Efficient Gaussian Selection for Continuous Density Acoustic Models

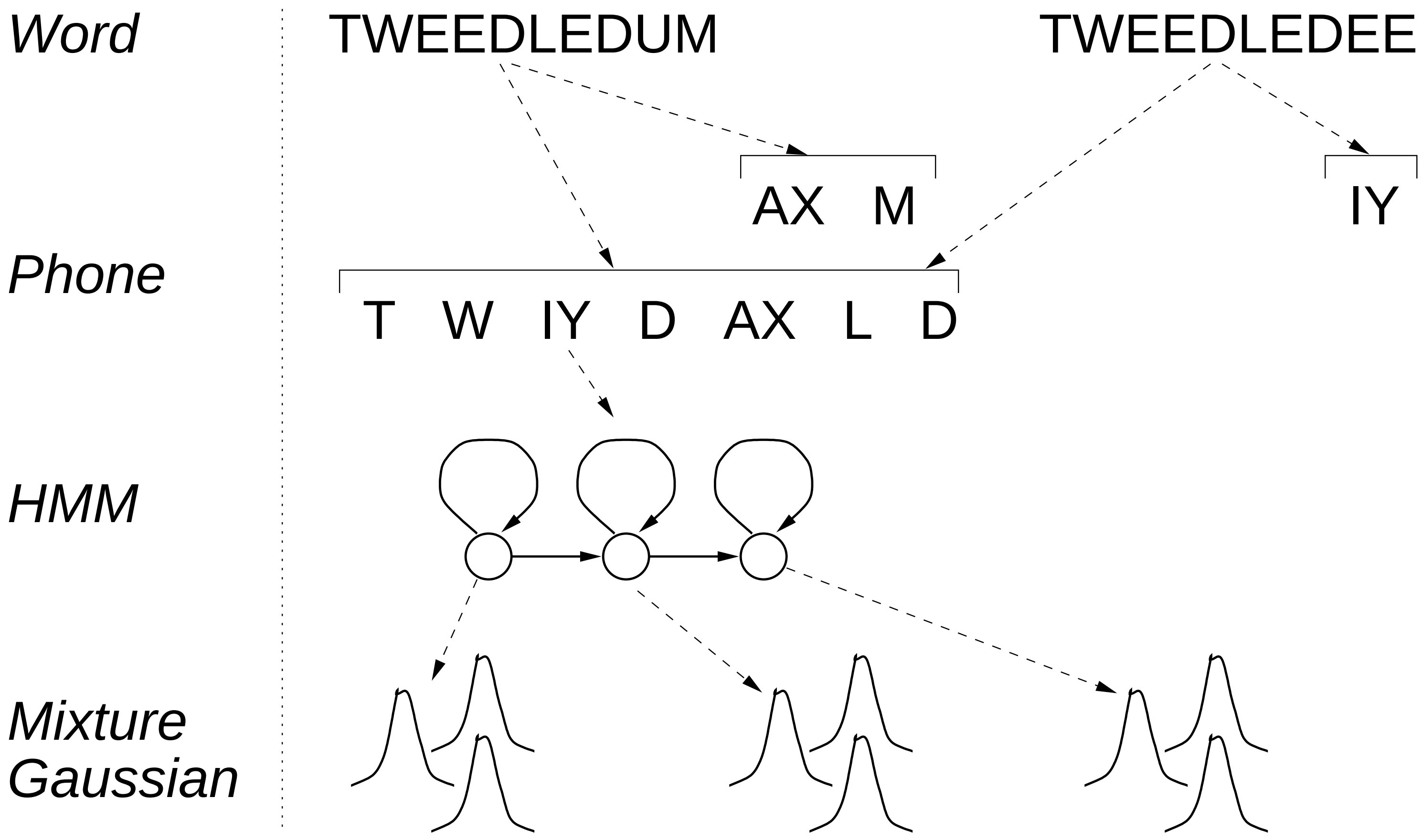
Mosur Ravishankar

Carnegie
Mellon

School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

April 22, 1999

Continuous Density Acoustic Models



Continuous Density Acoustic Models (contd.)

Gaussian density evaluation: Comparing an input speech vector $\overline{x}$ to a Gaussian density $\mathcal{N}(\overline{\mu}, \overline{\sigma})$:

$$-\sum_i \frac{(x_i - \mu_i)^2}{\sigma_i^2}$$

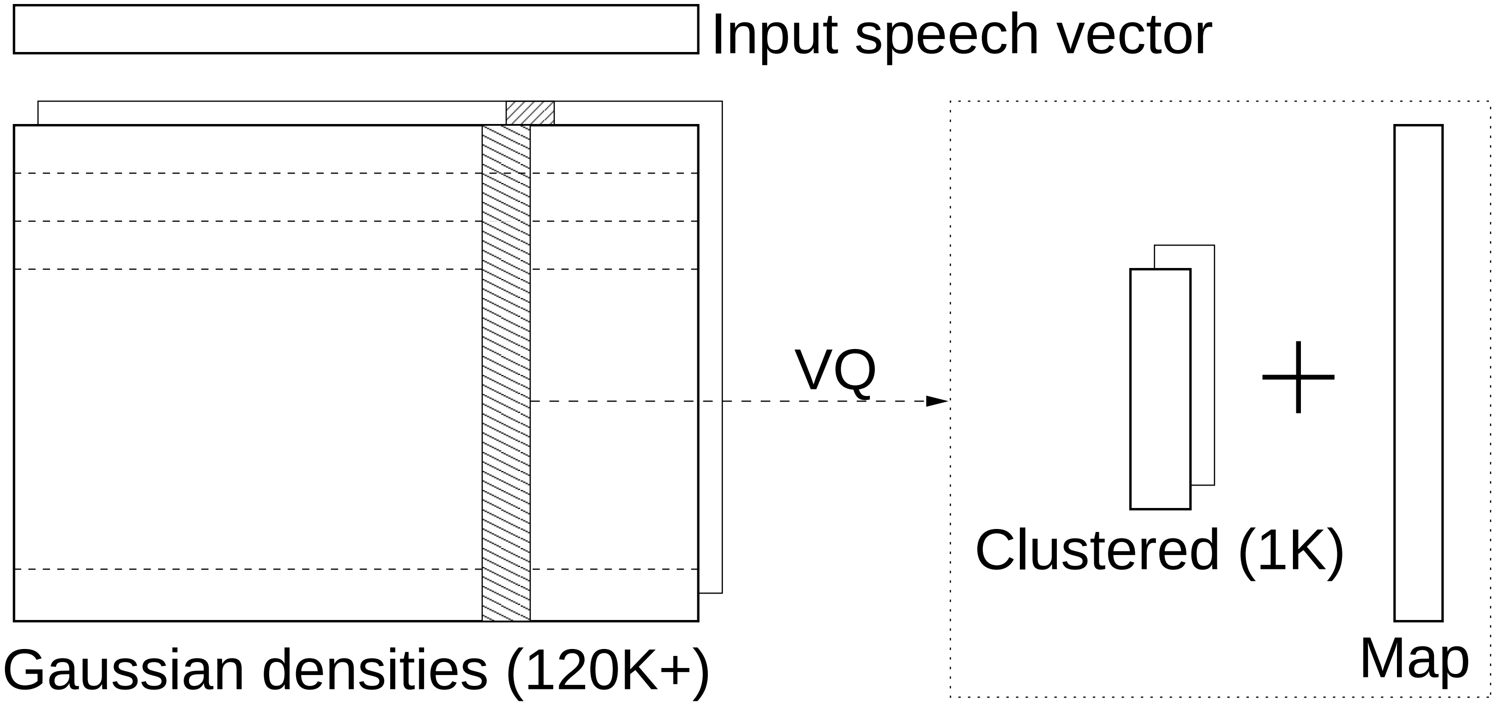
In the Nov97 Broadcast News model:

- 39-dimensional vectors
- 20 densities per mixture (*i.e.*, per state)
- 6290 states in acoustic model

So, in each 10msec frame:

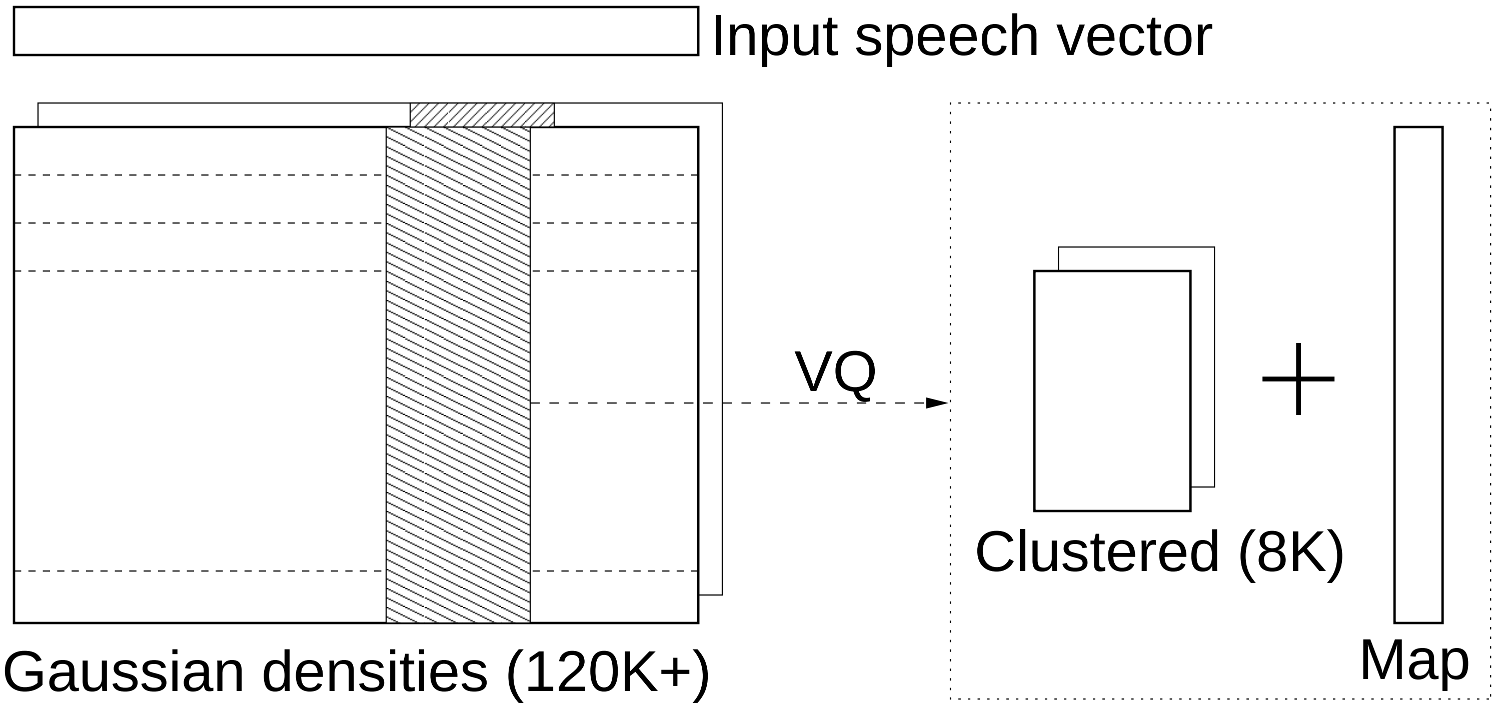
- 6290 x 20 x 39 x 4 (19.6M) arithmetic operations.

The Eurospeech 1997 Idea: Sub-Vector Clustering



- 4 arithmetic operations replaced by $1(+\epsilon)$
- Recognitions (accuracy) unaffected by quantization errors
- Bonus: Smaller acoustic model (throw away the original)

Even Better: Wider Sub-Vectors



- Only one addition operation per sub-vector per density ($+\epsilon$)
- But larger clustered tables: larger ϵ
- Quantization errors do affect recognition accuracy
- Bigger bonus: Even smaller acoustic model

The Catch: Memory Access

The devil is in the details:

- Hidden cost: one extra memory access per sub-vector per density
- Memory access more costly than floating point arithmetic operation (unless memory cache is humming along)
- Cache performance much less effective with sub-vectors
- Memory access latency limits speed-up, unless no. of sub-vectors small
- But *fewer* $\Rightarrow$ *wider* sub-vectors; higher quantization and recognition errors

We need:

- Small number of sub-vectors for speed
- Accuracy of original model for recognition performance

Sub-VQ Based Gaussian Selection (New Stuff)

Gaussian selection:

- In most (99+%) cases, 1 or 2 Gaussians in mixture determine state score
  - i.e., only about 10% Gaussians are *active* in a frame
- Problem: Identifying this *shortlist* of active Gaussians

Solution:

1. As before, use sub-vector codebook to evaluate each Gaussian approximately
2. In each mixture, select 1 or 2 top Gaussians (the shortlist)
3. Re-evaluate the shortlist using *original* model

Question: Accuracy of active Gaussian selection

Sub-VQ Based Gaussian Selection (contd.)

Sub-vector codebook configuration:

- Three sub-vectors: cepstra, Δcepstra, $\Delta\Delta$cepstra
- 4K entries in each sub-vector cluster
- A *shortlist beam* used to pick the Gaussian shortlist for each state (mixture)

Gaussian Selection Evaluation: Accuracy

Test set: 100K frames (10%) from Hub4-1997 evaluation set

Shortlist Beam	Shortlist Size(%)	Selection Accuracy(%)
1.0	5.0	54
0.1	8.2	69
0.01	12.8	80
0.001	18.4	88
0.0001	24.9	92
0.00001	31.8	95
0.000001	38.8	97

- *Shortlist size*: No. of Gaussians in the shortlist
- *Selection accuracy*: No. of states for which true best Gaussian correctly selected

Gaussian Selection Evaluation: Cross Entropy

Is 80% shortlist accuracy good enough?

- If the 20% missed states are low scoring ones, the errors are unimportant
- If high scoring ones missed, however, bad for recognition

Measure test set *cross-entropy* between complete model and shortlist model. In each frame:

1. Evaluate state scores using all densities; normalize to sum to 1
2. Evaluate state scores using shortlist densities; normalize
3. Compute cross-entropy between the two sets
4. Normalize by test set entropy of complete model

Gaussian Selection Evaluation: Cross Entropy (contd.)

Test set: 100K frames (10%) from Hub4-1997 evaluation set

Shortlist Beam	Shortlist Size(%)	Selection Accuracy(%)	Cross Entropy
1.0	5.0	54	1.560
0.1	8.2	69	1.143
0.01	12.8	80	1.040
0.001	18.4	88	1.013
0.0001	24.9	92	1.003
0.00001	31.8	95	1.002
0.000001	38.8	97	1.001

Trivia: Entropy using complete model is about 4 ($\sigma = 1.7$)

- State scores are distributed very, very sharply

Gaussian Selection Evaluation: WER and Speed

Test set: Hub-4 1997 evaluation set, excluding segments tagged as F2 (telephone bandwidth)

Wide search beam, 0.1 shortlist beam:

WER:	F0	F1	F3	F4	F5	FX	Tot	Speed
Base	17.2	26.2	36.1	31.4	35.1	63.3	26.8	53x
Test	17.7	26.3	37.6	32.8	35.1	65.2	27.5	11x

Narrow search (TREC) beam, 0.1 shortlist beam:

WER:	F0	F1	F3	F4	F5	FX	Tot	Speed
Base	20.0	27.9	38.3	35.3	36.8	64.7	29.3	53x
Test	20.5	28.4	38.8	36.4	37.4	65.7	29.9	11x

- About 2.5% relative increase in word error rate
- About 5x faster acoustic evaluation (on 200MHz PPRO)

An Aside: Exploiting the Memory Hierarchy

Memory system:

- 256-512MB real slow memory: 50 cycles access time
- 256KB-8MB faster memory: 5-10 cycles L2 cache
- 8-32KB real fast memory: 1-2 cycles L1 cache

Acoustic model size: 30-40MB
Q: How can it be squeezed into the 8KB L1 cache?
A: A little bit at a time ($blocking$):

- Acoustic model size for 1 state: 20x39x2x4 = 6KB+
- Evaluate 1 state for several frames before evaluating the next state
- Speeds up evaluation by 1.5-2x

What Next?

- Try Gaussian selection with *blocking*
- Evaluation against other methods
- A little matter of speeding up search