PART IIB 2011 PROF PC WOODLAND

Solutions: 4F11 Speech and Language Processing, 2011

1. *HMM training*

(a) HMM assumptions: [15%]

- The observations accurately represent the signal. Speech is assumed to be stationary over the length of the frame. Frames are usually around 25msecs, so for many speech sounds this is not a bad assumption.
- Observations are independent given the state that generated it. Previous and following observations do not affect the likelihood. This is not true for speech, speech has a high degree of continuity.
- Between state transition probabilities are constant. The probability of from one state to another is independent of the observations and previously visited states. This is not a good model for speech.

(b)(i)The backward probability needs to be defined so that it can be combined with $\alpha_j(t)$. Hence

$$\beta_j(t) = p(\mathbf{o}_{t+1}, \mathbf{o}_{t+2}, \ldots, \mathbf{o}_T \,|x(t) = j, \lambda)$$

It can be computed recursively backwards in time:
Initialisation:

$$\beta_j(T) = a_{jN} \qquad 1 < j \leq N$$

Recursion: [20%]

```
for $t = T-1, T-2, \ldots, 2, 1$
...for $j = N-1, N-2, \ldots, 1$
```

$$\beta_j(t) = \sum_{k=2}^{N-1} a_{jk} b_k(\mathbf{o}_{t+1}) \beta_k(t+1)$$

(b)(ii) To find the posterior probability first note that multiplying the $\alpha_j(t)$ and $\beta_j(t)$ yields

$$p(x(t) = j, \mathbf{O} \,|\lambda) = \alpha_j(t)\beta_j(t)$$

and hence [10%]

$$L_j(t) = P(x(t) = j \,|\mathbf{O}, \lambda) = \frac{1}{p(\mathbf{O}|\lambda)} \alpha_j(t)\beta_j(t)$$

where $p(\mathbf{O}|\lambda)$ can be computed from either the α or β values e,g,

$$p(\mathbf{O}|\lambda) = \sum_{k=2}^{N-1} \alpha_k(T) a_{kN}$$

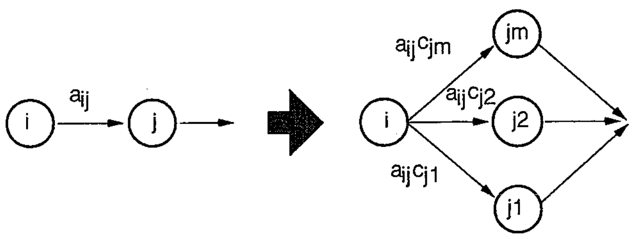
(b)(iii) Re-estimation formula for the mean parameter vector:

$$\hat{\boldsymbol{\mu}}_j = \frac{\sum_{t=1}^{T} L_j(t)\boldsymbol{o}_t}{\sum_{t=1}^{T} L_j(t)}$$

If multiple observation sequences need to sum the numerator statistics and the denominator statistics over the multiple training sequences:

$$\hat{\boldsymbol{\mu}}_j = \frac{\sum_{r=1}^{R}\sum_{t=1}^{T^{(r)}} L_j^{(r)}(t)\mathbf{o}_t^{(r)}}{\sum_{r=1}^{R}\sum_{t=1}^{T^{(r)}} L_j^{(r)}(t)}$$

(b)(iv) For Gaussian mixture distributions each of the component Gaussians may be considered as a separate state thus:



The posterior probability of a particular Gaussian component of a particular state is:

$$\begin{aligned} L_{jm}(t) &= P(x(t) = jm|\mathbf{O}, \mathcal{M}) \\ &= \frac{1}{p(\mathbf{O}|\mathcal{M})}\sum_{i=2}^{N-1} \alpha_i(t-1)a_{ij}c_{jm}b_{jm}(\mathbf{o}_t)\beta_j(t) \end{aligned}$$

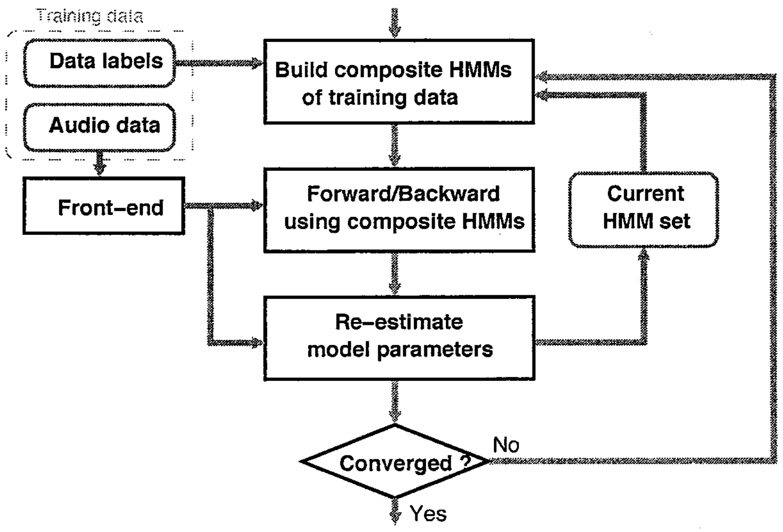
The estimates of the mean will be the similar to the single Gaussian case [20%]

$$\hat{\boldsymbol{\mu}}_{jm} = \frac{\sum_{t=1}^{T} L_{jm}(t)\mathbf{o}_t}{\sum_{t=1}^{T} L_{jm}(t)}$$

(c) Can extend the use to continuous speech by using composite models of complete utterances. These could be initialised from isolated word trained models or from a flat start (i.e. all models with equal arbitrary initial model parameters). Could use segmentation into word-units via the Viterbi algorithm (given current models) or (better) use sentence-level Baum-Welch training. [15%]

Q1 examiner's comment:

The question covered basic HMM assumptions and maximum likelihood estimation. It was generally well done, with all candidates showing that they understood the basic concepts.



Training data
Data labels
Audio data
Front-end
Build composite HMMs of training data
Forward/Backward using composite HMMs
Current HMM set
Re-estimate model parameters
Converged ?
No
Yes

2. *Features of a Large Vocab Speech Recogniser*

(a) MFCCs. Take the log energies of the filter-bank energies m_i and apply a discrete cosine transform.

$$c_n = \sqrt{\frac{2}{d}} \sum_{i=1}^{d} m_i \cos\left[\frac{n(i-\frac{1}{2})\pi}{d}\right]$$

where d is the number of filter-bank channels. This will reduce the dimensionality of the feature vector and (mainly) diagonalise it so that diagonal covariance matrices are more appropriate. It will reduce computation (smaller feature vector). lead to more robust estimation, and significantly improve recognition performance. [20%]

(b) Differential coefficients. Add time differentials of MFCCs and the normalised energy (or c_0). These can be added with a regression formula of the type:

$$\Delta \boldsymbol{y}_t = \frac{\sum_{\tau=1}^{D} \tau(\boldsymbol{y}_{t+\tau} - \boldsymbol{y}_{t-\tau})}{2\sum_{\tau=1}^{D} \tau^2}$$

where D determines the size of the *delta window* and the differential is taken as the best straight line through this window. The second differentials can be added in a similar way.

This adds information within a state on time evolution and overcomes (to an extent) the state-conditional independence assumption. There are more parameters (factor of 3), but much better recognition, so with pruning computational load can be lower. [20%]

(c) Full Covariance Matrix. This allows the correlations between feature elements to be explicitly modelled for each Gaussian component. For a feature vector of dimension d there $\frac{d(d+1)}{2}$ covariance parameters. If this $d = 39$ then there are 780 covariance parameters. The extra modelling capability is not normally needed for

MFCCs (although recently there has been renewed interest with very large training sets and more memory/computation available).

There is a large increase in storage, and computation requirements. Not normally needed when you have enough diagonal covariance models, but can bring a small improvement in performance if there is enough training data. [15%]

(d) Cross-Word Triphones. Convert each monophone to a cross-word context-dependent phone model (model depends on the immediate left and right phone context as well as the phone itself and the context extends across word boundaries). It allows co-articulation to be explicitly modelled. This complicates training as the number of models and potential parameters are increased.

The number of models is greatly increased and some type of smoothing or parameter sharing is required. The training data is very unevenly distributed and to get the correct trade-off between allocation of parameters and available data need to reduce the number of contexts. Note that there will be some contexts with no training data. A common method is state-tying via phonetic decision trees are used. These group contexts so that models can be robustly estimated for the grouped contexts and unseen triphones can be dealt with. The phonetic decision tree is grown automatically from training in a top-down fashion with questions chosen so as to maximise an approximate likelihood of the training data. The questions for splits are chosen from a pool of linguistic questions which yield generalisation ability. The advantage of this method is that it generalises including linguistic information and allows unseen contexts to be properly dealt with. It is widely used in state-of-the-art systems.

Overall there is a large increase in the number of parameters but it can be scaled to the amount of training data. Performance will significantly improve but the network structure in recognition is much more complex and hence needs to be designed to deal with a bigram if used in conjunction with other features such as cross-word triphones. [25%]

(e) A bigram model models the conditional probability of the next word on just the previous word.

$$P(w(k)|w(1)\dots w(k-1)) \approx P(w(k)|w(k-1))$$

whereas a unigram ignores all context.

Assuming that there is sufficient data to estimate the bigram and that suitable discounting and back-off strategies are employed then large gains in performance will flow (e.g. halve error rate on some tasks, and further gains with higher-order N-grams).

While a unigram model allows a monophone network to be tree-structured, the bigram adds complexity but the increase in accuracy and hence ability to perform pruning are very important here and so N-grams (now normally tri-gram or 4-gram) will be used in all modern speech recognition systems. [20%]

This question covered a discussion of various components of a large vocabulary speech recognition system including front-end analysis; full covariance models; cross-word triphones, decision-tree state tying and bigram language models. It was generally well answered.

3. *Machine Translation and HMM alignment models*

(a) Morphologically rich languages and the differences in word order between languages make machine translation difficult because [10%]

- Morphologically rich languages may require complex morphological analysis to be integrated into translation, and surface variability due to morphological variants can lead to sparsity of individual tokens in text translations
- Movement due to characteristic word order and lead to exponential complexity growth with sentence length

(b)(i) The BLEU score is calculated as follows (key points to mention, can be less detailed): [20%]

- Set N to be the order of the highest n-gram to be considered; here N=4
- For each sentence i, and for $n = 1, \ldots, N$, gather the following n-gram counts:
  - c_n^i : the number of hypothesized n-grams
  - $\bar{c}_n^i$: the number of correct n-grams, where the contribution of each distinct n-gram is *clipped* to the maximum number of occurrences in any one reference
  - Compute the precision for each n-gram , $n = 1, \ldots, N$: $p_n = (\sum_i \bar{c}_n^i) / (\sum_i c_n^i)$
  - Calculate the Brevity Penalty
    - Compute the shortest reference length : $r = \sum_i \min\{|E^i_{(1)}|, |E^i_{(2)}|, |E^i_{(3)}|, |E^i_{(4)}|\}$
    - Compute the hypothesis length : $c = \sum_i |E^i|$

$$BP = \begin{cases} 1 & c > r \\ \exp(1 - \frac{r}{c}) & c <= r \end{cases}$$

  - The BLEU score is

$$\text{BLEU} = BP * \exp\{\sum_{n=1}^{N} \log \frac{p_n}{N}\}$$

(b)(ii) The second hypothesis *mt2* has a slightly higher BLEU score, because it matches one more unigram. [10%]

	1-gram	2-gram	3-gram	BLEU
mt1	5/9	2/8	1/7	$\sqrt[3]{10/504}$
mt2	6/9	2/8	1/7	$\sqrt[3]{12/504}$

(c) Making simplifying conditional independence assumptions, we have that:

$$\begin{aligned} P(f_1^J, a_1^J, J|e_1^I) &= P(f_1^J|a_1^J, J, e_1^I)P(a_1^J, |J, e_1^I)P(J|e_1^I) \\ &= \prod_{j=1}^{J} P_T(f_j|e_{a_j})P_A(a_1^J|J, I)P_L(J|I) \end{aligned}$$

For Model 1:

$$P(f_1^J, a_1^J|e_1^I) = \frac{1}{I^J}\, p_L(J|I) \prod_{j=1}^{J} p_T(f_j|e_{a_j})$$

For Model 2:

$$P(f_1^J, a_1^J|e_1^I) = p_L(J|I) \prod_{j=1}^{J} p_T(f_j|e_{a_j})\, p_{M2}(a_j|j, J, I)$$

Differences:

- Model-1 assumes that the link distribution is entirely flat (all positions equally probable)
- Model-2 assumes that the link distribution depends on the foreign word location j [25%]

(d)Step 1 : Accumulate position alignment statistics over the word-aligned sentences

$$\#_{M2}(i, j, J, I) = \sum_{r=1}^{R} 1(J = J^{(r)}, I = I^{(r)}) \sum_{j=1}^{J} 1(i = a_j^{(r)})$$

Step 2 : Compute the Position Alignment Distribution

$$p_{M2}(i\,|\,j, J, I) = \frac{\#_{M2}(i, j, J, I)}{\sum_{i''=1}^{I} \#_{M2}(i'', j, J, I)}$$

Examples: [25%]

$$p_{M2}(i = 1|j = 2, J = 4, I = 5) = \frac{\#_{M2}(i = 1, j = 2, J = 4, I = 5)}{\#_{M2}(j = 2, J = 4, I = 5)} = \frac{1}{2}$$

$$p_{M2}(i = 0|j = 2, J = 4, I = 5) = \frac{\#_{M2}(i = 0, j = 2, J = 4, I = 5)}{\#_{M2}(j = 2, J = 4, I = 5)} = \frac{1}{2}$$

(e) The 'hard counts' are replaced by posterior probabilities.

$$\#_{M2}(i, j, J, I) = \sum_{r=1}^{R} 1(J = J^{(r)}, I = I^{(r)}) \sum_{j=1}^{J} P(i = a_j^{(r)}|E^{(r)}, F^{(r)})$$

This means than rather than performing a Viterbi-based estimation given a single alignment, all possible alignment are considered under a certain probability distribution. This will normally generate better estimates. [15%]

This question covered machine translation, estimation of the BLEU score and calculations on HMM alignment for MT. There was a very wide range of quality of attempts including one script with full marks and one with zero.

4. *Machine Translation and Weighted Finite State Networks*

(a) Source-Channel Formulation: Input - a Foreign sentence F , Output - an English sentence $\widehat{E}$ [20%]

$$\widehat{E} = \operatorname*{argmax}_E P(E|F) \;= \operatorname*{argmax}_E \frac{P(F|E)\,P(E)}{P(F)} = \operatorname*{argmax}_E \;\underbrace{P(F|E)}_{\substack{\textit{Translation}\\ \textit{Model}}} \quad \underbrace{P(E)}_{\substack{\textit{Source}\\ \textit{Language Model}}}$$

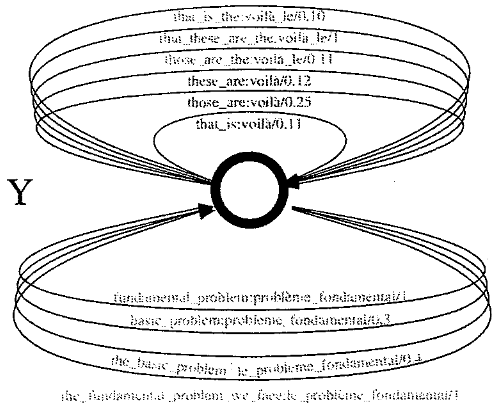
The Language Model $P(E)$ can be estimated using *monolingual text* using the same techniques developed for use in speech recognition. The Translation Model $P(F|E)$ can be estimated from *parallel texts.*

(b)(i) The steps are: [20%]

$$\underbrace{E = e_1^I}_{\textit{English Words}} \longrightarrow \underbrace{u_1^K}_{\textit{English Phrases}} \longrightarrow \underbrace{v_1^K}_{\substack{\textit{Foreign Phrases,}\\ \textit{English Order}}} \longrightarrow \underbrace{v_1^K, a_1^K}_{\substack{\textit{Foreign Phrases,}\\ \textit{Foreign Order}}} \longrightarrow \underbrace{F = f_1^J}_{\textit{Foreign Wor}}$$

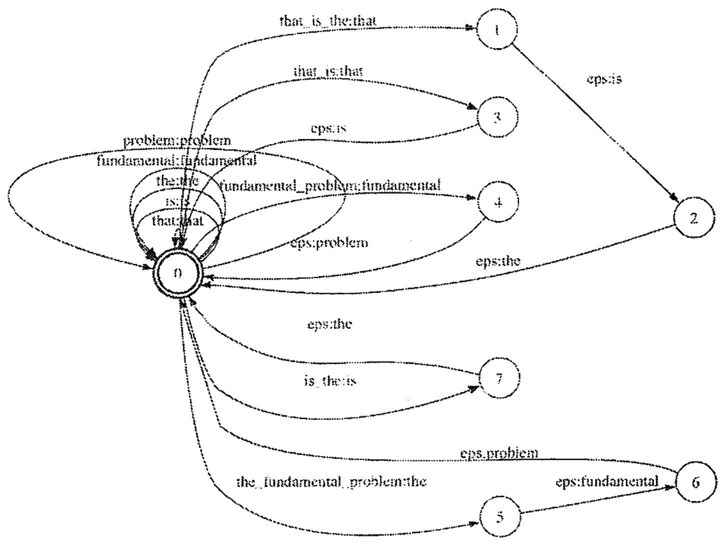
- Source Phrase Segmentation – $P(u_1^K|e_1^I)$ maps word sequences to phrase sequences
- Phrase Translation – $P(v_1^K|u_1^K)$
- Phrase Alignment (or Reordering) – $P(a_1^K|v_1^K, u_1^K)$ specifies the foreign phrase order, i.e. the sequence a_1^K specifies that $v_1 \ldots v_K$ is reordered as $v_{a_1} \ldots v_{a_K}$
- Target Phrase Segmentation – $P(f_1^J|v_{a_1} \ldots v_{a_K})$ maps phrase sequences to words

(b)(ii) An example could be:


that_is_the:voilà_le/0.10
that_these_are_the:voilà_le/1
those_are_the:voilà_le/0.11
these_are:voilà/0.12
those_are:voilà/0.25
that_is:voilà/0.11
Y
fundamental_problem:problème_fondamental/1
basic_problem:problème_fondamental/0.3
the_basic_problem:le_problème_fondamental/0.4
the_fundamental_problem_we_face:le_problème_fondamental/1

but smaller versions with fewer states/arcs would also be valid. [10%]

(b)(iii) An example could be:



where 'eps' represents ϵ-arcs. Smaller versions with fewer states/arcs would also be valid. [10%]

(b)(iv) Translation: maximum likelihood alignment under the language model

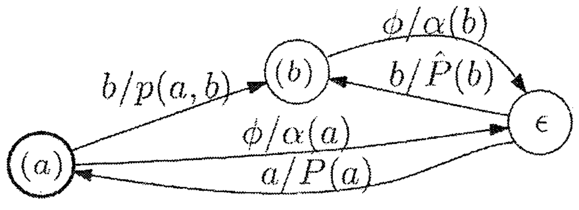
$$\widehat{e_1^I} = \underset{e_1^I}{\text{argmax}} \max_{v_1^K} [\![V \circ R \circ Y \circ T \circ G]\!](v_1^K, e_1^I)$$

where: V is the phrase-segmented foreign acceptor, R is the foreign phrase reordering transducer, Y is the phrase translation transducer, T is the English segmentation transducer and G is the English language model, implemented as a Weighted Acceptor. [15%]

(b)(v) With a language model vocabulary : $\Sigma = \{a, b\}$.
And cutoff statistics : $f(a,b) > C$ but $f(b,a) < C$
The bigram probabilities would be:

$P(b|a) = p(a,b)$ $\leftarrow$ discounting, but no back-off
$P(a|b) = \alpha(b)\hat{P}(a)$ $\leftarrow$ back-off

And the acceptor would be as shown, where ϕ implements a failure-transition for the back-off (i.e. only taken if no other leaving arc can be taken): [15%]



This question covered the models used in statistical machine translation and their implementation as weighted finite-state transducers. It was the least popular question with only two attempts, one of which was very poor.