Solutions: 4F11 Speech and Language Processing, 2013

1. *HMM training*

   (a) HMM assumptions: [10%]

   - 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. [10%]

   (b)(i) 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. Use of Mel-fileter spacing approximates the resolution of the human ear. [10%]

   (b)(ii) 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:

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

   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. [10%]

   (c)(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 $j = N-1, N-2, \ldots, 1$
```

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

(c)(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,

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

(c)(iii) Re-estimation formula for the transition probabilities involve estimating counts for all transitions leaving a state and normalising by the estimate of the number of times that a state is occupied (so that the transition probabilities leaving a state automatically are normalised to unity). In terms of $\alpha_j(t)$, $\beta_j(t)$, and $L_j(t)$ this can be written: [20%]

$$\hat{a}_{ij} = \frac{\frac{1}{p(\mathbf{O}^{(r)}|\lambda)} \sum_{t=1}^{T-1} \alpha_i(t) a_{ij} b_j(\mathbf{o}_{t+1}\beta_j(t+1)}{\sum_{t=1}^{T} L_i(t)}$$

(d) Computational underflow. The magnitude of $\alpha_i(t)$ decreases at each time step(since the values involved are normally all less than unity, and sometimes orders of magnitude less) this leads to computational underflow, even if double precision is used.. Two approaches have been used:

- Scale the $\alpha_j(t)$ at each time step so that $\sum_{j=1}^{N-1} \alpha_j(t) = 1$. The product of the scale factors can be used to calculate $p(\mathbf{O}|\lambda)$. Since this underflows, find $\log p(\mathbf{O}|\lambda)$ as the sum of the logs of the scale factors.
- Use a logarithmic representation of $\alpha_i(t)$. Many practical HMM systems (including HTK) use this solution. However, the forward recursion requires both multiplication and addition of log-represented numbers. Use the following method to evaluate $\log(A + B)$ when knowing $\log A$ and $\log B$. Using

  $$\log(A+B) = \log A\left(1 + \frac{B}{A}\right) = \log A + \log\left(1 + \frac{B}{A}\right)$$

  Only need to evaluate if $\frac{B}{A}$ is sufficiently large.

2. *Contex-Dependent Phone Models*

(a) Triphones are models of individual phones that take into account the immediately preceding and following phonetic context. If the context is blocked at word boundaries these are within-word triphones (effectively have a word-boundary context at word ends). If they take account of phonetic context across word boundaries they are cross-word triphones. Both model model co-articulation effects and since they can be smoothed or tied with other phone contexts are good models for speech. [15%]

(b)(i) Phonetic decision trees, A tree grown at the state level (normally for each of three states in left-to-right phone models). Simple binary tree with yes/no questions at each node to split contexts into equivalence classes. The questions are typically about groups of phone contexts and are aimed at generalisation through for instance phonetic knowledge of similar phones. The trees are automatically grown and at each stage the question chosen is that which gives the largest increase in approximate log likelihood and/or by an occupancy threshold. At the leaf nodes there are groups of contexts which are tied and used to build models. The advantage is that there is no need to back off to shorter contexts as any triphone context will fall into one class or another. Furthermore the technique can be applied to contexts beyond triphones. Disadvantages are that for efficiency a very simple model of log likelihood using a single Gaussian approximation at all levels and that the tree building is greedy - i.e. it makes only locally optimal data splits. [30%]

(b)(ii) Agglomerative bottom-up clustering, Assume all contexts are distinct and simple models of all those found in the training data can be estimated. However to ensure reliable estimates of more complex output distributions then parameter sharing is required via tying. The basic procedure is to take the simple models for all contexts and then iteratively merge similar states (using a between-state distance metric) until there is enough data for all states. This is simple but unreliable for contexts that occur rarely in training. It si also unable (without back-off) to provide suitable models for contexts not seen in training. [15%]

(c)(i) A straightforward phone network without tree-structuring is not expanded with word-internal triphones. However the ability to tree-structure is greatly reduced but there is an improvement in modelling accuracy and so with beam search this can be a good trade-off for both speed and accuracy. For cross-word triphones, models that cross word-boundaries must be duplicated to take into account all the possible end phones and start phones. For single phone words this can lead to N^3 expansion is all triphones are present where N is the size of the phone network. This means that it is infeasible to use a fully-expanded cross-word model for large tasks due to both memory and computational requirements. [20%]

(c)(ii) Two approaches required. First is via multi-pass recognition, This first forms an intermediate reduced search space using simpler acoustic (& often language) models than in the first search. Typically word lattices are produced and then recognition is run with cross-word triphones on the reduced space. An alternative is to use WF-

STs to optimise the network structure and have a minimal network (which might be an order of magnitude smaller than a naive implementation). A further choice would be via a dynamic network decoder in which the network is only generated when the extensions are within the search beam. [20%]

3. *Machine Translation*

(a) Some possible reasons

- Translation should respect document domain and genre. These differ in
  - Specialized vocabularies
  - Stylistic differences, e.g. active vs. passive voice
  - Use of headlines, passage and chapter numbers, web addresses, proper names, ...
  - Variable sentence lengths, e.g. weather forecasts vs. news stories Consequently The same translation system cannot be used for all genres and ext from one genre may not be suitable for building systems in another genre
- Translation must depend on word sense, which varies across languages depending on context
- 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

[10%]

(b) Making simplifying conditional independence assumptions:

$$\begin{aligned} P(f_1^J, a_1^J, J|e_0^I) &= P(f_1^J|J, a_1^J, e_0^I) \quad P(a_1^J|J, e_0^I) \quad P(J|I) \\ &= \textstyle\prod_{j=1}^J \; P_T(f_j|e_{a_j}) \quad P_A(a_1^J|J, I) \quad P_L(J|I) \end{aligned}$$

where:

- $P_L(J|I)$ is the Sentence Length Distribution
- $P_T(f|e)$ is the Word Translation Distribution
- $P_A(a_1^J|J, I)$ is the Word Alignment Distribution

[20%]

(c) Formulae:

- Model-1:

$$P_A(a_1^J|J, I) = \frac{1}{I^J}$$

- Model-2:

$$P_A(a_1^J|J, I) = \prod_{j=1}^{J} p_{M2}(a_j|j, J, I)$$

- HMM Model:

$$P_A(a_1^J|J,I) = \prod_{j=1}^{J} p_{HMM}(a_j|a_{j-1},I)$$

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
- HMM-Model assumes that the link distribution depends on the link of the previous word (that is, it has a history of one link)

[20%]

(d) This follows the derivation of probabilities for Model 2 in example paper 2.

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

and the calculation of the posterior simplifies to

$$P(a_j = i|e_1^I, f_1^J) = \frac{p_T(f_j|e_i)}{\sum_{i'=0}^{I} p_T(f_j|e_{i'})}$$

[20%]

(e) The process can be started with $P_T(f|e)$ set to a flat distribution. At each iteration the following quantity can be computed:

$$\#(f \leftrightarrow e) = \sum_{j=1}^{J}\sum_{i=1}^{I} P(a_j = i|e_1^I, f_1^J)\, 1(e = e_i)1(f = f_j)$$

The numerator and denominator statistics can be gathered over multiple sentence pairs, prior to the normalisation step. The distribution is reestimated as $P_T(f|e) = \frac{\#(f\leftrightarrow e)}{\sum_f' \#(f'\leftrightarrow e)}$ [20%]

(f) This iterative approach to estimation proceeds by seeding more complex models with model components, or alignments, computed using simpler models. Model 1 contains only a t-table. This is copied into model-2, with model-2 alignment distribution set initially either to a flat distribution or initialised via counts from Viterbi alignments (for example). The HMM is seeded with the t-table from model 2, with the parameterised probability distributions seeded as was done for model 2. [10%]

4. *Weighted Finite State Transducers*

(a) A weighted acceptor over a finite input alphabet Σ is a finite directed graph with a set of nodes Q (states) and a set of arcs E (edges).

- Each arc (or edge) e has an initial (or start) state $s(e)$ and a final state $f(e)$.
- Each arc e is labeled with an input symbol $i(e)$ and a weight $w(e)$.
- The weights take values in K

The main difference between an Acceptor and a Transducer is that each arc e of a Transducer has also an output symbol $o(e) \in \Delta$, where Δ is the output alphabet. [20%]

(b)(i) The pronunciation lexicon provides the pronunciation of each word in the ASR vocabulary in terms of the phone inventory of the ASR system. A WFST can be used to map from word sequences to sequences of phone symbols, corresponding to all the ways in which the words can be pronounced. An example is given in lecture. probabilities can be assigned to pronunciation alternatives, after suitable mapping depending on the chosen semiring. [15%]

(b)(ii) A CI-to-CD transducer maps **monophone sequences** to **triphone sequences**.
- e.g. the transducer should map '... d ae t ax ...' to '... d-ae+t ae-t+ax ...' States are added to keep track of the phonetic context, as follows:
For every three monophones, p_1, p_2, p_3 :
- add the states (p_1, p_2) and (p_2, p_3) to the transducer
- for the triphone $t = p_1\text{-}p_2\text{+}p_3$, add the following arc between the two states:

$$(p_1, p_2) \xrightarrow{p_3:t} (p_2, p_3)$$

- Silence models, monophones, etc must be handled differently [15%]

(c)(i) $d(r)$ is the discount coefficient. Its role is to reduce the counts of seen events to allow some probability mass for unseen events. In this case it is applied directly to the relative-frequency based bigram probability.
$\alpha(\cdot)$ is the back-off weight. It is chosen to ensure that $\sum_{j=1}^{V} \hat{P}(w_j|w_i) = 1$ for the vocabulary of size V.
C is the N-gram cut-off point t (i.e. only N -grams that occur more frequently than this are retained in the final model). The value of C also controls the size of the resulting language model. [20%]

(c)(ii) There is one state for every word, plus a unigram back-off state ϵ :

$$Q = \{(w_1), \ldots, (w_V), \epsilon\}$$

There is an arc for each pair of words w and w' for which $f(w, w') > C$

$$(w) \xrightarrow{w'/p(w'|w)} (w')$$

There is a back-off arc (with failure transition ϕ) from every word state (w) to the backoff state ϵ

$$(w) \xrightarrow{\phi/\,\alpha(w)} \epsilon$$

There is a unigram arc from the back-off state ϵ to every word state (w')

$$\epsilon \xrightarrow{w'/\hat{P}(w')} (w')$$

The scores are either probabilities or negative log likelihoods, depending on the semi-ring. [30%]