoNCE loss

$$\mathcal{L}_i = -\log \frac{\exp\left(\mathbf{v}_{qi} \cdot \mathbf{v}'_{ki}/\tau\right)}{\exp\left(\mathbf{v}_{qi} \cdot \mathbf{v}'_{ki}/\tau\right) + \sum_{j=1}^{K} \exp\left(\mathbf{v}_{qi} \cdot \mathbf{v}_{kj}/\tau\right)}, \tag{2}$$

When $\mathcal{L}_i$ satisfies two conditions:

$$\mathbf{v}_{qi} \cdot \mathbf{v}_{kj} > 0.8 \times \mathbf{v}_{qi} \cdot \mathbf{v}'_{ki}, j \in [1, K], \tag{3}$$

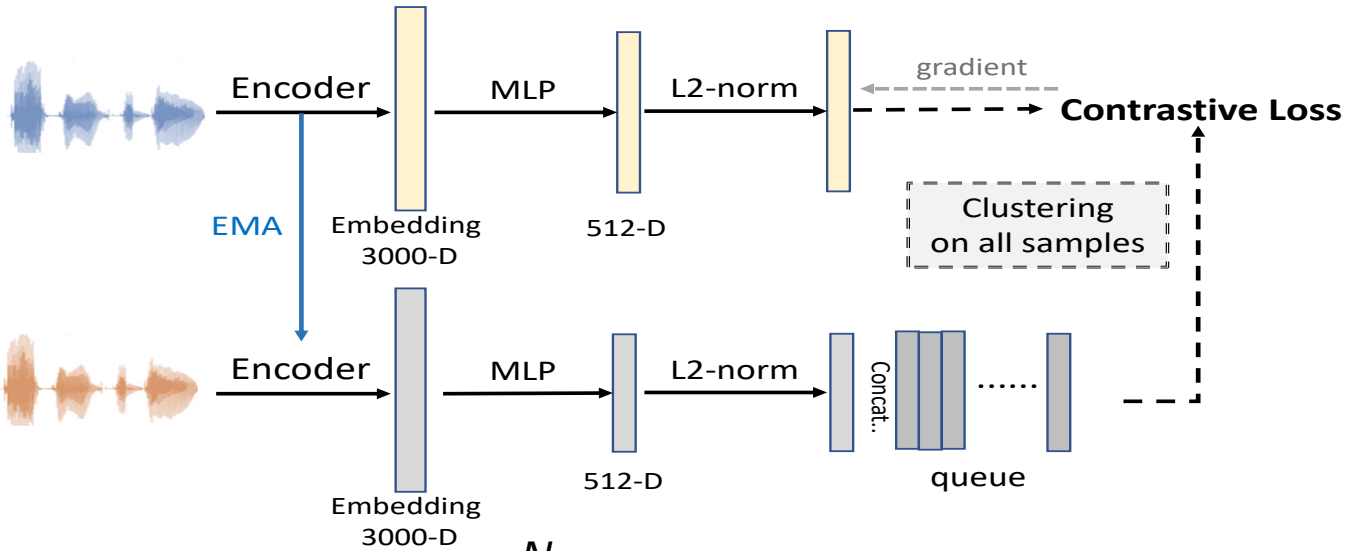
$$\mathbf{v}_{qi} \cdot \mathbf{v}'_{ki} > 0.4, \tag{4}$$

We can "predict" $\mathcal{L}_i$ contains "false negatives", and re-weight the MoCo loss $\mathcal{L}_{MoCo}$ as:

$$\mathcal{L}_{re\_MoCo} = \omega_1 \mathcal{L}^{tn}_{MoCo} + \omega_2 \mathcal{L}^{fn}_{MoCo}, \tag{5}$$

# Class-Collision Correction (C3) for MoCo systems

**Proposal 2**: ProtoNCE loss

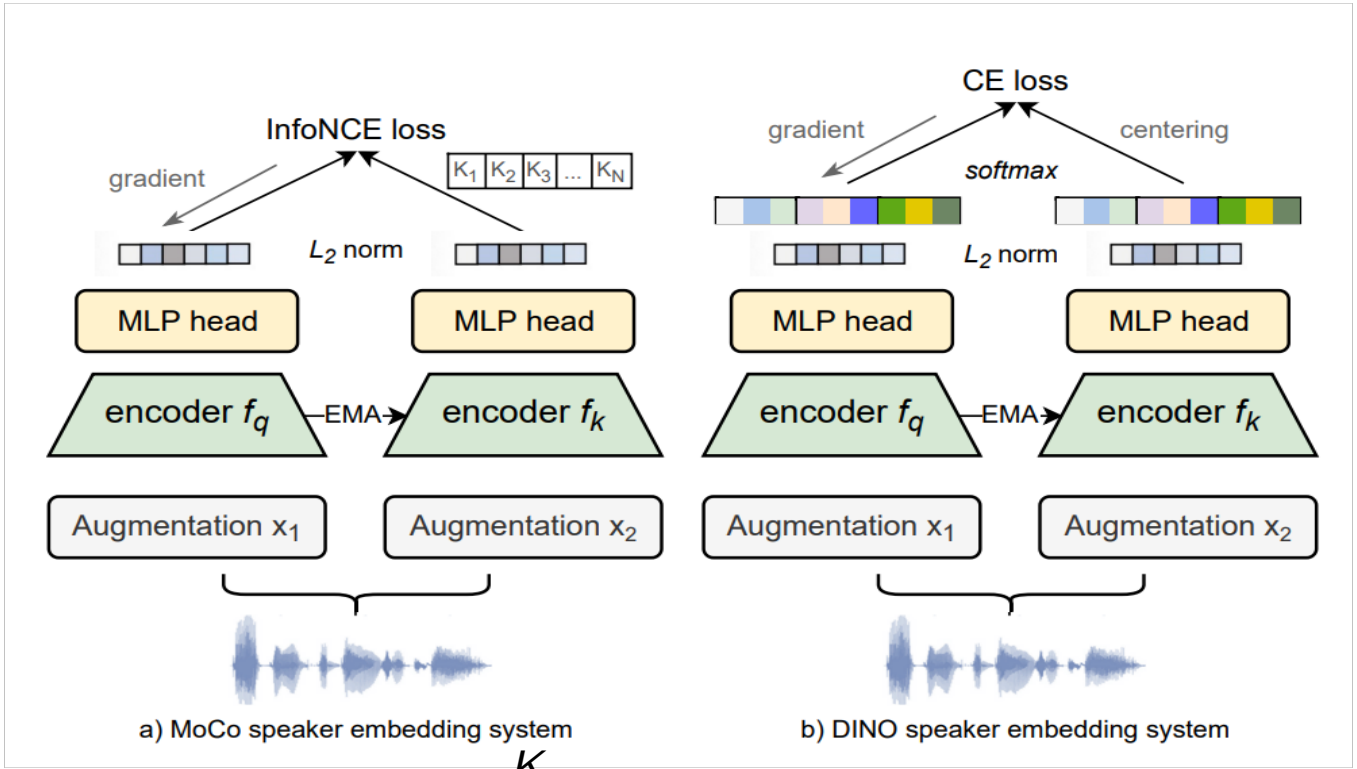



$$\mathcal{L}_{pNCE} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp\left(\mathbf{v}_{qi}\cdot\mathbf{c}_s/\phi_s\right)}{\sum_{j=0}^{R}\exp\left(\mathbf{v}_{qi}\cdot\mathbf{c}_j/\phi_j\right)}, \tag{6}$$

$$\mathcal{L}_{C3\_MoCo} = \mathcal{L}_{re\_MoCo} + \alpha\mathcal{L}_{pNCE}, \tag{7}$$

- Introduce a prototype memory bank. Perform clustering on all samples (with cluster IDs) and only sample negatives prototypes from different classes. Positives are from the same class prototypes.
- Dynamically estimate the cluster density for each epoch
- Later, extended to queue based ProtoNCE loss [Peng et .al. 2022]

# DINO speaker embedding




$$\mathcal{H}(X, Y) = -\sum_{i=1}^{K} p((x_i - c_i)/\tau_t) \log q(y_i/\tau_s), \qquad (8)$$

- A softmax layer with **K** dimension is added after the embedding layer in each branch
- Centering: $C = mC + (1-m)\frac{1}{B}\sum_{i=1}^{B} X^i$, representing the EMA of the batch centers ($m = 0.99$)
- $\tau_t$ and $\tau_s$ is the teacher/student branch temperature coefficient.

# DINO speaker embedding

$\tau$ is the key that determines the direction of the self-distillation process.

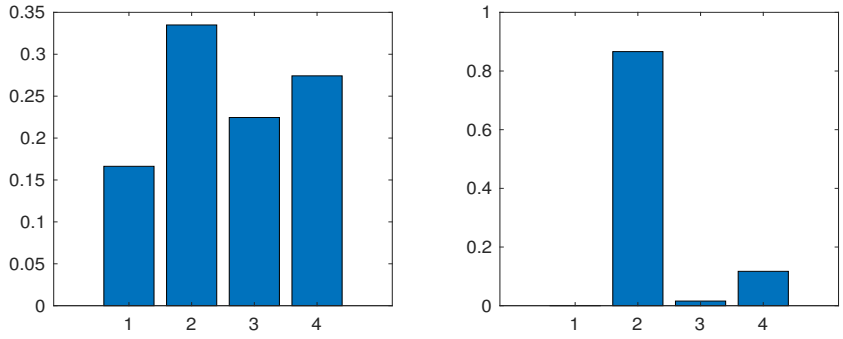



Figure 1: A $\tau$=0.1 applied to logits [0.2 0.9 0.5 0.7]

Avoid training collapse:

- Without centering, the training may end up with one position (class) dominates for all samples.
- Without distribution sharping, the training may end up with a flatten distribution for all samples
- K is important, the initial center C is important

# A multi-stage C3-DINO training framework

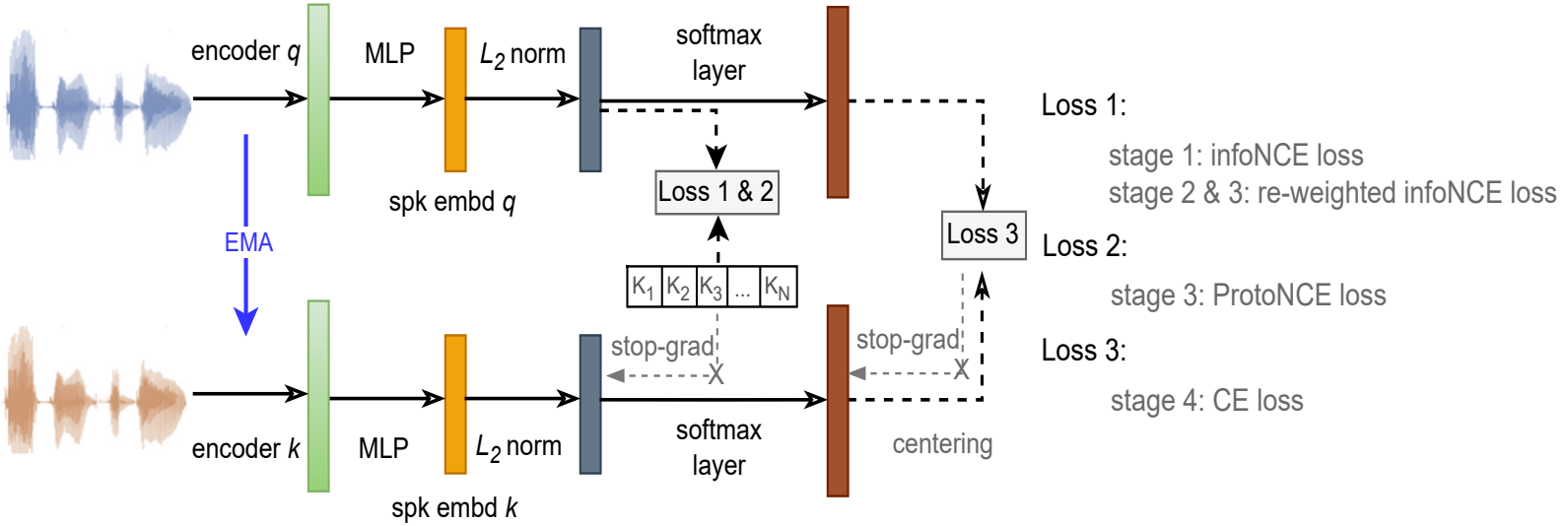



Figure 2: A multi-stage training framework.

- For the C3-MoCo training, the first stage is trained the infoNCE loss.
- Then, replace the InfoNCE loss with the re-weighted InfoNCE loss for the second stage, and jointly train with the ProtoNCE loss for stage 3.
- Finally, using the pretrained C3-MoCo model as the initial, and apply the DINO loss.

# Experimental results

**Baseline MoCos and DINOs:**

Table 1: The SV performances of MoCo and DINO baseline systems w.r.t. three main stream architectures on Voxceleb1 test set. We use R34L and E-TDNN to represent ResNetSE34L and ECAPA-TDNN.

| Training Data | | **EER**(in%) | | | **minDCF** | | |
|---|---|---|---|---|---|---|---|
| | | TDNN | R34L | E-TDNN | TDNN | R34L | E-TDNN |
| Vox1 | MoCo | 11.3 | 10.8 | **9.8** | 0.76 | 0.72 | **0.67** |
| Vox2 | MoCo | 8.5 | 8.3 | **7.3** | 0.63 | 0.62 | **0.61** |
| Vox1 | DINO | 7.2 | 6.8 | **6.1** | 0.76 | 0.72 | **0.52** |
| Vox2 | DINO | 5.6 | 4.6 | **4.0** | 0.51 | 0.50 | **0.48** |

- K=10000, encoders and **C** are randomly initialed

# Experimental results

**Impact of K in the DINO systems:**

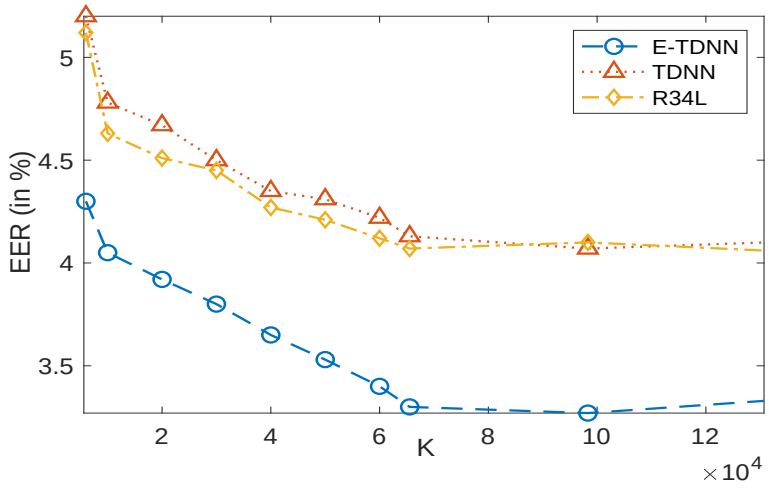



Figure 3: The SV performance (EER) of the DINO speaker embedding systems w.r.t. different dimension $K$. Specifically, $K$ is [6000, 10000, 20000, 30000, 40000, 50000, 60000, 65536, 98304, 131072].

- The best DINO baseline system with ECAPA-TDNN achieves 3.3% EER.

# Experimental results

**Progressive system development:**

Table 2: Progressive results with the proposed techniques (EER in% ), reported with the best ECAPA-TDNN model.

| system | method | Vox1 test | VoxSRC21 Dev |
|---|---|---|---|
| 1 | MoCo | 7.3 | 17.5 |
| 2 | + re-weighted infoNCE | 6.9 | 16.9 |
| 3 | ++ProtoNCE (C3-MoCo) | **6.4** | 16.4 |
| 4 | DINO | 3.3 | 12.7 |
| 5 | C3-DINO$_1$ (sys.3+4 multi-task) | 3.4 | 13.2 |
| 6 | C3-DINO$_2$ (sys. 1 initial) | 2.9 | 12.5 |
| 7 | C3-DINO$_2$ (sys. 2 initial) | 2.7 | 12.4 |
| 8 | C3-DINO$_2$ (sys. 3 initial) | **2.5** | **12.2** |
| 9 | C3-DINO$_2$ (sys. 4 initial) | 3.0 | 12.7 |

- Better MoCo initial results in better DINO system
- Multi-task with MoCo and DINO does not outperform the single objective DINO system

# Experimental results

Table 3: Ablation study on Vox1 test set (EER in%).

| supervision | method | Vox1 test |
|---|---|---|
| self-sup | i-vector [Huh et.al. 2020] | 15.3 |
| self-sup | AP+AAT[Huh et.al. 2020] | 8.6 |
| self-sup | MoCo+ProtoNCE[Xia et.al. 2022] | 8.2 |
| self-sup | MoCo (E-TDNN) [ID Lab 2020] | 7.3 |
| self-sup | Siaseme+SSR [Sang et.al. 2021] | 7.0 |
| self-sup | C3-MoCo (system 3) | **6.4** |
| self-sup | DINO [Cho et.al. 2021] | 4.8 |
| self-sup | DINO [Heo et.al. 2022] | 4.8 |
| self-sup | C3-DINO$_2$ (system 8) | **2.5** |
| self-sup | C3-DINO$_2$ (system 8 + LDA) | **2.2** |
| semi-sup | GCL+PLDA (15% label)[Inoue et.al. 2020] | 6.0 |
| semi-sup | MoCo+SupCon (15% label) [Xia et.al. 2020] | 4.3 |
| sup | x-vector [Snyder et.al. 2018] | 3.1 |
| sup | ECAPA-TDNN [ID Lab et.al. 2020] | **0.9** |

- Our results represent SOTA for both contrastive and non-contrastive SSL speaker systems.

# Open questions

**1. Self-training on SSL embeddings instead of on the raw features?**

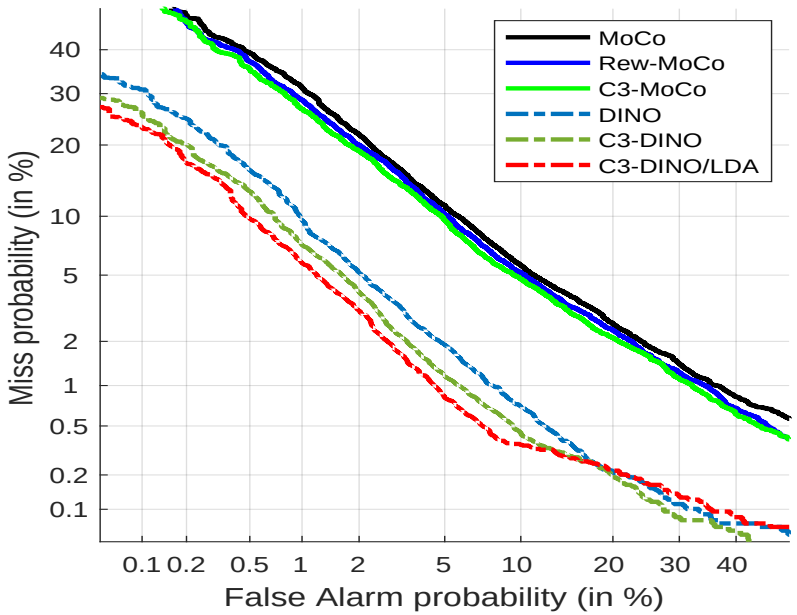



Figure 4: DET curves of the proposed SV systems on Vox1 test set.

- Pseudo-labeling/iterative training works well for Voxceleb task, but not necessary good for a much larger scale data with unknown speaker number prior information?
- Train a large scale SSL initial embedding, and build "supervised" back-end classifiers are more efficient and proper?
- Initial SSL embedding are sufficiently good and back-end classifiers are working great again [Brummer et.al 2022, Wang et.al. 2022].

# Open question

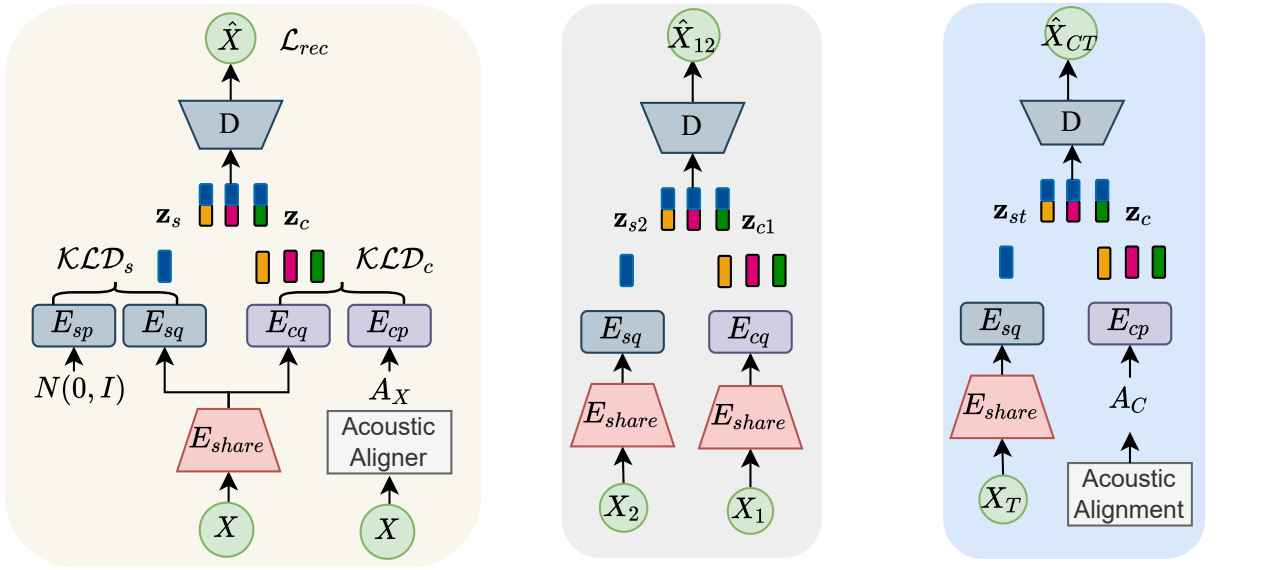
**2. DINO is all your need in SSL speaker embedding training ?**

Table 4: The SV performance (EER in %) conditioned on different scales of training data. With Tencent in-house training and test data.

| method | 2k | 4k | 6k | 8k | 10k | 12k |
|---|---|---|---|---|---|---|
| MoCo | 8.5 | 7.5 | 6.8 | 6.3 | 5.7 | 5.4 |
| DINO | 6.3 | 5.4 | 4.9 | 4.5 | 4.4 | 4.3 |
| SupCon | 5.2 | 4.6 | 4.2 | 3.9 | 3.7 | 3.6 |

- Looks like DINO is still better than MoCo on all conditions.
- MoCo does not seem to be saturating.

# Others – Disentangled representation learning




a) C-DSVAE training b) voice conversion c) alignment driven voice generation

Figure 5: The system diagram of conditional DSVAE.

# Applications

| model | content embd | | spk embd |
|---|---|---|---|
| | Phn ACC % | SV EER % | SV EER % |
| DSVAE [Lian et.al. 2022] | 30.5 | 40.5 | 4.8 |
| C-DSVAE(UA) | 53.6 | 43.2 | 3.9 |
| C-DSVAE(+MP) | 68.2 | 41.6 | 4.1 |
| C-DSVAE(++LibriTTS) | 72.1 | 44.5 | 3.6 |
| C-DSVAE(FA) | 54.6 | 43.4 | 3.8 |
| C-DSVAE(+MP) | 70.5 | 42.3 | 3.6 |
| C-DSVAE(++LibriTTS) | 73.3 | 45.6 | 3.3 |

Table 5: Objective analysis of content embeddings and speaker embeddings.

- Speaker embedding for speaker verification, voice conversion, zero-shot voice cloning in TTS
- ASR pretraining, like HuBERT, WavLM and BEST-RQ etc.
- Voice conversion and TTS.

# Reference


## SSL for speaker embedding:

- Wei Xia, Chunlei Zhang, Chao Weng, Meng Yu, and Dong Yu. "Self-supervised Text-independent Speaker Verification using Prototypical Momentum Contrastive Learning." In IEEE ICASSP 2021.
- Junyi Peng, Chunlei Zhang, Jan "Honza" Černocký, Dong Yu. "Progressive contrastive learning for self-supervised text-independent speaker verification". In ISCA Odyssey 2022.
- Chunlei Zhang, Dong Yu. "State-of-the-Art Self-Supervised Learning Based Speaker Verification". Submitted to IEEE J. Sel. Topics Signal Process., 2022.

## Disentangled speech representation learning:

- *Jiachen Lian, *Chunlei Zhang, Gopala K. Anumanchipalli and Dong Yu". UTTS: Unsupervised TTS with Conditional Disentangled Sequential Variational Auto-encoder". Submitted to NeurIPS 2022. (*Equal contribution).
- *Jiachen Lian, *Chunlei Zhang, Gopala K. Anumanchipalli and Dong Yu". Towards Improved Zero-shot Voice Conversion with Conditional DSVAE". Submitted to ISCA Interspeech 2022. (*Equal contribution).
- Jiachen Lian, Chunlei Zhang, Dong Yu. "Robust Disentangled Variational Speech Representation Learning for Zero-shot Voice Conversion". In IEEE ICASSP 2022.

Thanks for your attention!