fMPE & pMPE – A Discriminative Semi-parametric Trajectory Model

Khe Chai Sim & Mark Gales

27 February 2006

Cambridge University Engineering Department

MIL Speech Seminar

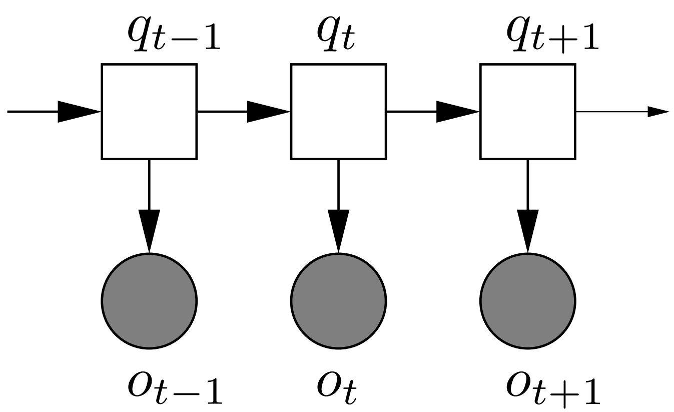
Outline

- Trajectory models for speech recognition
- A *semi-parametric* trajectory model
  - trajectory mean – fMPE
  - trajectory variance – pMPE
- Discriminative training – Minimum Phone Error (MPE)
- Experimental results on Conversational Telephone Speech (CTS) tasks
- Summary

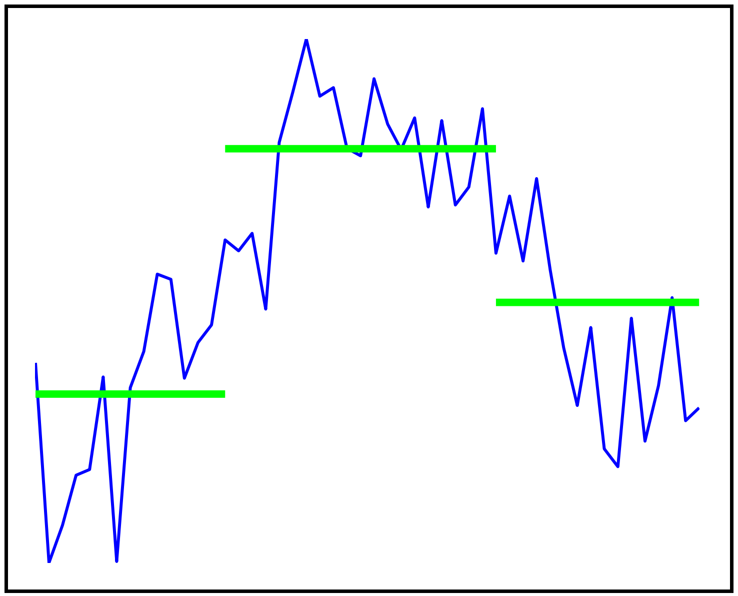
Motivation

- HMMs – commonly used for speech recognition
- Limitations of HMMs:
  - Conditional independence assumption of observations
  - Instantaneous state transition
  - Poor duration modelling
- Conditional independence assumption implies:
  - *constant* state output probability
- Ways to overcome this problem:
  - trajectory models (*e.g.* buried Markov Model, trajectory HMMs)
  - segment models (*e.g.* stochastic segment models, segmental HMMs)
  - switching linear dynamical systems

Standard HMM

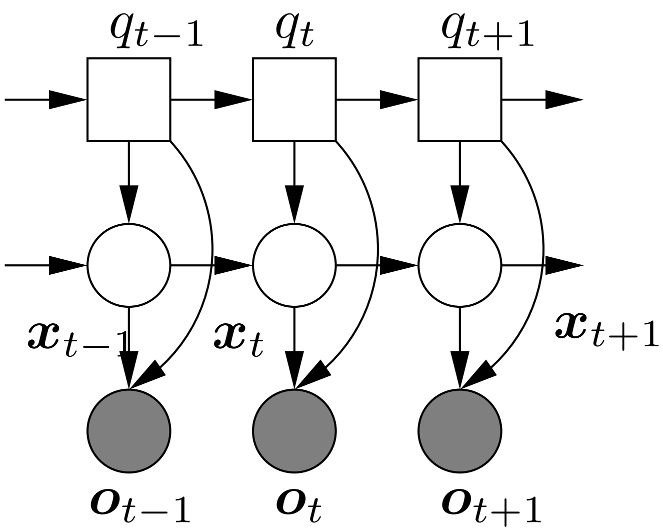


$$
\begin{aligned}
q_t &\sim P(q_t|q_{t-1}) \\
\boldsymbol{o}_t &\sim \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{\mu}_{q_t}, \boldsymbol{\Sigma}_{q_t})
\end{aligned}
$$

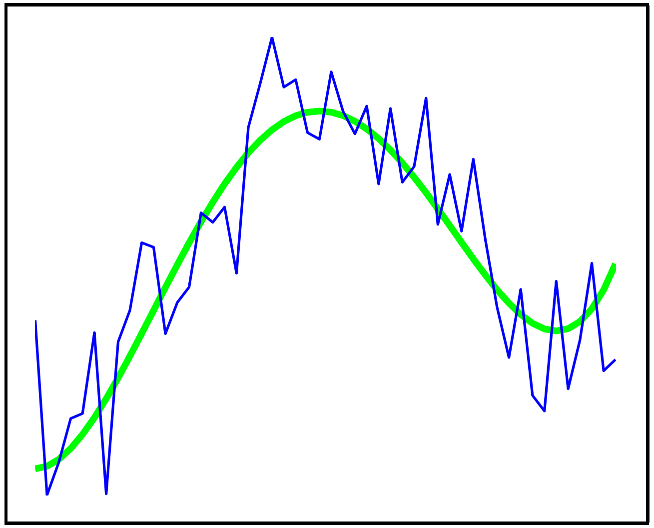


- Conditional independence assumption
- *Piece-wise constant* trajectory within state
- Poor trajectory modelliing via HMM state transition

Switching Linear Dynamical System



$$
\begin{aligned}
q_t &\sim P(q_t|q_{t-1}) \\
\boldsymbol{x}_t &= \boldsymbol{A}_{q_t}\boldsymbol{x}_{t-1} + \boldsymbol{u}_{q_t} \\
\boldsymbol{o}_t &= \boldsymbol{C}_{q_t}\boldsymbol{x}_t + \boldsymbol{v}_{q_t}
\end{aligned}
$$



- A state-space formulation
- Model smoothed trajectory via latent variables, $\boldsymbol{x}_t$
- *Time varying* mean: $\boldsymbol{\mu}_t = \boldsymbol{C}_{q_t}\boldsymbol{x}_t$

A Generic Trajectory Model Formulation

- A generic form of trajectory model: *non-stationary* state output distribution

$$\boldsymbol{o}_t \sim \sum_{m=1}^{M} c_{mt}\mathcal{N}\left(\boldsymbol{\mu}_t^{(m)}, \boldsymbol{\Sigma}_t^{(m)}\right)$$

- Consider time varying mean and covariance matrix
- Model time variation as a function of current (and neighbouring) observations:

$$\begin{aligned}\boldsymbol{\mu}_t^{(m)} &= f\left(\boldsymbol{\mu}^{(m)}, \mathcal{M}, \boldsymbol{o}_1, \ldots, \boldsymbol{o}_T\right)\\ \boldsymbol{\Sigma}_t^{(m)} &= g\left(\boldsymbol{\Sigma}^{(m)}, \mathcal{M}, \boldsymbol{o}_1, \ldots, \boldsymbol{o}_T\right)\end{aligned}$$

- What form should $f(.)$ and $g(.)$ take?
- How many frames around $\boldsymbol{o}_t$ should be considered?

Trajectory Mean – fMPE

- Apply a time dependent bias, b_{tj}, to jth element of mean

$$\mu_{tj}^{(m)} = f(\mu_j^{(m)}, b_j^{(i)}, \boldsymbol{o}_t) = \mu_j^{(m)} + b_{tj}$$

- Weighted interpolation of a set of bias vectors, $b_j^{(i)}$

$$b_{tj} = \sum_{i=1}^{n} h_{ti} b_j^{(i)} \quad \rightarrow \quad h_{ti} : \text{time varying interpolation weights}$$

- Equivalent to fMPE (Povey .et .al ICASSP 2005):

$$\hat{\boldsymbol{o}}_t = \boldsymbol{o}_t - \boldsymbol{M}\boldsymbol{h}_t \quad \text{where} \quad \boldsymbol{M} = \begin{bmatrix} \vdots & & \vdots \\ b_j^{(1)} & \cdots & b_j^{(n)} \\ \vdots & & \vdots \end{bmatrix} \quad \text{and} \quad \boldsymbol{h}_t = \begin{bmatrix} h_{t1} \\ h_{t2} \\ \vdots \\ h^{tn} \end{bmatrix}$$

Trajectory Covariance Matrix – pMPE

- Assume diagonal covariance matrix

- Apply a time varying *positive* scale factor, a_{tj}, to jth variance element

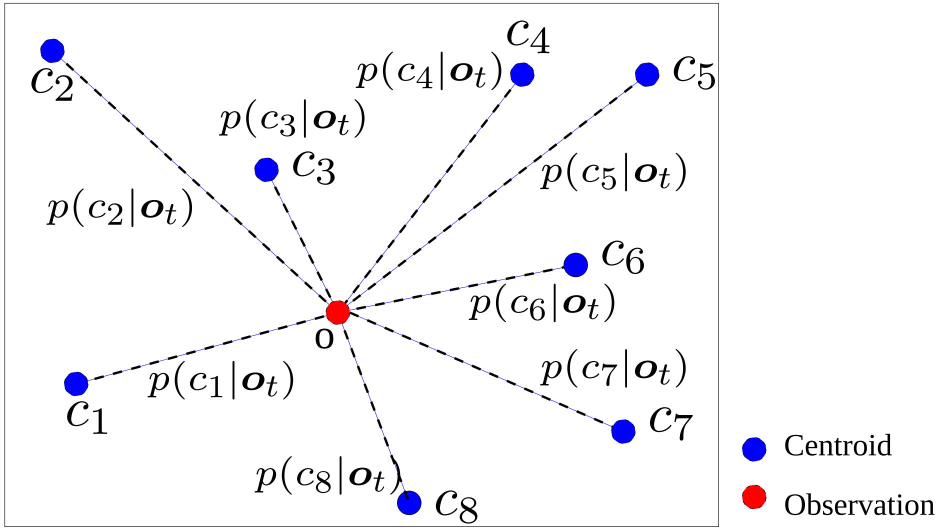
$$\sigma_{tj}^{2(m)} = g(\mu_j^{(m)}, a_j^{(i)}, \boldsymbol{o}_t) = \sigma_j^{2(m)} / a_{tj}$$

- Again, use weighted contribution:

$$a_{tj} = \left(1 + \sum_{i=1}^{n} h_{ti} a_j^{(i)}\right)^2 \quad \longrightarrow \quad \text{taking squared to ensure positive scale}$$

A Semi-parametric Representation

- Represent acoustic space with many *centroids*:

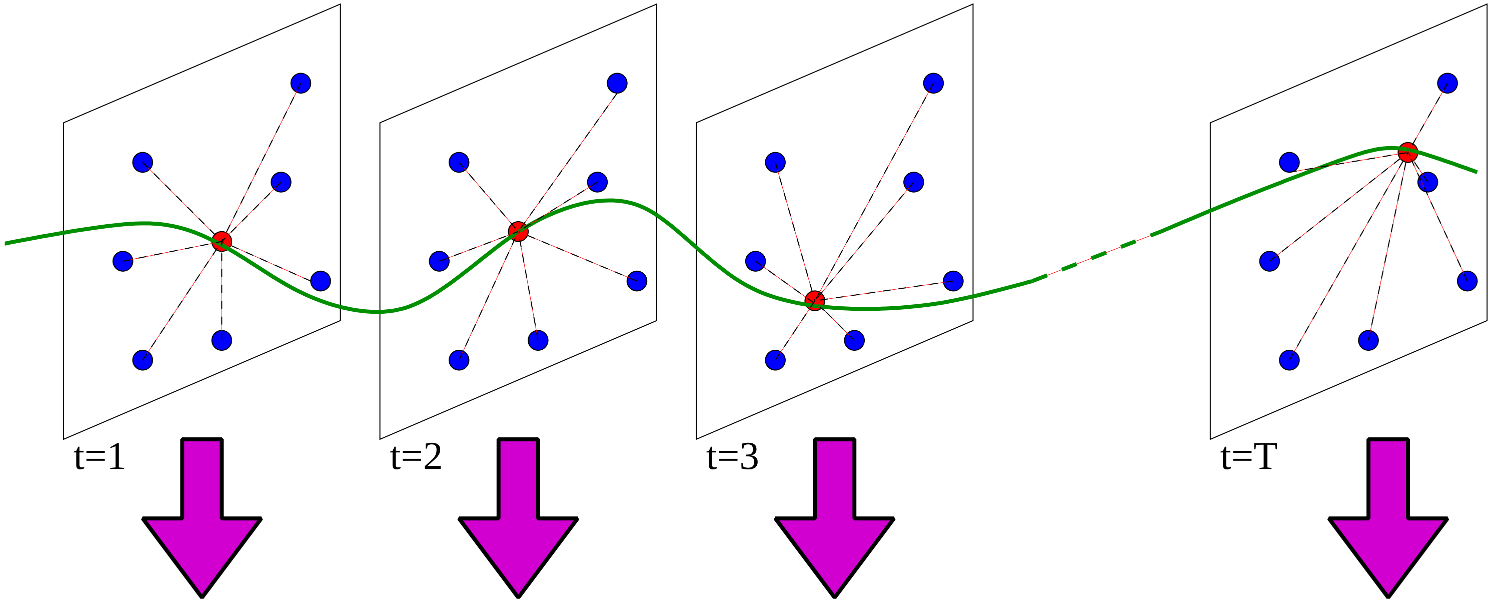


- Use centroid *posteriors* given $\boldsymbol{o}_t$ as weights:

$$h_{ti} = P(c_i|\boldsymbol{o}_t) = \frac{\mathcal{N}(\boldsymbol{o}_t; c_i)}{\sum_{j=1}^{n} \mathcal{N}(\boldsymbol{o}_t; c_j)}$$

- Parameters to be trained for each centroid: bias $b_j^{(i)}$ & scale $a_j^{(i)}$

Overall Smoothed Trajectory Parameters



$$(\boldsymbol{\mu}_1^{(m)}, \boldsymbol{\Sigma}_1^{(m)}) \; (\boldsymbol{\mu}_2^{(m)}, \boldsymbol{\Sigma}_2^{(m)}) \; (\boldsymbol{\mu}_3^{(m)}, \boldsymbol{\Sigma}_3^{(m)}) \; \cdots \; (\boldsymbol{\mu}_T^{(m)}, \boldsymbol{\Sigma}_T^{(m)})$$

- Overall trajectory – *weighted* contribution from each centroid
- Effective mean bias and precision scaling – smoothed over *centroids* and *time*
- May also include contributions from preceding and succeeding observations

Contexts Expansion

- Consider contributions from C observations on either sides ($2C+1$ frames)

$$b_{tj} = \sum_{i=1}^{n} \sum_{\tau=-C}^{C} w_i(t-\tau) b_{\tau j}^{(i)} \quad \text{and} \quad a_{tj} = \left(1 + \sum_{i=1}^{n} \sum_{\tau=-C}^{C} w_i(t-\tau) a_{\tau j}^{(i)}\right)^2$$

$$w_i(t-\tau) = \begin{cases} h_{ti} & \tau = 0 \\ h_{(t-\tau)i}/2 & \tau = \pm 1, \pm 2 \\ h_{(t-\tau)i}/3 \quad \text{for} & \tau = \pm 3, \pm 4, \pm 5 \\ h_{(t-\tau)i}/4 & \tau = \pm 6, \pm 7, \pm 8, \pm 9 \\ \vdots & \end{cases}$$

- For, $C = 9$, there are 7 biases and scales for each centroid
- Parameterisation: *static* $(\mu_j^{(m)}, s_j^{(m)})$ & *dynamic* $(b_{\tau j}^{(i)}, a_{\tau j}^{(i)})$

Minimum Phone Error (MPE) Training

- A discriminative training method — good improvement on LVCSR systems
- The MPE objective function:

$$\mathcal{F} = \sum_{H} P(H|\mathcal{O},\mathcal{M})l(H,\tilde{H})$$

  - $P(H|\mathcal{O},\mathcal{M})$ – posterior of hypothesis, H
  - $l(H,\tilde{H})$ – loss function of H given reference, $\tilde{H}$ (measure of phone error)

- MPE training of dynamic and static parameters together is complex
- Two gradient descent based training schemes:
  - Interleaved *dynamic-static* parameters update
  - Direct *dynamic* parameters update

Interleaved Dynamic-Static Parameters Update

- Key element — gradient (*complete* differential):

$$\frac{d\mathcal{F}}{d[b_j^{(i)}, a_j^{(i)}]} = \frac{\partial\mathcal{F}}{\partial[b_j^{(i)}, a_j^{(i)}]} + \sum_{m=1}^{M}\left(\frac{\partial\mathcal{F}}{\partial\mu_j^{(m)}}\frac{\partial\mu_j^{(m)}}{\partial[b_j^{(i)}, a_j^{(i)}]} + \frac{\partial\mathcal{F}}{\partial\sigma_j^{(m)2}}\frac{\partial\sigma_j^{(m)2}}{\partial[b_j^{(i)}, a_j^{(i)}]}\right)$$

- $\mu_j^{(m)}$ and $\sigma_j^{(m)2}$ depends on $a_j^{(i)}$ and $b_j^{(i)}$
  - to simplify update: use ML update formulae
  - if only use partial differential – gain lost after ML training
- Interleave between:
  - Dynamic parameters update – MPE (gradient-based optimisation)
  - Static parameters update – ML (simple closed-form)
  - Slow – requires 3 passes over training data

Direct Dynamic Parameters Update

- Start from a MPE trained system:
  - Assume static parameters are well-trained
  - Gradient w.r.t. static parameters $\approx$ zero
  - Only update the dynamic parameters

- Complete differential simplifies to a partial differential:

$$\frac{d\mathcal{F}}{d[b_j^{(i)}, a_j^{(i)}]} \approx \frac{\partial \mathcal{F}}{\partial [b_j^{(i)}, a_j^{(i)}]}$$

- Quicker to train – requires 1 pass over training data

Implementation Issues

- Likelihood calculation (given model parameters, $\mathcal{M}$):

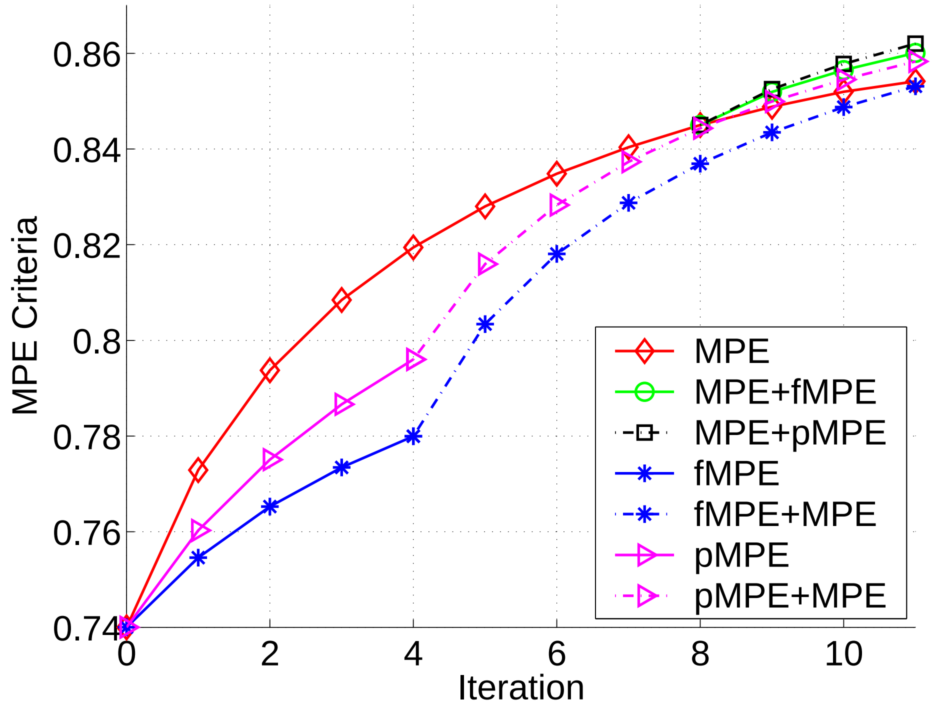
$$p(\boldsymbol{o}_t|\mathcal{M}) = K - \frac{1}{2}\sum_{j=1}^{d}\left(\log(\sigma_j^{2(m)}) - \log\left(a_{jt}^2\right) + a_{jt}^2\frac{\left(o_{jt} - b_{tj} - \mu_j^{(m)}\right)^2}{\sigma_j^{2(m)}}\right)$$

- For fMPE:
  - No additional cost (cache shifted observation $o_{jt} - b_{tj}$)
- For pMPE:
  - cache a_{jt}^2 and $\sum_{j=1}^{d}\log(a_{jt}^2)$
  - extra d multiplications and 1 addition
- Precision scale flooring:
  - more likely to overtrain pMPE parameters
  - apply *flooring* to the scaling factor

Experimental Setup – CTS English

- Acoustic model data sets:
  - **Training data**: 297 hours
  - **Test data**: 6 hours `eval03`
- Front-end: PLP (static, `C0`, Δ, Δ^2, Δ^3) + CMN + CVN + VTLN + HLDA
- HMM acoustic models (gender independent)
- Baseline acoustic models:
  - 16 component GMMs
  - Decision tree state clustered triphones (~ 6000 states)
- Trigram language models
- ~ 100k centroids without context expansion

MPE Criterion Gain – CTS English



- Interleaved update: fMPE+MPE & pMPE+MPE
- Direct update: MPE+fMPE & MPE+pMPE
- MPE gain: MPE > pMPE > fMPE
- Final systems have similar MPE criteria (0.85-0.86)

Interleaved Update – CTS English

System	Iter 0	Iter 8
MPE	31.9	28.6
fMPE+MPE	30.1	27.8
pMPE+MPE	30.7	28.4
fMPE+pMPE+MPE	29.9	27.9

Unadapted WER performance of 16-component models on `eval03`

- Gains over ML: *1.8%* (fMPE) and *1.2%* (pMPE)
- Gains over MPE: *0.8%* (fMPE+MPE) and *0.2%* (pMPE+MPE)
- Combinining fMPE and pMPE gave *0.2%* gain over fMPE
- Gain disappears after subsequent MPE training
  - possibly due to over training

Direct Update – CTS English

System	Training Method	% WER
MPE	—	28.6
fMPE+MPE	Interleaved	27.8
MPE+fMPE	Direct	28.0
pMPE+MPE	Interleaved	28.4
MPE+pMPE	Direct	28.3

Unadapted WER performance of 16-component models on `eval03`

- Quicker to train compared to interleaved update
- Yield similar performance (slighly better for pMPE systems)
- Direct updates yield smaller gains for systems with *context expansion*
  - *partial differential* approximation not valid

CTS English State-of-the-art Performance

System	Iter 0	Iter 8
MPE	27.5	22.8
fMPE+MPE	24.5	21.6

Unadapted WER performance of 36-component models on `eval03`

- Trained on approx. 2200 hours of `fsh2004h5etrain03b`
- Standard MPE alone — 4.7% absolute WER reduction
- fMPE (with $C = 9$ context expansion) gain — 3.0% absolute over ML
- Overall fMPE+MPE gain — 5.9% (1.2% over standard MPE)
  - Increasing # components in standard system gives only small improvements

Experimental Setup – CTS Mandarin

- Acoustic model data sets:
  - **Training data**: 72 hours
  - **Test data**: 2 hours `dev04`
- Front-end: PLP (same as English system) + pitch + Gaussianisation
- HMM acoustic models (gender independent)
- Baseline acoustic models:
  - 1 and 16 components GMMs
  - Decision tree state clustered triphones (~ 4000 states)
- Trigram language models
- ~ 4k centroids with a window of 1 and 19 frames ($C = 9$)

Single Component CTS Mandarin Results

System	Frames	
	1	19
MPE	44.4	44.4
fMPE+MPE	42.1	40.1
pMPE+MPE	43.3	41.3
fMPE+pMPE+MPE	41.6	38.9

Unadapted CER performance of 1-component models on `dev04`

- Gains over MPE:
  - 1 frame: gains for fMPE (*2.3%*) and pMPE (*1.1%*)
  - 19 frames: more gains for fMPE (*4.3%*) and pMPE (*3.1%*)
- Good improvement from fMPE+pMPE+MPE:
  - gave a further *0.5%* (1 frame) and *1.2%* (19 frames) over fMPE+MPE
- More parameters for fMPE/pMPE systems
  - *but*, only one active Gaussian component per state in decoding

16-component CTS Mandarin Results

System	Frames	
	1	19
MPE	36.0	36.0
fMPE+MPE	35.6	34.4
pMPE+MPE	35.9	35.4
fMPE+pMPE+MPE	35.3	34.7

Unadapted CER performance of 16-component models on `dev04`

- Gains over MPE:
  - 1 frame: small gains for fMPE (*0.4%*) and pMPE (*0.1%*)
  - 19 frames: larger gains for fMPE (*1.6%*) and pMPE (*0.4%*)
- Combining fMPE and pMPE:
  - 1 frame: gained further *0.3%*; 19 frames: 0.3% degradation
  - degradation may be due to over-training on limited data

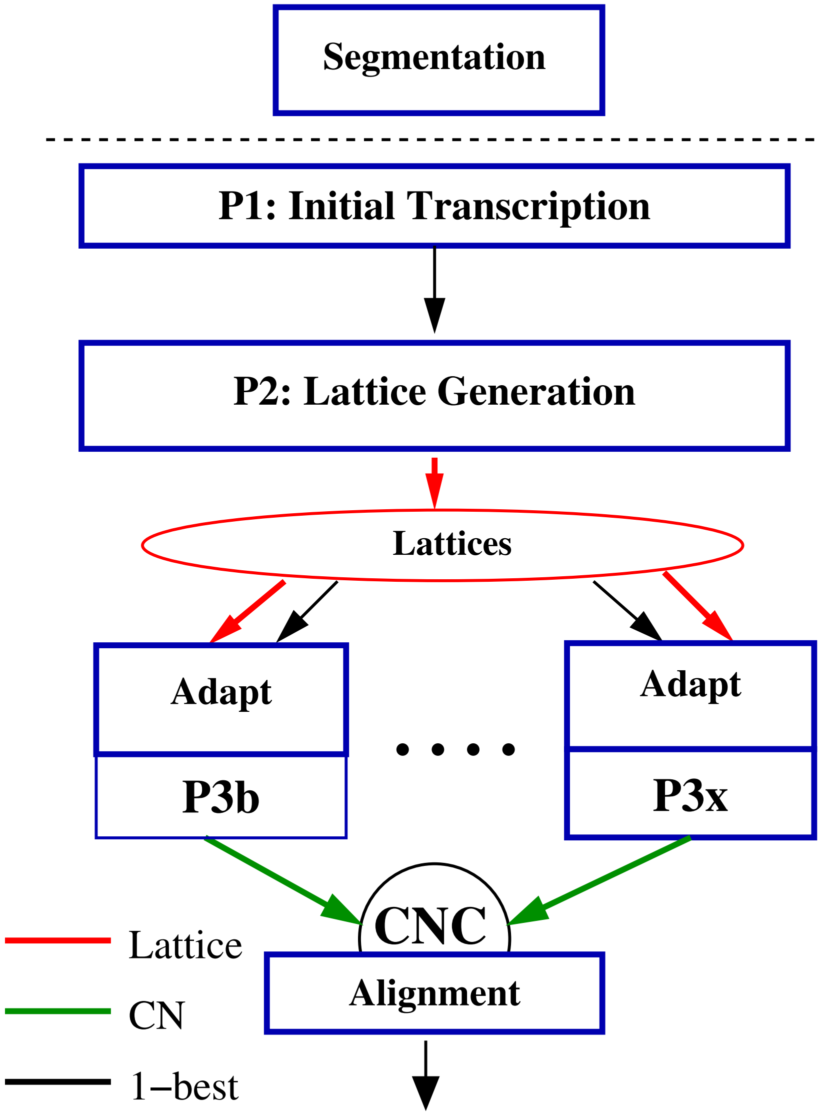
Confusion Network Combination CTS Mandarin Results

System	dev04		eval04	
	Viterbi	CN	Viterbi	CN
MPE	36.0	35.0	33.9	33.4
fMPE+MPE	34.4	33.9	32.5	32.2
fMPE+pMPE+MPE	34.7	34.0	33.1	32.6
CNC	—	33.3	—	31.6

Unadapted CER performance of 16-component models on `dev04` and `eval04`

- Using 19 frames window for fMPE and pMPE
- Confusion network (CN) decoding: *0.3%* average absolute gain
- Confusion network combination (CNC): further *0.6%* absolute improvement

10xRT Evaluation Framework



- Evaluation 10xRT framework:
- Multi-pass framework
- Confusion network generation
- Confusion network combination
- Adaptation in P3 stage:
  - 1-best CMLLR
  - lattice-based mean MLLR

State-of-the-art CTS Mandarin Performance

System	CER (%)		
	HLDA	+GAUSS	+fMPE
DIAGC	35.8	34.6	33.5
+SAT	35.0	33.7	33.0
+SPAM	34.2	33.2	32.7

Adapted CN performance of various systems evaluated on `dev04`

- Baseline HLDA frontend with DIAGC system: (*35.8%*)
- Different acoustic models:
  - SAT: Speaker Adaptive Training
  - SPAM: an efficient precision matrix modelling scheme
- Using improved frontends:
  - Gaussianisation: gain (*1.0%*) – (*1.3%*) absolute
  - fMPE: gain (*0.5%*) – (*1.1%*) absolute
- Trade-off between frontend and acoustic model refinements.

Summary

- Investigated semi-parametric trajectory model:
  - Trajectory mean (fMPE)
  - Trajectory variance (pMPE)
- Discriminatively trained using MPE criterion
- Gave improvement over baseline:
  - fMPE typically gave 1.0–1.5% absolute gain over MPE alone
  - Gains from pMPE smaller compared to fMPE
  - Gains of fMPE and pMPE not additive
    - disappears as system complexity increases
- Successfully applied on state-of-the-art CTS English and Mandarin systems