Proceedings of The Institute of Acoustics

BOLTZMANN MACHINES FOR SPEECH PATTERN PROCESSING:

J.S. Bridle (1) and R.K. Moore (2)

(1) Joint Speech Research Unit, Princess Elizabeth Way, Cheltenham
(2) Royal Signals and Radar Establishment,, St. Andrews Road, Malvern

INTRODUCTION

The aim of this paper is to increase awareness of some recent developments in visual perception modelling, and to discuss some implications for speech applications.

All modern speech recognition machines are based on variations of the same methods for representing speech knowledge and for searching for interpretations that are consistent with the data and the knowledge. These methods (Markov models and dynamic programming) provide: a basis for encoding speech knowledge in a structure with many levels of representation (e.g. spectra, phonetic segments, words, noun phrases..); a well-defined measure of the agreement of any particular interpretation with the data and the knowledge; and a very efficient method of searching for good interpretations.

There is great scope for development of these methods, and it is important that this development should go ahead. However, there is also good reason to look for alternatives. The main algorithms of the standard method are dynamic programming and the related 'forward-backward algorithm'. Both rely on the 'Markov property': this means that the speech model can have little memory of its past outputs, and consequently it is very difficult or impossible to include in the structure of such models many of the rich interdependencies between different parts of the pattern that many people assume are needed.

These limitations have caused some speech scientists to turn to "artificial intelligence" style symbolic reasoning methods for knowledge representation and for searching for good interpretations. We find this approach unconvincing for the following reasons. Speech signals are not symbolic, and any attempt to force a segmentation and labelling on the data before all knowledge sources have been applied is bound to be unsatisfactory. The symbolic methods of linguistics and current artificial intelligence should be useful for describing and designing natural and artificial perception systems, but at a level of description considerably 'above' the implementation method. The human speech perception process, which is the target of all natural speech, is unlikely to be based on symbolic reasoning. Rather, we prefer Hinton's vision of perception as "a parallel, distributed computation in which a large network settles into a particular state" under the influence of the sensory input [1].

This paper presents an introduction, for the speech technology research community, to the concepts of adaptive stochastic constraint satisfaction networks and "optimisation by simulated annealing". We indicate how these methods might be applied at various levels of speech pattern processing, and illustrate with some very simple networks. The next section is a very quick overview of the main ideas, some of which are re-introduced more gently later on. We strongly recommend reference [2].

BOLTZMANN MACHINES FOR SPEECH PATTERN PROCESSING

BOLTZMANN MACHINE OVERVIEW

In a series of eloquent and persuasive papers [2,3,4] Hinton, Sejnowski and others have recently presented a method for representing 'knowledge' in the pattern and strengths of the connections of a 'constraint satisfaction network' (CSN) composed of very simple units which can each be in either an 'on' state or an 'off' state. The input 'sensory' data constrains the state of some units of such a network.

An 'interpretation' of the data is a global state of the network (a pattern of ON and OFF states of the units). Each global state of the network can be assigned a single number called the 'energy' of that state. The energy can be interpreted as a measure of the 'implausibility' of the interpretation, given the data and the knowledge represented by the network.

The search for good (low energy) global states is done by a relaxation algorithm. A unit is selected (randomly) and the difference in the global energy is computed for the two possible states of that unit, given the current states of the other units. A simple relaxation algorithm would set the unit to the state for which the global energy is lower, but this procedure tends to get stuck in local minima. The solution is to make the decision probabalistic: uphill steps are then possible, and the system can find its way out of local minima. There are several ways of making the decision probabalistic. One method is to compute the difference in energy for the two possible states, add a random number from a Gaussian distribution, and compare with zero. We shall refer to this random number as 'noise'.

The amplitude of the noise is analogous to the temperature of a physical system of interacting particles. The recommended method for reliably finding good minima in a limited time is to start the system at a high temperature and reduce the temperature carefully. This technique is known as 'optimisation by simulated annealing' [5], and its applications are much wider than presented here. It has already been used successfully in the automatic design of layout and wiring for integrated circuits [5].

As in statistical mechanics, the probability of finding the system in a particular global state is related to the energy of that state, and this relationship is governed by the Boltzmann distribution. It turns out that the mathematical properties of the Boltzmann distribution permit analysis of the statistics of the search process, and lead to a method of adapting the strengths of interconnections (weights) so that the behaviour of the network can capture the essential properties of classes of 'training' patterns. Such an adaptive constraint satisfaction network is refered to as a 'Boltzmann machine' (BM) [2].

In this paper we concentrate on examples of very simple CSN's and BM's which seem to us to be relevant to speech. Some details of the BM algorithms are introduced where appropriate, but we do not attempt to improve on the excellent presentation of Hinton et.al. We do not deal with weight adaptation, although this is seen as essential for tuning BM's in any practical application.

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.

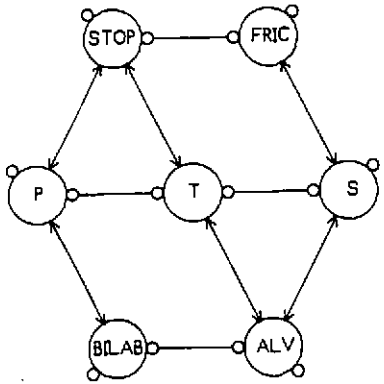

Fig.1: A very simple constraint satisfaction network

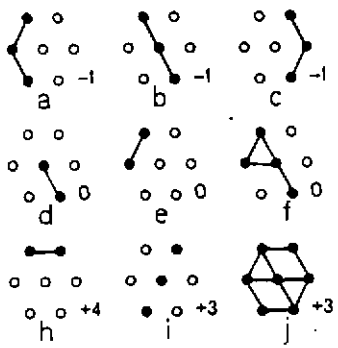

Fig.2: Energies for some global states

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

SELF ORGANISING ARRAYS FOR SPEECH RECOGNITION

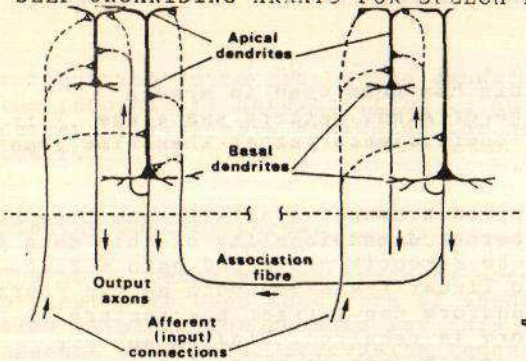

Fig3. Pyramidal neurons in the cortex.

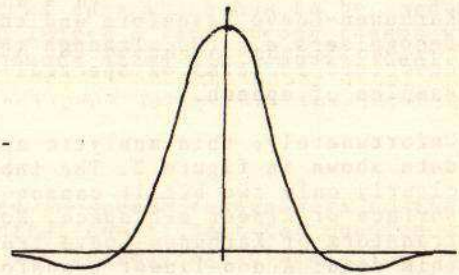

Fig4. Lateral excitation function between neurons in the array.

The operation of this type of neural array when exposed to input pattern vectors via the afferent fibres is largely speculative but if just a few apparently reasonable assumptions are made then the array is found to have powerful feature extraction properties.

The first assumption is that when an input vector is applied to the array each neuron will tend to be excited in proportion to the closeness of its synaptic weight vector to the input pattern vector.i.e Assuming the neurons to be weighted input summing amplifiers, the neuron output will be the scalar product of the input vector and that neuron's synaptic weight vector. The second assumption is that the outputs of each neuron are non-linearly scaled such that the output of just one neuron will dominate in a particular locality. The final assumption is that the synaptic weight vectors of each neuron are moved towards or away from the input pattern vector depending on whether the overall excitation of that neuron is positive or negative after all the effects of lateral excitation and inhibition are taken into account.

With these assumptions a simple computational algorithm for the neural array can be formulated as follows:

1) Set up a two dimensional array of elements(neurons),each with storage for an initially random synaptic weight vector order k.

2) Take an input pattern vector of order k and find the neuron with the closest stored synaptic weight vector.

3) Define an excitory and inhibitory neighbourhood around that neuron in the array and modify the synaptic weight vectors of each neuron in the neighbourhood such that they move towards or away from the input pattern vector depending on whether it lies in an excitory or inhibitory part of the neighbourhood.

SELF ORGANISING ARRAYS FOR SPEECH RECOGNITION

i.e $$Wn = Wn + Kn * (X - Wn) \quad \ldots\ldots\ldots\ldots 1$$

Where Wn is the weight vector of the nth neuron in the array ,X is the input vector and Kn is the factor determined from the "mexican hat" lateral excitation function.

Properties of the Neural Array Model:

The properties of the neural array model are most easily demonstrated by generating an artificial data set of random two dimensional vectors having a uniform probability distribution at any radius from the centre of their pattern space and a Gaussian distribution along a radius. The scatter plot of such points is shown in figure 5. If vectors are drawn at random from this distribution and applied to a one dimensional neural array it is found that the synaptic weight vectors associated with each neuron start to cluster along the ridge of the data's probability distribution. More startling, neurons which are adjacent to each other in the array take on synaptic weight values which are adjacent in the pattern space. In other words ,the array becomes topologically related to the data as is shown in figure 6. and the data is projected through a complex non-linear transform onto the array.

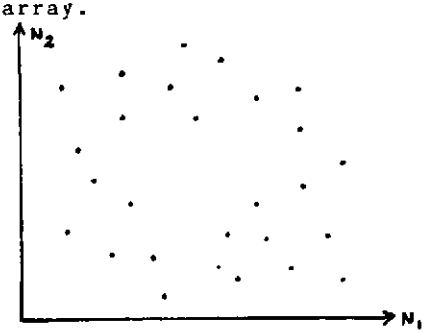

Fig5. Scatter plot of 2-D data.

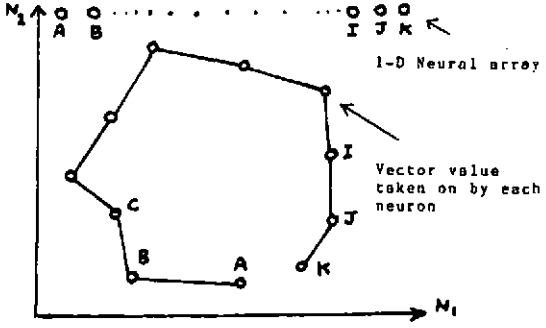

Fig6. 1-D neural array and its associated synaptic weight vector after exposure to the data of Fig5.

In general data which is embedded in a very high dimensional space can be projected onto a neural array of low dimensionality as long as the inherent dimensionality of the data is not greater than the dimensionality of the array.This is feature extraction. If the inherent dimensionality of the data exceeds that of the array then the array will fold itself so as to fill the subspace occupied by the data. Of course, when this happens ,the topological ordering of the map is disturbed. However,this in itself is a useful property,

because it means that the inherent dimensionality of a set of data can be determined by increasing the dimensionality of the array onto which it is projected until the array is seen to be topologically ordered.

Neural Arrays and Speech Recognition:

A two dimensional neural array has been used by Kohonen (1) for the recognition of Finnish phonemes described in a pattern space of thirty DFT coefficients .Such a system could form the basis of a speech recogniser. However it is perhaps more interesting at this stage to use the array as a tool for investigating the properties of speech sounds. To this end a high speed hardware neural array model has been built at BTRL .In the long term it is intended to expose the array directly to sequences of time domain samples of speech to see if speaker independent features can be found which do not depend on spectral analysis. However,in the short term an attempt is being made to set a bench mark by applying spectral coefficients of speech to the array in the following experiments:

Single Speaker Clusters:

A phrase from a single speaker, "Why were you away a year Roy?" will be segmented into blocks of 256 samples and spectrally analyzed to yield sixteen spectral coefficients equally spaced in frequency. Each of the 16 dimensional vectors will be applied many times in random order to a 20*20 neural array such that the total number of "training passes " is 20000. The values of the synaptic weight vectors in the array will then be analyzed to see if the array is topologically ordered and also to measure the proximity of adjacent neuron's synaptic weight vectors over the entire array.This should give a measure of cluster density and cluster separation in the original 16 dimensional pattern space. It is of course expected that the clusters will correspond to particular speech primitives. The cluster density and separation should indicate how reliably a recogniser working on these types of spectral analysis could operate.

Multi Speaker Clustering :

The same experiment will be repeated except that the speech will be taken from several different speakers. The values taken on by the neural array will be analyzed to see if discernible clusters still exist and if so, how their separation has changed.

All the previous tests will be repeated for spectral analysis block lengths ranging from 4ms to 32ms and using an enlarged speech test set containing nasals and fricatives as well as vowel sounds.

SELF ORGANISING ARRAYS FOR SPEECH RECOGNITION

Inherent Dimensionality of Speech:

In this experiment arrays of various dimensionalities will be exposed to the sets of spectral coefficients and the minimum array dimensionality required for a topologically ordered map determined. This test would provide an interesting piece of circumstantial evidence for or against the speculative idea that the ear produces spectral features which are initially projected onto a two dimensional neural array.

Hardware for Neural Array Model:

In order to train the neural array ,it must be exposed to very large numbers of input patterns .For each input the neuron with the nearest synaptic weight vector must be found and then all the neuron vectors in the array updated using equation(1). This is a computationally time consuming task when done in software on a mini computer and so a the neural array has been implemented in hardware form under the control of a micro computer.

The hardware consists of thirty identical rack mounted cards each communicating with the controlling micro computer via a common bus.Each card consists of 32kbytes of memory to store synaptic weight vectors and their corresponding difference vectors (Xn - Wn) along with logic to determine the position of the neuron in the array with the smallest difference vector and logic to update all neuron values in the array according to equation (1). The definition of the lateral exctitation function is software controlled.

The total memory capacity available for storing synaptic weight vectors is 240 kbytes and this can be partitioned under control of the microcomputer between array size and vector order. For example ,a 32*32 array could be set up which could deal with input vectors of order 2048.

Observations from using a Neural Array:

a) Nearest Neuron Metric:

Computationally ,the simplest metric for finding which synaptic weight vector is nearest to the current input vector is "city block". This metric does not actually match the Euclidean space in which we wish to generate the array map,but it has been found that the measure will enable the array to become roughly ordered. At this stage the vector distances are so small that there is very little difference between Euclidean distance and city block distance, and the system will continue to full convergence.

b) Neighbourhood Metric:

SELF ORGANISING ARRAYS FOR SPEECH RECOGNITION

Since it is required that the map be topologically ordered, the metric which is used to determine the neighbourhood of neurons which are updated must be consistent with the spatial distances between neurons in the physical array. Thus if the array is rectangular, a Euclidean metric should be used. If the array is based upon a diamond shape ,then the city block metric should operate.

c)Neighbourhood as a Function of Time:

At the start of training the neighbourhood must be set wide in order that topological ordering can occur. However,if it is not reduced as time progresses it is difficult for the synaptic weight vectors to converge to values which accurately mirror the statistics of the input patterns .A typical result is that all the synaptic weight vectors are pulled towards the average of all the pattern vectors to which the system has been exposed. The result is a shrunken map. The solution is to linearly decrease the neighbourhood size as training progresses.

d) Lateral Excitation Function:

The original software simulations of the system done at BTRL showed that very low levels of inhibition aided rapid ordering of the array and also gave convergence without reducing the neighbourhood size. The necessary ratio between excitation and inhibition values being about 100 to 1 while the excitory neighbourhood size was about one eigth of the pattern space dimension and the inhibitory neighbourhood about one half. However ,it has been found that when using 8 bit integer arithmetic in the hardware ,inhibition leads to instability and has therefore been abandoned.

It has also been found that a computational simplification can be made at the cost of increased ordering time: The expression for updating the neuron values (equation (1)) can be modified so that the synaptic weight vector is moved by an incremental amount away or towards the current input vector.

$$Wn = Wn + (X - Wn)/|X - Wn| \quad \ldots\ldots\ldots 2$$

In the computation this is implemented merely by adding the sign of the difference between the ith element of X and the ith element of Wn to the value of the ith element of Wn.

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

References:

1) T.Kohonen, Clustering Taxonomy, and Topological Maps of Patterns, Proc. 6th International Conf. on Pattern Recognition, IEEE, October

SELF ORGANISING ARRAYS FOR SPEECH RECOGNITION

1982.

2) Hirai, A Template Matching Model for Pattern Recognition: Self Organisation of Templates and Template Matching by a Disinhibitory Neural Network, Biol Cybernetics 38,91-101,1980.

3) T.Kohonen et al, A Thousand Word Recognition System Based on The Learning Subspace Method and Redundant Hash Addressing, Proc. IEEE 5th International Conf. on Pattern Recognition, December 1980.

4) M.J.Carey, The classification of phonemes by a self ordering network, Institute of Acoustics Autumn Conf. ,1984.