Proceedings of The Institute of Acoustics

REASONING ABOUT THE ACOUSTIC REALISATION OF SEMIVOWELS USING AN INTERMEDIATE REPRESENTATION - THE 'SPEECH SKETCH'.

P.D. Green and A.R. Wood

Department of Computing, North Staffordshire Polytechnic,
Blackheath Lane, Stafford

ABSTRACT

We argue for an approach to acoustic-phonetic reasoning in speech recognition based on the 'Speech Sketch' - a data structure in which the behaviour of spectral parameters is explicitly described. We represent acoustic-phonetic knowledge in frame-like structures whose terminals map onto the descriptors of the speech sketch. Part of a frame expresses how the manifestation of the event the frame describes is expected to change with context. The speech sketch describes what has happened and the frames describe, in an equivalent way, what is expected to happen in particular speech events. 'Recognition' within this paradigm, is achieved by a recursive matching function which resembles the 'inference engine' of an expert system. We discuss first results from a testbed implementation of the above ideas, in which a speech sketch for formant frequency data is used to distinguish between /l/, /r/, /w/ & /j/ in various environments.

INTRODUCTION

Our interest is in acoustic-phonetic analysis in computer speech recognition or speech understanding systems - in the relationship between the acoustic evidence derived from an utterance by one or more forms of signal processing and the fundamental perceived units in terms of which speech is understood. We see this as a problem for Artificial Intelligence techniques - we would like to capture a phonetician's expertise about how speech events manifest themselves and to be able to reason with this knowledge base.

Our approach to acoustic-phonetic reasoning is based on an intermediate, descriptive data structure which we call the Speech Sketch (SS) previously introduced in [4,5,6]. In summary, our argument for the SS is as follows:-

1. Acoustic-phonetics must link two data structures - acoustic parameters and speech knowledge - which are represented in quite different ways. The parameters are continuous measurements while the knowledge is best organised in terms of the discrete perceived entities which phoneticians talk about - phones, allophones, distinctive features, formant behaviour and so on.

2. It only makes sense to compare like with like. One can achieve this, and avoid the reprepresentation problem, by making the knowledge look like the parameters (template technology) or by making the data look like the knowledge (segmentation and labelling). Both these extremes, however, have serious limitations as general schemes for acoustic-phonetic processing, template technology because it is not an explicit reasoning process and segmentation and labelling because speech is to a large extent, a flow rather than a separable sequence of events. We would contend that the interpretation of speech as a string of phonemes is one of the products of the whole speech perception process rather than an early step along the way.

REASONING ABOUT THE ACOUSTIC REALISATION OF SEMIVOWELS USING AN INTERMEDIATE REPRESENTATION - THE 'SPEECH SKETCH'.

3. We propose to directly confront the representation problem, as vision researchers have been forced to do [8] by introducing an intermediate data structure, the SS, whose purpose is to make it possible to represent parametric behaviour and phonetic knowledge in compatible ways.

4. From the point of view of the incoming data, the SS is a data structure composed of a number of descriptors, each of which explicitly describes some fragment of parametric behaviour. The SS is constructed using only very general, low level constraints, for instance that formants, in voiced sounds, are continuous. No attempt is made to interpret the significance of a descriptor: the aim is simply to describe.

5. From the point of view of the stored knowledge, the SS corresponds to the terminal level of the representation, so that the lowest level operation in speech recognition is a matching between an individual descriptor in the SS and an expected pattern in memory.

The application of expert system methodology in acoustic-phonetics is a central theme of the nascent Edinburgh-Plessey Alvey speech transcription demonstrator project [10]. Here the system is organised around active chart parsing, and it would seem natural to think of the SS as being one level, perhaps the lowest level, of the chart. The group at Turin, formerly led by De Mori, have conducted the most extensive study of acoustic-phonetic reasoning to date [2,3]. In their work, individual parameters are described as a sequence of dips, valleys, steady regions and the like, in a language which maps onto their knowledge formation, based on grammars. We envisage that the SS should be a far richer description than this, and we feel it is important to keep a clear distinction between description and interpretation. In later work, De Mori has reported using frames (the knowledge representation scheme we use) for at least part of his system. Interest in spectogram reading [7,13] is also related to our approach - here the visual representation of the spectrum and the experts' visual expertise play the role of the SS. Darwin proposes to use operators like Marr's 'Mexican Hat', and grouping processes, to produce a description of a critical band spectrogram.

A TESTBED

We do not have the facilities to construct a full scale speech recognition system. We have developed a minimal demonstration of the rather ambitious ideas outlined in the previous section. In this testbed, we attempt to distinguish between the semivowels or glides (/l/, /r/, /w/ and /j/) on the basis of formant frequency estimates derived by an LPC algorithm [1], which provides up to 6 Formant values every 10ms. We do not have any great loyalty to this form of preprocessing and our software could use formant data derived by any other means. We chose to look at the semivowels primarily because we wanted to show that we could represent the effects of heavy context-dependancy. To provide different contexts, we have been looking at vowel-semi-vowel-vowel sounds like /iw3/ (...we were going...) or /aju/ (...are you going...). Figure 1 shows an example of formant frequency data for an utterance /3wa/. A human expert can identify the vowels and the semivowel quite readily, though not infallibly, from such data.

THE SPEECH SKETCH FOR FORMANT FREQUENCY DATA

We wish to make changes in formant frequencies explicit, since we want to reason

in terms like 'falling F1', sharply rising F2' and so on. Each descriptor in the SS should represent a reasonable description of some time fragment of some formant behaviour. However,

1. It is not necessary to track formants explicitly since the interpretation of a descriptor as part of, say, F2, can be left to the matcher.
2. With our data, it is not desirable to track formants since the LPC algorithm makes errors: transitions are sometimes not followed well and spurious formants may be introduced. While it is undoubtedly possible to improve on the quality of the preprocessing, it is important to be able to cope with noisy parametric data, as it is with messy edge information in vision.
3. It is not necessary for the SS to be unambiguous, or for every descriptor it contains to be of eventual significance. If there is more than one distinct way of describing some parameter fragment, the SS should include them all.

The above considerations are largely satisfied by a scheme based on straight-line descriptors, though it may be necessary, in future work, to move to a more general representation of the shape of formant trajectories. We have tried a variety of ways of producing the SS for our testbed, but our current SS builder works as follows:-

1. Each point in frequency-time space is associated with its nearest neighbour in the preceding and succeeding time frames. This captures the constraint that formants should be continuous without committing us to any hard and fast tracking decisions. Figure 2 illustrates. Note that we have cut off the SS above 3.5 KHz, since only F1, F2 and F3 are significant for recognition.
2. For each point, we find the best-fitting line that can be obtained by extension along its nearest-neighbour path or paths. The SS descriptors are formed from all such lines which are sufficiently different, as shown in figure 3.

If anything, this version of the SS builder is too conservative, producing fewer, shorter descriptions than we would like - typically, about 100 descriptors are produced per second of speech. It may be advantageous to perform a limited amount of grouping on the results of 1/ and 2/., adding new descriptors to represent neighbouring, similar, existing ones. In general, the SS builder should produce a richer description than the scheme we have implemented - for instance there should be descriptors to represent cases where several events have happened more or less concurrently.

REPRESENTING ACOUSTIC-PHONETIC KNOWLEDGE IN FRAMES

Our knowledge representation scheme is based on one of the best-known formalisms, that of frames, first proposed by Minsky [9]. A frame represents what the system knows about how a particular entity is expected to be manifested in the SS, including knowledge about the ways in which this manifestation will change with context. We chose frames rather than production rules or grammars because we wanted to describe a relatively small number of relatively complex objects, and because it was important to be able to represent class inheritance - for instance to have a generic frame representing vowels which specialises into frames for /i/, /e/ etc., which inherit the characteristics which are common to all vowels.

REASONING ABOUT THE ACOUSTIC REALISATION OF SEMIVOWELS USING AN INTERMEDIATE REPRESENTATION - THE 'SPEECH SKETCH'

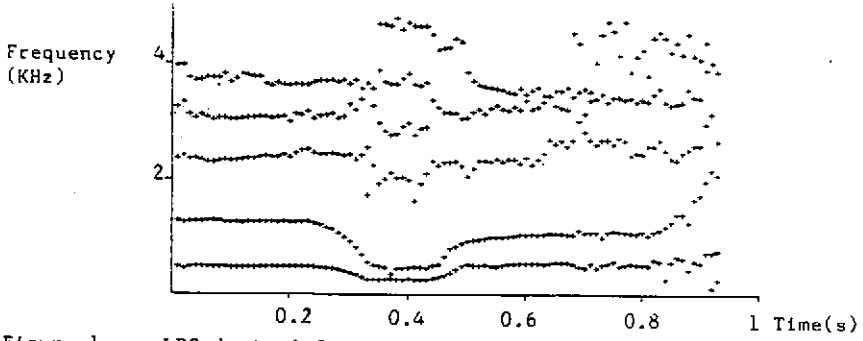

Figure 1: LPC-derived formant frequency estimates for an utterance /ʒwa/.

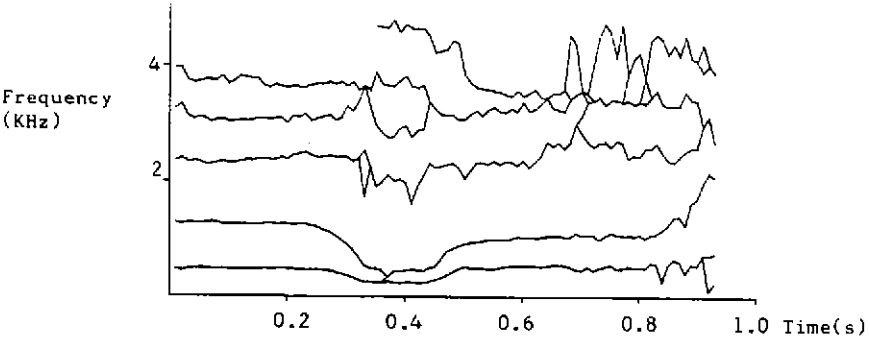

Figure 2: Nearest neighbour paths for the data of figure 1

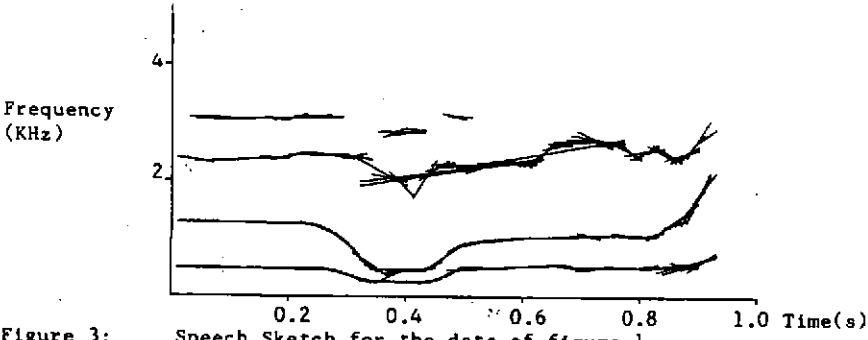

Figure 3: Speech Sketch for the data of figure 1

REASONING ABOUT THE ACOUSTIC REALISATION OF SEMIVOWELS USING AN INTERMEDIATE REPRESENTATION - THE 'SPEECH SKETCH'.

A frame is a list of 'slots'. One slot specifies the frame's 'components' - the things one expects to find in the speech event the frame describes. An *AND frame* requires all its components to be matched while an OR frame requires a match for one of its alternative components. We can also represent the idea of a NOT - a component whose presence inhibits a match for the frame. A component may itself be a frame or, at the terminal level, a pattern to match against the SS. Further slots in the frame specify how the components relate to each other, the relative importance of finding a match for each component, and any other knowledge which may be useful. For instance, we might have a generic frame for a glide, one of whose specialisms is the frame for a /w/. The /w/ frame might have 3 components, representing expected F1, F2 and F3 behaviour. These components must be concurrent. The frames describing the individual formants would have 3 sequentially related terminal components representing an initial transition, a steady state and a following transition.

A frame is always matched within a developing context which supplies timing information, any preceding or following formant values already established, and the like. Our mechanism for representing coarticulation effects is to associate one or more functions with a component which trigger on information in the current context and change the expected form of the component. Thus, if we are looking for a match for the initial transition in F2 for a /w/ and we know the preceding F2 value, we can say that the transition should start at this value or, if the preceding F2 value is close to the steady state frequency for a /w/, we should not expect to see the transition at all.

'SPEECH RECOGNITION' - MATCHING FRAMES AGAINST THE SPEECH SKETCH

The fundamental 'recognition' operation within our system is the 'instantiation' of a frame - the discovery of evidence, within the SS, for an arrangement of descriptors consistent with the expectations in the frame and the surrounding context. This region can then be interpreted as an instance of the entity that the frame represents - ie we have segmentation by recognition rather than recognition by segmentation. In a complete system, one would need a variety of computational tools for instantiating frames, and a control strategy to deploy these tools. This procedural and structural knowledge might itself be represented in frames, as reported in later work by De Mori et al [3], where a frame-based scheme is used to control the formation of pseudo-syllable hypotheses. For our testbed, we have developed a single top-down algorithm FMATCH, which is essentially a backwards-chaining inference engine whose structure reflects the structure of a frame.

FMATCH is called to try to instantiate a given frame in a given context. It schedules its activity by means of an event queue. An event is a partial instantiation of a frame in which some components have been matched, some have been tried and failed to match and some have not yet been tried at all. Each event carries its own context, derived from the initial context for the FMATCH call modified by the work which led up to this event. The event Queue is ordered by a shortfall scoring metric [12], which preserves admissability - the first complete instantiation FMATCH finds is guaranteed to be the best-scoring one.

FMATCH selects the best-scoring event from the event Queue and tries to instantiate its next untried component, first modifying the component using the context-sensitive functions associated with it. The component instantiation is achieved

by a recursive call of FMATCH (if the component is itself a frame) or (if the component is a terminal) by a direct match against the SS. In this case, the component and the current context define an envelope which is used to select candidate descriptors from the SS, and suitable descriptors in the envelope (for instance, those with about the right slope) form terminal instantiations.

To help to prevent an explosion in the search space of partial instantiations, we make much use of a facility called 'resume functions'. An FMATCH call will not, in general, proceed to completion but will suspend itself when the score of its best event has dropped significantly, returning this partial result and a function of zero arguments which, when called,has the effect of resuming the FMATCH call. Thus we can ensure that, at all times, we are working on the most promising partial solution so far found. Resume functions are particularly easy to program in our implementation language, POLYLISP-83 [11], as a consequence of its purely static scoping.

INITIAL TESTING

At the time of writing, we are deeply engaged in development and evaluation of our testbed. We are not in a position to present formal results but we can report briefly on progress so far. For our first real test, we designed a frame library, by hand, from examination of 32 vowel-semivowel-vowel utterances by a single speaker. We then tried to identify the semivowels in a further 16 utterances by the same speaker, in different combinations, using this knowledge base. We first used FMATCH to find the surrounding vowels, which it did very well, and then tried to instantiate a semivowel frame in the context thus provided.

In English, the system's knowledge about how to distinguish between semivowels is, without going into all the details,

'The primary discrimination between the semivowels will be in F2, which is low for /w/ (about 530 Hz for this speaker), high for /j/ (about 2200Hz), and medium for /r/ and /l/ (about 1120Hz). In order to distinguish between /r/ and /l/ it is necessary to look at F3 which should be higher for /l/ (about 2400Hz) than for /r/ (1690Hz). All four semivowels have about the same F1 (240Hz) but it may be useful to look at F1 in order to get timing information. For each formant, expect, ideally, to see a transition from the preceding vowel positon to the natural value, then a brief steady state, then a final transition to the following vowel position. However, the initial transition will not appear if the preceding vowel position is close to the natural value, conversely for the final transition, and the steady state may not appear at all'.

Apart from programming difficulties, problems can arise from inadequate evidence in the SS and inadequate knowledge in the frames. An example of the former is in making the /r/-/l/ distinction. This depends on F3, which is often not picked up clearly by the LPC algorithm. Although the SS builder does not require high performance in formant estimation, it does need some systematic evidence. A problem which requires more knowledge arises from the fact that, for this speaker, the formant positions for the semivowel /j/ are very close to those for the vowel /i/. It is possible to interpret an initial transition from an /i/ to, say, a /w/ as a final transition from a /j/ to the following vowel, and vice versa. To fully overcome this problem it is necessary to deploy more global knowledge, for instance about speaking rate, or to look at parameters other than formant frequencies.

BOLTZMANN MACHINES FOR SPEECH PATTERN PROCESSING

A SIMPLE CONSTRAINT SATISFACTION NETWORK

In this section we illustrate some properties of CSNs and the noisy relaxation search, using an extremely simple example which should not be taken too seriously.

Fig.1 shows how a few terms familiar in phonetics might be related in a CSN. Connections with arrows are positive weights, (reinforcing, excitatory) which tend to make both units come on together. Connections with blobs are negative weights, (inhibitory) which tend to suppress one unit if the other is on.

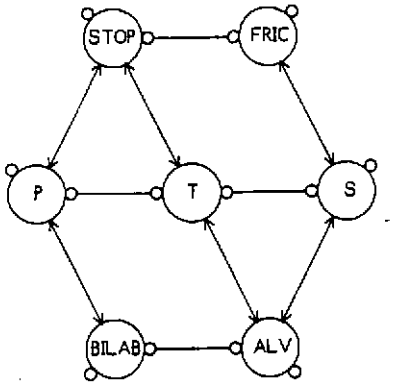

Fig.1: A very simple constraint satisfaction network

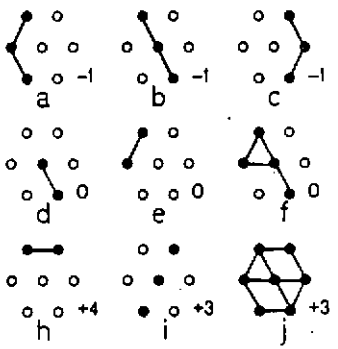

Fig.2: Energies for some global states

The network could be thought of as expressing logical relationships, such as: an /s/ is a stop, something cannot be both velar and alveolar, etc. However, the global evaluation principle simply scores different configurations as more or less plausible. The energy is minus the sum of the weights on connections joining two units that are both 'on'. Each unit has a bias, which can be thought of as a weight joining it to a permanently 'on' unit.

Fig.2 shows the energies for some representative global states, for the case that all weights are +2 or -2, and the biases are -1. The minimum energy states correspond to complete statements that are acceptable (2a-c). Slightly higher energy states correspond to partial or somewhat conflicting 'interpretations' (2d-f), while ridiculous states have high energy (2g-i).

Left to itself at some non-zero temperature, the noisy relaxation algorithm will spend most time in 'meaningful' states, but will move from one to another at random. (The mean time between movements will be controlled by the temperature and the height of the potential barriers between low energy states.)

If we constrain the states of some of the units, the relaxation algorithm will attempt to complete the pattern. For instance, if we turn on STOP and ALV, the minimum energy configuration is Fig.2b. If we turn on P then STOP and BILAB will tend to turn on. If we just turn on ALV, then the system will alternate between 2b and 2c.

The units of a BM are in an 'on' state or an 'off' state at any particular time, and in general the proportion of time spent in the on state in equilibrium represents the system's confidence in the elementary hypothesis that the the unit deals with. One method of applying an input pattern to a BM is simply to 'clamp' the states of some of the units, which are then in effect regarded as input connections.

CONTINUITY OF INPUT VALUES

In the above example, the input patterns are essentially binary. However, in most speech recognition systems the raw pattern data is an array of values such as spectrum amplitudes, which in principle take continuous values. In this situation there are several options for applying such values to a more practical BM.

The speech pattern values could be 'binarised' in some way, and applied directly to an input layer of the network. Hinton suggests that values be represented by sets of units, each covering a range of values.

The continuous-valued inputs could also be applied as biases, direct to the 'sensory units'. These units will then act as noisy, context sensitve, threshold units, and for small input values relative to the noise standard deviation, will encode the input value as a probability.

It is possible to treat a continuous input value as if it were the probability of a (fictional) unit being 'on'. In this case the search and the weight adaptation involve a little more arithmetic, but the same formulae can be used. This 'fictional input unit' technique can lead to an arrangement in which a real unit forms a weighted sum of individual measurements. In the case of a time-spread array, this is equivalent to an FIR filter, and the resulting weight adaptation method is closely related to that used in adaptive equalisers.

Another possibility is that the input data could use a more complex binary code, in which correlations between 'spectral' channels would be important. There is some evidence that auditory nerve data encoding has such a property.

CONTINUITY AND UNIFORMITY OF TIME

Boltzmann machines were devised primarily for processing static visual images, including stereo pairs. Speech patterns, on the other hand, are essentially functions of time, and any method of dealing with speech patterns should explicitly account for temporal behaviour. Markov models include time in their formulation, but it is not obvious how to include time in a BM. Hinton argues strongly against the natural temptation to use the dynamics of the relaxation process to handle time-varying input.

For dealing with acoustic patterns, it is perhaps most useful to treat time as another dimension of the pattern (like frequency) and spead out our data and our network across each such dimension. CSN's that apply to an instant of time (eg Fig.1) are repeated regularly along the time axis, and knowledge about

relationships between one moment in time and the next is encoded in connections which join units at different time locations. We consider that it is very important for the network to be homogeneous with respect to time, so that the behaviour of the network will be independent of time (except for the influence of the input). This also means that the number of different weights may be much smaller than the number of different units.

For simple time-spread networks, the connections and weights are all repeated at each time instant, and all we need to know about the network is the spacing along the time axis, and the connections and weights for units in a single time-slice. More complex networks will need different time-spacings for units dealing with different levels of representation.

A similar procedure may be appropriate for the frequency axis, but the weights will probably need to be functions of frequency, perhaps expressible in terms of the first few coefficients of a frequency-axis basis function set such as that used in the cosine transform.

A VERY SIMPLE SPREAD NETWORK EXAMPLE

Fig.3 shows a very simple, one-dimensional, regular network, with the same pattern and values of weights for every one of the units. Each unit receives an input via a weight of value a, has a lateral connection of value c to each of its immediate neighbours, and has a bias b. We can imagine that, with appropriate values for a,b and c, a network like this might be used to pick peaks in a spectrum cross-section, or respond to interesting features of a short-term-power-versus-time profile. More interesting behaviour would be possible with a more general version, with lateral connections to more than the adjacent units, input connections to more than the local input value, and connections to 'higher-level' units of various kinds.

The energy of a global state of the network of fig.3 is

$$E = -\sum_{i} s(i).[b+a.d(i)+c.(s(i-1)+s(i+1))/2]$$

where d(i) is the ith input value and s(i) is the state (0 or 1) of the ith unit.

The local decision rule is

If $F + N(0,T) > 0$ then set $s(i)=1$ else set $s(i)=0$

where $F = b+a.d(i)+c.s(i-1)+c.s(i+1)$
and N(m,s) is a sample from a Gaussian distribution of mean m and standard deviation s.

Let us assume that a and c are positive and b is negative, as implied by the arrowheads and blobs. The local decision function is a noisy, context-sensitive threshold. The threshold, which is -b if the neighbours are off, reduces to -b-c with one neighbour on, and -b-2c if both neighbours are on. The result is that isolated input values of less than -b will not lead to stable

BOLTZMANN MACHINES FOR SPEECH PATTERN PROCESSING

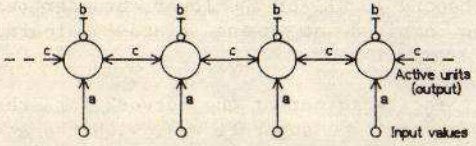

Fig.3: A very simple spread network

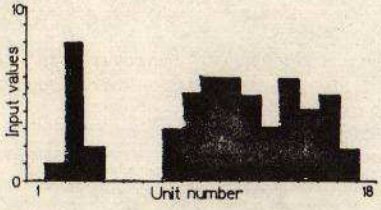

Fig.4: Input values for the spread network

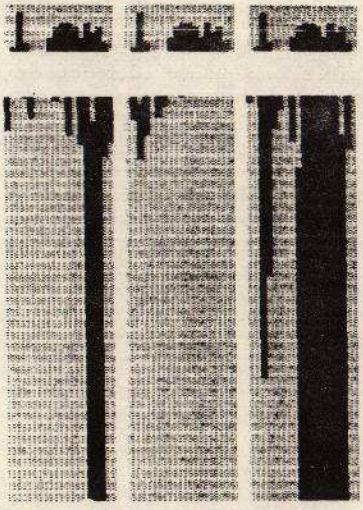

Fig.5: Three runs at zero temperature

Fig.6: Three runs at higher temperature

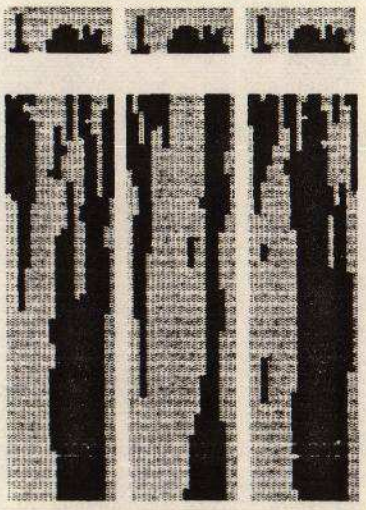

Fig.7: Three annealed runs

configurations of 'on' units, but 'clumps' of high-ish values may well do so, even if they contain a few, isolated low-spots.

Fig.4 shows an input pattern with such features. The desired response is to ignore the isolated peak and respond to all of the lower, broader peak. Suitable weights (a=10,b=-86,c=53) were arrived at using Hinton's learning algorithm [2].

Figures 5,6 and 7 show typical settling behaviour of the network for three different conditions. Each vertical column is a separate run, with the array started in a random state at the top of each column. Each row of a column shows the states of all the units (black is 'on'). A new row is displayed after each (random) selection of a unit and local decision. The input pattern is shown again above each column for comparison.

Fig.5 shows three runs at zero temperature, so only 'improvements' to the score are accepted. Note that the system soon settles into some local optimum state.

Fig.6 shows three runs at a steady non-zero temperature. The state of the network tends to hover round the desired global minimum, irrespective of the starting state.

Fig.7 shows three runs using 'annealing', starting at a high temperature and reducing to almost zero at the bottom of the column. In all the runs shown the state falls into the region of the global minimum before the end of the column.

REAL-TIME BOLTZMANN NETWORKS?

If the above example is regarded as a time-spread network, then the annealing process has to be applied after all the signal has arrived. However, we would like the annealed search to proceed while the input is arriving, with interpretations of the recent past becoming available as soon as it is sufficiently unambiguous.

One interesting possibility is to introduce a temperature gradient across the time dimension. As it arrives, data is applied to a 'hot' part of the network, which cools as the data gets older. This 'continuous flow annealing' is analogous to certain industrial processes, and also to the process of magnetic tape recording, where the decay of the AC bias field as the tape leaves the recording head gap provides the equivalent of a temporal temperature gradient.

An interesting consequence of processing the data as it arrives is that the units implementing higher-level constraints will tend to pre-condition the lower-level units in advance of the data, thereby reducing the settling time in most cases.

DISCUSSION

The examples of this paper are of limited interest because every unit is an input or an output. It is a well-established practice to describe the regularities of the sound patterns of speech using a hierarchy of intermediate levels of representation. Hinton et.al. have shown that a BM can learn to use 'hidden' units to represent intermediate-level concepts [2]. In practice we have the choice of how many assumptions and how much structure to build into such an adaptive network.

Boltzmann machines are rather inefficiently implemented in a conventional serial digital computer. However, the possibilities for asynchronous, parallel architectures are obvious, and it is also worth considering the possibility of a special new type of integrated circuit using analogue computation and thermal noise.

Although we have presented BMs as an alternative to hidden Markov models etc., there are interesting relationships between the two methods. For instance, the array of fig.4 can compute the maximum likelihood sequence of states from the output of a two-state, two output symbol, HMM. An exploration of such relationships should prove fruitful.

CONCLUSIONS

We feel that the Boltzmann Machine approach offers so many desirable properties that it should be pursued, not only for the potential benefits to speech technology, but also for the insights which it may provide into the mechanisms of speech perception.

REFERENCES

1. G.E.Hinton, "Inferring the meaning of direct perception", The Behavioral and Brain Sciences (1980), 3, pp.387,388.

2. G.E.Hinton, T.J.Sejnowski and D.H.Ackley, "Boltzmann machines: constraint satisfaction networks that learn" Technical report CMU-CS-84-119, Carnegie-Mellon University, May,1984.

3. G.E.Hinton and T.J.Sejnowski, "Analysing cooperative computation", Proc. 5th Ann. Conf. Cognitive Science Soc., Rochester, NY, May 1983

4. G.E.Hinton and T.J.Sejnowski "Optimal perceptual inference", Proc. IEEE Computer Soc. on Computer Vision and Pattern Recognition, Washington, DC, June 1983, 448-453.

5. S.Kirkpatrick, C.D.Gelatt and M.P.Vecchi, "Optimisation by simulated annealing", Science, 1983, 220, pp.671-680.