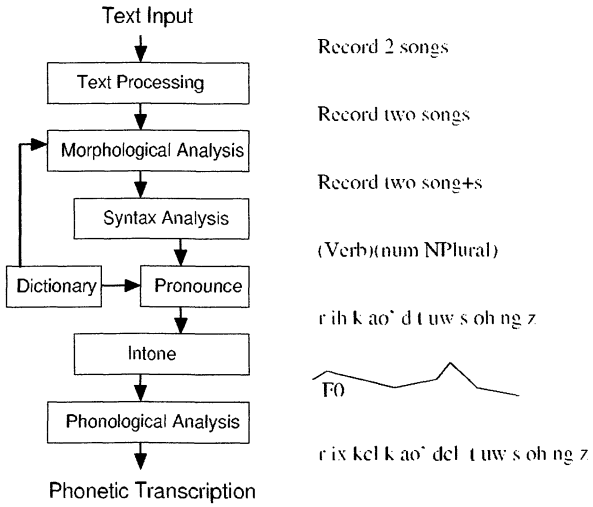
Module 4F11 Exam Lent 2004 - Speech Processing Answers

Question 1

(a) The input text can come from arbitrary sources and needs to be normalised. Secondly the text needs to be translated into a high level description of speech sounds that can be used to synthesize signals. The output of the linguistic analysis stage is a sequence of phonetic symbols together with intonation information, the output of the second stage is the synthesized speech signal. [10%]

(b) (i) This diagram is directly from the lecture handouts: [15%]



(b) (ii) The text preprocessing stage is need for text normalisation, the morphological analysis is on the one side as preparation for the pronunciation generation stage, and secondly for syntactic analysis. Syntactic analysis essentially performs a sentence parsing in order to provide parts of speech tagging.The pronunciation generation stage generates pronunciations for each word in isolation. The information form the syntactic analysis and the pronunciation generation is used to generate intonation patterns (energy,F0,duration). The final stage performs a phonological analysis to account for between-word articulatory effects. [15%]

(b) (iii) Practically all stages require expert knowledge. [20%]

- Text-preprocessing for example rules how to deal with special numbers
- Morphological analysis: pattern/action rules
- Syntactic analysis: grammars describing the language

- Pronunciation generation: Expert made pronunciation dictionary
- Intonation: for example rules on stress
- Phonological analysis: common coarticulation patterns (rules)

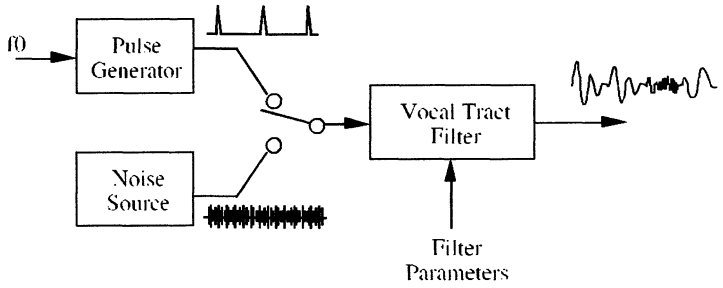
(c)(i) Computational power, memory, speech quality. [10%]

(c)(ii) A synthesis unit is the fundamental entity for which parameters are stored in the speech synthesis stage. Having larger units is better as long as the number of parameters to describe such a unit is equally increased. This allows for example the detailed representation of the sound of a whole syllable. Consequently no or little artifacts are to be expect inside this unit. [15%]

(c)(iii) Formant synthesis uses a model for speech production, i.e. the source filter model and describes the speech signal in terms of very few parameters which are usually manually chosen. Consequently the memory consumption is relatively low, but equally, due to the fact of a simplistic model shorter units have to be used, resulting in poor synthesis quality. In contrast PSOLA stores original speech signals and is totally non-parametric. As such it can provide a very quality synthesised speech signal, but with high memory and computational costs. [15%]

Question 2

(a) The block-diagram can be found in the first handout:



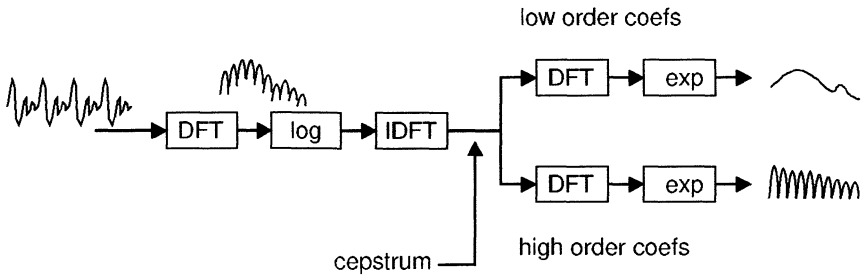
The filter represents the human vocal tract. This is the location where specific speech sounds are formed. Consequently the shape of the human vocal tract varies quickly over time (by movement of the articulators) and the filter parameters need to be updated frequently (e.g. every 10ms). The model for the source is in a simple version switched between a regular pulse train representing the air puffs from the glottis and white noise representing friction for example at the teeth. [20%]

(b) *Formants* are the resonance frequencies of the vocal tract, i.e the frequencies at which we can identify clear peaks in the speech spectrum. Formants are only the centre frequencies of these resonances and they are modelled but the filter in the source filter model. *Pitch* is the perceptual equivalent of the fundamental frequency which is by definition the lowest frequency of a periodic waveform. The excitation generated by the glottis vibration is pseudo-periodic and consequently its frequency of vibration determines the pitch. [15%]

(c) (i) Cepstral analysis is a purely heuristic approach which aims at separation of rapid periodic events from envelope filtering effects to change slowly. As such it filters out the harmonic content of a spectrum (for formant detection purposes). Cepstral analysis is not based on a model for the signal but on practical observations. As such for example the selection of the filter threshold is difficult. For formant detection one is left with peak picking (search) of the cepstrally filtered spectrum.

In contrast linear prediction uses a model for the vocal tract, namely a lossless tube model, represented in an all-pole filter. The filter is characterised by parameters and these are obtained by a minimum squared error optimisation procedure. Further the resonance frequencies of an all-pole filter can be derived from the filter coefficients themselves by finding the roots of the polynomial. No search is necessary. [30%]

(c) (ii) The cepstrum can be computed as follows: [20%]



(d) First detection of formants on telephone speech is difficult due to the filtering of the telephone channel. F1 can be severely suppressed and F4 may be undetectable. The relatively high spacing between speech frames will cause substantial discontinuities from one frame to the next, given that windows are small, or bad detection if windows are large. In consequence the important smoothing of formant tracks over time will be poor and many outliers will be generated. [15%]

Question 3

(a) [20%]

1. The features (*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.
2. 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.
3. 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) **Backward probability** ($\beta_j(t)$) defined as

$$\beta_j(t) = p(\boldsymbol{y}_{t+1} \dots \boldsymbol{y}_T | s(t) = j, \mathcal{M})$$

[10%]

(b) (ii) The slightly asymmetric definitions allow $L_j(t)$ to be easily computed from the $\alpha_j(t)$) and $\beta_j(t)$:

$$\begin{aligned} \alpha_j(t)\beta_j(t) &= p(\boldsymbol{y}_1 \dots \boldsymbol{y}_t, s_t = j \,|\mathcal{M}) p(\boldsymbol{y}_{t+1} \dots \boldsymbol{y}_T \,| s_t = j, \mathcal{M}) \\ &= p(\mathbf{Y}, s_t = j \,|\mathcal{M}) \\ &= p(\mathbf{Y} \,|\mathcal{M}) P(s_t = j \,|\mathbf{Y}, \mathcal{M}) \\ &= p(\mathbf{Y} \,|\mathcal{M}) L_j(t) \end{aligned}$$

Hence,

$$L_j(t) = \frac{1}{p(\mathbf{Y}\,|\mathcal{M})}\alpha_j(t)\beta_j(t)$$

[20%]

(b) (iii)

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

[10%]

(c) In this case can still apply Baum-Welch, but need to make a sentence HMM for each training utterance and train the whole HMM set together. For each utterance form a sentence-level HMM and do forward-backward at the sentence level. The statistics are then computed in parallel for all the models being trained. [20%]

(d)(i) [Note that (d) is not covered in lectures]. Here the maximum α value would be used to set a beam in a similar way to recognition (calculate the most likely state at each frame and only retain those within a log-likelihood threshold of this). This would be fairly broad but would cut the computation on the forward pass (esp for sentence-level training) and also on the backward pass, since only states/times that were active on the forward pass would be considered.. [10%]

(d)(ii) Here the posterior pruning would be used to reduce the computation in accuulating statistics. The beamwidth canb be very tight since have information from both the forward and backward passes. [10%]

Question 4

(a) (i) Basic search will use the Viterbi algorithm. HMMs are joined according to the lexicon to form word models, and then the network is completed by adding the unigram probabilities at the start of words. Expect very brief description of Viterbi here. Key points are that all paths are explored in parallel and dynamic programming is used to make search feasible. It is time synchronous and the head of each path can be represented as a token. [20%]

(a) (ii) Beam pruning. Basic idea is for each frame find the most likely token and then only exntend paths on next frame that are within a beam (log likelihood difference) of the best token. This can simply added to the previous algorithm, by keeping an active list of tokens. Beam pruning helps enormously in practical large vocabulary systems. [15%]

(a) (iii) Tree structuring. Due to pruning search effort is asymmetric. If tree-structure the start of words can save a lot of computation, although unigram probabilities can only be applied when word is unique (or incrementally through word). [10%]

(b) (i) Word-internal triphones. Make each phone dependent on immediate left and right phone context: context does not cross phone boundaries. Little impact on search, but tree-strutring not so effective. [10%]

(b) (ii) Cross-word triphones. Make each phone dependent on immediate left and right phone context: context does cross phone boundaries. Need to expand search network at the first and last phones of each word and thus search space is increased, but pruning tends to be more effective since the HMMs are more accurate. Sometimes do this multi-pass with first pass using word-internal triphones. [15%]

(b) (iii) Syllable models. Introduces syllable level context since each model now is at the syllable level. Key issue is estimation of parameters since parameter sharing is more difficult. As far as computation concerned tree-structuring is much less effective since 10k syllable models. It fits in

with the genral single pass search however and is similar to word-intenal triphones so multi-pass search is not required. [15%]

(b) (iv) Trigram. Now have word-triple probabilities. This can be done in single pass but this was not discussed in lectures. Search space greatly expanded so that the various 2-word histories are kept unique and hence mutli-pass approaches are often used, often with a bigram on the first pass (and possibly word-internal models). [15%]

Question 5

(a) Language models for speech recognition predict the probability of the next word from a history It is usual to restrict the size of the history to the previous $N-1$ words. This is the N-gram language model. Thus

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

[25%]

Most frequently used are the unigram ($N=1$), bigram ($N=2$) and trigram ($N=3$) LMs. They are effective because they are simple to use in the search, they capture the most important local dependencies (including semantics and syntax) and can be trained on large amounts of real text. [10%]

(b) Maximum likelihood estimation isn't used directly since this would be based purely on relative frequency estimates and so any N-grams not occuring in training would give rise to zero probability estimates which in turn would lead to recognition errors. [10%]

(c) Idea is to assign some probability "mass" to unseen events by reducing the counts from seen events (discounting). Then for seen events, relative-frequency based estimates of the discounted N-gram counts.

The N-gram estimate is modified to be

$$\hat{P}(w_k|w_i,w_j) \quad = \quad d(f(w_i,w_j,w_k))\frac{f(w_i,w_j,w_k)}{f(w_i,w_j)}$$

where $d(r)$ is a ***discount coefficient***. The amount by which the maximum likelihood estimate is altered depends on the frequency of the N-gram.

Typical discounting schemes include Good-Turing and absolute discounting.

Backing off is the use of a more general distribution suitably normalised (using a back-off weight) when the N-gram is not seen (often enough) in training.

E.g.

$$\hat{P}(w_j|w_i) = \begin{cases} d(f(w_i,w_j))\frac{f(w_i,w_j)}{f(w_i)} & f(w_i,w_j) > C \\ \alpha(w_i)\hat{P}(w_j) & \text{otherwise} \end{cases}$$

$\alpha(w_i)$ is the ***back-off*** weight, it is chosen to ensure that

$$\sum_{j=1}^{V} \hat{P}(w_j|w_i) \quad = \quad 1$$

and C is the N-gram cut-off point (i.e. only N-grams that occur more frequently than this are retained in the final model). [30%]

(d)(i) Use of a bigram will reduce the size but also increase the perplexity and hence the error rate. Note however that the search architecture is simpler for a bigram. For a well-trained trigram might reduce the error rate by about 20%. [10%]

(d)(ii) This is the usual way of controlling size and can lead to large reductions in size since N-grams that occur only once or twice are in the majority. Typically large reductions in size can be achieved with very little increase in perplexity and sometimes none in word error rate. [10%]

(d)(iii) This is what is known as entropy-based pruning, and is more effective still than count-based pruning in reducing size, since it least disturbs the probability estimates in the model. Hence it can have only very small increaacs in perplexity and word error rate from this process. [15%]