Discriminative Training of VBx Diarization

Dominik Klement, Mireia Diez, Federico Landini, Lukáš Burget, Anna Silnova
Marc Delcroix, Naohiro Tawara

Diarization

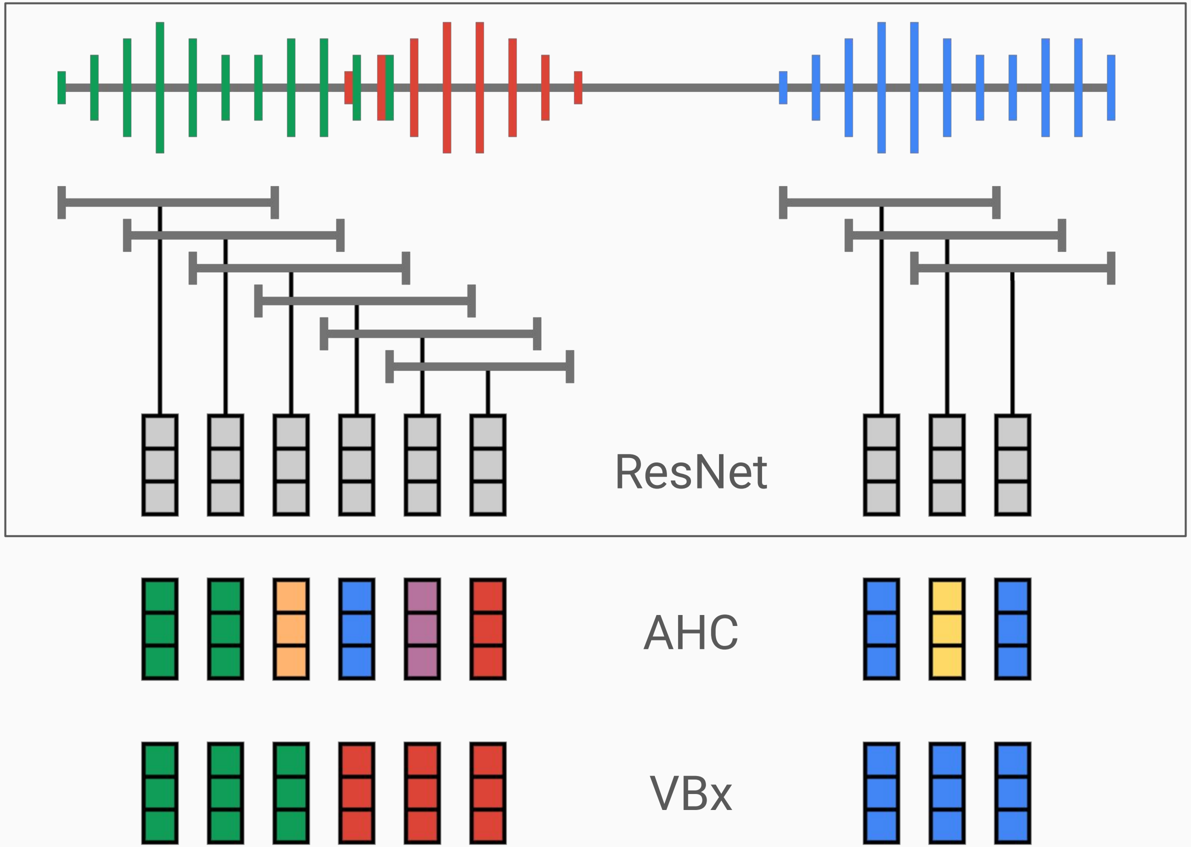
- Methods:
  - **Clustering-based**
  - End-to-end Neural Diarization (EEND)
  - Hybrid (combination of the previous ones)
- Currently dominated by EEND
- EEND Problems:
  - Trained on huge amount of simulated data
  - May not even model speaker information implicitly
  - Problems with determining the correct # of speakers

Why Clustering-based VBx?

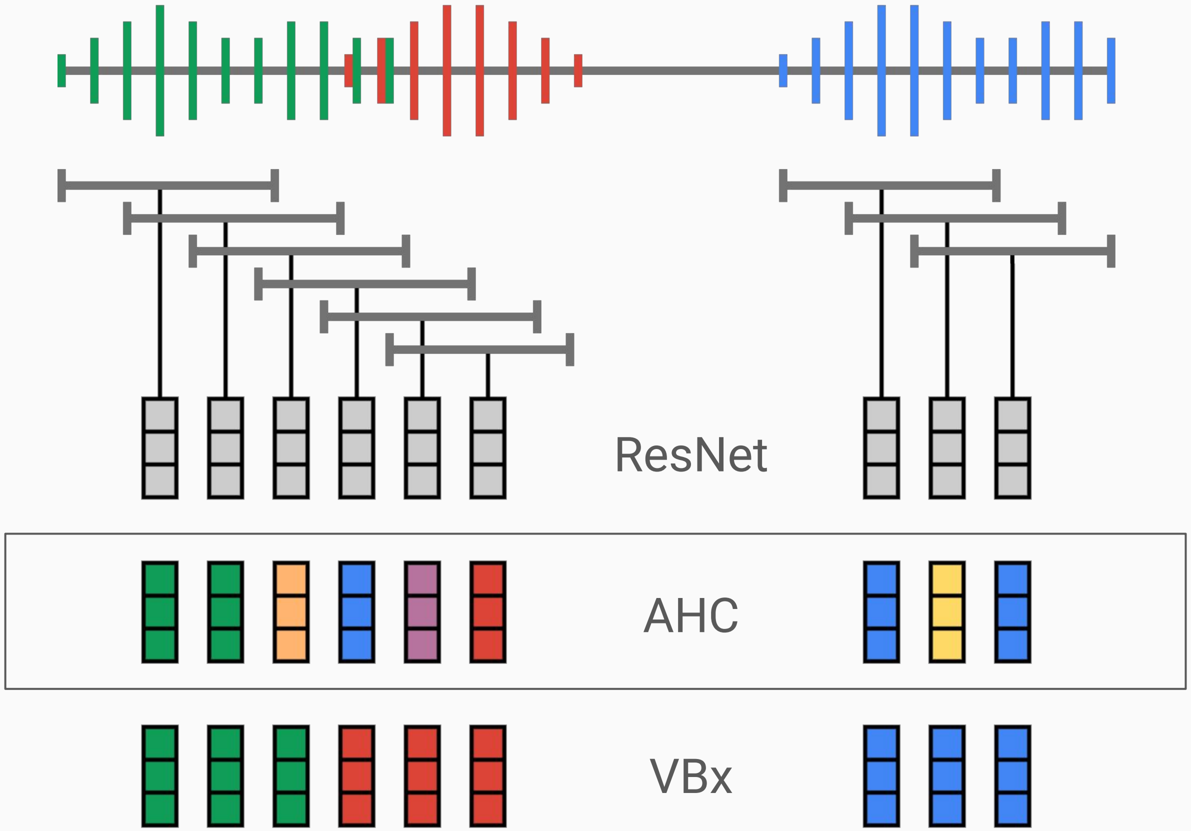
- Built on top of a pre-trained SID embedding extractor (ResNet)
- Implicit speaker-discriminative power in the embeddings
- Surpasses EEND models in estimating # of speakers
- Ability to use large **real** SID datasets to train the pipeline (real diarization data are scarce)
- Relevant baseline used in many research works till this date
- Still competitive on 16 kHz data

VBx Overview

1.
Per-segment
Embeddings

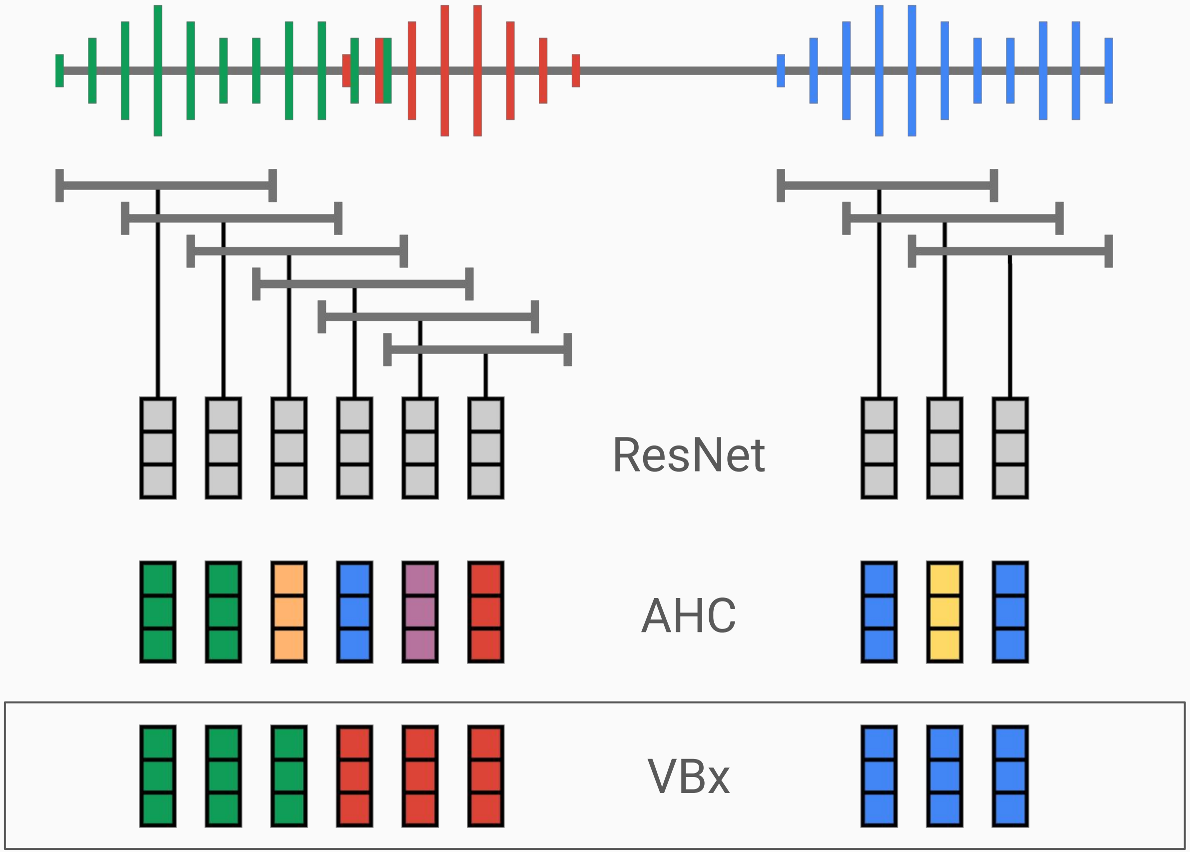


VBx Overview



2.
Pre-cluster

VBx Overview



3.
Refine

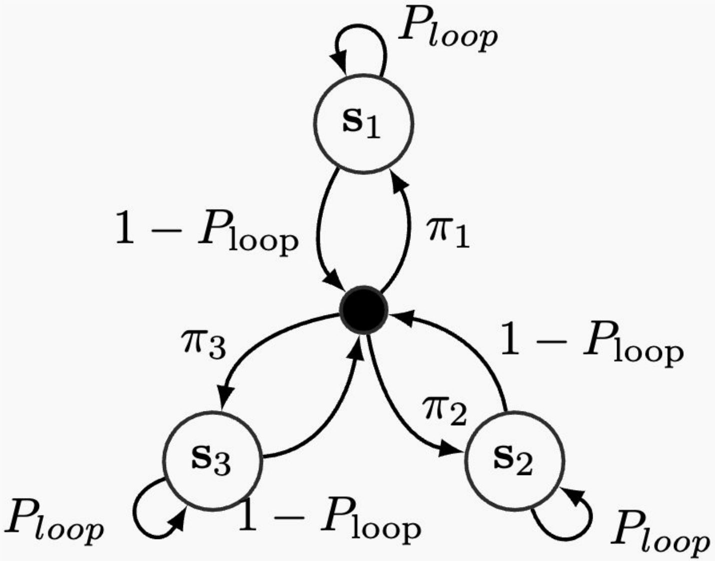
VBx - Basics

- Bayesian HMM-based model $p(z_t = s | z_{t-1} = s') = (1 - P_l)\pi_s + \delta(s = s')P_l$
- Two-covariance PLDA speaker models (HMM emissions):
    - $p(\hat{\mathbf{x}}_i) = \mathcal{N}(\hat{\mathbf{x}}_i; \hat{\mathbf{m}}_s, \mathbf{\Sigma_w}); p(\hat{\mathbf{m}}_s) = \mathcal{N}(\hat{\mathbf{m}}_s; \mathbf{m}, \mathbf{\Sigma_b})$
    - Transformed input x-vectors (std. normal within-class, diagonal across-class)
    - $\mathbf{X} = (\hat{\mathbf{X}} - \mathbb{1}\mathbf{m})\mathbf{E}, \mathbf{\Sigma_b}\mathbf{E} = \mathbf{\Sigma_w}\mathbf{E}\mathbf{\Phi}$
    - Standard normal prior on speaker variable

$$p(\mathbf{m}_s) = \mathcal{N}(\mathbf{m}_s; \mathbf{0}, \mathbf{\Phi})$$

$$p(\mathbf{x}_t | z_t = s) = \mathcal{N}(\mathbf{x}_t; \mathbf{V}\mathbf{y}_s, \mathbf{I})$$

$$\mathbf{V} = \mathbf{\Phi}^{\frac{1}{2}}, \mathbf{m}_s = \mathbf{V}\mathbf{y}_s, p(\mathbf{y}_s) = \mathcal{N}(\mathbf{y}_s; \mathbf{0}, \mathbf{I})$$



VBx - Inference

- $$p(\mathbf{X},\mathbf{Z},\mathbf{Y}) = p(\mathbf{X}|\mathbf{Z},\mathbf{Y})p(\mathbf{Z})p(\mathbf{Y}) = \underbrace{\prod_t p\left(\mathbf{x}_t|z_t\right)}_{\text{PLDA}} \underbrace{\prod_t p\left(z_t|z_{t-1}\right)}_{\text{HMM}} \underbrace{\prod_s p\left(\mathbf{y}_s\right)}_{\text{PRIOR}}$$
- Intractable posterior: $p(\mathbf{Z}|\mathbf{X}) = \int p(\mathbf{Z},\mathbf{Y}|\mathbf{X})d\mathbf{Y}$
- Let's approximate: $p(\mathbf{Z},\mathbf{Y}|\mathbf{X}) \approx q(\mathbf{Z},\mathbf{Y}) = q(\mathbf{Z})q(\mathbf{Y})$

  By maximizing:

$$\hat{\mathcal{L}}\left(q(\mathbf{Y},\mathbf{Z})\right) = F_A E_{q(\mathbf{Y},\mathbf{Z})}\left[\ln p(\mathbf{X}|\mathbf{Y},\mathbf{Z})\right] + F_B E_{q(\mathbf{Y})}\left[\ln \frac{p(\mathbf{Y})}{q(\mathbf{Y})}\right] + E_{q(\mathbf{Z})}\left[\ln \frac{p(\mathbf{Z})}{q(\mathbf{Z})}\right]$$

- We maximize $q(\mathbf{Y})$ given $q(\mathbf{Z})$ fixed and vice-versa iteratively
- F_A counteracts the independence assumption of HMM
- The higher the F_B, the more speakers are dropped

Gridsearch

- Hyperparameters need to be optimized **jointly**
- Gridsearch requires manual specification of search space
    - **F_a=0.2, F_b=6** - DIHARD II
    - **F_a=0.4, F_b=64** - AMI
- Precision of found parameters is **limited**
- Prior knowledge is **necessary** to find optimal parameters

Automatic Search

- **Advantages:**
  - User can treat VBx as a **blackbox** and optimize the hyperparameters for a new dataset
  - **Joint optimization** of the VBx pipeline (including ResNet)
- **Procedure:**
  - Hyperparameters are optimized while the rest of the pipeline is fixed
  - PLDA is fine tuned with fixed hyperparameters to potentially further boost the model performance

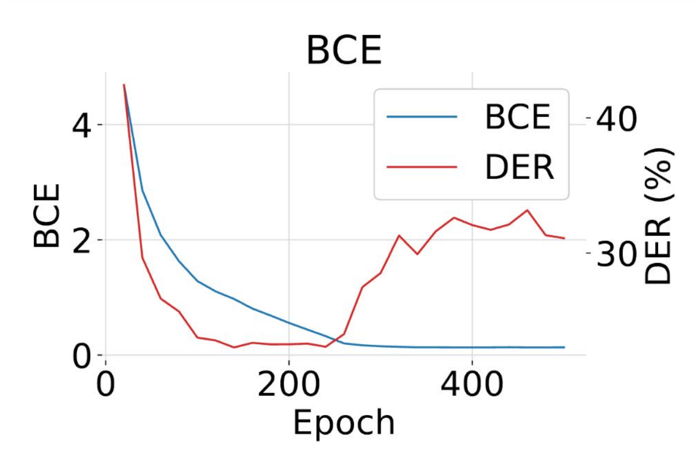
Training & Evaluation Setup

- We used Adam optimizer with different learning rates for F_a, F_b and loop probability
- Datasets:
  - CALLHOME
  - DIHARD II
  - AMI
- Metrics: Diarization Error Rate (DER)
- Selected the best-performing model based on the lowest validation DER

BCE loss

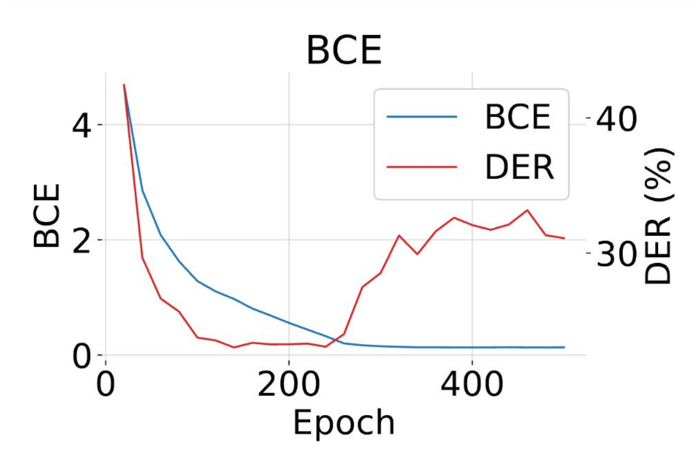
- $\boldsymbol{\gamma}_t^{\phi} = (\gamma_{t1}^{\phi}, \gamma_{t2}^{\phi}, \ldots, \gamma_{tS}^{\phi})^\top \in \langle 0, 1\rangle^S$ denotes VBx predictions
- $\hat{\mathbf{l}}_t = (\hat{l}_{t1}, \hat{l}_{t2}, \ldots, \hat{l}_{tS})^\top \in \{0, 1\}^S$ denotes ground truth labels

$$\mathcal{L} = \frac{1}{TS} \min_{\phi \in perm(S)} \sum_{t=1}^{T} H(\boldsymbol{\gamma}_t^{\phi}, \hat{\mathbf{l}}_t) \qquad H_B(\boldsymbol{\gamma}_t^{\phi}, \hat{\mathbf{l}}_t) = \sum_{s=1}^{S} \underline{-\hat{l}_{ts} log(\gamma_{ts}^{\phi})} - \underline{(1 - \hat{l}_{ts}) log(1 - \gamma_{ts}^{\phi})}$$



Overconfidence & BCE

- VBx produces overconfident posteriors
- BCE is trying to fix overconfident error during later stages of training
- We tried BCE+calib: $H_{B+C} = H_B(softmax(\tau \cdot \gamma_{ts}^{\phi}), \hat{\mathbf{l}}_t)$ with trainable or fixed scaling constant



EDE loss

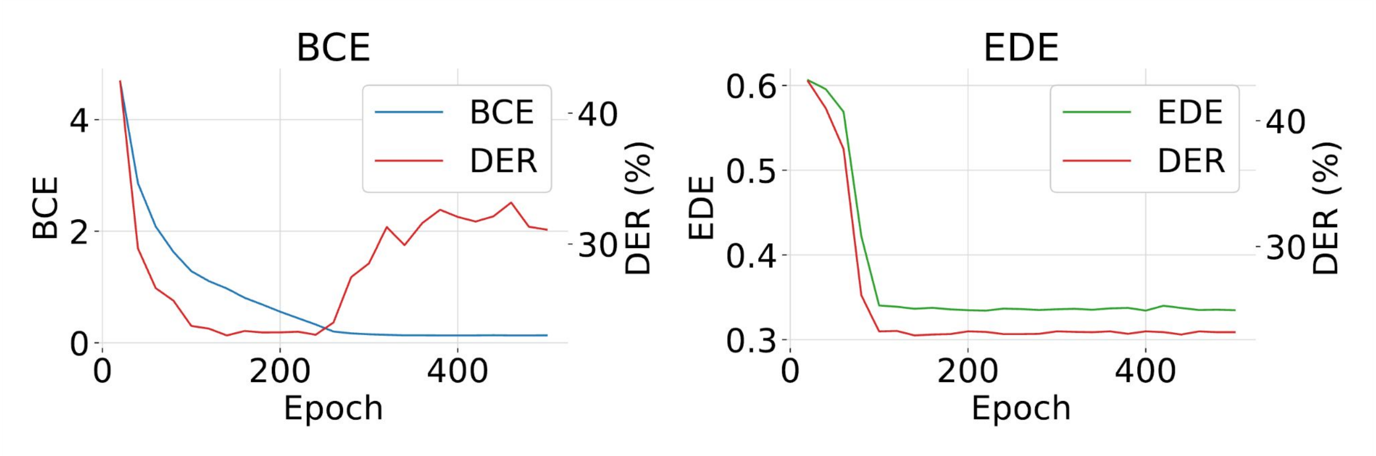
- BCE does not correlate with DER well as it tries to fix over-confidence errors instead of diarization-related errors
- We propose Expected Detection Error (EDE) loss:

$$H_E(\boldsymbol{\gamma}_t^{\phi}, \hat{\mathbf{l}}_t) = \sum_{s=1}^{S} \underbrace{\boxed{(1-\gamma_{ts}^{\phi})\hat{l}_{ts}}}_{\text{Expected FA}} + \underbrace{\boxed{\gamma_{ts}^{\phi}(1-\hat{l}_{ts})}}_{\text{Expected MISS}}$$

EDE loss

- BCE does not correlate with DER well as it tries to fix over-confidence errors instead of diarization-related errors
- We propose Expected Detection Error (EDE) loss:

$$H_E(\boldsymbol{\gamma}_t^{\phi}, \hat{\mathbf{l}}_t) = \sum_{s=1}^{S} (1 - \gamma_{ts}^{\phi})\hat{l}_{ts} + \gamma_{ts}^{\phi}(1 - \hat{l}_{ts})$$

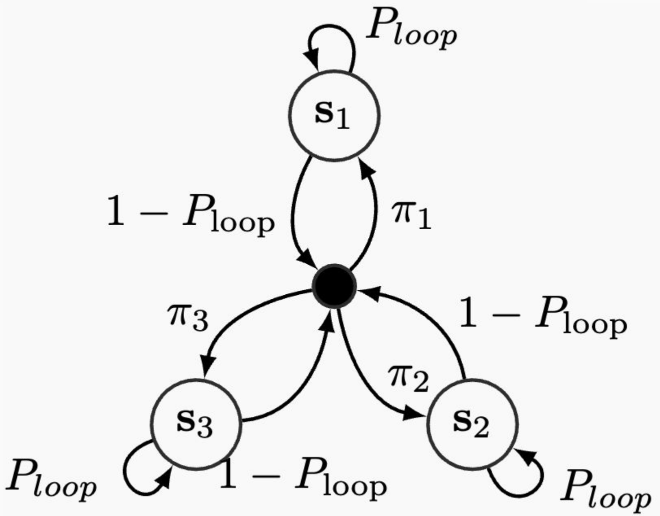


VBx - HMM to GMM

- Another important hyperparameter: **loop probability**
- Preliminary experiments showed the automatic search pushed it to **0**
- Effectively degrades **HMM to GMM**

$$p(z_t = s | z_{t-1} = s') = (1 - P_l)\pi_s + \delta(s = s')P_l \overset{P_l=0}{=} \pi_s$$

- Almost **no effect** on the performance



Optimization Results

- HMM VBx baseline (GS)
- GMM VBx baseline (GS)
- DVBx hyper parameters trained

- DVBx matches the baseline performance, which is the best we can do

Data	System	τ	F_A	F_B	DER
DH	HMM VBx [2]	7.00	0.20	6.00	**18.55**
	GMM VBx	7.00	0.20	5.00	18.93
	DVBx - BCE	2.90	0.25	4.38	18.98
	DVBx - BCE+calib.	12.88	0.43	10.14	18.84
	DVBx - EDE	9.62	0.33	9.64	18.76
CH	HMM VBx [2]	7.00	0.40	17.00	13.53
	GMM VBx	7.00	0.30	13.00	13.63
	DVBx - BCE	0.97	0.08	1.39	13.53
	DVBx - BCE+calib.	1.93	0.51	11.16	14.52
	DVBx - EDE	12.40	0.26	9.47	**13.48**
AMI	HMM VBx [2]	7.00	0.40	64.00	**20.84**
	GMM VBx	7.00	0.50	63.00	21.49
	DVBx - BCE	12.35	0.12	8.89	21.06
	DVBx - BCE+calib.	15.10	0.21	13.90	21.72
	DVBx - EDE	3.48	0.25	25.31	20.91

Optimization Results

- HMM VBx baseline (GS)
- GMM VBx baseline (GS)
- DVBx hyper parameters trained

- DVBx matches the baseline performance, which is the best we can do

Data	System	τ	F_A	F_B	DER
DH	HMM VBx [2]	7.00	0.20	6.00	**18.55**
	GMM VBx	7.00	0.20	5.00	18.93
	DVBx - BCE	2.90	0.25	4.38	18.98
	DVBx - BCE+calib.	12.88	0.43	10.14	18.84
	DVBx - EDE	9.62	0.33	9.64	<u>18.76</u>
CH	HMM VBx [2]	7.00	0.40	17.00	13.53
	GMM VBx	7.00	0.30	13.00	13.63
	DVBx - BCE	0.97	0.08	1.39	13.53
	DVBx - BCE+calib.	1.93	0.51	11.16	14.52
	DVBx - EDE	12.40	0.26	9.47	**<u>13.48</u>**
AMI	HMM VBx [2]	7.00	0.40	64.00	**20.84**
	GMM VBx	7.00	0.50	63.00	21.49
	DVBx - BCE	12.35	0.12	8.89	21.06
	DVBx - BCE+calib.	15.10	0.21	13.90	21.72
	DVBx - EDE	3.48	0.25	25.31	<u>20.91</u>

Optimization Results

- HMM VBx baseline (GS)
- GMM VBx baseline (GS)
- DVBx hyper parameters trained

- DVBx matches the baseline performance, which is the best we can do

Data	System	τ	F_A	F_B	DER
DH	HMM VBx [2]	7.00	0.20	6.00	**18.55**
	GMM VBx	7.00	0.20	5.00	18.93
	DVBx - BCE	2.90	0.25	4.38	18.98
	DVBx - BCE+calib.	12.88	0.43	10.14	18.84
	DVBx - EDE	9.62	0.33	9.64	18.76
CH	HMM VBx [2]	7.00	0.40	17.00	13.53
	GMM VBx	7.00	0.30	13.00	13.63
	DVBx - BCE	0.97	0.08	1.39	13.53
	DVBx - BCE+calib.	1.93	0.51	11.16	14.52
	DVBx - EDE	12.40	0.26	9.47	**13.48**
AMI	HMM VBx [2]	7.00	0.40	64.00	**20.84**
	GMM VBx	7.00	0.50	63.00	21.49
	DVBx - BCE	12.35	0.12	8.89	21.06
	DVBx - BCE+calib.	15.10	0.21	13.90	21.72
	DVBx - EDE	3.48	0.25	25.31	20.91

PLDA Fine Tuning Results

- PLDA FT further improves the model performance (substantially on AMI suggesting more data is needed)

System	DH	CH	AMI
a) GMM VBx	18.93	13.63	21.49
b) DVBx trained F_A, F_B	18.76	13.48	20.91
c) b) + PLDA FT	**18.66**	**13.38**	18.99
d) a) + PLDA FT	18.93	13.63	**18.88**

Conclusion

- Proposed a new technique for automatic hyperparameter finding without the requirement of prior knowledge
- Proposed a new loss that better correlates with DER metric
- Showed that we can further improve VBx performance by discriminative PLDA fine tuning
- Available on GitHub:
    - https://github.com/BUTSpeechFIT/DVBx

VBx - Speaker Models

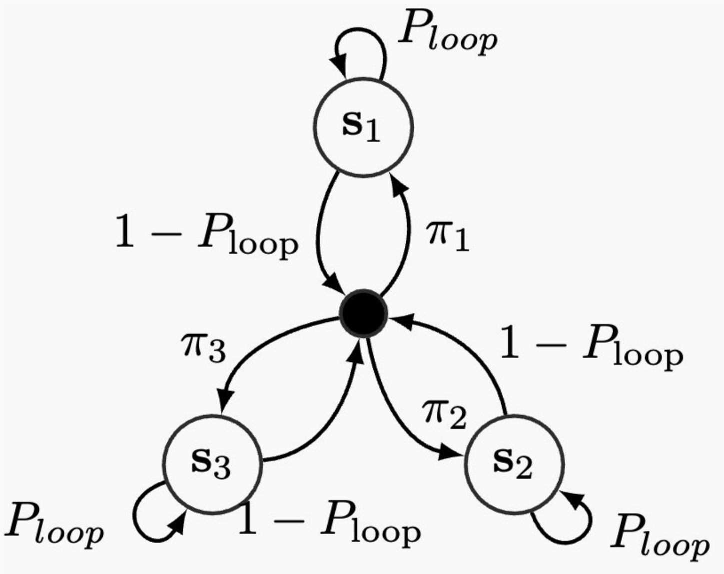
- $$q^*(\mathbf{Y}) = \prod_s q^*(\mathbf{y}_s) = \mathcal{N}\left(\mathbf{y}_s | \boldsymbol{\alpha}_s, \mathbf{L}_s^{-1}\right)$$

  $$\boldsymbol{\alpha}_s = \frac{F_A}{F_B}\mathbf{L}_s^{-1} \sum_t \gamma_{ts} \mathbf{V}^\top \mathbf{x}_t, \qquad \mathbf{L}_s = \mathbf{I} + \frac{F_A}{F_B}\left(\sum_t \gamma_{ts}\right)\boldsymbol{\Phi}$$

  - The higher the F_B, the closer spk. models are to the standard normal prior
  - The opposite holds for F_A

VBx - Basics

- Bayesian HMM-based model $p(z_t = s | z_{t-1} = s') = (1 - P_l)\pi_s + \delta(s = s')P_l$
- PLDA speaker models (HMM emissions):
  - Transformed input x-vectors (std. normal within-class, diagonal across-class)
  - Standard normal prior on speaker means



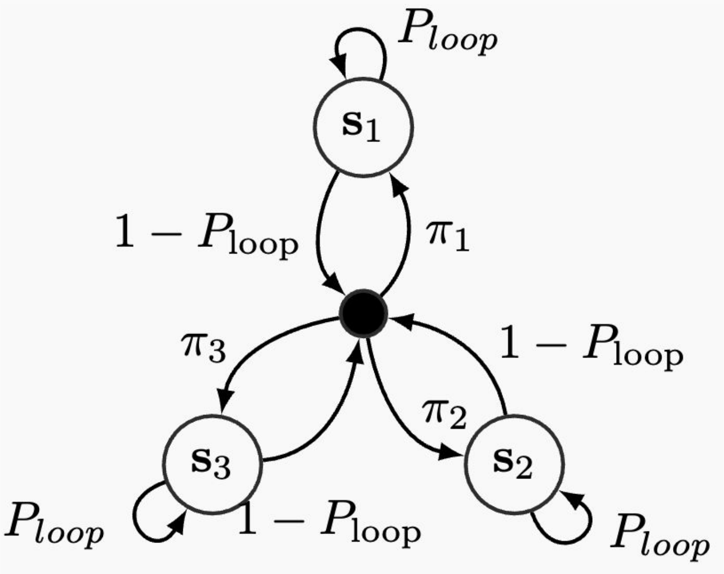
VBx - Basics

- Bayesian HMM-based model $p(z_t = s | z_{t-1} = s') = (1 - P_l)\pi_s + \delta(s = s')P_l$
- PLDA speaker models (HMM emissions):
  - Transformed input x-vectors (std. normal within-class, diagonal across-class)
  - Standard normal prior on speaker means

$$\mathbf{X} = (\hat{\mathbf{X}} - \mathbb{1}\mathbf{m})\mathbf{E} \quad \mathbf{\Sigma_b E} = \mathbf{\Sigma_w E \Phi}$$

$$p(\mathbf{x}_t | z_t = s) = \mathcal{N}(\mathbf{x}_t; \mathbf{V}\mathbf{y}_s, \mathbf{I})$$

$$\mathbf{V} = \mathbf{\Phi}^{\frac{1}{2}}, \mathbf{m}_s = \mathbf{V}\mathbf{y}_s, p(\mathbf{y}_s) = \mathcal{N}(\mathbf{y}_s; \mathbf{0}, \mathbf{I})$$



VBx - Hyper Parameters

- We only need $\gamma_{ts} = q(z_t = s) = \frac{A(t,s)B(t,s)}{\overline{p}(\mathbf{X})}$ instead of $q(\mathbf{Z})$, where $\ln \overline{p}(\mathbf{x}_t|s) = F_A\,[\ldots]$
- I.e. F_A also scales the distribution of the embeddings

- We also trained loop probability but it was being pushed to 0 by the training itself, thus we opted for GMM instead of HMM

$$p(z_t = s|z_{t-1} = s') = (1 - P_l)\pi_s + \delta(s = s')P_l \overset{P_l=0}{=} \pi_s$$

VBx - PLDA Fine Tuning Results

- Recall, we re-parametrized the PLDA model:
    - $\mathbf{X} = (\hat{\mathbf{X}} - \mathbb{1}\mathbf{m})\mathbf{E}$ d $\boldsymbol{\Sigma}_{\mathbf{b}}\mathbf{E} = \boldsymbol{\Sigma}_{\mathbf{w}}\mathbf{E}\boldsymbol{\Phi}$
- We train the transformation matrix $\mathbf{E}$ and between-class covariance matrix in the transformed space $\boldsymbol{\Phi}$