Proceedings of the Institute of Acoustics

FINDING TEMPORAL FEATURES OF SPEECH USING KOHONEN NETWORKS

G.D. Tattersall and P.W. Linford
University of East Anglia, Norwich NR4 7TJ

1. INTRODUCTION

This paper describes a technique for selecting commonly occurring temporal features of speech and examines techniques by which the features can be probabalistically associated with different phonemes. The work has been inspired by the phoneme based speech recognition system invented by Kohonen [1] which uses unsupervised neural arrays to learn speech features. This system is apparently successful at recognising Finnish and Japanese languages but the authors have had only limited success in implementing an English language version.

A key problem appears to be finding an appropriate vector description of acoustic events which are associated with perceptual entities such as the stop consonants and of reliably mapping the vectors to an appropriate phonetic label. The techniques used in the Kohonen recognition system are not disclosed in detail, but appear to be based on the use of concatonated spectral frames to represent transitory events coupled with the use of hand crafted high level rules to define a mapping from the spectral vectors to a phonetic label.

The work described in this paper attempts to systematically search for natural temporal features which could be subsequently mapped to phonemes or sub word units. The features are represented by a number of concatonated spectral frames whose values occur frequently in speech.

We have assessed the usefulness of the features by examining the probability of each feature mapping to a particular phoneme label. A good feature will map unambiguously to a single type of phoneme, whereas a poor feature will map to many types of phoneme with similar probability. Our experiments have used phonemes as the feature labels, but it should be possible to apply the same techniques to other types of sub word unit if a suitable transcription of the training utterances is available.

2. FINDING TEMPORAL FEATURES

2.1 The Nature of Temporal Features

If speech is represented by a series of N-dimensional spectral frames, the progression of an utterance can be visualised as a trajectory through the N-space in which different speech sounds are associated with particular shapes of trajectory sub-section.

Natural temporal features can be derived from the N-space trajectory by searching for sub-section shapes that occur very frequently and this is done very simply by using an unsupervised Kohonen Network [2] to perform a cluster analysis. The network is exposed to vectors consisting of a concatonation of successive frames taken at spatially equidistant points along the trajectory and, eventually, the weight vectors of the 'neurons' in the network will tend to reflect the values of

frequently occurring training vectors. The 'neuron' weight vectors thereby become the required natural temporal features of the speech.

2.2 Trace Segmentation

The previous discussion has assumed that an N-space trajectory corresponding to the speech is available. However, a succession of spectral frames taken at uniform intervals in time do not properly define the trajectory because different time warps in the speech may change the times at which the trajectory is sampled and hence lead to a different set of frame values even though the trajectory shape remains the same.

The trajectory must be sampled at equal spatial intervals if its shape is to be unambiguously encoded and this can be done using *Trace Segmentation* [5] by selecting frames at equal spatial intervals along the trajectory. Other intermediate frames are not used in the trace representation of the speech.

The simple premise behind trace segmentation is that the phonetic description of speech is directly related to the *shape* of its N-space trajectory and that the shape is encoded by samples taken at uniform intervals along the trajectory. The original samples taken at uniform time intervals do not lie at equal intervals on the N-space trajectory and therefore do not form an unambiguous coding of its shape.

All the experiments described in this paper have been done using trace segmented speech with a fixed spatial sample interval adjusted so that average number of frames per word is approximately 20.

2.3 The Spectral Representation

A conventional spectral representation was chosen for the speech used in the temporal feature experiments. The speech utterances were sampled at 10KHz and processed to form frames consisting of 8^{th} order LPC derived Mel frequency Cepstral Coefficients (MFCCs). A pre-emphasis of 6dB per octave is included with the 3db point set at 1KHz. Each frame consists of nine coefficients comprising the usual MFCCs C_1 to C_8 and an additional delta energy coefficient whose value is the logarithm of the ratio of the speech energies in adjacent frames. The coefficients do not have normalised variances.

This representation of speech has been used very successfully in HMM and DTW speech recognition systems, and no reason is seen to handicap the Kohonen Net system by using other types of spectral description. The frame rate was set at 4ms to ensure that no information about the speech articulation was lost during the spectral sampling process.

3. THE KOHONEN NETWORK

3.1 Operation of Kohonen Network

Kohonen proposed a neural system [2] consisting of a rectangular array of neural elements, which are all supplied with the same N-dimensional input pattern vector, $X=[x.... x_N]$. Each element contains storage for its own set of synaptic weights. Thus for the i^{th} element in the array a weight vector, W_i can be defined, where $W_i = [w_{i1}...w_{iN}]$. The output of the i^{th} element is given by a measure of similarity, $S(X,W_i)$, between W and X and in this paper, it is defined by the Euclidean distance between X and W_i.

Training of the array takes place as follows. A large representative set of pattern vectors, X, are collected and are applied without supervision and in random order, to the neural array. Every time a vector is applied, the element with the largest output or greatest similarity between X and W is found. A spatial neighbourhood is defined around the element and the synaptic weight vectors of all neural elements lying within the neighbourhood are updated such that:

$$W_i^{n+1} = W_i^n + k * (X - W_i^n) \quad \text{..... 1}$$

3.2 Kohonen Network as a Vector Codebook

It can be seen from equation (1) that the array's weight vectors will tend to take on values which match the values of commonly occurring input vectors and it is this behaviour which allows the net to be used as a vector codebook, and, in fact, the weight vectors distribute themselves such that the mean square error between the training set vectors and nearest weight vector is minimised.

It should be noted that other techiques, such as k-means clustering [3] may be computationally simpler for generating the codebook However, the authors have found experimentally that codebooks generated using the Kohonen net generally give better performance in recognition tasks using k-nearest neighbour classification.

3.3 The Fast Kohonen Net

In the standard Kohonen Net, training is started with a large neighbourhood and its size is reduced as training progresses. An initially wide neighbourhood allows global ordering to be established, and subsequent slow reduction allows the weight vectors to expand to cover the pattern space.

An alternative technique which has recently been used by many workers is the *Fast Kohonen Net* first suggested informally by S.Lutterall of RSRE in which the size of the array is initially set at 2*2 or 4*4. Such small arrays can be made to globally order and converge very rapidly using a fixed spatial neighbourhood of one. Once convergence has been obtained in the small net, its size is doubled, with a new neuron being inserted between every existing neuron. The synaptic weight of each newly introduced neuron is set at an intermediate value between the weights of its immediately surrounding pre-existing neighbours and its value is normally calculated by linear interpolation. Since the array is already globally ordered there is no need to increase the spatial neighbourhood size and the process of net enlargement and training is performed repeatedly until

the required final array-size is reached. We have found experimentally that the number of training iterations at any particular array size should be roughly proportional to the current number of neurons in the array.

The Fast Kohonen Net algorithm is computationally much faster than the original training scheme using a shrinking neighbourhood and has been used to implement the larger nets described in these experiments.

4. CLUSTERING OF TEMPORAL FEATURES

4.1 The Speech Database

Ideally, the database for these experiments would be phonetically very diverse and based on utterances from many speakers. Unfortunately only a very restricted database is available at present which consists of 90 utterances of British place names spoken by a single speaker. The place names were deliberately chosen to be polysyllabic and have been pronounced according to a BBC Pronunciation Dictionary. An analysis of the frequency of occurrence of each of the phonemes defined in the dictionary shows that only about 50% of all possible phonemes are represented with significant frequency. Although this is plainly undesirable, it is believed that sufficient diversity exists to test for the existence of temporal primitives as described in this paper.

4.2 The Clustering Experiments

The clustering experiments are designed to find if tight clusters corresponding to particular features exist for certain numbers of concatonated spectral frames taken at random from a diverse set of speech utterances. A number of training sets have been generated which each consist of 5000 examples of vectors formed by the concatonation of between one and five MFCC frames drawn randomly from trace segmented isolated utterances. The dimensionality of the training vectors ranges from 9 to 45.

Two approaches have been used to detect the presence of clusters. In the first, large (32*32) networks are exposed to each of the training sets and the local density of the resultant weight vectors measured. A basic property of a Kohonen net, trained as described earlier, is that the distribution of weight vectors mirrors the probablility distribution of the data to which the net has been exposed. Thus, a cluster in the data space should produce a cluster of 'weight vectors in the 'weight space' which can be detected by measuring the local density of weight vectors in the array. The vector lying at the cluster centre can then be selected as a feature. Typically the sixteen tightest clusters are selected to provide a set of sixteen features.

Examples of this type of cluster detection are shown in Fig.1a and 1b , in which the weight vector density distributions for two 32*32 arrays are plotted. Fig.1a shows the clusters obtained using single frames as training vectors and Fig.1b shows the results using a concatonation of three frames.

The second approach uses a very small network (4*4) in which it is intended that every neurons's weight vector become a feature vector. In this situation, it is desirable that the weight vectors spread to cover the extremities of the data space but at the same time be attracted to the regions in

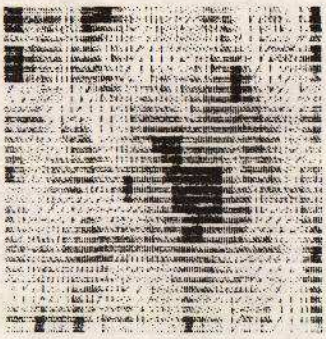

Fig.1a Weight vector density in 32*32 net trained on single frames.

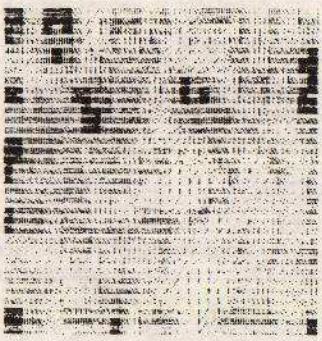

Fig.1b Weight vector density in 32*32 net trained on five concatonated frames.

which the data pdf is *locally higher* than the surrounding pdf. In this way, the array is able to detect features even if their absolute frequency of occurrence is low.

The desired behaviour can be obtained by modifying the learning algorithm of the network so that the amount by which a weight vector is updated is proportional to some power, p, of the distance between the attracting training vector and the weight vector. i.e

$$W_i^{n+1} = W_i^n + k * (X - W_i^n) * |(X - W_i^n)|^p \quad \ldots\ldots\ldots\ldots \quad 2$$

Numerous experiments have shown that using a power of 2 causes good convergence and that the features provided by very small nets correlate better with phoneme types than the features selected from cluster centres in the large 32*32 nets. The following description of feature labelling will therefore be restricted to features derived from small nets.

5. ASSESSING FEATURE USEFULNESS

5.1 The Labelling Problem

The usefulness of the features produced using the Kohonen nets can be assessed very crudely by measuring the relative density of the weight vectors in each cluster. However, a more meaningful measure of is the degree of correlation between the occurrence of a feature and the presence of a particular phonetic event in the speech. In other words, can the feature be reliably labelled as a phoneme or other phonetic unit.

A common method of feature labelling is by excising from speech the spectral vectors which are thought to be associated with a particular phonetic event and then searching for the feature which best matches the excised vector. This approach is very unreliable because, except for vowel sounds, it is very difficult to reliably excise spectral vectors from speech which correspond to particular events. This paper considers some alternative approaches to labelling of the features.

5.2 Phoneme-Feature Correlation Estimation

In this technique the phonetic transcription $[p_1...p_m]$ of an utterance and the sequence of features $[f_1...f_n]$ it causes to be emitted from a trained kohonen net are aligned linearly such that the i^{th} phoneme is aligned with the $(i*n/m)^{th}$ feature in the sequence. A small group of features in a window around the $(i*n/m)^{th}$ point are then associated with the i^{th} phoneme and the probability, $P(p_i,f_k)$, of the joint occurrence of each of the phoneme and feature types is estimated by counting the number of times a particular phoneme and feature are associated over the whole of the utterance database. The joint probability is then normalised to both the apriori probability of the phoneme and feature type to provide a correlation factor $C(p_i,f_k)$.

$$C(p_i, f_k) = \frac{P(p_i, f_k)}{P(p_i) * P(f_k)} \quad \text{..... 3}$$

The value of the correlation factor will be greater than one if a positive correlation exists between a particular phoneme and feature type. An example is given in Table 2 which shows the correlation values for the features produced by a 4*4 net trained on vectors consisting of triples of concatonated vectors.

The correlation values produced by this technique are very approximate because linear segmentation has been used. However, it does provide a guide to the way in which features derived from different numbers of concatonated frames correlate with different phoneme types and this information is summarised in Table 3 which shows the number of concatonated frames which provide the best phoneme-feature correlation value for each kind of phoneme.

i	4	4	3	0	5	5	3	0	8	5	11	0	0	0	0	7
e	0	13	9	0	6	3	1	0	0	0	12	0	2	0	0	3
a	5	14	17	0	2	1	2	0	0	2	3	0	2	0	0	0
o	4	4	0	0	4	1	17	0	0	0	0	0	23	0	0	3
aaa	5	3	4	0	4	6	1	0	11	8	4	0	1	0	0	0
t	5	0	15	18	3	5	1	0	15	14	0	0	0	90	0	5
d	3	3	14	0	4	9	3	0	0	3	14	0	4	0	0	3
k	17	4	6	48	5	1	5	0	2	0	2	0	5	0	0	0
m	3	6	0	0	8	3	6	0	0	1	9	0	4	0	0	2
n	2	1	0	0	8	9	4	0	9	4	7	0	3	0	0	0
l	7	3	3	0	3	4	14	0	0	2	6	0	5	0	0	5
r	6	8	0	11	3	3	3	0	2	3	0	0	9	0	0	0
s	8	1	0	0	1	3	3	0	19	12	0	0	0	0	0	17

Table 2. Correlation values using features from 4*4 net trained on triples of concatonated frames. (Correlation values scaled by 5).

5.3 Probability Estimate Using Viterbi Optimisation

The previous technique can be refined by non linear alignment of the phonetic and feature sequences using the Viterbi [4] algorithm. The Viterbi algorithm allows the most likely alignment of a sequence of phonemes and a sequence of features to be found if the feature conditional

probabilities, $P(p_i|f_k)$, of each phoneme are known. In other words the algorithm computes the probability, P_{max}, of the most likely alignment where:

$$P_{max} = \prod_{max\ path} P(p_i | f_k) \quad \ldots\ldots\ldots\ldots 4$$

Phoneme \ No. of Conctonated Frames	1	2	3	4	5
i	3.4	2.2	2.2	1.4	2.0
e	2.6	2.6	2.6	1.6	2.8
a	2.6	20.2	3.4	10.2	10.2
o	33.0	7.8	4.6	4.8	7.4
aaa	3.2	2.8	2.2	2.6	5.4
t	3.0	5.2	18.0	2.6	2.4
d	3.6	2.4	2.8	1.8	1.6
k	12.0	3.2	9.8	2.6	2.4
m	2.0	2.0	1.8	8.6	8.4
n	1.6	1.6	1.8	4.0	4.0
l	6.4	4.2	2.8	3.0	4.4
r	3.0	2.4	2.2	1.8	3.6
s	5.4	5.6	3.4	11.0	5.6

Table 3. Maximum phoneme-feature correlation for different numbers of concatonated frames

Initially, only poor estimates for the conditional probabilites are available, but these can be used in conjunction with the Viterbi algorithm to define a tentative alignment path and the probability estimates can then be updated using gradient descent. A new alignment path is then computed and the process repeated until the conditional probability values are optimized to maximise the aggregated probability, P_{total} of the alignment paths of all utterances in the training set where :

$$P_{total} = \sum_{i=1}^{\text{no of utterances}} P_{max,i} \quad \ldots\ldots\ldots\ldots 5$$

The derivitive $d\,P_{total}/d\,P(p_i|f_k)$ required for gradient descernt optimization is obtained from equations 4 and 5:

$$\frac{d\,P_{total}}{d\,P(p_i|f_k)} = \sum_{\text{all utterances}} \frac{P_{max,i}}{P(p_i|f_k)} \quad \ldots\ldots\ldots\ldots 6$$

The modification to the conditional probabilities must be done under the constraint that their sum over all phonemes in the utterance is unity, and the constraint is applied by renormalising all the conditional probabilities after they have been updated.

The optimization algorithm should cause the phoneme_feature correlations to become more pronounced and make each of the features more discriminatory because the features and phonemes are being optiimally aligned in time. As an example, the algorithm has been tested by taking the conditional probability values from which the correlation values in Table 3 have been estimated and

optimizing them using gradient descent and the Viterbi algorithm. Table 4 shows the correlation values obtained using the optimized conditional probabilities, and it can be seen that in comparison with the values in Table 3, the correlations are much greater for specific phoneme-feature pairs.

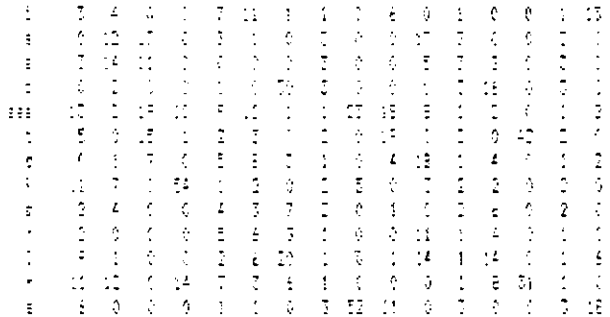

Table 4. Viterbi optimized correlation values using features from 4*4 net trained on triples of concatonated frames. (Correlation values scaled by 5).

6. CONCLUSIONS

This paper has sought to find if natural temporal features exist which can be described by various numbers of concatonated spectral frames taken from speech traces. The paper has also attempted to assess the usefulness of such natural features by mapping them probabalistically to phonemes and examining the peakiness of the phoneme-feature probability distribution. Finally an optimization algorithm has been demonstrated which allows non-linear alignment of a sequence of features and phonemes in order to better estimate the probabalistic mapping from phoneme to feature.

The results of the investigation into natural features are rather disappointing because no clear optimal length of feature is evident for different phoneme types as indicated in Table 2. This may be because the selection of frames by trace segmentation is inappropriate, the Kohonen nets used for cluster detection are not fully converged, or most likely, the phonemes do not map unambiguously to natural primitives of speech.

More encouragingly, the use of the Viterbi Optimization algorithm to align the phoneme and feature sequences does appear to work correctly and an improvement in the correlations between specific phonemes and features is observed using this technique.

7. REFERENCES

[1] Kohonen T., *A neural phonetic typewriter*, IEEE Computer, pp11-22, March 1988.
[2] Kohonen T., Clustering, *Taxonomy, and Topological Maps of Patterns*. Proc. Int. Conf. on Pattern Recognition, October 1982.
[3] Wilpon J. , Rabiner L. , *A modified K-means clustering algorithm for use in isolated word recognition* . IEEE Trans. on ASSP, Vol. 33 No. 3, June 1985.
[4] Viterbi A.J. *Error bounds for convolutional codes and an asymptotically optimum decoding algorithm*. IEEE Trans. Information Theory IT-13, pp260-269. 1967.
[5] Khun M.H., *Fast nonlinear time alignment for isolated word recognition*, Proc. ICASSP, pp736-740, 1981.