Proceedings of the Institute of Acoustics

# THE STRUCTURE, STRATEGY AND PERFORMANCE OF A MODULAR CONTINUOUS SPEECH RECOGNITION SYSTEM

F R McInnes, D McKelvie & S M Hiller

Centre for Speech Technology Research, University of Edinburgh

## 1. INTRODUCTION

This paper describes a continuous speech recognition system developed at the Centre for Speech Technology Research (CSTR) as part of the Alvey/IED Integrated Speech Technology Demonstrator Project. Results are given for experiments with isolated words and connected speech from several speakers. Companion papers [1,2] describe work by CSTR's industrial partners in this project, in which the same training and evaluation data have been used.

The organisation of the paper is as follows. Section 2 describes the design philosophy and structure of the recognition system, and the system parameter settings currently in use. Section 3 describes experiments conducted to evaluate its performance, and presents the results, in the form of per-phone entropy for the acoustic-phonetic front end, and utterance and word error rates for the system as a whole. Section 4 contains some concluding remarks, assessing the system and its performance results and indicating directions for further development.

## 2. DESCRIPTION OF THE RECOGNITION SYSTEM

### 2.1 Overall Structure and Design Philosophy

The speech recognition system described here has been designed to be used as a tool for research into the enhancement and optimisation of recognition techniques. The requirements for such a purpose are rather different from those for a 'production' system, in which utterance recognition accuracy is of the first importance and real-time operation is strongly desirable. In a research system, real-time recognition is not required since most experiments are performed on prerecorded data, and it may be acceptable to sacrifice some optimality in recognition accuracy for the sake of ease of examining the performance of system components and isolating and correcting causes of error.

The main feature of the design of this system which contrasts with that of most successful continuous speech recognition systems [3,4] is its modular structure.

Systems designed for real-time recognition usually rely on an integrated top-down strategy in which syntactic and lexical constraints control the operation of the acoustic-phonetic matching. Only signal processing (which may include vector quantisation) is carried out in a bottom-up direction; the modelling of individual speech units (such as phonemes) and the language modelling (imposing constraints on the sequence of such units) are integrated into what is effectively a large transition network, from which the best match to the output of the signal processing is found by a dynamic programming operation (typically limited by a beam-search strategy for computational efficiency).

# A MODULAR CONTINUOUS SPEECH RECOGNITION SYSTEM

In the CSTR modular recognition system, the recognition processing is divided into two components, the *front end* and the *back end*. The front end takes in the speech to be recognised, and generates a lattice of probabilistically scored hypotheses of acoustic-phonetic units (APUs). (In the simplest case, the APUs are phonemes, but it is possible to refine this, as described below.) This is done without any use of higher-level linguistic information, as to the words in the vocabulary or the syntactically permitted sequences of words. The back end takes in the APU lattice and finds a path through it corresponding to an estimated best-matching sequence of words permitted by its language model. The back end makes use of lexical information (a representation of each word in the vocabulary in terms of the APUs) and syntactic information (defining what word sequences are permissible); it also incorporates APU substitution, insertion and deletion probabilities, which can be trained on lattices generated by the front end for known utterances.

The modular structure allows the front end and the back end to be evaluated and optimised separately. The performance of the front end can be evaluated by an analysis of the lattices it produces. A mathematically well-founded evaluation measure, which is an estimate of the entropy (per phone) of a lattice, has been devised and is described in a previous paper [5]. This measure can be used to determine optimal values for various parameters in the front end. The lattices can also be examined in detail to identify specific aspects of the front end's performance which need to be improved. Then, once lattices have been constructed for a set of utterances, the back end can be run repeatedly with these lattices as input, using different parameter settings, different lexicons, and different language models, to test the effects of these variations, without any need to repeat any of the front-end processing.

The front end and back end currently implemented are described in next two subsections.

### 2.2 The Front End

The front end in this system is a refinement of that described in [5], based on the use of discrete hidden semi-Markov models (HSMMs).

The speech to be recognised is sampled at 10kHz. The start and end times of the utterance are found using adaptive thresholds on signal magnitude, and regions of the data beyond these points are discarded. The signal processing, performed in a 20ms Hamming window every 5ms, consists of a 14th-order LPC analysis to generate cepstral coefficients, together with the estimation of three log formant frequencies using a generalised centroid algorithm [6]. For each 5ms frame, a 28-dimensional acoustic feature vector is derived, consisting of 11 cepstral coefficients (from the 0th to the 10th) and three log formant frequencies from the frame 20ms before the current time, together with the corresponding values from the frame 20ms after the current time. This vector implicitly incorporates both static information (with some smoothing in the time dimension) and dynamic information. A linear transformation, based on a discriminant analysis among a set of acoustic-phonetic classes, is applied to this vector, yielding a 10-dimensional discriminant feature vector. The sequence of such feature vectors representing the input utterance is converted into a sequence of integer labels (in the range from 0 to 255) by vector quantisation (VQ).

The HSMM component performs the two tasks required to generate a lattice. Firstly, it determines the start and end times of the segments in which APU hypotheses are to appear. The basic segmentation algorithm is a one-stage connected Viterbi algorithm, which determines the acoustically best-matching sequence of APU models together with the corresponding sequence of segment boundary times. Each APU is represented by a three-state model, with a simple left-to-right

topology (without skip transitions) and a Gaussian duration distribution for each state. APU sequence probabilities of the form $P$(next APU|current APU), estimated from a corpus of transcribed speech, are used to guide the segmentation. In the traceback stage of the segmentation, multiple segmentations of regions of the input can be derived by an algorithm which finds local maxima of the overall probability up to the current segment boundary time as a function of the most recent previous boundary time. Secondly, once the segments have been defined, probability scores for all the APUs are derived in each segment. (In fact, if the multiple segmentation option is in use, these scores are obtained as a by-product of the local-maxima computation.)

Various postprocessing operations [5] are applied to the scores computed by the HSMM component, to optimise the relative probability estimation. The output of the front-end processing is a lattice of segments each with a start time, an end time and a set of scaled negative log probability scores for all the possible APUs. The lattice is well-formed in the sense that each segment has abutting left and right neighbours, so that a continuous path can be traced through the lattice.

The set of APUs currently in use consists of phonemes, allophones (released and unreleased stops, clear and dark /l/, etc.), and some phoneme sequences such as /tr/ and /el/ where modelling at the phoneme level is unsatisfactory because of the strong interaction between successive phonemes. There are 98 units altogether, including silence (which is needed because the endpoint detection may leave non-speech regions at the beginning and end of the utterance, and because some utterances may contain pauses between phrases).

The probability scoring can be enhanced by a hierarchical probability estimation technique [7]. Once the start and end times of a segment have been defined, revised scores are computed for the APUs in each of a set of broad classes, using a discriminant feature space, VQ codebook and models specific to the broad class. The conditional probabilities of the form $P$(APU|broad class) derived from these scores (by normalising so that the probabilities sum to 1 within each broad class) are combined with the probabilities $P$(broad class) obtained by summation of the APU scores estimated in the original feature space, to yield revised estimates of the probabilities $P$(APU) for all APUs. The set of APUs is currently partitioned into two broad classes, of 45 and 53 APUs respectively, corresponding roughly to the classifications 'sonorant' and 'non-sonorant'. The power of the hierarchical scoring technique lies in the discriminant transformations applied to derive the acoustic feature spaces: the feature space specific to each broad class is optimised to discriminate among a set of fine acoustic-phonetic classes *within that broad class*, and will thus capture more effectively the information that is useful for APU discrimination within that class (e.g. formant frequencies rather than cepstral coefficients, or dynamic rather than static information).

### 2.3 The Back End

The back-end program reads the APU lattice and attempts to find the best sentence which matches this lattice. This sentence must consist of words in a dictionary (or lexicon) of known words and must be grammatical with respect to a grammar. This matching is done by a chart based dynamic programming algorithm (implemented in Common Lisp). The DP algorithm is left-to-right, beam-searched, breadth-first search. It is carried out in two stages which run in parallel: word recognition and sentence recognition. Word recognition consists of finding the best phonetic match to words in the lexicon which the grammar allows. Sentence recognition then combines these found words with existing partial sentence hypotheses to form longer partial sentences. The output of the back end is a list of likely recognised sentences, with scores indicating their estimated relative probabilities.

The lexicon of known words is implemented as a directed graph, with edges labelled with APUs. Nodes which correspond to words have a pointer to information about those words. Edges also have attached a set of tags or word classes, which specify the set of words which can be reached by following this edge. This information is used to dynamically restrict the size of the lexicon under the control of the grammar, so that words which cannot be used by any of the current sentence hypotheses are never looked for. Multiple APU pronunciations of words are stored in the lexicon and are generated automatically from a set of phonemic transcriptions written by a phonetician.

The interface between the back end and the syntax is a general procedural interface to a grammar. The present experiments use a simple finite-state grammar because of the simplicity of the test language, but previous work has used probabilistic grammars [9] or GPSG style phrase structure grammars.

In order to match lexical word pronunciations against realistic lattices, the DP algorithm has to allow insertions, deletions and substitutions of APUs. In contrast to some systems, these insertions etc. are not coded in the lexicon but are handled separately. Each pair of lexical APU and lattice APU has a confusion cost, e.g. c(t:d) would be the penalty paid by the DP algorithm for allowing a lattice /d/ to match a lexical /t/. These costs are trained by a standard Viterbi algorithm. Unlike the normal DP algorithms deletions of lattice material are handled by only allowing the deletion of a special /DELETE/ APU which is present in each segment of the lattice. One oversimplification of this approach is that e.g. the probability of deleting a segment does not depend on the surrounding context.

## 3. EXPERIMENTS AND RESULTS

### 3.1 Linguistic Domain and Speech Data

The linguistic domain selected for these experiments was taken from the field of air traffic control (ATC). A finite state syntax was defined and a set of 100 sentences conforming to this syntax was generated. Each sentence consisted of an aircraft call sign followed by one or two other phrases. The average number of words per sentence was 10.9, or 12.6 including pauses (which occurred between phrases, and were counted as words for the purpose of the back-end processing).

A list of words occurring in the ATC domain was also generated; there were 98 distinct words (though the list was 101 words long because of three duplications). A subset of the 98-word vocabulary consisted of the digits from 'zero' to 'nine'.

Four male speakers of non-rhotic British English each spoke the 100 sentences once, and also provided five repetitions of the words in isolated word format for use as training data, and two repetitions of all the words and 10 further repetitions of the digits for use as test material for isolated word recognition. There were also available utterances by each of the speakers of 200 sentences (designed for good coverage of every phoneme in a range of contexts) as training material for continuous speech recognition; and of a further 170 sentences, drawn from a domain of dictated cytology reports, which were used in training the substitution, insertion and deletion probabilities for use in lexical access.

Each speaker was recorded in a sound-treated booth using a Shure SM10A unidirectional dynamic head-worn microphone connected to a Shure FP11 microphone-to-line amplifier. The signal was

oversampled at 80 kHz with 16-bit resolution using an analogue anti-aliasing filter, then filtered using a digital 64-tap low-pass filter with cutoff at 7.5 kHz and downsampled to 20 kHz. The speech data used in the recognition experiments were low-pass filtered with a digital 256-tap low-pass filter set to a cutoff of 4.75 kHz and downsampled to 10 kHz.

**3.2 Design of Experiments**

Three sets of recognition trials were conducted, one on the connected sentence utterances and two on the isolated word utterances (the ATC words and the digits respectively). In each case the recognition system operated in speaker-dependent mode.

The front end parameter settings were determined by experiments on the cytology utterances from one of the speakers (GSW), using his training sentences with a manual phonemic segmentation to train the discriminant analysis, codebook and APU models. These preliminary experiments confirmed that the extended APU set (in contrast to the basic set of 44 phonemes plus silence), the extended acoustic feature set with discriminant analysis and the hierarchical probability scoring improved the performance of the front end. (This was assessed using the per-phone entropy criterion.)

The discriminant transformation, codebook and models derived from GSW's hand-segmented training utterances were used to initialise the training procedure for continuous speech recognition for each speaker. First the target speaker's training utterances were segmented by an automatic procedure using Viterbi alignment with a transcription of each utterance in terms of the APUs. (The transcriptions were generated automatically from a phonemic lexicon, and were structured as directed graphs, rather than simple sequences, so as to include multiple variant pronunciations of each word and optional word-boundary assimilation and reduction effects.) The resulting segmented data were used to construct a discriminant transformation, a codebook and APU models for the target speaker. These were then used to segment the training utterances again, and the APU models were retrained on the basis of this revised segmentation. Also broad-class-specific discriminant transformations, codebooks and models were derived from the same segmentations.

Lattices were constructed, using the front end with the chosen parameter settings and the speaker-specific models, for each speaker's cytology utterances, and these were used to estimate the APU substitution, insertion and deletion probabilities for the back end (by running the lexical access process with a syntax which allowed only the correct recognition for each utterance, and iterating over the set of utterances). This was done for each speaker separately. Lattices were then constructed for the ATC sentences, and the back end was applied to recognise these.

A similar procedure was followed in the case of the isolated words. In this case, the initialisation was based on a set of isolated words recorded from GSW, which had been hand-segmented, and the training for each speaker used the five training repetitions of the ATC words. (At first the models trained for connected speech were applied to the isolated word recognition task, but the entropy results were so much poorer than those for continuous speech that this approach was discontinued.) The discriminant transformations and VQ codebooks used for the continuous speech recognition were retained for the isolated words. The back-end probabilities estimated on the cytology sentences were likewise retained.

Because the system had been developed on continuous speech, the endpoint detection parameters were not optimal for isolated-word utterances, and, since endpoint detection accuracy is particu-

larly important for good recognition performance in the case of isolated word recognition, some adjustment of the parameters for isolated words was performed.

For the ATC sentences, the back end used a finite state syntax, of perplexity 2.40 at the word level. For the isolated words, the syntax permitted a pause (corresponding to 0 or more segments in the lattice), any single word from the current vocabulary, and then a pause.

### 3.3 Results

For each of the three experiments, two types of results are reported: entropy results for the front end, and recognition results for the system as a whole. In the case of sentence utterances, the recognition results subdivide into utterance recognition accuracy and a word-based accuracy measure. The word-based measure adopted is the *weighted total error* measure [8], defined by

$$T_W = 100 \times \frac{N_S + N_D/2 + N_I/2}{N_P}$$

where $N_S$ is the number of word substitutions, $N_D$ the number of word deletions, $N_I$ the number of word insertions, and $N_P$ the number of words in the correct recognitions of the utterances.

The results on the ATC sentences are shown in table 1. The lattice entropy is expressed in bits per phone; the smaller it is, the better the lattices. The percentage figures for utterance and word errors are based on the best-ranked sentence hypothesis (i.e. the first in the list generated by the back end) for each utterance, with those based on the most nearly correct hypothesis in the top 10 for each utterance given in parentheses. Many of the errors were confusions among the digits, such as recognition of 'one' as 'niner', and 'two' as 'zero'. The truncation of some utterances during endpoint detection meant that sentence-final 'minutes' was also not detected reliably, especially for speaker HB.

Table 1: Continuous speech recognition results (100 ATC sentences per speaker)

| speaker | GSW | HB | JMR | PMS |
|---|---|---|---|---|
| entropy | 2.028 | 2.115 | 2.532 | 2.487 |
| utterance error rate | 5.0 (4.0) | 17.0 (11.0) | 22.0 (17.0) | 8.0 (6.0) |
| word error ($T_W$) | 0.5 (0.3) | 2.7 (1.9) | 3.6 (2.9) | 1.1 (0.7) |

The results for the isolated word recognition experiments on the ATC vocabulary are shown in table 2. Some of the errors were confusions between pairs of words such as 'five'/'fife', 'tree'/'three' and 'leff'/'left', which would not be significant in practice because the words are synonyms in the ATC domain. Another source of errors was when in words such as 'oh' and 'eh' the long vowel was split into two segments by the front end.

Table 2: Isolated word recognition results (98-word ATC vocabulary; 202 words per speaker)

| speaker | GSW | HB | JMR | PMS |
|---|---|---|---|---|
| entropy | 1.590 | 3.623 | 2.334 | 2.183 |
| utterance error rate | 3.9 (0.5) | 5.9 (1.5) | 6.4 (1.0) | 6.9 (1.0) |

The results for the isolated digit recognition experiments are shown in table 3. The two errors

were on utterances of 'eight' (recognised as 'three', because of noise at the beginning) and 'five' (recognised as 'nine').

Table 3: Isolated word recognition results (digits; 100 words per speaker)

| speaker | GSW | HB | JMR | PMS |
|---|---|---|---|---|
| entropy | 3.820 | 2.650 | 3.764 | 2.221 |
| utterance error rate | 0.0 (0.0) | 1.0 (0.0) | 0.0 (0.0) | 1.0 (0.0) |

## 4. CONCLUDING REMARKS

The modular recognition system has proved a useful research tool and has served as a testbed for a number of innovative techniques, including hierarchical probability scoring and the extended acoustic-phonetic unit set in the front end, and various options for substitutions, insertions and deletions in the back end. Having an intermediate level of representation between acoustic features and the recognised sentence — namely the lattice of APU hypotheses — allows detailed examination of the performance of the front end and the back end, which is very useful during the development of speech modelling techniques. Also the entropy measure computed at the APU lattice level is a more sensitive criterion for optimising front-end parameters than utterance or word recognition accuracy.

In the results for the ATC sentences, some degree of correlation is evident between the trend in per-phone entropy across the speakers and the corresponding trend in recognition error rates. This is as might be expected. The entropy is a measure of the probability assigned to the correct recognition (expressed as a sequence of APUs, or a network of such sequences to allow for variant pronunciations), relative to the total probability for all possible APU sequences. The lexical and syntactic constraints cut down the set of possible APU sequences from which the correct one is to be chosen. The more constraining the lexicon and syntax are, the less closely correlated the probabilities of the correct recognition with and without the constraints will be. In this case the lexicon and syntax are fairly strongly constraining, and so only a weak correlation is observed. The entropy results are more general in that they measure the front end's performance without the assumption of any particular lexicon and syntax.

The correlation between entropy and recognition error does not hold good in the case of the isolated words, especially the digits, where two speakers (GSW and JMR) have very poor entropy scores but no recognition errors. The reason for the poor entropies lies in the endpoint detection. For good word recognition performance, the endpoint detection parameters must be set so that there is little danger of cutting off the end (or the beginning) of a word. Given the difficulty of distinguishing between low-amplitude speech sounds and breath noise, this means that non-speech intervals will tend to be included. This may not introduce errors in the recognition of the words, since the 'silence' hypothesis usually scores well in non-speech segments and this is allowed for in the isolated word syntax. It does, however, seriously affect the entropy value, since this is computed without any constraints on APU sequences, and so initial and final sequences of several consonant hypotheses, which often have good scores in non-speech segments, are permitted as competing incorrect recognition candidates which substantially worsen the entropy. This effect is particularly strong for short utterances. This is one instance in which APU lattice entropy is an

unreliable guide to likely recognition accuracy. The practical conclusion is that endpoint detection parameters should not be optimised using the entropy criterion.

Some problems remain in the estimation of the substitution, insertion and deletion costs. Some of the possible pairs of lexical and lattice APUs do not occur in the estimation data, and this can cause errors when the unseen substitutions are required in the test data. Also, the training algorithm produces deletion costs which are far too low; this problem will be addressed, but in the meantime the deletion cost is set to an empirically derived constant value.

Various further enhancements to the system are under development, including the introduction of context-specific APU modelling (a variant of interpolated triphone modelling) in the front end, and a context-specific APU deletion mechanism in the back end [10].

## 5. ACKNOWLEDGEMENTS

This work was supported by the Information Engineering Directorate/Science and Engineering Research Council as part of the IED/SERC Large Scale Integrated Speech Technology Demonstrator Project (SERC grants D/29604, D/29611, D/29628, F/10309, F/10316) in collaboration with Marconi Speech and Information Systems and Loughborough University of Technology. The recognition system described here is the result of the contributions of a large team of researchers at CSTR.

## 6. REFERENCES

[1] L C Wood & D J B Pearce, 'Sub-word HMM recognition. An investigation of phone-context modelling and improved discrimination', these proceedings (1990)
[2] M J Hunt, S M Richardson & M G Abbott, 'The use of linear discriminant analysis in isolated and continuous word speech recognition experiments', these proceedings (1990)
[3] K-F Lee, *Large-Vocabulary Speaker-Independent Continuous Speech Recognition: The SPHINX System*, PhD dissertation, Computer Science Department, Carnegie Mellon University (1988)
[4] P Bamberg, Y Chow, L Gillick, R Roth & D Sturtevant, 'The Dragon Continuous Speech Recognition System: A Real-Time Implementation', *Proc. DARPA Speech and Natural Language Workshop* (1990)
[5] F R McInnes, Y Ariki & A A Wrench, 'Enhancement and Optimisation of a Speech Recognition Front End Based on Hidden Markov Models', *Proc. Eurospeech 89*, 2, pp461–464 (1989)
[6] A Crowe & M A Jack, 'Globally optimising formant tracker using generalised centroids', *Electronics Letters*, 23, 19, pp1019-1020 (1987)
[7] Y Ariki, F R McInnes & M A Jack, 'Hierarchical Phoneme Discrimination by Hidden Markov Modelling Using Cepstrum and Formant Information', *Proc. IEEE ICASSP 89*, pp663–666 (1989)
[8] M J Hunt, 'Figures of Merit for Assessing Connected-Word Recognisers', to be published in *Speech Communication*
[9] C A Matheson, J C Foster, A W Black & I A Nairn, 'Dualgram: an efficient method for representing limited-domain language models', Nato ASI Workshop, Cetraro, Italy (1990)
[10] H S Thompson, D McKelvie & F R McInnes, 'Robust Lexical Access for Continuous Speech Using Dynamic Time Warping and Finite-State Transducers', *Proc. Eurospeech 89*, 2, pp59-62 (1989)