SIMPLIFICATION AND OPTIMIZATION OF I-VECTOR EXTRACTION

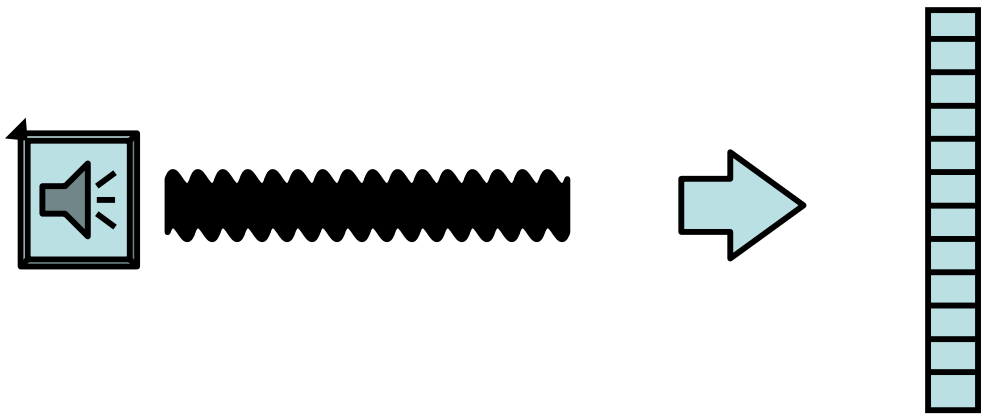
Ondrej Glembek, Lukas Burget, Pavel Matejka,
Martin Karafiat, Patrick Kenny

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

ICASSP
Prague
May 25th 2011

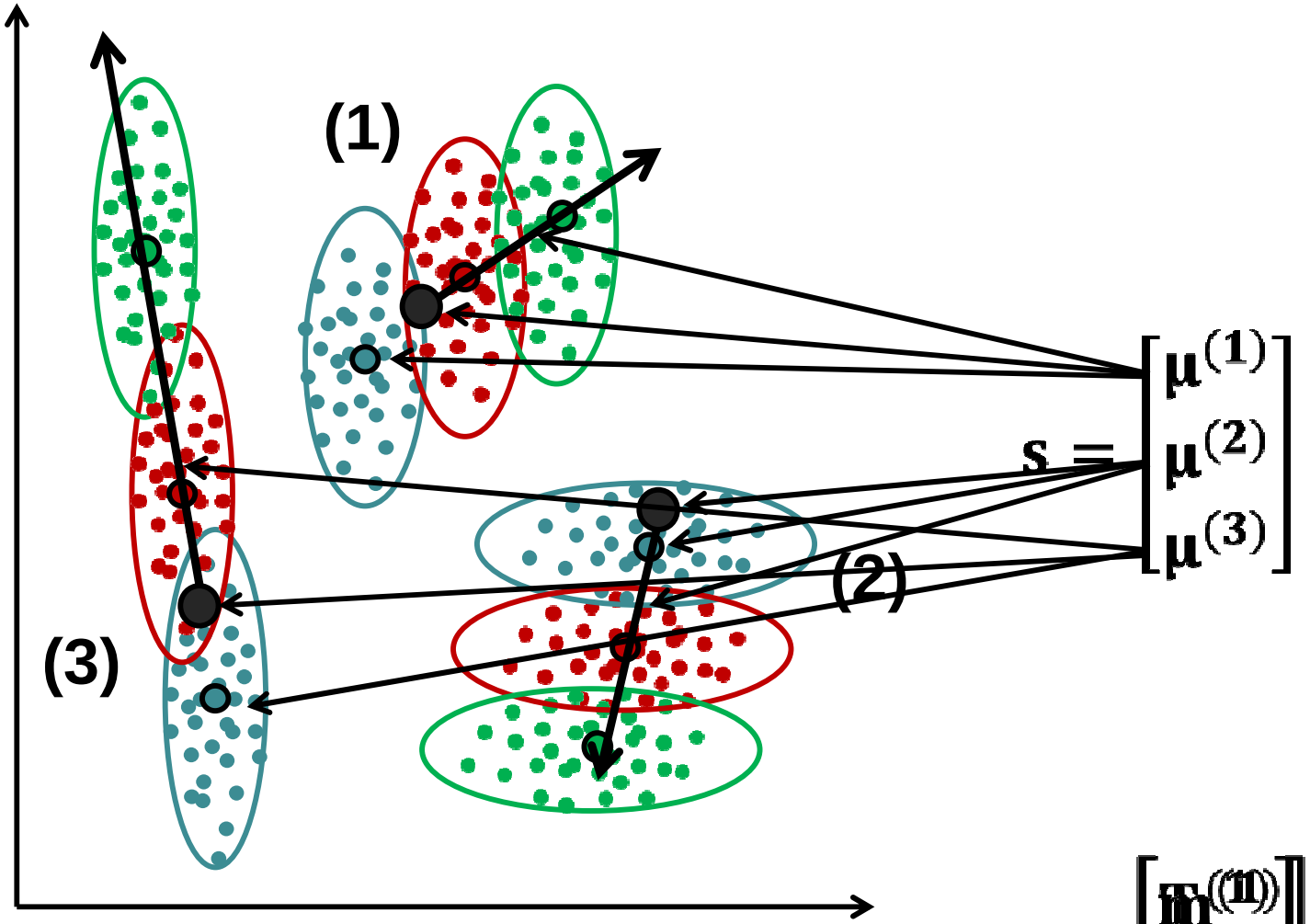
i-vectors

- An i-vector is an **information-rich, low-dimensional fixed-length** vector representing a voiceprint of an arbitrarily long sequence of speech frames. [Dehak 2010]



- We like these little units, because they turn the speaker verification task to pattern recognition problem

Subspace modeling

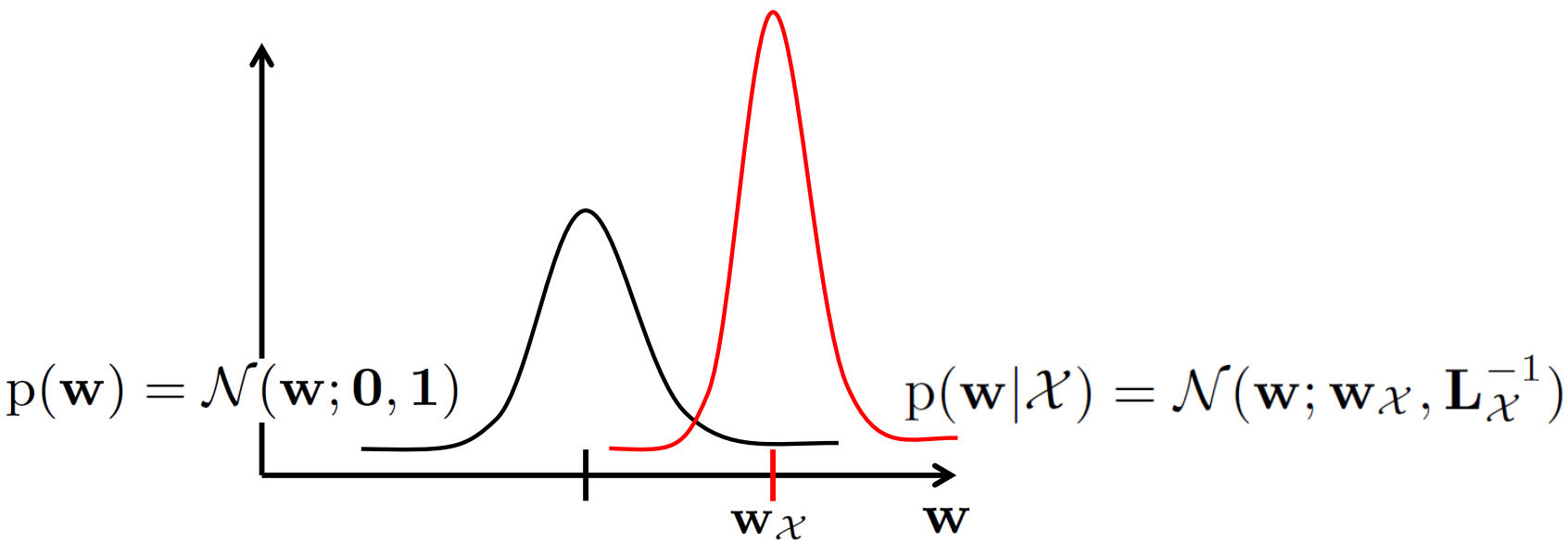


$$\mathbf{s} = \mathbf{m} + \mathbf{T}\mathbf{w}$$

$$\mathbf{m} = \begin{bmatrix} \mathbf{m}^{(1)} \\ \mathbf{m}^{(2)} \\ \mathbf{m}^{(3)} \end{bmatrix}$$

s = m + Tw

- Matrix **T** describes the directions of the highest variability of **s**
- Vector **w** is a hidden variable and as such, we can impose a prior distribution on it
- Getting adaptation data *X*, the posterior of **w** can be computed
- What we call an **i-vector** is the mean of this posterior $\mathbf{w}_X$



Statistics extraction

$$N_{\mathcal{X}}^{(c)} = \sum_t \gamma_t^{(c)}$$

$$\mathbf{f}_{\mathcal{X}}^{(c)} = \sum_t \gamma_t^{(c)} \mathbf{o}_t$$

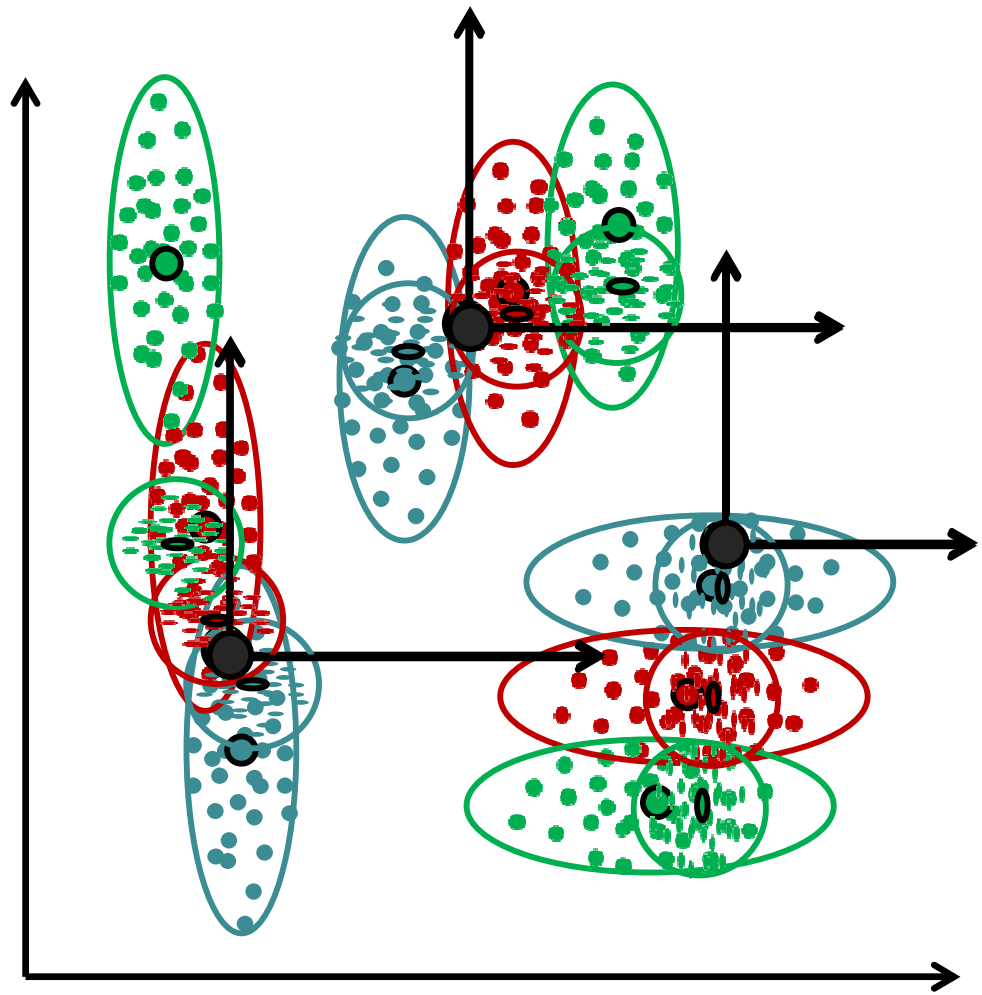
$$\mathbf{f}_{\mathcal{X}}^{(c)} \leftarrow \mathbf{f}_{\mathcal{X}}^{(c)} - N_{\mathcal{X}}^{(c)} \mathbf{m}^{(c)}$$

$$\mathbf{m}^{(c)} \leftarrow \mathbf{0}$$

$$\mathbf{f}_{\mathcal{X}}^{(c)} \leftarrow \mathbf{\Sigma}^{(c)-\frac{1}{2}} \mathbf{f}_{\mathcal{X}}^{(c)}$$

$$\mathbf{T}^{(c)} \leftarrow \mathbf{\Sigma}^{(c)-\frac{1}{2}} \mathbf{T}^{(c)}$$

$$\mathbf{\Sigma}^{(c)} \leftarrow \mathbf{I}$$



- With such data structures, the i-vector is computed as

$$\mathbf{w}_{\mathcal{X}} = \mathbf{L}_{\mathcal{X}}^{-1}\mathbf{T}'\mathbf{f}_{\mathcal{X}}$$

$$\mathbf{L}_{\mathcal{X}} = \mathbf{I} + \sum_{c=1}^{C} N_{\mathcal{X}}^{(c)}\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)}$$

$$O(CFM + CM^2 + M^3)$$

computational complexity

$$O(CFM + CM^2)$$

memory complexity

C … # of GMM components
F … Feature dimensionality
M … Subspace dimensionality

Motivation for simplifications

- Port the application to small-scale devices
- Prepare the framework for discriminative training

$$\mathbf{w}_{\mathcal{X}} = \mathbf{L}_{\mathcal{X}}^{-1}\mathbf{T}'\mathbf{f}_{\mathcal{X}}$$

$$\mathbf{L}_{\mathcal{X}} = \mathbf{I} + \sum_{c=1}^{C} N_{\mathcal{X}}^{(c)}\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)}$$

Simplification 1

$$\mathbf{L}_{\mathcal{X}} = \mathbf{I} + \sum_{c=1}^{C} N_{\mathcal{X}}^{(c)} \mathbf{T}^{(c)\prime} \mathbf{T}^{(c)}$$

- Simplifying 0th order stats:

$$\bar{N}_{\mathcal{X}}^{(c)} = \omega^{(c)} N_{\mathcal{X}} \qquad \mathbf{W} = \sum_{c=1}^{C} \omega^{(c)} \mathbf{T}^{(c)\prime} \mathbf{T}^{(c)}$$

$$\bar{\mathbf{L}}_{\mathcal{X}} = \mathbf{I} + N_{\mathcal{X}} \mathbf{W}$$

$$O(CFM + \cancel{CM^2} + M^3)$$

computational complexity

$$O(CFM + \cancel{CM^2})$$

memory complexity

Simplification 2

$$\mathbf{L}_{\mathcal{X}} = \mathbf{I} + \sum_{c=1}^{C} N_{\mathcal{X}}^{(c)} \mathbf{T}^{(c)\prime} \mathbf{T}^{(c)}$$

- Let us assume, that there exist a linear transformation **G** that would diagonalize all $\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)}$.

$$\hat{\mathbf{L}}_{\mathcal{X}} = \mathbf{G}'\mathbf{G} + \sum_{c=1}^{C} N_{\mathcal{X}}^{(c)} \mathbf{G}'\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)}\mathbf{G}$$

$$\mathbf{L}_{\mathcal{X}} = \mathbf{G}^{(-1)\prime} \hat{\mathbf{L}}_{\mathcal{X}} \mathbf{G}^{-1}$$

- Also, $\hat{\mathbf{L}}_{\mathcal{X}}$ is now diagonal, so the inversion is trivial
- Also, $\mathbf{L}_{\mathcal{X}}$ is now diagonal, so the inversion is trivial, which can be implemented effectively

Simplification 2 - Continuing

- Effectively written

$$\hat{\mathbf{L}}_{\mathcal{X}} = \mathrm{Diag}\left(\mathrm{diag}(\mathbf{G}'\mathbf{G}) + \mathbf{V}\mathbf{n}_{\mathcal{X}}\right)$$

$$\hat{\mathbf{w}}_{\mathcal{X}} = \mathbf{G}\hat{\mathbf{L}}_{\mathcal{X}}^{-1}\mathbf{G}'\mathbf{T}'\mathbf{f}_{\mathcal{X}}$$

- where **V** packs diagonal matrices into column vectors:

$$\mathrm{diag}(\mathbf{G}'\mathbf{T}^{(c)\prime}\mathbf{T}^{(c)}\mathbf{G})$$

$$O(CFM + \cancel{CM^2} + \cancel{M^3})$$

computational complexity

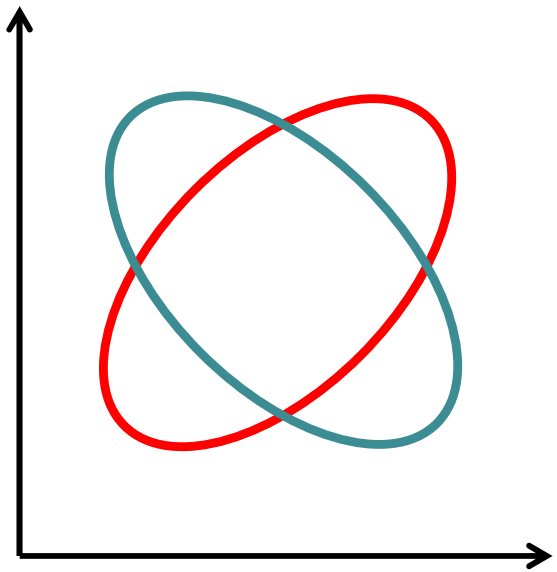
$$O(CFM + M^2 + CM)$$

$O(CFM+CM^2)$

memory complexity

Simplification 2 – Estimating **G** matrix

- The simplest approach to estimate the orthogonalization matrix **G** is PCA
- Inspired by other fields, we also tried Heteroscedastic Linear discriminant analysis (HLDA) [Kumar 1997, Gales 1999]



Obviously, we want the transform to rotate the space 45 degrees, but the average within-class covariance matrix would be diagonal ➔ PCA fails

i-vector extractor training

- i-vector extractor **T** training procedure comprises collecting the following accumulators

$$\mathbf{C} = \sum_i \mathbf{f}_i \mathbf{w}_i'$$

$$\mathbf{A}^{(c)} = \sum_i N_i^{(c)} \left(\mathbf{L}_i^{-1} + \mathbf{w}_i \mathbf{w}_i' \right)$$

- The update is the given as

$$\mathbf{T}^{(c)} = \mathbf{C} \mathbf{A}^{(c)^{-1}}$$

- We see that the simplifications can be applied even in training

Experimental setup

- Features

  MFCC 19+E, short time cepstral mean and variance normalization over 300frames, Δ+ΔΔ
- Training set

  SWII, phase 2 and 3, SW cellular, NIST2004-6, Fisher English 1,2
- Test set

  NIST SRE 2010

  Extended core condition 5 – tel-tel, female only
- Performance set
  - MATLAB, single core, Intel Xeon CPU X5670, 2.93GHz
  - 50 randomly picked utterances from MIXER corpus
- UBM

  Diagonal covariance, 2048 component UBM

Experimental setup

- Summary in numbers:

  C = 2048

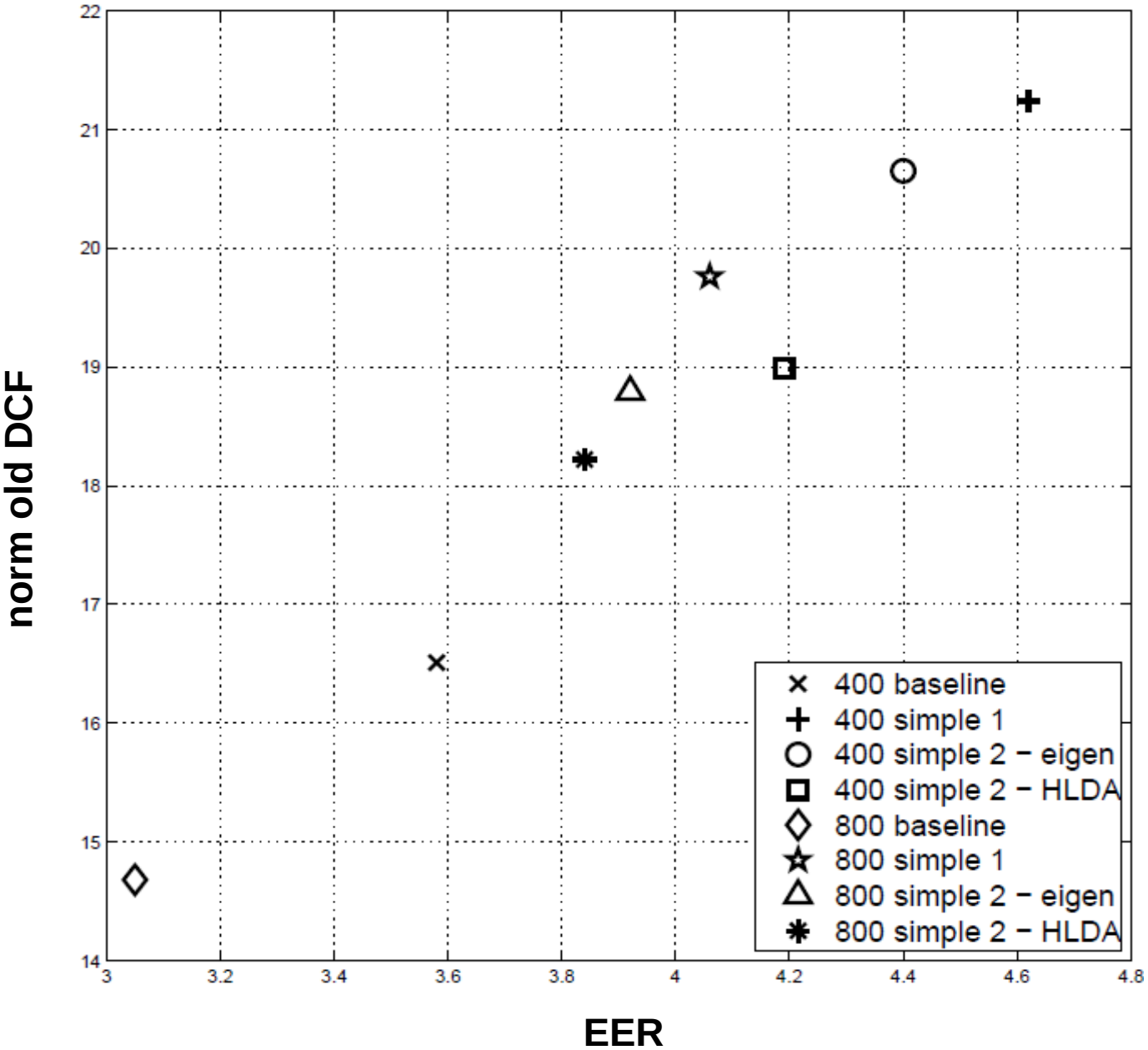
  F = 60

  M = 400

- M = 400 has been chosen as a tradeoff between performance and technical conditions.

- With simplification 1, we can afford to use M = 800 with the same memory consumption

Results


norm old DCF
EER
22
21
20
19
18
17
16
15
14
3
3.2
3.4
3.6
3.8
4
4.2
4.4
4.6
4.8
400 baseline
400 simple 1
400 simple 2 – eigen
400 simple 2 – HLDA
800 baseline
800 simple 1
800 simple 2 – eigen
800 simple 2 – HLDA

Results – Comparison of speed

	absolute [sec]	relative to 400 baseline
400 baseline	13.70	100.00%
400 simple 1	1.01	7.37%
400 simple 2	0.54	3.94%
800 baseline	65.75	480.00%
800 simple 1	3.64	26.57%
800 simple 2	1.11	8.10%

BASELINE $O(CFM + CM^2 + M^3)$

SIMPLE 1 $O(CFM + M^3)$

SIMPLE 2 $O(CFM)$

Results – Comparison of memory usage

	constant	algorithm specific	total
400 baseline	422.96	2,500.00	2,923.00
400 simple 1	”	1.22	424.18
400 simple 2	”	7.47	430.43
800 baseline	802.84	10,000.00	10,802.84
800 simple 1	”	4.88	807.83
800 simple 2	”	17.38	820.23

BASELINE $O(CFM+CM^2)$

SIMPLE 1 $O(CFM+M^2)$

SIMPLE 2 $O(CFM+M^2+CM)$

Results – Simplification 1 in i-xtractor training

	DCF_{new}	DCF_{old}	EER
400 baseline	0.5460	0.1722	3.40
400 simple 1	0.5376	0.1729	3.42

Conclusions

- We managed to significantly simplify the state-of-the-art technique in terms of **speed and memory** with a sacrifice of slight degradation in recognition performance

- We have simplified the formulas so that they are easily differentiable and usable for numerical optimizations for **discriminative training**

- We managed to fit the i-vector based SRE system into a real **cell-phone application** (EC-sponsored MOBIO project)

Thank you