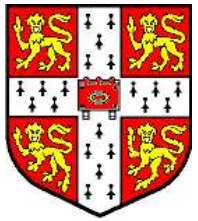
Discriminative Training for Speech Recognition

Dan Povey

May 2002



Cambridge University Engineering Department

IEEE ICASSP'2002

Overview

- MMI & MPE objective functions
- Optimisation of objective functions
  - Strong & weak-sense auxiliary functions
  - Application to Gaussians and weights
- Prior information: I-smoothing
- Lattices and MMI & MPE optimisation
- Other issues to consider in discriminative training
- Some typical improvements from discriminative training

Objective functions: MMI & ML

ML objective function is product of data likelihoods given speech file $\mathcal{O}_r$

$$\mathcal{F}_{\mathrm{ML}}(\lambda) = \sum_{r=1}^{R} \log p_\lambda \left(\mathcal{O}_r | s_r\right), \tag{1}$$

MMI objective function is posterior of correct sentence:

$$\begin{aligned} \mathcal{F}_{\mathrm{MMIE}}(\lambda) &= \sum_{r=1}^{R} \log \frac{p_\lambda \left(\mathcal{O}_r | s_r\right)^\kappa P(s_r)^\kappa}{\sum_s p_\lambda \left(\mathcal{O}_r | s\right)^\kappa P(s)^\kappa} \\ &= \sum_{r=1}^{R} \log P^\kappa \left(s_r | \mathcal{O}_r, \lambda\right) \end{aligned} \tag{2}$$

Objective functions: MPE

Minimum Phone Error (MPE) is the summed "raw phone accuracy" (#correct - #ins) times the posterior sentence prob:

$$\begin{aligned}\mathcal{F}_{\mathrm{MPE}}(\lambda) &= \sum_{r=1}^{R} \frac{\sum_s p_\lambda(\mathcal{O}_r|s)^\kappa P(s)^\kappa \mathrm{RawPhoneAccuracy}(s, s_r)}{\sum_s p_\lambda(\mathcal{O}_r|s)^\kappa P(s)^\kappa} \\ &= \sum_{r=1}^{R} \sum_s P^\kappa(s_r|\mathcal{O}_r, \lambda)\, \mathrm{RawPhoneAccuracy}(s, s_r) \qquad (3)\end{aligned}$$

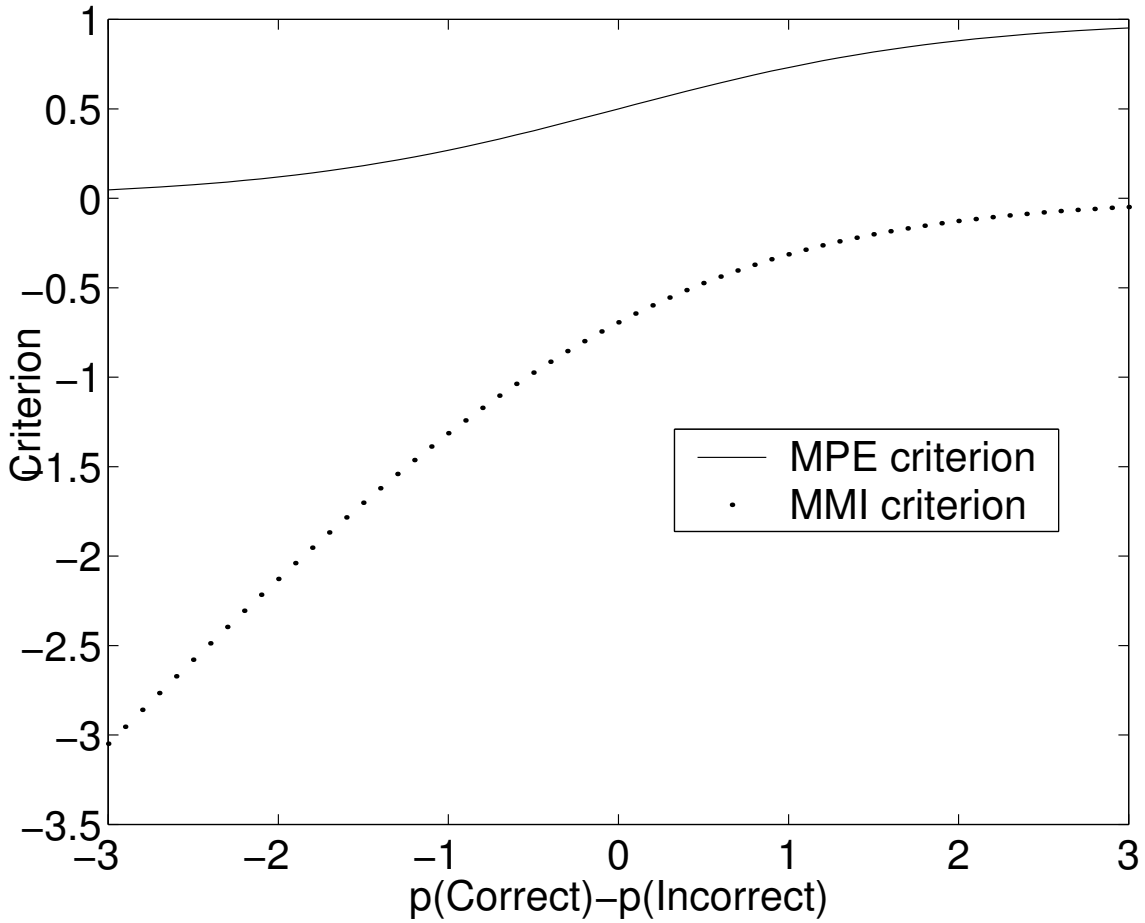
Equals the expected phone accuracy of a sentence drawn randomly from the possible transcriptions (proportional to scaled probability).

Objective functions: Simple example

- Suppose correct sentence is "a", only alternative is "b".
- Let $a = p_\lambda(\mathcal{O}|\text{"a"})P(\text{"b"})$ (acoustic & LM likelihood), b is same for "b".
- ML objective function $= \log(a) +$ other training files.
- MMI objective function $= \log(\frac{a}{a+b}) +$ other training files.
- MPE objective function $= \frac{a \times 1 + b \times 0}{a+b} +$ other training files.

Objective functions: Simple example (Continued)

Criteria shown graphically: MPE and MMI criteria as a function of $log(\frac{a}{b})$.



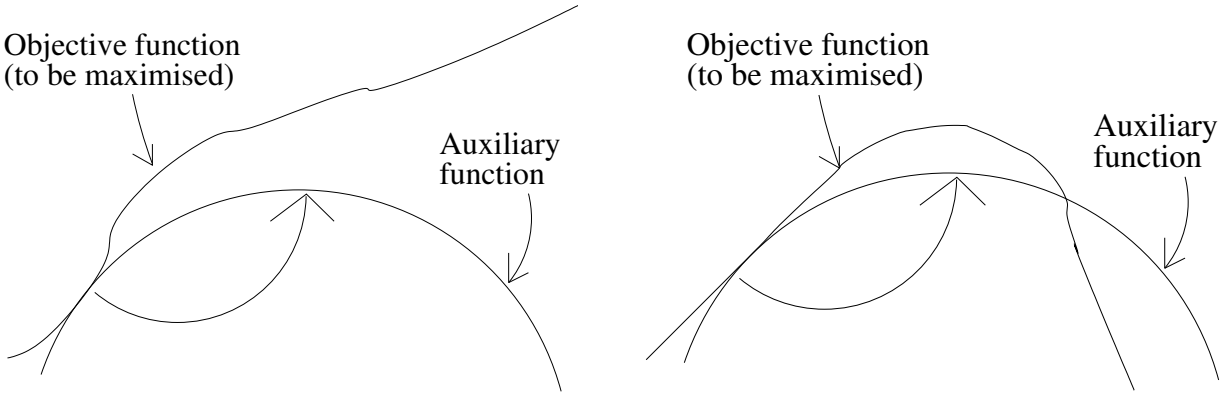
Objective functions: Further remarks on MPE

- MPE is sensitive to the "degree of wrongness" of wrong transcriptions.
- There is a related criterion, MWE, where we calculate accuracy based on a word level.
- (MWE doesn't work quite so well).

Optimisation of objective functions: preliminary remarks

- With ML training, there is a fast method available (Expectation-Maximisation)
- For MMI and MPE training, optimisation is more difficult
- Two general kinds of optimisation available: gradient based, and Extended Baum-Welch (EB)
- Be careful, because criterion optimisation $\neq$ test-set recognition !!
- Need to optimise the objective function in a "smooth" way
- Extended Baum-Welch (EB) is nice because it doesn't need second-order statistics

Auxiliary functions



(a) (b)

Use of (a) strong-sense and (b) weak-sense auxiliary functions for function optimisation

- Auxiliary functions are a concept used in E-M. Functions of (eg) HMM parameters λ
- Strong-sense auxiliary function: has the same value as real objective function at a local point $\lambda = \lambda'$, but $\leq$ objf everywhere else
- Weak-sense auxf has same differential around local point $\lambda = \lambda'$

Auxiliary functions & function maximisation

- To maximise a function using auxiliary functions, find the maximum of the auxiliary function, find a new auxiliary function around the new point and repeat
- With strong-sense auxiliary function, this is guaranteed to increase the function value on each iteration unless a local maximum has been reached (e.g. as in E-M)
- With weak-sense auxiliary function, there is no guarantee of convergence
- ... but if it $does$ converge it will converge to a local maximum
- Similar level of guarantee to gradient descent (which will only converge for correct speed of optimisation)
- Note– "weak-sense" and "strong-sense" are my terminology, normal terminology is different also involves the term "growth transformation."

Strong-sense auxiliary functions- beyond E-M

Example of using strong-sense auxiliary function to maximise something (not E-M):

- Suppose we want to maximise $\sum_{m=1}^{M} A_m \log x_m + B_m x_m$ for constants A_m, B_m, with constraint $\sum_{m=1}^{M} x_m = 1$ (will mention reason later)
- Suppose the current values of x_m are x'_m (for $m = 1 \ldots M$).
- For each m, add a +ve constant k_m times the function $(x'_m \log(x_m) - x_m)$ to the objective function.
- Function $k_m(x'_m \log(x_m) - x_m)$ for +ve k_m is convex with a zero gradient around the current values x'_m
- ... so can add this function to objf for each m & will get a strong-sense auxf
- Add this using appropriate values of k_m to make coeffs of x_m all the same, hence constant (due to sum-to-one constraint).
- Reduces to something of the form $\sum_{m=1}^{M} A_m \log x_m$ which can be solved

Weak-sense auxiliary functions– Mixture weights

Example of weak-sense auxf for MMI

- Optimising mixture weights for MMI
- For ML, we can get a (strong-sense) auxiliary function which looks like $\sum_{j=1}^{J} \sum_{m=1}^{M} \gamma_{jm} \log c_{jm}$ (plus other terms for Gaussians & transitions
- ... as in normal E-M. The above is a strong-sense auxiliary function for the log HMM likelihood
- For MMI, the objective function is one HMM likelihood (OK) minus another (Not OK)
- Call these $numerator$ (num) and $denominator$ (den) HMMs

Weak-sense auxiliary functions– Mixture weights (cont'd)

- Try $-\sum_{j=1}^{J}\sum_{m=1}^{M}\gamma_{jm}^{\mathrm{den}}\log c_m$ as a *weak-sense* auxf for second term
- But total auxf $\sum_{j=1}^{J}\sum_{m=1}^{M}(\gamma_{jm}^{\mathrm{num}}-\gamma_{jm}^{\mathrm{den}})\log c_m$ would not give good convergence (would set some mixtures to zero).
- Instead use $\sum_{j=1}^{J}\sum_{m=1}^{M}\gamma_{jm}^{\mathrm{num}}\log c_m-\gamma_{jm}^{\mathrm{den}}\frac{c_m}{c'_m}$.
- Same differential w.r.t. mixture weights where they equal old mixture weights c'_m.
- Can be maximised easily (see previous slide)
- Gives good convergence

Weak-sense auxiliary functions– Gaussians

- Normal auxiliary function for ML is
  $\sum_{j=1}^{J}\sum_{m=1}^{M} -0.5\left(\gamma_{jm}\log\sigma_{jm}^2 + \frac{\theta_{jm}(\mathcal{O}^2)-2\mu_{jm}\theta_{jm}(\mathcal{O})-\gamma_{jm}\mu_{jm}^2}{\sigma_{jm}^2}\right)$
  where $\theta_{jm}(\mathcal{O})$ and $\theta_{jm}(\mathcal{O}^2)$ are sum of data & data squared for mix m of state j.

- Abbreviate this to $\sum_{j=1}^{J}\sum_{m=1}^{M} Q(\gamma_{jm}, \theta_{jm}(\mathcal{O}), \theta_{jm}(\mathcal{O}^2)|\mu_{jm}, \sigma_{jm}^2)$.

- For MMI, a valid *weak-sense* auxiliary function for objf is
  $\sum_{j=1}^{J}\sum_{m=1}^{M} Q(\gamma_{jm}^{\mathrm{num}}, \theta_{jm}^{\mathrm{num}}(\mathcal{O}), \theta_{jm}^{\mathrm{num}}(\mathcal{O}^2)|\mu_{jm}, \sigma_{jm}^2)$
  $-Q(\gamma_{jm}^{\mathrm{den}}, \theta_{jm}^{\mathrm{den}}(\mathcal{O}), \theta_{jm}^{\mathrm{den}}(\mathcal{O}^2)|\mu_{jm}, \sigma_{jm}^2)$.

Weak-sense auxiliary functions– Gaussians (cont'd)

- Would not have good convergence, so add "smoothing function" $\sum_{j=1}^{J}\sum_{m=1}^{M} Q(D_{jm}, D_{jm}\mu'_{jm}, D_{jm}({\mu'_{jm}}^2, {\sigma'_{jm}}^2)|\mu_{jm}, \sigma^2_{jm})$ for a positive constant D_{jm} chosen for each Gaussian.

- This function has zero differential where the parameters equal the old parameters $\mu'_{jm}, {\sigma'_{jm}}^2$, so local gradient unaffected.

- Solving this leads to the EB update equations, e.g. (for the mean):

$$\mu_{jm} = \frac{\left\{\theta^{\text{num}}_{jm}(\mathcal{O}) - \theta^{\text{den}}_{jm}(\mathcal{O})\right\} + D_{jm}\mu'_{jm}}{\left\{\gamma^{\text{num}}_{jm} - \gamma^{\text{den}}_{jm}\right\} + D_{jm}}$$

- For good convergence set D_{jm} to $E\gamma^{\text{den}}_{jm}$ for e.g. $E = 1$ or 2

MPE optimisation

- For MPE, we dont have a difference of HMM likelihoods as in MMI.
- For Gaussians– Work out differential w.r.t. MPE objective function of each log Gaussian likelihood at each time t.
- Define $\gamma_{jm}^{\mathrm{MPE}}(t)$ as that differential.
- Use $\sum_{r,t,j,m} \gamma_{jm}^{\mathrm{MPE}}(t) \log \mathcal{N}(\mathbf{o}_r(t)|\mu_{jm}, \sigma_{jm}^2)$ as basic auxiliary function. Obviously has same differential as real objective function locally (where $\lambda = \lambda'$)
- The functional form of this is equivalent to the $Q(\dots)$ functions referred to above, with similar statistics required.

MPE optimisation

- Ensure convergence by adding "smoothing function"
  $\sum_{j=1}^{J} \sum_{m=1}^{M} Q(D_{jm}, D_{jm}\mu'_{jm}, D_{jm}({\mu'_{jm}}^2, {\sigma'_{jm}}^2)|\mu_{jm}, \sigma^2_{jm})$.

- Leads to EB equations, except statistics are gathered in a different way

- Set the constant D_{jm} based on a further constant E, in a similar way to MMI.

I-smoothing

- I-smoothing is the use of a prior distribution over the Gaussian parameters
- Mode of prior is at the ML estimate
- Prevents extreme parameter values being estimated based on limited training data
- Prior is $Q(\tau, \tau\frac{\theta_{jm}^{\mathrm{mle}}(\mathcal{O})}{\gamma_{jm}^{\mathrm{mle}}}, \tau\frac{\theta_{jm}^{\mathrm{mle}}(\mathcal{O}^2)}{\gamma_{jm}^{\mathrm{mle}}}|\mu_{jm}, \sigma_{jm}^2)$
- ... where mle refers to the ML statistics, and τ is a constant (e.g. 50)
- Very simple to implement in the context of the EB equations (all the terms inside the various $Q(\dots)$ functions can just be added together)
- Important for MPE: unless I-smoothing is used for robustness, MPE is worse than MMI
- I-smoothing can also improve MMI, but only slightly

Lattices and MMI/MPE optimisation

- Lattices are generated once and used for a number of iterations of optimisation
- 2 sets of lattices-
  - Numerator lattice (= alignment of correct sentence)
  - Denominator lattice (from recognition). [Needs to be big, e.g beam > 125]
- Lattices need time-marked phone boundaries:
- Can't do unconstrained forward-backward because:
  i) slow and ii) interferes with the probability scaling which is done at whole-model level

Lattices and MMI/MPE optimisation (cont'd)

- Optimisation involves two phases, as in ML: i) get statistics, ii) reestimate.
- Gathering statistics initially involves a forward (/backward) alignment of time-marked models, to get whole-model acoustic likelihoods
- For MMI, a forward-backward algorithm is done over the lattice at the phone level to get model occupation probabilities, and then stats are accumulated (for each of the 2 lattices separately)
- For MPE, see next slide...

Lattices and MPE optimisation

- For MPE, only align denominator lattice (numerator lattice is used to work out how correct den-lattice sentences are)
- Each phone HMM in the lattice has a given start and end time, use q to refer to these "phone arcs"
- Need to work out of differential of MPE objective function w.r.t. log acoustic likelihood of each arc q (can then work out differentials w.r.t. individual Gaussian likelihoods)
- Define $\gamma_q^{\mathrm{MPE}} = \frac{1}{\kappa}$ times this differential
- Can use $\gamma_q^{\mathrm{MPE}} = \gamma_q(c(q) - c_{\mathrm{avg}})$ where γ_q is occupation probability
- $c(q)$ is average correctness of sentences passing through arc q, weighted by scaled probability
- c_{avg} is average correctness of entire file
- Hence, differential is positive for arcs with higher-than-average correctness

Lattices and MPE optimisation (cont'd)

Can calculate $c(q) =$ correctness of q in two ways:

- (Both of these ways involve an algorithm similar to a forward-backward algorithm over the lattice)

- Approximate method:
  - Use a heuristic formula based on overlap of phones to calculate the approximate contribution of an individual phone arc to the correctness of the sentence
  - This method makes use of the time markings in the correct-sentence (numerator) lattice
  - Gives a value quite close to the "real" phone accuracy of paths

Lattices and MPE optimisation (cont'd)

- Exact method:
  - Turn the numerator (correct sentence) lattice into a sausage (in case of alternate pronunciations)
  - Do an algorithm which is like a forward-backward algorithm combined with token-passing algorithm as used for recognition (not quite as complex as normal token passing)
  - Token-passing part corresponds to getting the best alignment to the lattice; forward-backward part follows from the need to get a weighted sum over sentences encoded in the lattice
- In both cases, generally ignore silence/short pause phones for calculating accuracy
- Difference in recognition performance between approximate & exact versions is not consistent

Optimisation regime

- Generally use 4-8 iterations of EB, typically 4 for MMI and 8 for MPE
- Very quick– some discriminative optimisation techniques reported in the literature use 50-100 iterations
- Recognition is the aim, not optimisation! Too-fast optimisation can lead to poor test set performance
- "Smoothing constant" E (=1 or 2) and number of iterations of training are set based on recognition (on development test set)
- For MMI on Broadcast News (hub4), criterion divided by #frames typically increases from, say, -0.04 to -0.02 during training (0.0 = perfect)
- MPE on hub4: MPE criterion divided by #words increases from 0.78 to 0.88 during training (1.0 = perfect)

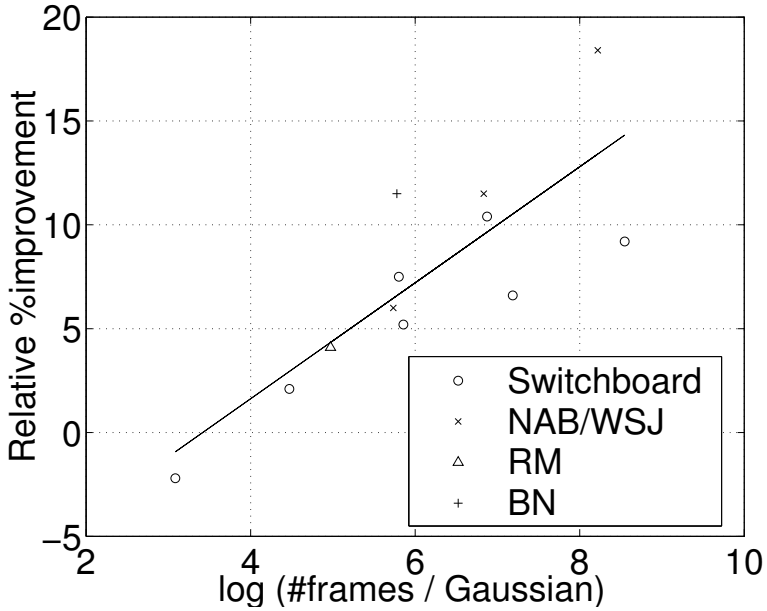
Practical issues for discriminative training

- Need to recognise all the training data– takes a long time
- Need to get phone marked lattices $\rightarrow$ need right software
- Important to use the scale κ rather than using unscaled probabilities; otherwise test set accuracy may not be very good
- κ typically in the range $\frac{1}{10}$ to $\frac{1}{20}$: generally equal to inverse of normal language model scale
- Essential to have a language model available (in HTK it is in the lattices)
- Unigram language model is best (generates more confusable words than a bigram)

MMI or MPE?

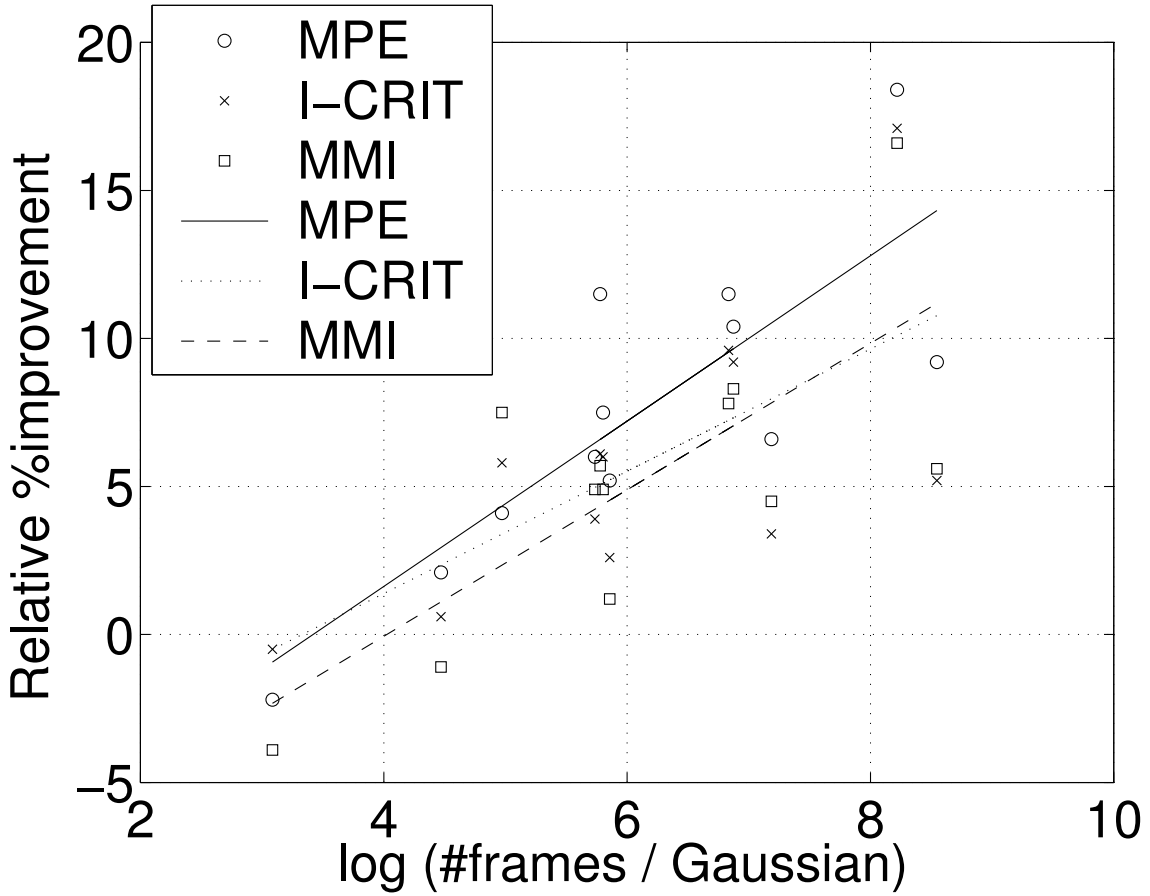
- MPE generally gives more improvement than MMI, especially where there is plenty of training data (see later)
- Compute time is similar for both criteria
- But MMI is easier to implement
- MPE implementation is built on top of MMI implementation so best to start with MMI

Improvements from MPE on various corpora



- Figure shows relative improvements from MPE on various corpora
- Shows that once we know the amount of training data available per Gaussian, improvement is predictable
- For typical systems as used for evaluations: 6% (WSJ), 11% (Swbd), 12% (BN) relative improvement

MMI & MPE on various corpora



- Figure shows relative improvement from MMI, I-smoothed MMI and MPE

- MPE best, but I-smoothed MMI nearly as good for limited training data (or too many Gaussians)

Interaction with other techniques

- How is the relative improvement from discriminative training affected by other techniques?
- Discriminative training gives most improvement for small HMM sets and large amounts of training data
- MLLR can sometimes (but not always) decrease improvement from discriminative training
- Discriminative training can be combined with SAT, which helps restore any lost improvement
- Discriminative training gives nearly as much improvment when tested on a different database

- Improvement slightly reduced when combined with HLDA
- Interaction with VTLN, CMN, clustering etc not investigated

Summary & conclusions

- Discriminative objective functions described (MMI and MPE)
- Mentioned the use of probability scaling (κ) in the objective functions
- Explained meaning of strong-sense & weak-sense auxiliary functions
- Described how weak-sense auxiliary functions justify EB update equations
- Described in general terms how the same approach is applied to MPE
- ... and how MPE objective function is differentiated within the lattice
- Mentioned I-smoothing (priors over Gaussian parameters)
- Gave typical results over various corpors, showing that improvement is a predictable of function of log(#frames/Gaussian)