Proceedings of The Institute of Acoustics

SPEECH RECOGNISERS BASED ON N-TUPLE SAMPLING

G D TATTERSALL & R D JOHNSTON

BRITISH TELECOM RESEARCH LABORATORIES

Abstract:

The rationale for using extracted features in speech recognisers is discussed and the conclusion is reached that it may be desirable to avoid any explicit feature extraction. N-tuple sampling systems operating on the raw time domain samples of the speech are proposed and analyzed statistically.

Introduction:

In this paper we examine the problems of finding suitable features for extraction in automatic speech recognisers and propose a recognition scheme which avoids explicit feature extraction.

Many features have been used in recognisers, ranging from the ratio of formant frequencies, to the use of expert systems to spot the sequential or simultaneous occurrence of low level features and derive from them a higher level feature. However, the generation of all these features rely upon models of speech generation and the functioning of the ear and brain. Current knowledge of these processes is still limited and it is likely that features chosen in this way will be rather arbitrary.

To avoid this problem we propose a recognition system which uses a probability map of the pattern space spanned by the time domain samples of speech utterances. No explicit feature extraction is used in the system. Three possible implementations of the probabalistic map approach are described which are derived from the n-tuple sampling techniques first described by Bledsoe (1) and latterly investigated by Aleksander (2). These implementations require large amounts of RAM but negligible computation, and consequently hold the prospect for economic recognisers for mass application.

The Rationale for Feature Extraction:

Analysis of the feature extraction process and the operation of probabalistic maps is conveniently done by representing sequences of sampled data as single points in a multidimensional pattern space. From hereon we will refer to the pattern space constructed from the time domain samples as H1.

In general there will be an extremely large number of time domain waveforms corresponding to a single meaning because the same sequence of words can be uttered with many accents, speeds, stresses etc. Consequently there are many points in the original pattern space, H1, which have the same meaning. This implies redundancy in H1. An obvious approach to recognizing patterns in H1 is to try and map points in H1 into a new hyperspace, H2 which has a reduced number of dimensions and contains just one resolvable point for each possible meaning. If such a mapping function can be found, recognition becomes trivial;

the utterance to be recognized is mapped to H2 in which each resolvable point has been previously labelled with the appropriate meaning by training. The label located by the mapped utterance is the meaning of the utterance. The dimensions of the space H2 must contain all the necessary information to enable different meanings to be perfectly distinguished and can be considered as speech "features" which are strongly related to particular meanings.

The foregoing description indicates that the process of feature extraction is a mapping or transformation from H1 to the feature space, H2, with the object of removing redundancy and reducing dimensionality. The beneficial result of the process is that many points belonging to the same class of pattern, distributed over a wide volume of the space, H1, are mapped into a small volume in H2. Thus feature extraction appears to produce compact class clusters which are more easily represented by a small number of templates than patterns of the same class in H1. However, as will be discussed in the next section, no extra information has actually been gained by feature extraction, and the existence of features merely reflects a particular cluster topology in the pattern space H1. This topology can be detected by the techniques to be described later in the paper thereby making explicit feature extraction unnecessary.

The Nature of Feature Extraction.

Feature extraction is the process of mapping through a transform from H1 to H2. The transform can be linear or non-linear depending on the nature of the data in H1. The process of linear mapping is performed by choosing a sparse set of basis vectors or "features" whose weighted sum is believed to be capable of efficiently representing the speech waveform. The basis vectors must be orthonormal and, to reduce dimensionality, must be fewer in number than the time domain samples of the speech. The latter requirement can only be fulfilled if they are matched in some way to the characteristics of the speech. For example, the basis vectors could be formant frequencies.

A linear transform is only effective in reducing the dimensionality of H2 if all the patterns of one class in H1 have a large component along just a few of the chosen basis vectors: This implies that the class clusters in H1 must be cigar shaped for a linear transform to effectively work as a feature extractor. The best sparse set of feature vectors based upon a linear transform can be found from the co-variance matrix of the patterns in H1 using the Karhunen-Loeve Transform (3). In general the cluster of patterns in H1 will not be cigar shaped although their inherent dimensionality may be much less than the dimensionality of the space H2. For example, in a 2-dimensional pattern space, all members of a class could lie on a complex curved filament. The inherent dimensionality of the data in the filament is only one, but this will not be detected by a linear transform because there is no single rotated axis onto which the data in the filament can be projected without significant loss of information. However, a non-linear transform which projected the data on to a curved axis parallel to the locus of the filament would enable the dimensionality of the feature space to be reduced. Unfortunately, there is no simple technique for deriving an appropriate non-linear transform.

Two aspects of feature extraction of speech need to be emphasized at this point:

The existence of a class feature reflects the existence in the pattern space, H1, of a bounded volume containing most of the pattern vectors belonging to that class. In other words, if a good feature exists for a class, then members of that class lie within a cluster in H1 although its shape may be hard to visualize. For example, the cluster maybe a filament or surface wandering through the dimensions of H1. The second aspect of feature extraction is that class separability is not fundamentally enhanced: There is just as much overlap between the different class clusters in the feature space H2 as there is in the pattern space, H1.

These points suggest that an alternative approach to feature extraction is to actually store a map of the pattern space, H1, containing information about the locations of the boundaries of each class cluster. An unknown pattern in H1 is then recognized by seeing which class label is associated with its position. At first sight it would appear that a training pattern would be required at each location in the map in order to conduct an effective "survey". However, in the map system to be described, spatial interpolation can be provided which enables the map to be formed from relatively few training patterns.

Probability Maps:

The optimal technique for assigning a pattern vector to one of several classes, Cj, is by comparing the conditional probabilities of each class given that X has occurred. i.e. X belongs to class Cj, if:

$P(Cj/X) > P(Ci/X)$ for $i \neq j$

In order to employ this decision rule it is necessary to know the values of P(Cj/X) for all i and X. These values can be obtained by training and the use of Bayes Theorem.

The process of training involves presenting the recogniser with examples of pattern vectors whose class is known. From these examples of each class, it is possible to estimate the conditional probability, P(X/Cj). Bayes Theorem could then be used to calculate the aposteriori class probability, P(Cj/X):

$$P(Cj/X)=P(X/Cj).P(Cj)/P(X)...1$$

$$P(X)=\sum_{J} P(X/Cj).P(Cj).....2$$

The class of X is Cj if:

$$P(Cj/X) > P(Ci/X) \text{ for } i \neq j...3$$

However, assuming equiprobable classes and using equations (1) and (2), it can be shown that the rule of equation 3 gives the same result as assigning X to Cj if:

$$P(X/Cj) > P(X/Ci) \text{ for } i \neq j...4$$

One way of storing the values of P(X/Cj) would be as a set of contour maps of the pattern space H1 in which X lies; the height of the contours being the values of P(X/Cj). Hence the term probability map.

The values of P(X/Cj) for each point in H1 could be derived by obtaining a small number of examples of X for each class; calculating their mean position and variance and then imposing some chosen parametric statistical distribution on the map. This approach is satisfactory if there are good reasons to believe that the data belongs to a particular type of distribution. However, in general such reasons are not evident and the distribution of P(X/Cj) must be estimated empirically from a larger number of training examples. This can be done using a Parzen Estimator (4).

N-tuple Recognition Schemes:

It will now be demonstrated that n-tuple recognisers can be designed so that they derive an implicit map of the class conditional probabilities, P(X/Cj), using an approximation to the Parzen Estimator. A Bayes Classification can subsequently be performed by reading the maps as described in the previous section. Since the n-tuple recognition scheme requires almost no computation, it appears to be an attractive way of implementing a Bayes Classifier.

It is proposed that the n-tuple system operate directly on the time domain samples of speech for the reasons explained earlier, although in principle there is no reason why it should not operate on extracted features, with a consequent decrease in the number of training examples required to obtain a certain confidence in the recognition result.

Basic n-tuple Recogniser:

The n-tuple recogniser shown if Fig.1 has been proposed by Aleksander (2). The sample sequence or feature to be recognized is stored as a 2-dimensional array of binary elements with successive samples stored in successive columns and the value of the sample represented by a coding of the binary elements in each column. One of several possible codings is to represent the sample value by a "bar" of binary '1's; the length of the bar being proportional to the value of the sample.

Random connections are made onto the elements of the array; n such connections being grouped together to form an n-tuple which is used as an address to a random access memory (RAM) having one bit per location. A large number of RAMs are grouped together to form a class discriminator whose output is the sum of all the RAM's outputs. This configuration is repeated to give one discriminator for each class of pattern to be recognised.

The system is trained by storing examples of patterns from each class in the array. A logical '1' is written into the RAMs in the discriminator associated with the class of the training pattern at the locations addressed by the n-tuples. This is repeated many times for each class.

In recognition mode, the unknown pattern is stored in the array and the RAMs of every discriminator put into READ mode. The pattern class is then assigned to the class of the discriminator producing the highest score.

Pattern Space Interpretation of Basic n-tuple Recognizer:

Insight can be gained into the operation of the n-tuple recognizer by considering the simple example of a pattern consisting of two sequential samples S1 and S2. All the possible patterns can be represented by points in a 2-dimensional pattern space, with the values of S1 and S2 as the co-ordinate values. It is assumed that the sample values are "bar" coded as described in the previous section.

In this simple two dimensional case, a large number of n-tuples are formed by making random connections onto the array. This results in the generation of an irregular grid of elements in the pattern space, each element corresponding to a particular n-tuple address. An example of the grid produced by a 3-tuple is shown in Fig.2.

During training a pattern S1,S2 is placed in the array and is seen in a particular element of each n-tuple's grid. The elements are remembered by storing a logical '1' in the RAM location addressed by the n-tuple. The intersection of the grid elements defines the small region of pattern space in which the training example occurred; the spatial resolution being governed by the density of n-tuples across the space.

If an unknown pattern vector lies at the same point in space as a training vector then each 2-tuple will cause a '1' to be produced by each RAM and the discriminator will produce a maximum score, Smax, equal to the number of n-tuples in the space. If the unknown pattern moves to some point away from the training point, some of the n-tuples will see the pattern in a new element which has not previously been encountered. These n-tuples will not address '1's in their RAMs and the discriminator score will fall. The score at different points in the pattern space caused by a single training example has been determined by computer simulation for different orders of n-tuple. These results, shown in Fig.3, indicate that the score approximates to a conical kernel function whose base width is reduced by increasing the n-tuple order.

If the system has been trained with two prototypes, it can be shown that the score along a line joining the two training points is given by:

Score(X)=Sig(Score1(X)+Score2(X)).....5

Where Sig(z)=z for z $<$ Smax and Sig(z)=Smax for z $>$ Smax and Score1 and Score2 are the scores resulting from the kernels around the two prototypes.

i.e. The score is given by the linear addition of the kernel functions with the constraint that the score cannot exceed Smax. This is just the operation to produce a score that is proportional to the conditional probability, P(X/Cj), based on the principle of the Parzen Estimator. At points in the space which are displaced from the lines joining training points the addition of the kernel functions becomes non-linear, such that at a large lateral distance from the lines it appears that only one kernel function is present centred on the mean position of the training points. Similarly, at points in the space which have no projection onto the lines between training points, the score is controlled only by the kernel of the nearest training point. In spite of these undesirable properties it is believed that a reasonable estimate of $P(X/Cj)$ will be given if the training points are sparse. An example of the score obtained on a section through the pattern space when the system has been trained with two examples is shown in Fig.4.

The width of the kernel function is an important parameter in a Parzen Estimator. A wide kernel allows a smooth estimate of the probability distribution to be obtained from a sparse set of training examples, but prevents the estimate following a complex distribution. To obtain a good estimate of a complex probability surface it is necessary to use a narrow kernel and many training examples.

In the simulation of the basic n-tuple system it is evident that the kernel function is very broad for n-tuple orders of six and less. Unfortunately it is impractical to increase the order greatly above six because the RAM memory size increases exponentially.

The conclusion is drawn that the basic n-tuple system is incapable of forming an accurate map of a complex probability surface without using a massive amount of memory.

Type-1 Modified n-tuple Recogniser:

The n-tuple system can be simply modified to obtain a narrower kernel function for a given order of n-tuple. It is done by choosing each n-tuple's connections on the sample array such that a regular grid of elements is imposed on the pattern space i.e. The element corresponding to each n-tuple address is made equal in size. The connections for each separate n-tuple are also chosen so that the regular grids are displaced from each other in a random manner with a uniform spatial probability distribution ranging from zero to Q, where Q is the width of each element in the grid.

Q=Pattern space width/$(n/2+1)$....6

This system produces a perfectly conical kernel function whose base width is 2Q. Simulation results for this system are shown in Fig.5.

Type-2 Modified n-tuple Recognisers:

The relationship between kernel width and n-tuple order can be improved further by treating each n-tuple as a bearing compass in pattern space. This can be clarified by the example of a 2-tuple:

The 2-tuple is formed by making connections onto the sample array at value ,a, on S1 and ,b, on S2. An address of '00' will be formed if S1 is less than a and S2 less than b; an address of '01' if S1 is greater than a and S2 less than b, etc. In the 2-dimensional pattern space the 2-tuple can be considered to be acting as a bearing compass located at co-ordinates (a,b) as shown in Fig.6. If a pattern lies to the "south-west" of (a,b) then the 2-tuple address is '00'. If it lies to the "north-west", then the address will be '10' etc.

The ability of the probability map to track complex distributions would be enhanced by using a narrow kernel function. This can be achieved by increasing the number of divisions around each of the "bearing compasses". This is also shown in Fig.6 where each n-tuple gives rise to eight possible bearings. With this arrangement the probability of moving from a set bearing to an unset bearing is large in the neighbourhood of a training point. Consequently, the score will decrease rapidly as the test point is moved. If the score is tested at a large distance from the training point, the only n-tuples contributing a '1' to the score, will be distant from the test point and it will need to be moved through a large distance to change the bearing and hence the score. Thus the score will only change slowly outside the immediate neighbourhood of the training point. These properties suggest that the kernel will have a "square law" appearance whose width is decreased by increasing the number of angular divisions around each n-tuple's location. This is verified by the simulation results shown in Fig.7.

The major problem with the type-2 system is finding a sample coding and n-tuple connection pattern that give rise to the star shapes of Fig.6. One possibility is to connect 2-tuples as for the type-1 system, to obtain four divisions. Finer divisions are obtained by testing the values of the pattern against the equations of each of the star dividing lines for each n-tuple: This involves significant amounts of computation.

Type-3 Modified n-tuple Recognisers:

A third method of controlling the kernel width which does not suffer from the computational difficulties of the type-2 system is as follows: The samples S1 and S2 are stored in the array in binary code and n-tuple connections are made with n/2 connections to S1 and n/2 to S2. The connections always start on the most significant bits of the samples as shown in Fig.8. The n-tuple is connected to a RAM in the normal way.

Patterns lying in different regions of the space will give rise to n-tuple addresses in accordance with a uniform grid of 2^n elements covering the entire space. The dimension of each element being P.

Where: P= Pattern Space width $/2^{n/2}$

Ideally, many n-tuples producing the same grid are required, with uniformly distributed positions over the length (P) of one element of the grid. This is shown in Fig.9 for a system using 4-tuples. In such a system, the maximum possible score is obtained when the test point co-incides with a training point. As the test point is moved, the probability of changing the address of any one n-tuple is uniform within a range of P, of the training point. Beyond distance P from the training point the score will be zero. This implies that a triangular kernel function will be obtained around each training point, whose base width will be equal to 2P. In this way the kernel width can be controlled by choosing the order n of the n-tuple.

The position of each n-tuple grid cannot be directly perturbed by changing the connection pattern. Instead, random variables, Ri1,Ri2, of peak value P are added to S1 and S2 respectively and each perturbed pattern Si1,Si2 is stored in a separate array connected to its own RAM by identical n-tuple connections; This has the same effect as perturbing the grid positions as shown in Fig.9. Examples of the kernel functions obtained with different orders of n-tuple are shown in Fig.10 and the complete system in Fig.11.

Conclusions:

This paper has sought to show that it is difficult to choose optimal explicit features for use in speech recognisers. It has been proposed that the problem be avoided by deriving a probabalistic map of the pattern space spanned by the time domain samples. Bayes Classification is then used to assign a pattern to a particular class. The penalty for this approach is that more training examples are required.

Three possible implementations of the probabalistic map recogniser have been proposed which are based upon n-tuple sampling techniques. These techniques are memory intensive but require negligible computation.

References:

1) W W BLEDSOE & I BROWNING: Proc. Eastern Joint Computer Conf. Boston, Mass. 1959, 'Pattern Recognition & Reading by Machine'.

2) I ALEKSANDER & T J STONHAM: Computers and Digital Techniques, Feb 1979, Vol 2 No 1, 'Guide to Pattern Recognition Using Random-Access Memories'.

3) P A DEVIJVER & J KITTLER: Prentice Hall International, 1982, Page 301, 'Pattern Recognition, A Statistical Approach'.

4) P A DEVIJVER & J KITTLER: Prentice Hall International, 1982, Page 243, 'Pattern Recognition, A Statistical Approach'.

Acknowledgement:

Acknowledgement is made to the Director of Research of British Telecom for permission to publish this paper.

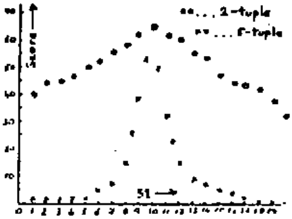

Fig.7 Type-2 Modified n-tuple system score on a section along S2=10 when trained at (10,10).

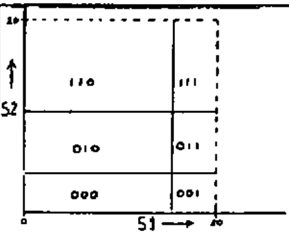

Fig.2 Representation of an n-tuple in 2-D pattern space.

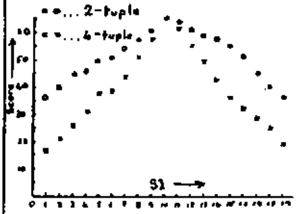

Fig.3 Basic n-tuple Recogniser score on a section along S2=10 when trained at (10,10). 64 n-tuples used.

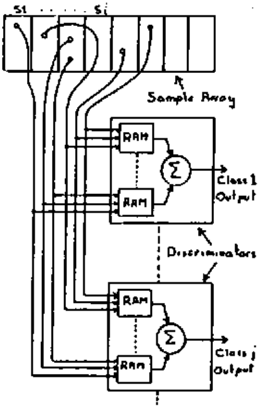

Fig.1 Schematic of Basic n-tuple recogniser.

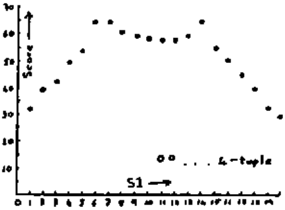

Fig.4 Basic n-tuple System score on a section along S2=10 when system trained at (6,10) and (14,10).

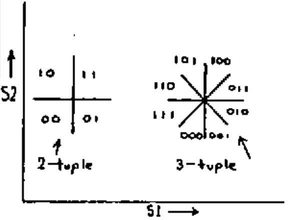

Fig.6 Representation of a 2-tuple and desired representation of a 3-tuple in 2-D pattern space.

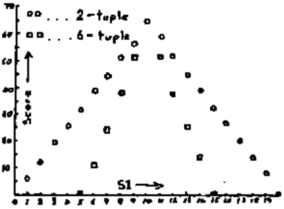

Fig.5 Type-1 Modified n-tuple system score on a section along S2=10 when system trained at (10,10).

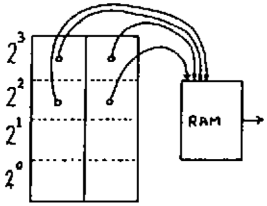

Fig.8 Connection of a 4-tuple onto a binary sample array in a Type-3 Modified n-tuple system.

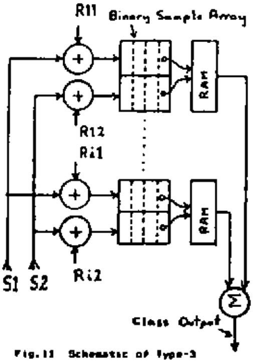

Fig.11 Schematic of Type-3 Modified n-tuple system.

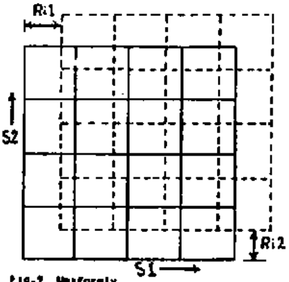

Fig.9 Uniformly distributed n-tuple grids in Type-3 Modified system.

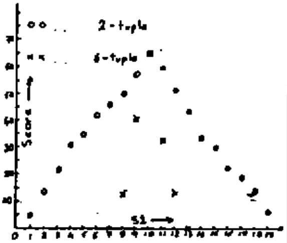

Fig.10 Type-3 Modified system score along a section at S2=10 when trained on (10,10).