NIST SRE 2006 Workshop

STBU

SDV + TNO + BUT + SUN

STBU

SDV (Spescom DataVoice, South Africa)

TNO (Netherlands)

BUT (Brno University of Technology, Czech Republic)

SUN (Stellenbosch University, South Africa)

STBU

Niko Brümmer

David van Leeuwen

Pavel Matejka, Lukas Burget, Petr Schwarz, Ondrej Glembek, Martin Karafiat, Frantisek Grezl and Jan Cernocky

Albert Strasheim, Johan du Preez

Thanks to

- The NIST evaluations that have made our progress possible. The drivers of this progress include:
  - The *challenge* posed by NIST and the *data* they make available.
  - The open *sharing* of knowledge by participants in previous years. I hope that those who have contributed so much to our knowledge (but whos C_{DET}'s didn't make it to the bottom of the pile this year) enjoy seeing their ideas flourish in the work of others.
- My hard-working partners TNO, BUT and SUN.

Good news

- Much of what we did is (after the fact) relatively easy to implement.
  - David will do a *6% EER in 24 hours* slide.
- Most methods allow very fast implementations.

STBU

- Each site developed their own systems, but we worked very closely together, with some common design principles.
- We shared (via wiki, email, sms) papers, advice, ideas, formulas, code, supervectors, scores, EER's etc., over a period of about 10 weeks. Over 600 emails were sent.
- Finally, we fused a selection of all our sub-systems.

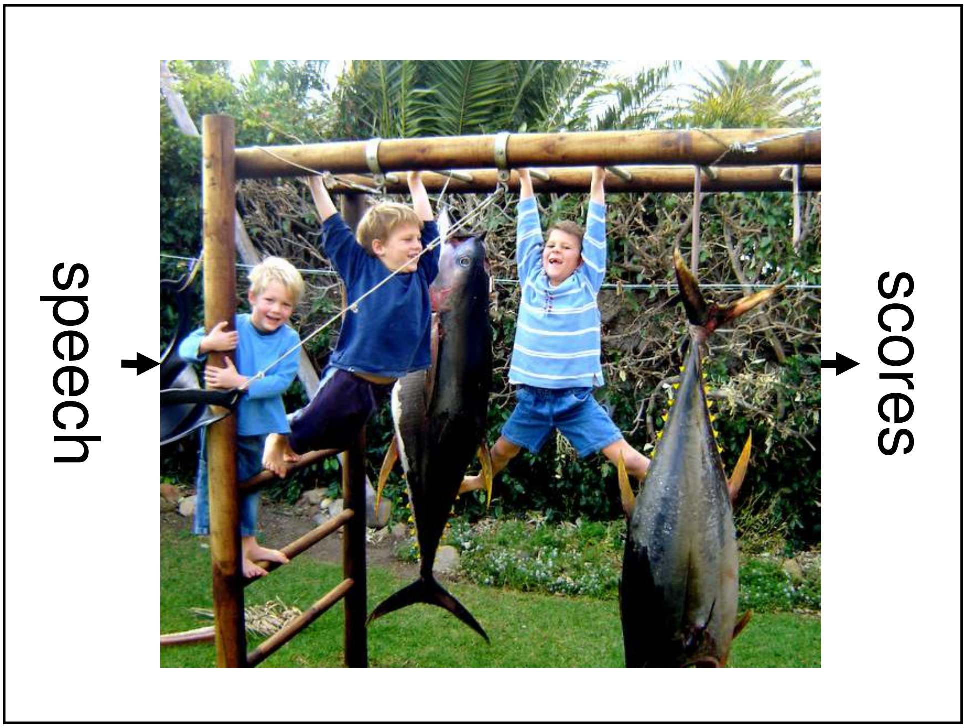
BRNO UNIVERSITY OF TECHNOLOGY
FIT FACULTY OF INFORMATION TECHNOLOGY

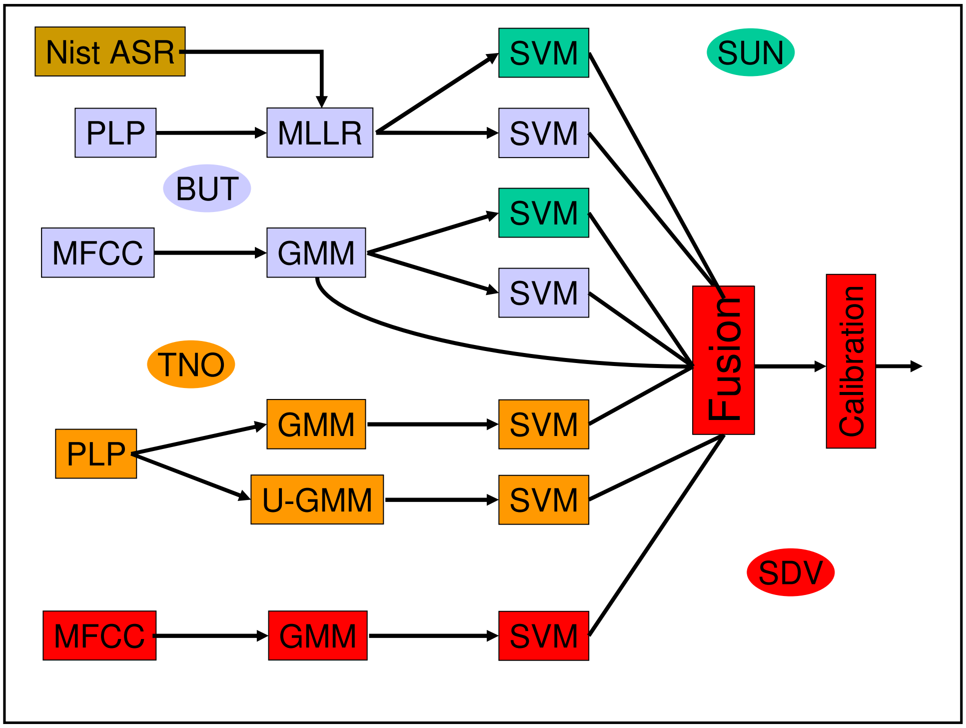
System Skeleton Overview

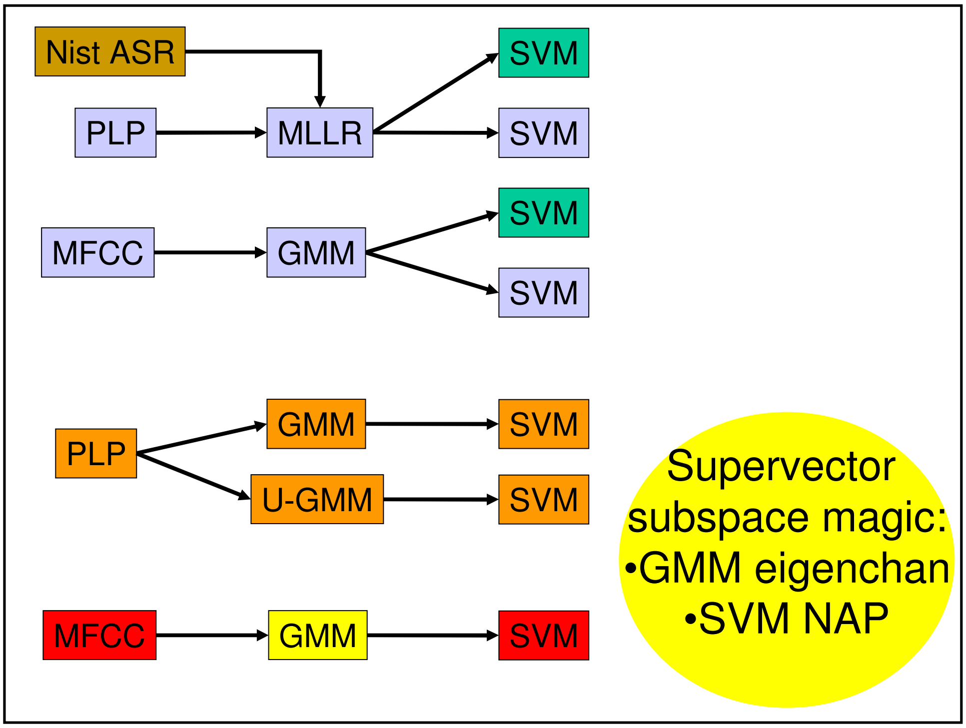
Unifying design principle

- Express speaker information as a variety of different forms of *supervector*.
- Do *inter-session subspace compensation* in supervector space.

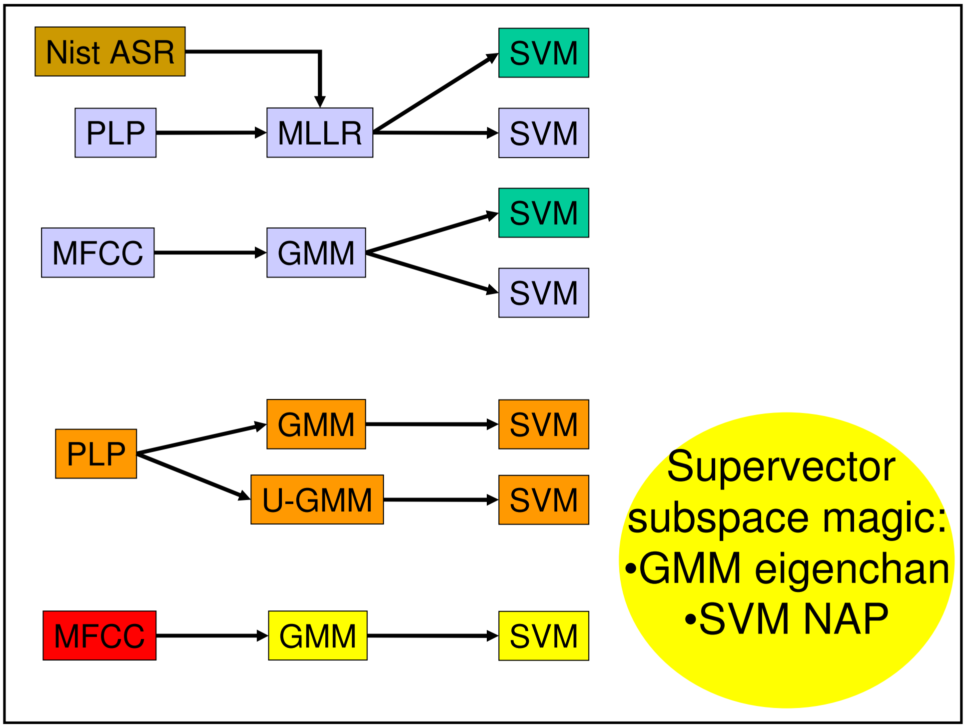
lead to neat system design


speech
scores

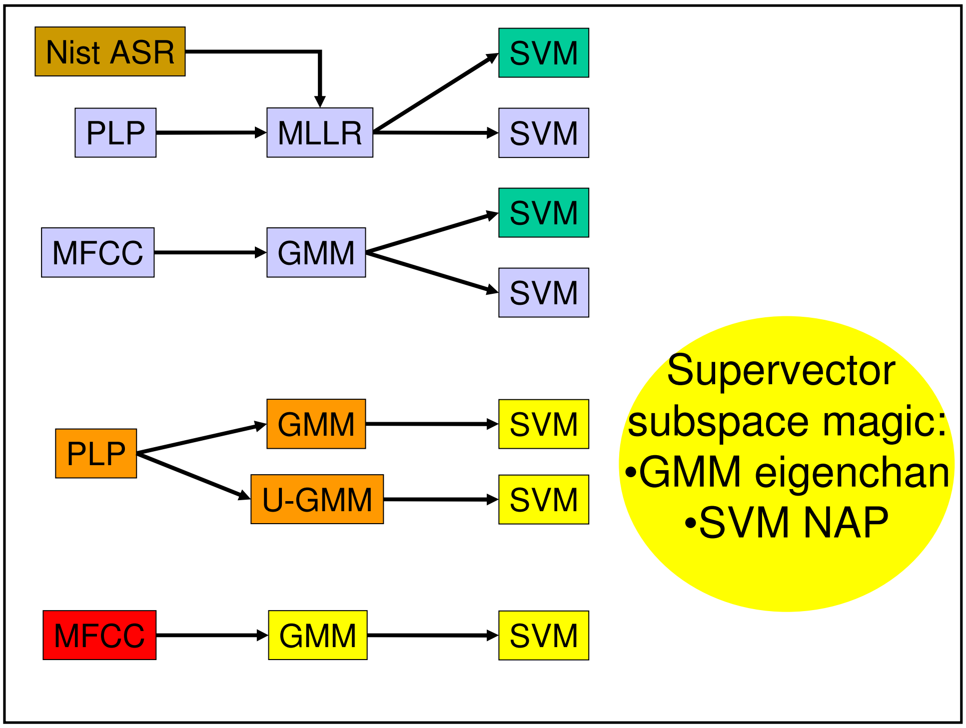

Nist ASR
PLP
MLLR
SVM
SVM
SUN
BUT
MFCC
GMM
SVM
SVM
TNO
PLP
GMM
U-GMM
SVM
SVM
Fusion
Calibration
SDV
MFCC
GMM
SVM

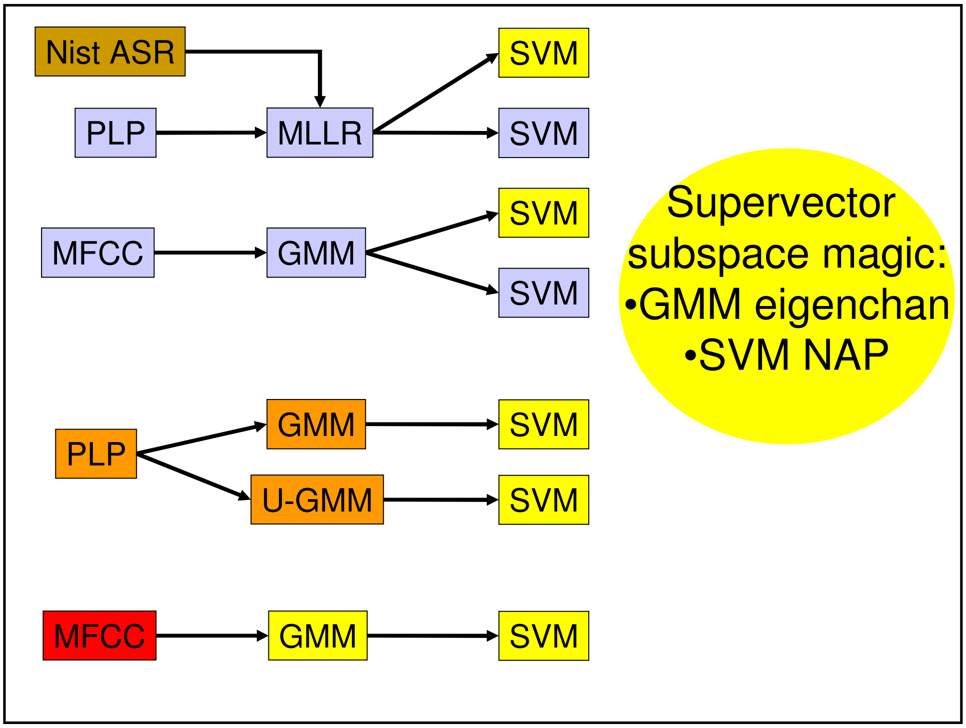


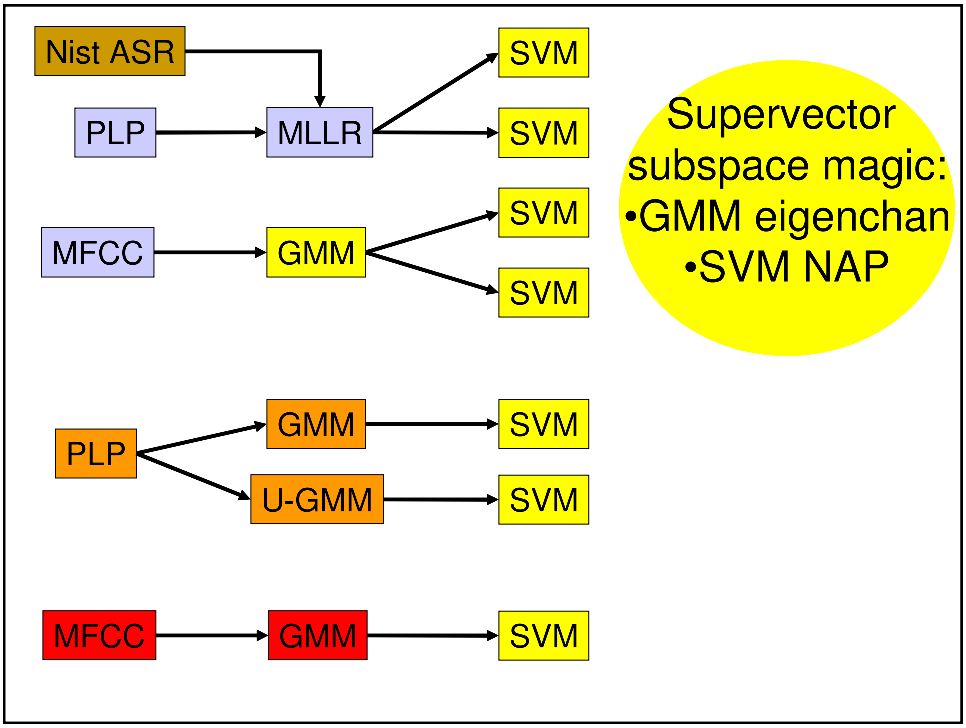
Nist ASR
PLP
MLLR
SVM
SVM
MFCC
GMM
SVM
SVM
PLP
GMM
SVM
U-GMM
SVM
MFCC
GMM
SVM
Supervector
subspace magic:
•GMM eigenchan
•SVM NAP



Nist ASR
PLP
MLLR
SVM
SVM
MFCC
GMM
SVM
SVM
PLP
GMM
SVM
U-GMM
SVM
MFCC
GMM
SVM
Supervector
subspace magic:
•GMM eigenchan
•SVM NAP


Nist ASR
PLP
MLLR
SVM
SVM
MFCC
GMM
SVM
SVM
PLP
GMM
U-GMM
SVM
SVM
MFCC
GMM
SVM
Supervector subspace magic:
•GMM eigenchan
•SVM NAP


Nist ASR
PLP
MLLR
SVM
SVM
MFCC
GMM
SVM
SVM
PLP
GMM
U-GMM
SVM
SVM
MFCC
GMM
SVM
Supervector subspace magic:
•GMM eigenchan
•SVM NAP



Main strengths

- Diversity (front-ends, data, implementations)
- Development data organization
- Many development experiments (*fast* SVM, and BUT's 100 CPUs)
- Supervector subspace channel compensation (SVM-NAP, GMM-Eigenchannel)
- Fusion
- Calibration

Agenda

- Diversity
- Development data
- *Fast* SVM
- Supervectors
- Sub-system performance
- Fusion
- Calibration

Agenda

- Diversity
- **Development data**
- *Fast* SVM
- Supervectors
- Sub-system performance
- Fusion
- Calibration

Development data

- Switchboard (UBM, feature mapping)
- Fisher (SVM background, t-norm, UBM, feature mapping)
- SRE-04 (NAP + Eigenchannel subspace training, t-norm, feature mapping, UBM)
- SRE-05 (development test set, fusion/calibration training)

SPESCOM

Agenda

- Diversity
- Development data
- ***Fast* SVM**
- Supervectors
- Sub-system performance
- Fusion
- Calibration

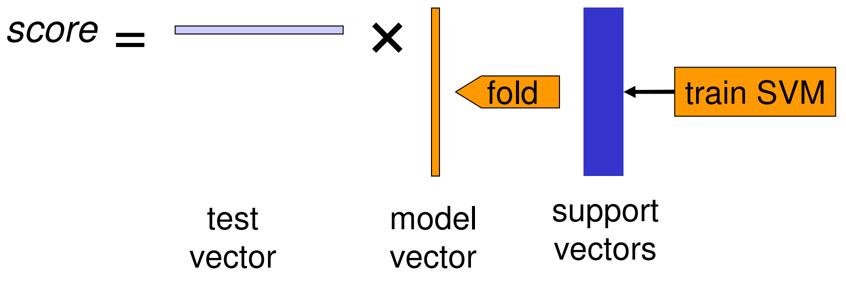
BRNO UNIVERSITY OF TECHNOLOGY
FIT FACULTY OF INFORMATION TECHNOLOGY

Fast SVM

- It is known that linear SVMs can *score* test segments really fast when all support vectors are folded into a single model supervector.
- But *training* can also be done really fast.
  - To train an SVM it needs to form *all* of the dot-products between all (thousands of) training vectors. This product matrix is called the *Gram* or *kernel* matrix.
  - When the same background set is used for every speaker model, *almost the whole* Gram matrix stays constant. We implemented our SVM training code to make use of this fact.

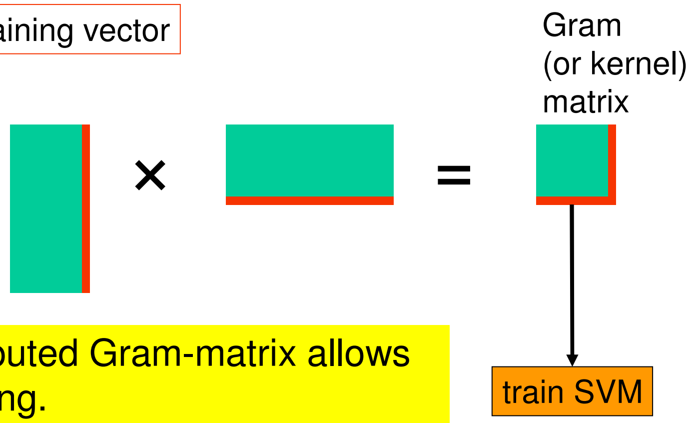
Fast SVM

Linear SVM allows *fast* single dot-product scoring:



thousands of background vectors
(this is constant and can be done *once only*)

single training vector



Pre-computed Gram-matrix allows fast training.

(0.75s per model)

Fast SVM

- With fast SVM train/test, we were able to run whole development test cycles in a few minutes.
- This allowed us to:
  - re-do and re-do and re-do our NAP transforms
  - experiment with the selection of background and T-norm sets

  until we eventually got our various SVMs working well.

Fast SVM

- See [1] for Albert's (Google sponsored) Python code which builds a fast SVM implementation on top of LibSVM [2].

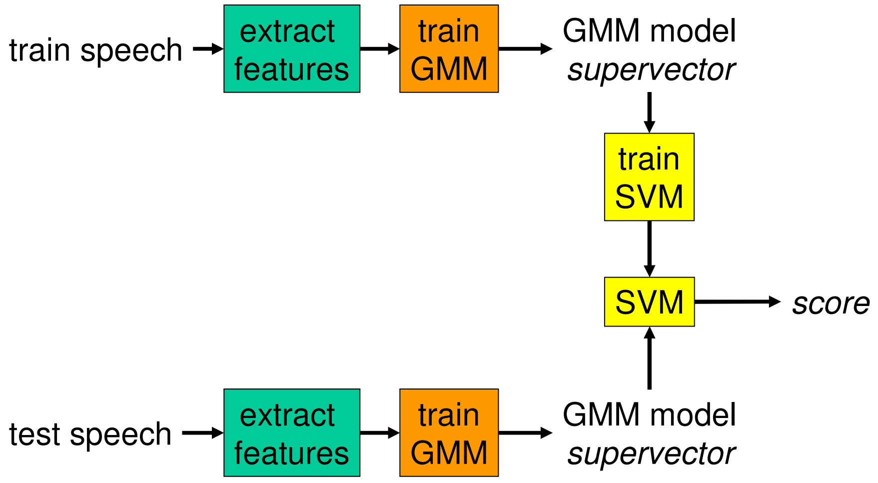
Agenda

- Diversity
- Development data
- *Fast* SVM
- **Supervectors**
- Sub-system performance
- Fusion
- Calibration

Supervectors

- We used 2 kinds of supervector:
  - **MLLR-transform** supervectors (due to SRI [3][4]).
  - **GMM-mean** supervectors (has origins in speech recognition, see e.g. [5]. You will see a lot more of this idea at NIST'06 and Odyssey'06.)

GMM-SVM



Supervector Subspace Magic

- Two types of inter-session sub-space compensation:
  - **GMM**: ***eigen-channel*** MAP-adaptation
  - **SVM**: nuisance attribute projection *(**NAP**),* applied to both GMM and MLLR supervectors.

GMM eigen-channel

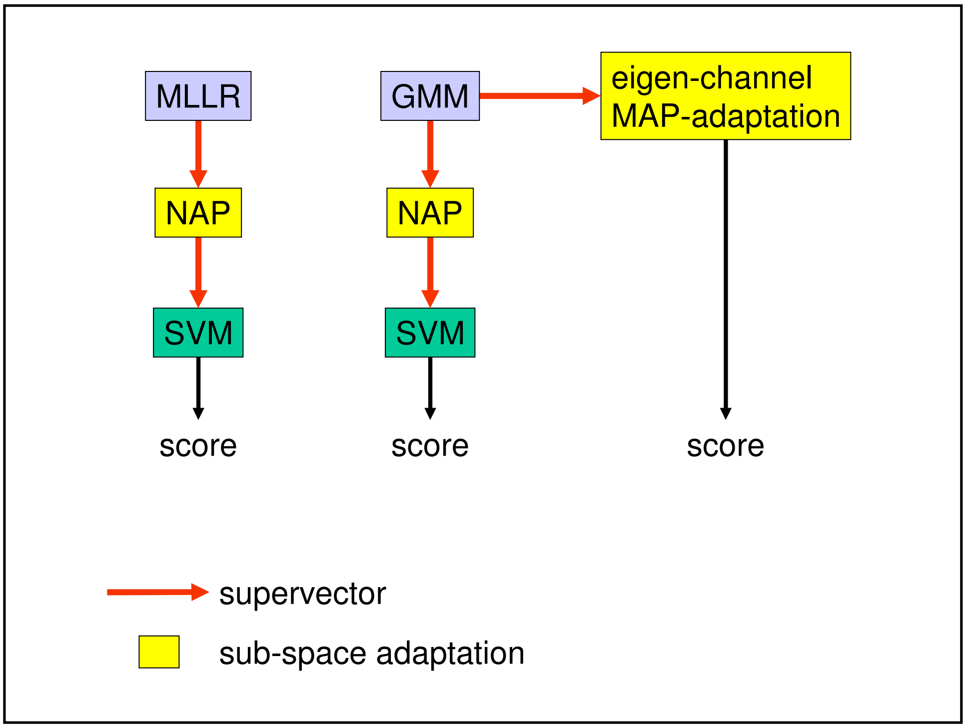
- There are a few different variants and as many different names.
- Was brought to Odyssey’04 by CRIM [5] and NIST SRE ’04 by SDV [6].
- CRIM, SDV and QUT [7][8] again fielded eigen-channel systems for NIST SRE’05.

GMM eigen-channel

- I’m going to leave detailed explanations of GMM subspace compensation to other presenters today: BUT, QUT, LPT, CRIM and MIT.
- Variants include:
  - Model MAP-adaptation during test (BUT,MIT), or train and test (QUT).
  - Feature adaptation, prior to test and train (LPT).
  - Integration over all possible adaptations (CRIM).

SVM NAP

- Due to MIT in 2004 [9][10]
- Variants, based on:
  - Supervised discrete channels
  - Unsupervised continuous inter-session
- We used the latter variant, which is very similar to eigen-channel.



Our SVM-NAP recipe

- SRE-04 works really well as training data!
- Get multiple recording sessions of all speakers (about 310 speakers in 2004).
- Create a supervector per session.
- Calculate speaker mean supervectors.
- Subtract each speaker mean from all sessions of that speaker. This leaves a data matrix with most speaker info removed, but inter-session variability still present.
- Simply do a principal component analysis (PCA) on this 'channel' data. MATLAB's `eigs()` works well for this.

SVM NAP recipe

- (For *eigenchannel* you need principal *eigenvectors* and *eigenvalues*.)
- For NAP, you need only the principal eigenvectors. You need to optimize for the number of eigenvectors. We used more for large GMM supervectors and fewer for smaller MLLR vectors.
- Make sure your eigenvectors are nicely ortho-normal. If they are not, they don't project completely away. Singular-value-decomposition (SVD) is good for ensuring ortho-normality.

SVM NAP recipe

- Project NAP subspace (principal eigenvectors) away from all supervectors involved in SVM training. You *have* to do it prior to training.
- You don't need to project NAP subspace away during test. (But it does no harm).

SVM NAP recipe

- NAP projection, where
  - *v* is original supervector,
  - *w* is projected supervector
  - *S* is matrix of principal (ortho-normal) inter-session covariance eigenvectors
  - ^{T} is transpose

$$w = v - S(S^T v)$$

Eigenchannel vs NAP?

- Is it better to do
  - eigenchannel GMM, or
  - to do straight GMM and then SVM-NAP?
- SDV did both and found the latter to work better.
- BUT did both and found the former to work better.
- TNO and SUN did the latter and found it to work well.
- Who cares. They fuse!

Warning

- Methods that use database-wide training such as NAP and eigen-channel can wreck your calibration, because performance becomes overoptimistic.
- Make sure you train fusions and calibrations on fresh data which has not been exposed to such training.

Subspace methods vs Feature mapping

- Both BUT and TNO used traditional discrete-channel-and-gender-based *feature mapping,* in combination with their subspace methods.
- However, BUT's post-eval experiments show that simpler (2-class) male-female mapping is sufficient, letting subspace methods alone deal with channel mismatch.

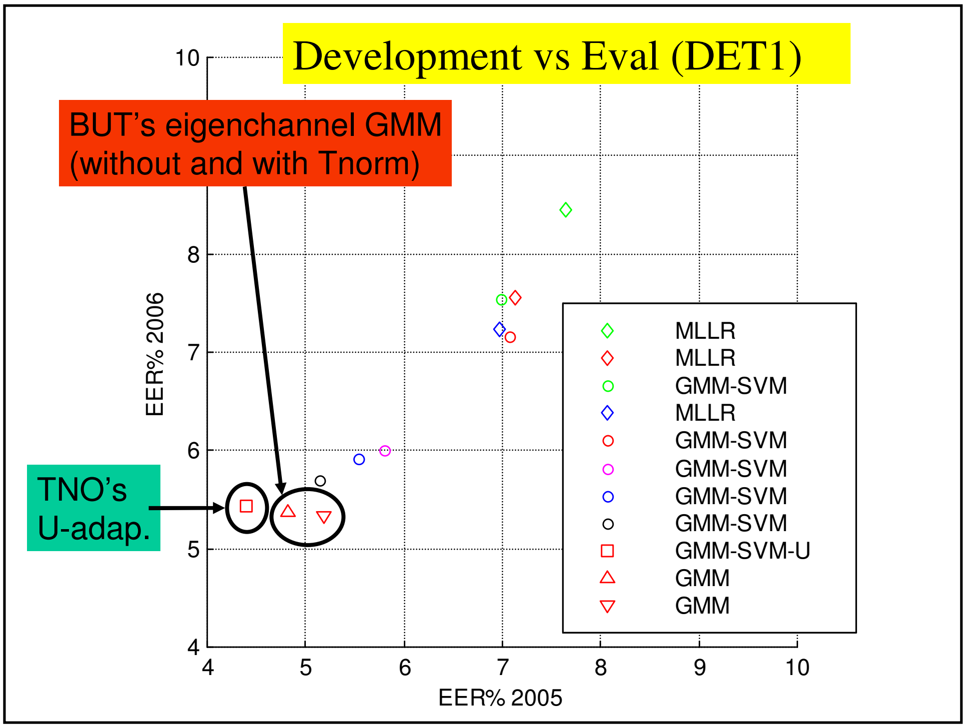
Agenda

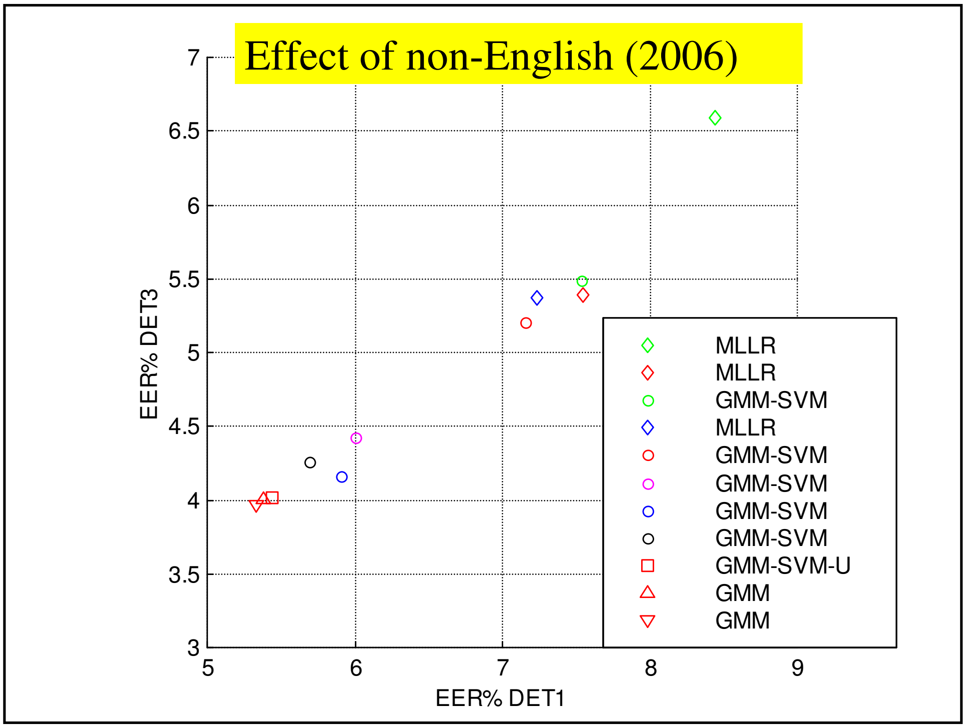
- Diversity
- Development data
- *Fast* SVM
- Supervectors
- **Sub-system performance**
- Fusion
- Calibration

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Sub-system result analysis

- 11 sub-systems:
  - 2 GMM systems (± Tnorm)
  - 5 GMM-SVM systems
  - 3 MLLR-SVM systems
  - 1 unsupervised adaptation GMM-SVM


Development vs Eval (DET1)
BUT's eigenchannel GMM
(without and with Tnorm)
TNO's
U-adap.
EER% 2006
EER% 2005
10
8
7
6
5
4
4
5
6
7
8
9
10
MLLR
MLLR
GMM-SVM
MLLR
GMM-SVM
GMM-SVM
GMM-SVM
GMM-SVM
GMM-SVM-U
GMM
GMM


Effect of non-English (2006)
EER% DET3
EER% DET1
7
6.5
6
5.5
5
4.5
4
3.5
3
5
6
7
8
9
MLLR
MLLR
GMM-SVM
MLLR
GMM-SVM
GMM-SVM
GMM-SVM
GMM-SVM
GMM-SVM-U
GMM
GMM

Observations

- There is a mild increase in error-rate from development (2005) to evaluation (2006).
  - Worst affected was unsupervised adaptation.
- On 2006 data, all systems do somewhat better on DET3 than on DET1.

Observations

- MLLR, which is partly dependent on English ASR input, seems not to be affected by the DET3 / DET1 difference any more than the short-time cepstral systems.

Agenda

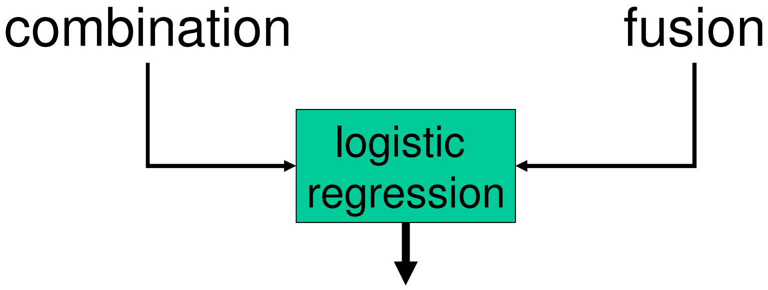
- Diversity
- Development data
- *Fast* SVM
- Supervectors
- Sub-system performance
- **Fusion**
- Calibration

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Terminology

- *Combination* in Western USA
- *Fusion* elsewhere.

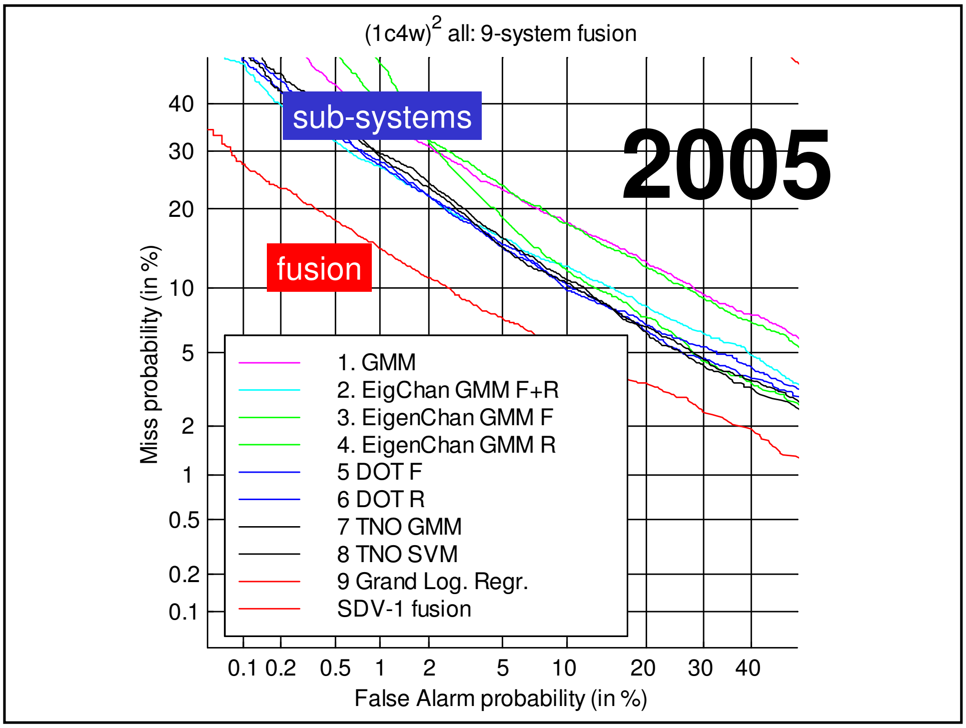
Proposal for a unified terminology:



comfusion?

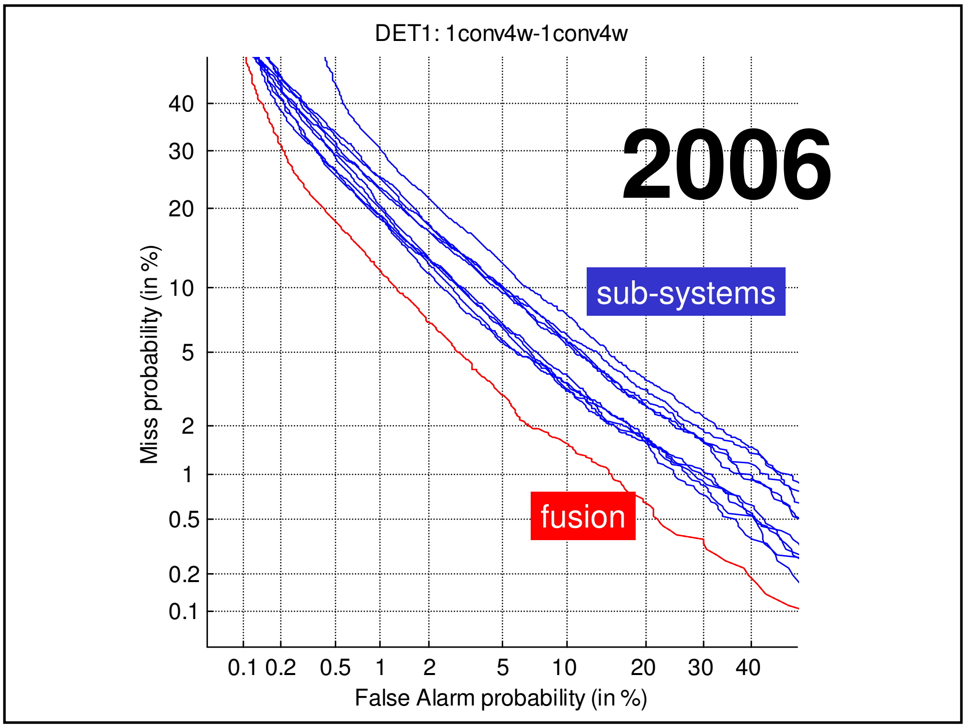
Fusion

- In SRE'05, the SDV+TNO fusion proved that multiple *mediocre* (EER 10%) systems can produce *good* (EER 6%) results.



Fusion

- This year we wanted to repeat that exercise over four sites, but then my partners started cheating! Suddenly they were contributing *good* (5%, 6% EER) systems to the fusion.
- This makes the fusion more difficult. Does fusion still work when development error-rates are low?



Fusion

- There are many ways to fuse, including both *generative* and *discriminative* methods.
- (In language recognition generative methods like LDA seem to be more popular.)
- In speaker detection discriminative methods like Logistic Regression, MLP and SVM are popular.
- Simple equal-weight summation of T-normed scores is also not a bad idea at all. (Requires no training.)

Fusion

- I chose to use logistic regression again this year.
- But I did a 10-fold cross-validation as a sanity check to see if the fusion would remain stable on unseen data, which it did.
- Finally, I used the same procedure as last year, which indeed:
  - Gave a nice gain over the whole DET-curve (EER 2.3% for English; 3.3% all)
  - Gave reasonable calibration

Logistic Regression Fusion

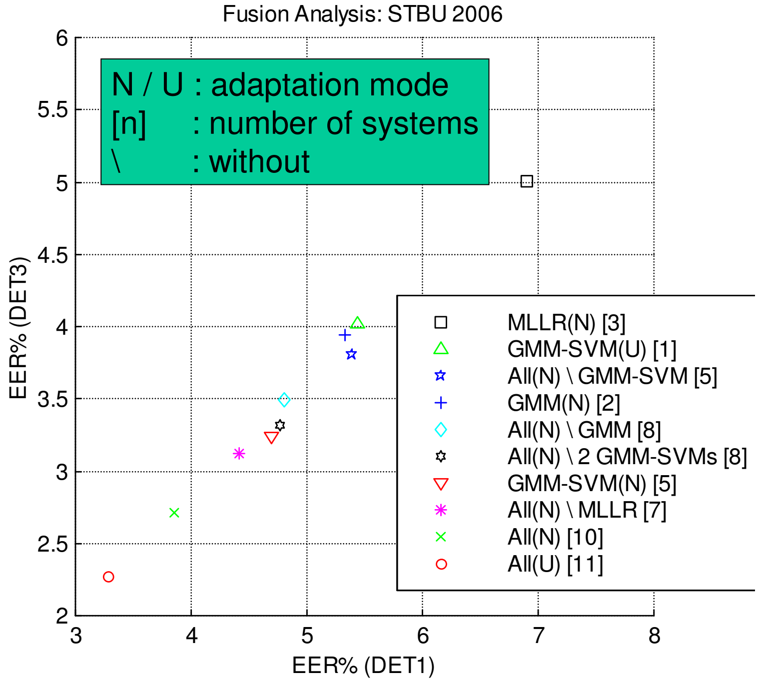
- Logistic regression fusion is an affine transform from multi-score space to log-likelihood-ratio.
- It is easy to implement: see Focal toolkit [11].
- In addition to fusion, it also *calibrates*. We had no need to choose an empirical decision threshold.
- It is robust

Why Robust?

- Training is a convex optimization problem: there is a unique global optimum.
- It has a low number of parameters, *N+1*, for *N* scores.
- It optimizes the fusion over the *whole* DET-curve, not specifically at the CDET operating point.
- Logistic regression also helped QNI, BUT, CRIM and TNO to perform successful fusion and/or calibration this year.

Fusion analysis

- We analyze DET1 and DET3 EER's for a few different fusions of **subsets** of our original 11-system fusion.
- All fusions are trained on 2005 and tested on 2006.

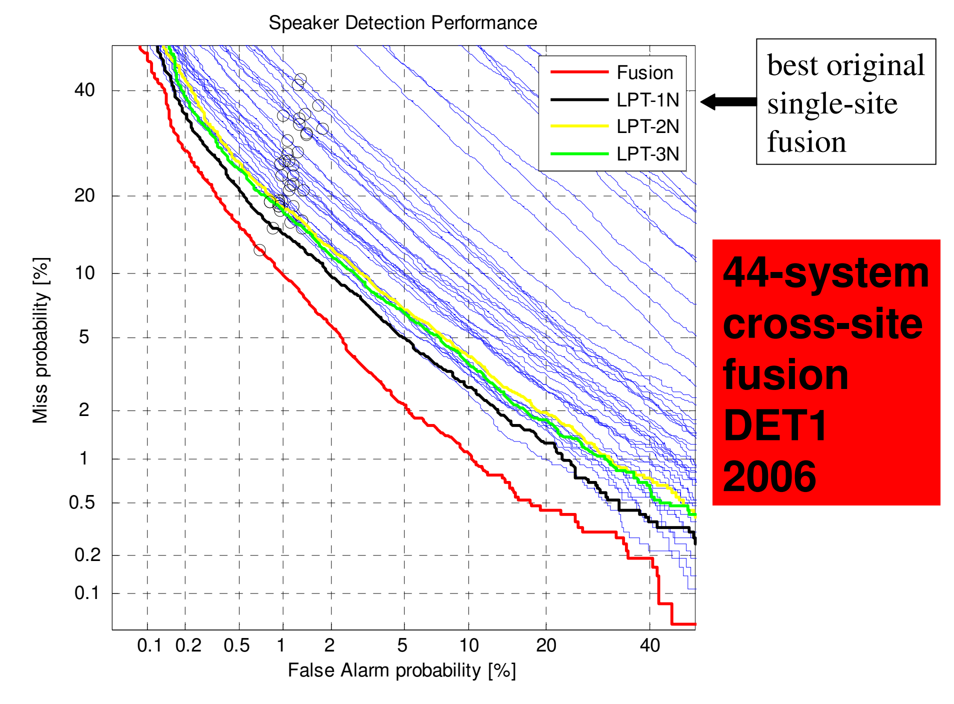


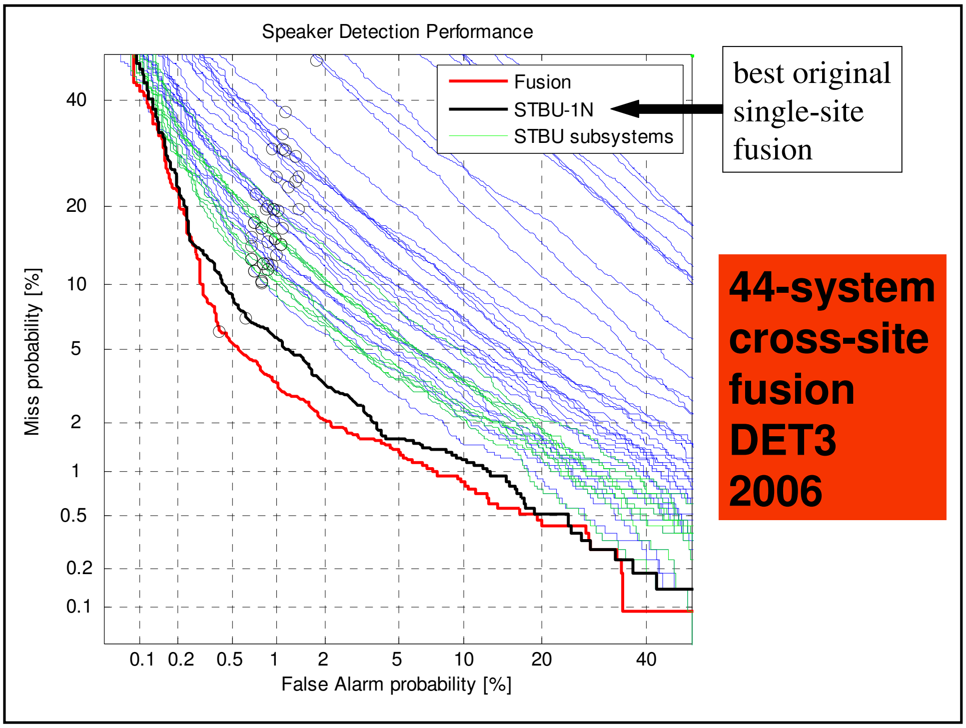
Observations (for DET1, N-mode)

- All systems: EER = 3.9%
- All fast systems (no MLLR): EER = 4.4%
- Different *methods* fuse, but different *implementations* of same methods also fuse:
  - Effect of removing 2 weakest of the 5 GMM-SVM systems increases EER from 3.9% to 4.8%

Cross-site fusion result

- As a post-eval experiment, we collected 1conv4w-1conv4w scores from multiple sites.
- In the end we fused a whole 44 sub-system scores from 12 different sites.
- (No unsupervised adaptation was used.)
- Logistic regression fusion was trained on 2005 scores.
- Results are for 2006: DET1 and DET3





Cross-site fusion

- Fusion still works when using very many systems and when error-rates are low!
- DET1 EER = 3.3%
- DET3 EER = 2.0%

Agenda

- Diversity
- Development data
- *Fast* SVM
- Supervectors
- Sub-system performance
- Fusion
- **Calibration**

Calibration

- If your speaker recognition ambitions end at doing well in the NIST SRE, then calibration means empirically choosing a decision threshold that optimizes for the *single operating point* represented by *actual* C_{DET}.
- Yes, you can go and re-optimize for other single operating points. But there are richer types of applications that require simultaneous operation over a wide range of operating points.

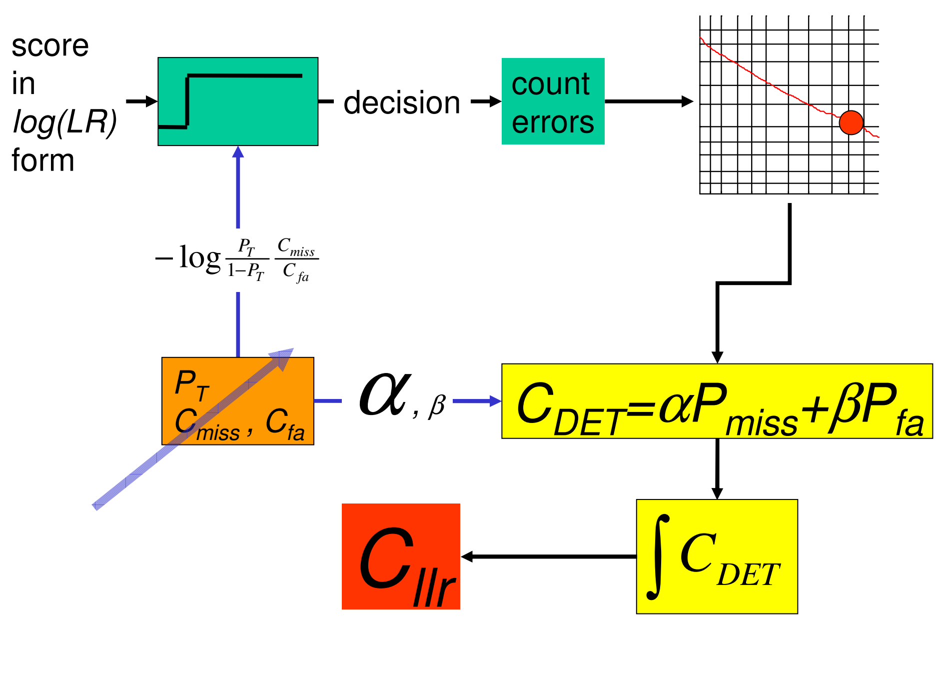
Examples

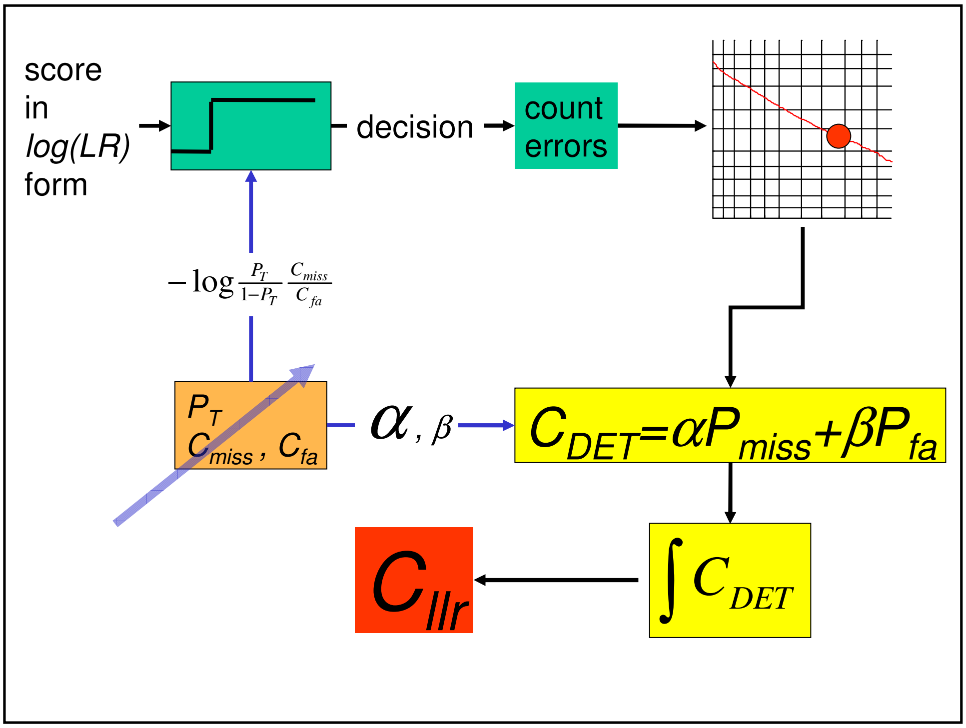
- A speaker recognizer that can advise the user of things like:
  - The *probability* of finding a target speaker in a given set of recordings which it has processed.
  - The *expected time* required of the user to find this speaker amongst those recordings, when sorted in order of likelihood.
- A forensic speaker recognition system that delivers not hard decisions, but detection *confidence* (weight-of-evidence) to the user.

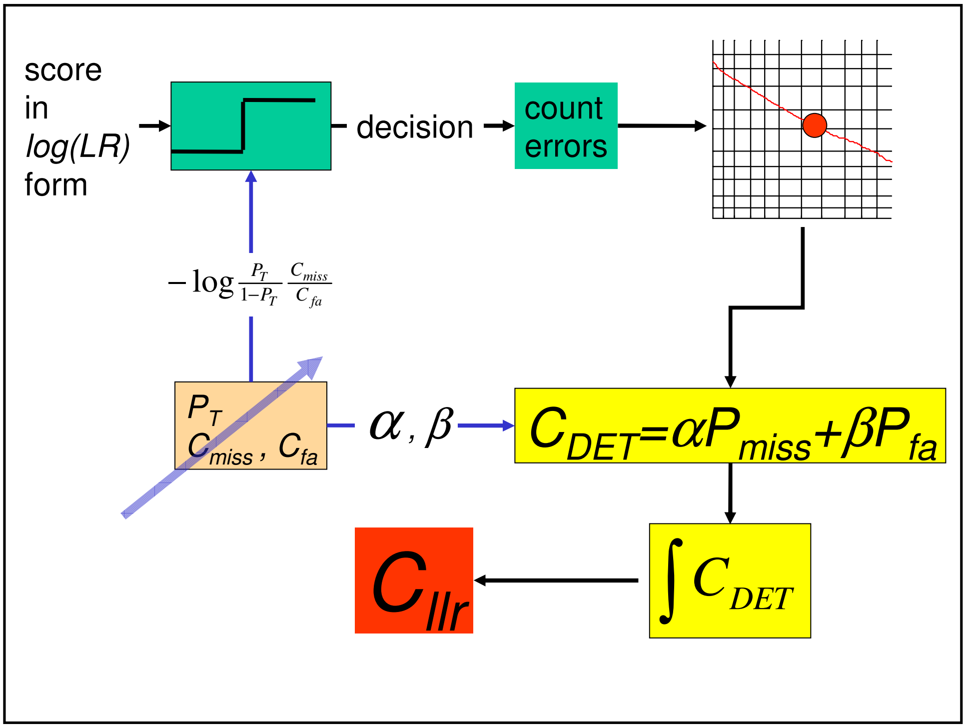
Calibration

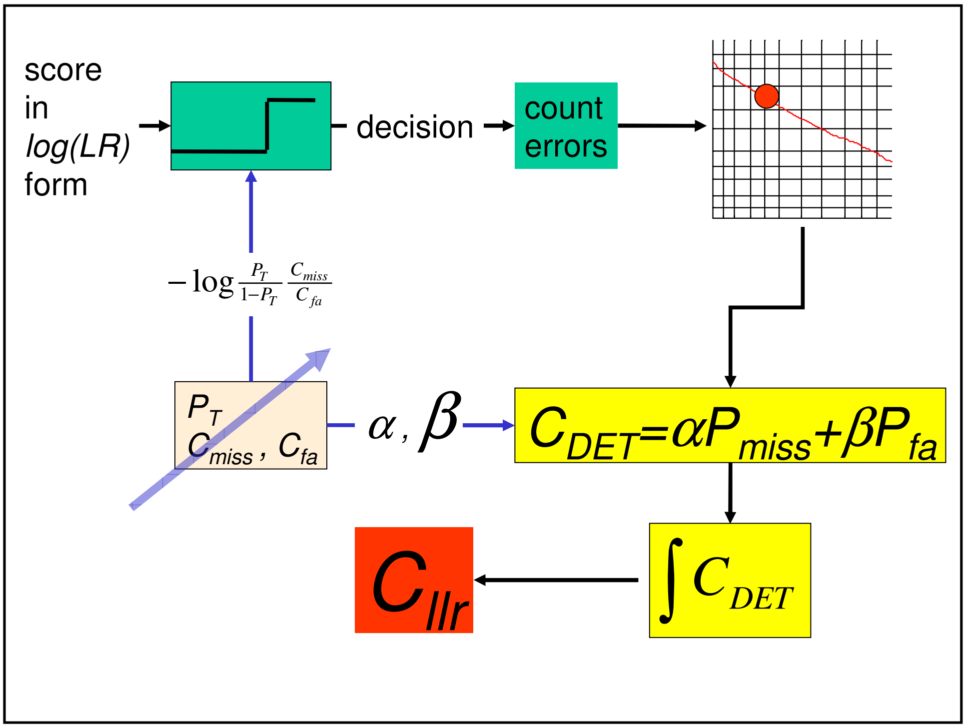
- If your application needs are more general, there are many good reasons (see my paper [12]) to use C_{llr} as evaluation objective rather than C_{DET}.
- C_{llr} represents performance over a wide range of operating points.
- *Calibration* is now the act of designing a mapping which outputs scores in log-likelihood-ratio format, such that C_{llr} (rather than C_{DET}) is optimized.

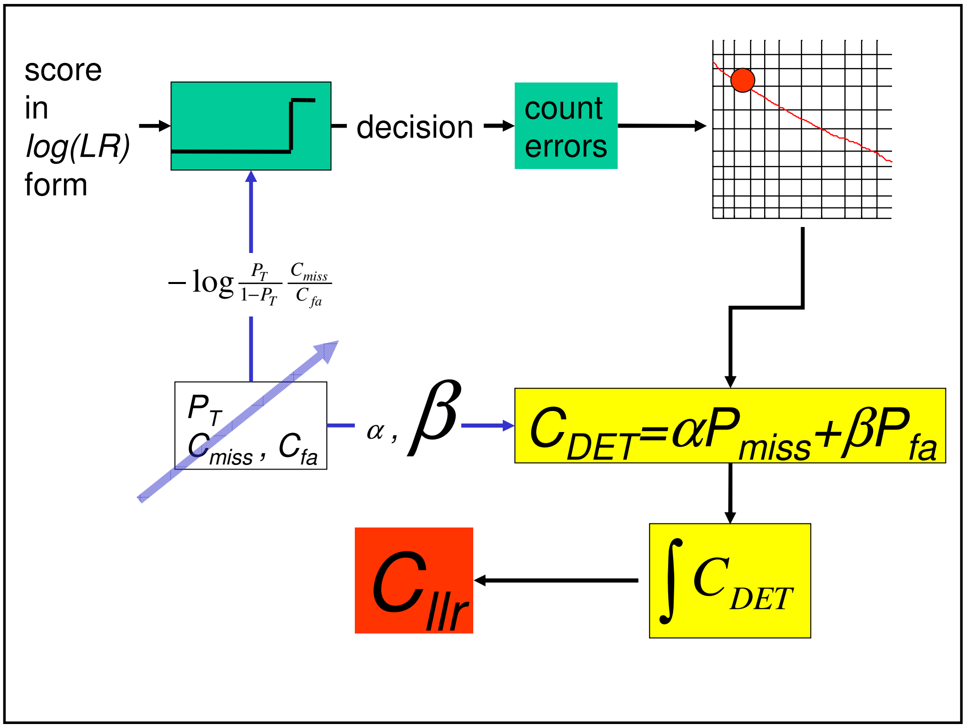
C_{llr} is simply C_{DET} integrated over a wide range of operating points.




score
in
log(LR)
form
decision
count
errors
$-\log \frac{P_T}{1-P_T}\frac{C_{miss}}{C_{fa}}$
P_T
C_{miss}, C_{fa}
α, β
$C_{DET}=\alpha P_{miss}+\beta P_{fa}$
C_{llr}
$\int C_{DET}$

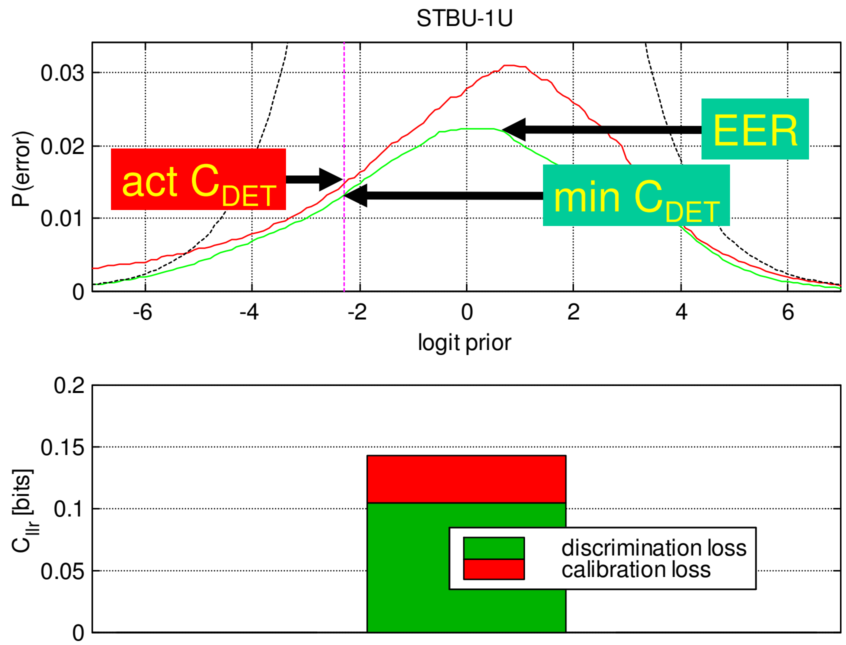

score
in
log(LR)
form
decision
count
errors
$-\log \frac{P_T}{1-P_T}\frac{C_{miss}}{C_{fa}}$
P_T
C_{miss}, C_{fa}
α, β
$C_{DET}=\alpha P_{miss}+\beta P_{fa}$
C_{llr}
$\int C_{DET}$

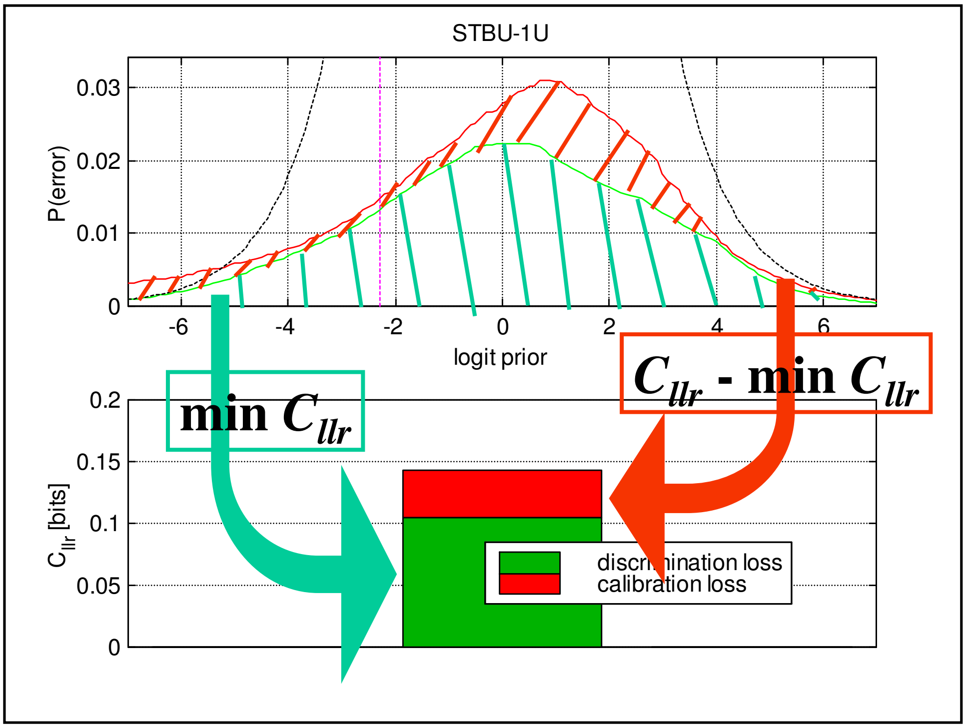

score in log(LR) form
decision
count errors
$-\log \frac{P_T}{1-P_T} \frac{C_{miss}}{C_{fa}}$
P_T
C_{miss}, C_{fa}
α, β
$C_{DET}=\alpha P_{miss}+\beta P_{fa}$
$\int C_{DET}$
C_{llr}


score in log(LR) form
decision
count errors
$-\log \frac{P_T}{1-P_T} \frac{C_{miss}}{C_{fa}}$
P_T
C_{miss}, C_{fa}
α, β
$C_{DET}=\alpha P_{miss}+\beta P_{fa}$
$\int C_{DET}$
C_{llr}

Measuring calibration

- **Point calibration** can be measured via the discrepancy between *actual* C_{DET} and *minimum* C_{DET} ,
- **General calibration** can be:
  - measured via discrepancy between *actual* C_{llr} and *minimum* C_{llr} .
  - analyzed with applied probability of error (APE) curves.





STBU-1 Calibration

11 sub-system scores

↓

logistic regression fusion ← prior = 0.0917 (to optimize wide operating range around C_{DET} operating point)

↓

log(LR) score

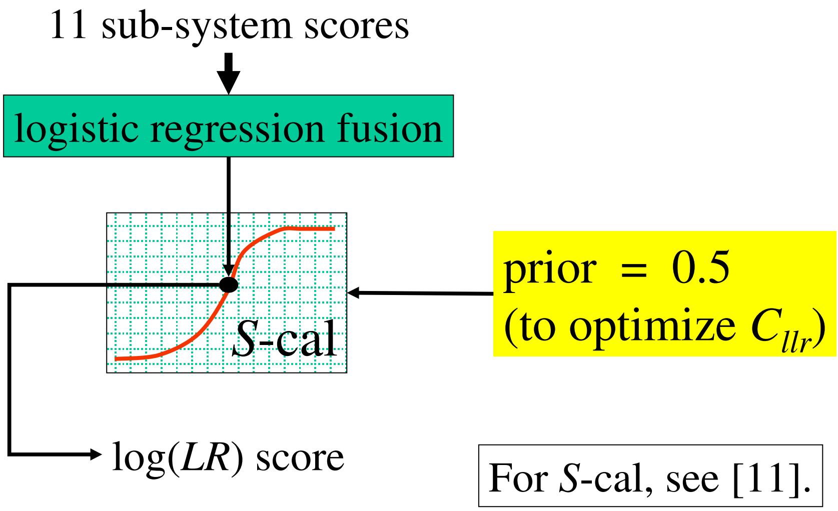
↓

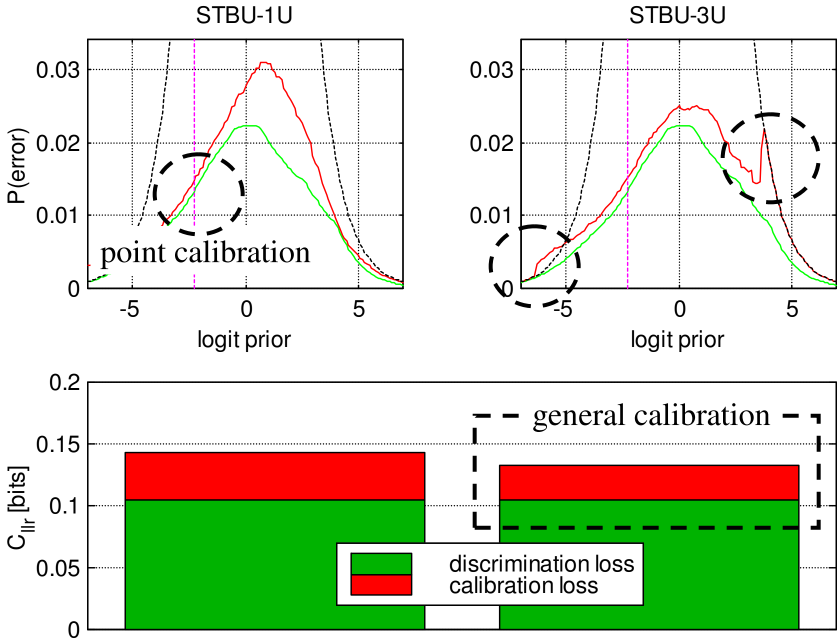
threshold @ 2.29

↓

decision

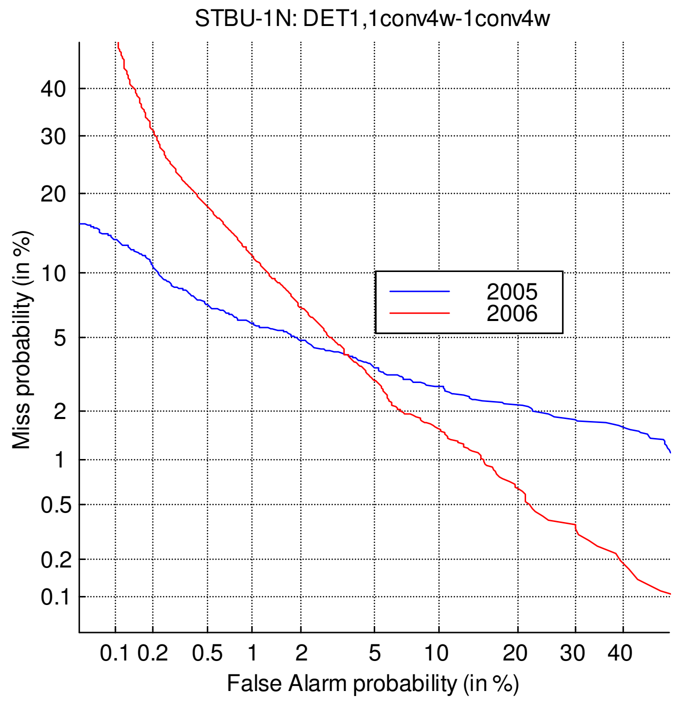
STBU-3 Calibration





Calibration complication

- For STBU-1, our development (2005) EER and eval (2006) EER are almost identical.
- But DET-curves are very different ….



Calibration complication

- The ratio of target and non-target score variances changed a lot between 2005 and 2006!
- This is part of the reason why our calibration is not quite as neat as last year.
- The challenge is therefore to get calibration more stable across different environments.

SPESCOM

Conclusion

- Fast short-time cepstrum systems are very effective when supervector subspace compensations are applied.
- No single system gets it quite right --- fusion between different systems usually helps.
- Slower ASR-based systems can add further value.

BRNO UNIVERSITY OF TECHNOLOGY
FIT FACULTY OF INFORMATION TECHNOLOGY

References

[1] Albert Strasheim's Python SVM code: See www.scipy.org (Available July 2006.)

[2] LibSVM: www.csie.ntu.edu.tw/~cjlin/libsvm/

[3] A. Stolcke et al. , "MLLR transforms as features in speaker recognition" in *Eurospeech*, 2005.

[4] A. Stolcke et al.,"Improvements in MLLR-tranform-based speaker recognition" in *Odyssey 2006.*

[5] Various papers by Patrick Kenny available here: www.crim.ca/perso/patrick.kenny/

References

[6] N. Brummer. "SDV NIST SRE'04 System description", 2004.

[7] R. Vogt et al. "Modelling session variability in text-independent speaker verification," in *Interspeech – Eurospeech* 2005.

[8] R. Vogt and S. Sridharan, "Experiments in session variability for modelling for speaker verification," in *ICASSP*, 2006.

[9] Alex Solomonoff et al."Channel Compensation for SVM Speaker Recognition", Odyssey 2004.

References

[10] W. M. Campbell et al., "SVM Based Speaker Verification Using a GMM Supervector and NAP Variability Compensation," ICASSP 2006.

[11] FoCal: Toolkit for Fusion and Calibration. See: www.dsp.sun.ac.za/~nbrummer/focal

[12] Niko Brümmer and Johan du Preez, "Application-independent evaluation of speaker detection",in *Computer Speech and Language,* 20, 2006: pp.230-275 .