Proceedings of The Institute of Acoustics 2A9

CONNECTED WORD RECOGNITION USING WHOLE WORD TEMPLATES

J S BRIDLE AND M D BROWN

JOINT SPEECH RESEARCH UNIT, CHELTENHAM

ISOLATED WORD RECOGNITION

It has been known for some time that it is possible to recognise isolated words from a small vocabulary from a known speaker using the method of whole word template matching. The machine's information about each vocabulary word is limited to one or more patterns ('templates') derived by acoustic analysis of utterances of the word, previously spoken by the person using the machine. The pattern derived from an unknown word is compared with all the templates, and the most similar determines the machine's decision about the identity of the particular word spoken. All commercially available automatic speech recognition machines use some form of whole word template matching.

Among the technical questions concerning the design of such a machine are the method of acoustic analysis, the representation of the template patterns, the method of computing the similarity between small portions of an unknown input speech pattern and of a template, and the method of mapping the timescale of a template on to that of an unknown word. A fairly standard method is to use a regularly sampled short-term spectrum representation, a Euclidean distance between log power spectra, and a Dynamic Programming algorithm (Ref.1) to compare complete patterns while allowing flexibility of the timescales. Such time-flexible matching is particularly useful for isolated polysyllabic words (Ref.2).

In Dynamic Programming algorithms for isolated word recognition, we compute the 'score', S(i,j), for the best way of matching the first i frames of a particular template with the first j frames of the unknown input, for all values of i and j in a predetermined region of the time-time plane (Fig. 1). The score for each point (i,j) is computed from the scores at points earlier in time in the two patterns, plus the similarity between the i'th template frame and the j'th input frame.

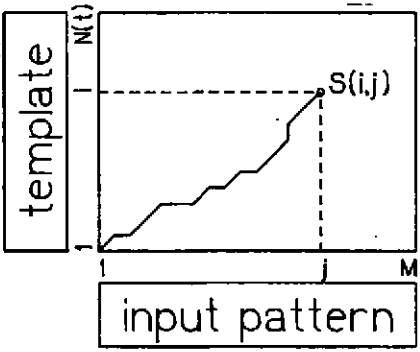

Figure 1

$$S(i,j) = \underset{a,b}{\text{Optimum}}(S(i-a,j-b) + sim(i,j))$$

There are several related algorithms which differ in the set of previous points examined, and in other details which are sometimes quite important. The set of previous points might typically be given by (a,b)=(0,1),(1,1),(1,0) as in figure 1. See reference 3 for a survey.

The score for the t'th template, of length L(t), matched with the complete

input pattern of length M is S(L(t),M). It is usual to consider the templates sequentially, and choose the one with the best score.

CONNECTED WORD RECOGNITION

It is reasonable to expect that automatic speech recognition would find significantly more applications if the restriction to isolated words could be relaxed. For many practical purposes it is not necessary to cope with fluent prose: even a facility that would only accept fluently spoken digits and isolated command words would be potentially useful in many cases.

At first sight whole word template techniques would seem to be limited to isolated word recognition, because we cannot expect to be able to segment an utterance into words.

In an earlier report (Ref.4) we introduced the idea of using a set of 'word detectors', one for each template, which scan the input speech pattern and indicate the positions of likely occurrences of each word. After experiencing difficulties with automatic procedures for 'making sense' of the resulting pattern of word detector outputs we devised the alternative method described below.

Perhaps the most natural way to extend the ideas of whole word matching to deal with connected words is to define the best word sequence for a given input utterance as the one for which the corresponding templates, when joined together end to end, make a 'composite template' which matches the input pattern best. There are two interesting questions concerning such an approach:

1. Can we devise an efficient algorithm for finding the best word sequence?
2. Will the best template sequence correspond to the spoken words often enough to be useful?

Sakoe (Ref.5) describes a two-stage Dynamic Programming algorithm in which the score is first computed for every template matched against every portion of the input. The second stage computes the best sequence of templates, using this information. The algorithm described below works through the input pattern in one pass, and gains efficiency by incorporating word sequence information in the process which matches words. The method is similar to that used in the Harpy speech understanding system (Ref.6), and like Harpy can benefit further by pruning away the least likely paths.

Our connected word recognition algorithm is very similar to the isolated word method outlined above. We compute the score, C(t,i,j), for the best way of matching the first j input frames with a permissible sequence of templates followed by the first i frames of the t'th template (Fig. 2). Within each template we perform the same basic operation as in isolated word recognition:

$$C(t,i,j) = \underset{a,b}{\text{Optimum}} \left(C(t,i-a,j-b) + \text{sim}(t,i,j) \right)$$

At the start of a template we must look at the ends of preceding words and

choose between them:

$$C(t,1,j) = \text{Optimum}_{r \text{ in } P(t)} \left(C(r,N(r),j-1) + sim(t,1,j) \right)$$

where P(t) is the set of words that the word sequence rules allow to preceed the t'th input word.

There are some details to sort out at word boundaries, depending on the set (a,b) used within words. We may also choose to skip the end frames of the template, and we may allow a skip of a few input frames between templates. e.g. use C(r,N(r)-c,j-d)

Computation proceeds for all templates in parallel, in one pass through the input pattern. At the end of the input we select the best score corresponding to the end of a word that is allowed to occur at the end of an input phrase. The sequence of templates which leads to this score is the desired recognition of the input pattern.

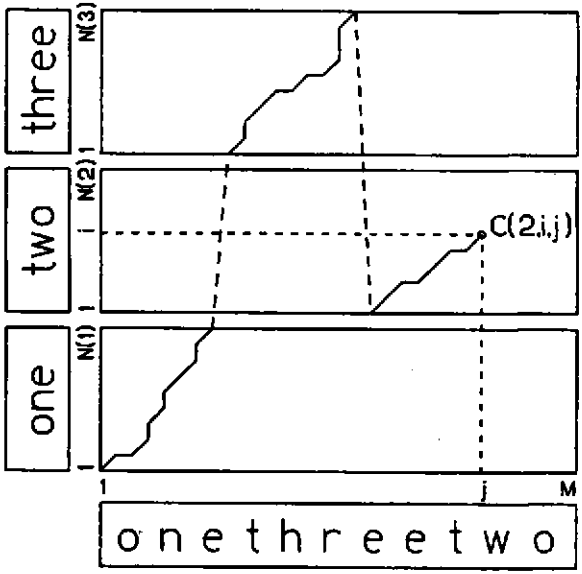

Figure 2

The above algorithm will by definition find the best 'explanation' of the input pattern in terms of a sequence of template patterns. It is fairly expensive in computation and storage. Compared with an isolated word recogniser with the same size vocabulary, using the same methods of pattern representation and Dynamic Programming matching algorithm, it takes about the same amount of computation per input frame, but needs a greater amount of working storage. Because all words must be considered in parallel the working storage is increased by a factor of the vocabulary size. Additionally, because some information must be propagated through the pattern to recover the sequence of templates there is a further increase in working store. The total amount of working storage will be about as much as is required to hold the template patterns themselves.

The amount of computation can be reduced by a factor of about ten by using Lowerre's 'Beam Search' modification of Dynamic Programming, as used in the Harpy speech understanding system (Ref.6). For each input frame we remove from further consideration all scores which are more than some 'beam width factor' away from the best score for that input frame. By this means we can avoid

considering relatively unlikely interpretations of the beginning of the input pattern, but keep the options open if there seems to be some ambiguity.

RESULTS

The question whether the variability of the patterns of words in strings is too great to permit so simple a philosophy as whole word matching has already been answered: The Nippon Electric Company's DP100 is a connected word recogniser which uses Sakoe's two stage algorithm, and it works well.

Our algorithm is currently implemented on a mini computer, using a channel vocoder filter bank for acoustic analysis because it was conveniently available (see Ref.4). We have tried to recognise recorded groups of three and five digits, using digit templates each derived from one utterance of a single digit. The algorithm would accept any number of digits in a group (the word sequence rules were particularly simple). On a total of 350 digits the error rate was 2.5% per digit.

This is neither a good test nor a particularly good result. In practice much depends on the speaker and the manner of speaking. The NEC machine made about as many errors on the same recorded material. To put the computer's performance in perspective, it is interesting that the speaker himself (JSB) made 1% digit errors in merely reading out the strings.

CONCLUSIONS

The above technique may have applications in simple connected word speech input. It remains to be seen how it compares in power, flexibility and cost with Sakoe's method and with Harpy.

The limitations of connected word recognition based on whole word matching have yet to be explored. The present technique has potential for elaboration to deal with some of the forseeable problems.

REFERENCES

1. V.M. VELICHKO and N.G. ZAGORUYKO 1970 Int.J.Man-Machine Studies 2 223-234. Automatic recognition of 200 words.
2. G. WHITE and R. NEELY 1976 IEEE ASSP-24 2 183-188. Speech recognition experiments using linear prediction,bandpass filtering and dynamic programming.
3. H. SAKOE and S. CHIBA 1978 IEEE ASSP-26 1 43-49. Dynamic Programming algorithm optimisation for spoken word recognition.
4. J.S. BRIDLE and M.D. BROWN 1974 JSRU Research Report 1003. An experimental automatic word recognition system.
5. H. SAKOE 1976 U.S. Patent 4,059,725. Automatic continuous speech recognition system employing Dynamic Programming.
6. B.T. LOWERRE 1976 Carnigie-Mellon University Dept. Comp. Sc. Dissertation. The HARPY Speech Recognition System.