Tattersall G.D, Foster S.M, *Generalisation and Single Layer Look Up Perceptrons*, IEE Proc. Part F: Radar and Sig. Proc., December 1990.

Abstract

This paper examines the generalisation properties of various types of neural network such as radial basis function systems and the multi layer perceptron (MLP). It is concluded that their behaviour can be explained in terms of low pass interpolation in which discrete training examples of a function are implicitly convolved with the impulse response of a low pass filter to produce an estimate of the function for previously unseen arguments.

A different form of neural network in the form of a single layer look up perceptron (SLLUP) is described, and this type of perceptron is shown to also generalise by low pass interpolation. However, the SLLUP can learn reliably and rapidly compared to the multi layer perceptron and experiments are described which show that it compares well with the MLP on problems such as speech recognition and text-to-speech synthesis.

1.0 Introduction

In this paper the nature of generalisation by neural networks is related to the established signal processing concepts of filtering, convolution and interpolation and it is argued that neural networks such as the multi layer perceptron [1] and radial basis function systems [2] are examples of low pass interpolating systems which can only generalise on data which exhibits simple clustering. Although limited, this form of generalisation is often very useful when the data being processed by the net represents physical quantities, and in this paper an alternative low pass interpolating perceptron is described which is able to perform this type of generalisation but with fast learning and little computation compared to the MLP and RBF.

The perceptron incorporates n-tuple pattern recognition techniques [3] in an single layer architecture to produce a *single layer look up perceptron* (SLLUP) whose hardware realisation could be almost identical to WISARD [4] but whose *modus operandi* is very different. The SLLUP is based on a non-linear adaptive filter proposed by Johnston [5] which has been used for echo cancellation and is functionally similar to the cerebullar model articulation controller (CMAC) [6] used in learning control systems. It is shown that this type of system can be used in most of the applications which are currently seen as the domain of neural networks such as the multi layer perceptron (MLP). The SLLUP can learn the same types of non-linear mappings as an MLP but with a fraction of the training and computation, and an additional very desirable property of the SLLUP is that it produces a quadratic error surface and so convergence to optimal performance is assured.

It will be shown that the SLLUP is basically an interpolation system which is able to generate an estimate of a continuous mapping function from a sparse set of training examples and, as will be demonstrated, is well suited to dealing with simple non-linear mappings such as parity detection.

The ability of the SLLUP to work on the very complex mapping problems of speech recognition and text to speech synthesis is also examined and compared with the performance obtainable using the multi-layer perceptron. It will be seen that the SLLUP can very nearly equal the performance of the MLP in these problems, suggesting that the MLP also does little more than a straightforward sample interpolation.

2.0 Generalistion by Neural Networks

Any type of neural network can be visualised as a vector transformer which accepts an input vector or pattern X and produces an output vector Y. The functional relationship between Y and X is learned from a relatively sparse set of examples of input-output pairs which are shown to the network during a supervised learning phase. A typical arrangement for performing the learning is shown in Fig.1 in which a training example of the required function, f(X), is used as an output target for the transformer. The error between the actual output of the transformer and the target is used to adapt the internal parameters of the transformer until the error is minimised. In the context of neural networks the internal parameters are the 'synaptic weight' values and the error is used in a gradient descent algorithm such as back propagation.

A desirable property of neural networks is *generalisation* which enables the network to produce a good estimate of an output Y given a previously unseen value of input X. In many cases, this is simply achieved by some form of interpolation.The process is illustrated in Figs. 2a and 2b for a one-dimensional case in which the function underlying a training set of discrete input-output example pairs is $y = g(x)$. Fig. 2a shows that the input-output examples are effectively samples of the function g(x) and although the samples are irregularly positioned across the pattern space, many of the ideas used in digital signal processing such as Nyquist Sampling, aliasing, filtering and function bandwidths are also relevant to the problem of finding this function.

In particular, if the input-output examples are samples of the underlying function, then the complete continuous function should be recoverable by passing them through a suitable low-pass interpolation filter in which they are convolved with the filter's impulse response, as illustrated in Fig. 2b. This is the essence of the radial basis function system and the single layer look up perceptron described in this paper.

Insight into the nature of the generalisation obtained using interpolative filtering is gained by viewing the process in the frequency domain . As an example consider a filter whose impulse response is a multivariate gaussian. The filter's frequency response is low pass and consequently the correct continuous function will only be generated by filtering the sampled function if the spectrum of the unsampled function is also low pass as illustrated in Fig.3. To emphasise this limitation, the generalisation will henceforth be called lowpass interpolation to indicate that it is only suitable if the spectrum of the function underlying the training samples is also low pass.

In contrast to Fig.3 , Fig.4 shows a situation in which the function underlying the samples has a bandpass spectrum. Low pass filtering the spectrum of the sampled function will not yield the correct continuous function because a frequency shifted version of the wanted continuous function is produced.

It may be argued that real world data will only arise from functions having a lowpass spectrum and this is often be true for data which has been generated by physical processes such as speech articulation, whose generating system contains mechanical inertia which tends to cause the spectral content of the function to be dominated by low frequencies. Such functions are appropriately generalised by low pass interpolation like the radial basis function and, as will be shown shortly, sigmoidal multi-layer-perceptrons.

However, functions which describe logical processes which relate a set of input conditions to an output action are often characterised by bandpass spectra. A simple example is the parity function, which contains no energy at zero frequency because the function oscillates in value as each input variable changes state. Low pass interpolation is quite inappropriate for generalising in this kind of situation and, predictably, it will usually be found that radial basis function systems or sigmoidal MLPs fail to work correctly when only trained on a subset of possible input-output examples. Correct generalisation is still possible with these types of function if a *bandpass or multiband* interpolation scheme is used. The Fourier Perceptron [7] is an example of such a bandpass interpolating system but, in this paper, the discussion will be limited to the case of low pass interpolation.

3.0 Nyquist Sampling, Filtering And Generalisation

Three important points are raised by viewing the process of function generalisation as interpolative filtering of a sparse set of samples of the function. These points are equally applicable to lowpass and bandpass interpolation and provide insight into the limitations of any generalising machine.

3.1 *Nyquist Sampling Criteria*

Sufficient training example must be given such that there is a minimum of two samples per cycle of the highest frequency in the mapping function. This suggests that the complexity of a mapping function be specified in terms of its bandwidth, B, and that the maximum interval between training examples should be no greater than 1/2B.

3.2 *Bandwidth of Interpolation Filters*

To obtain a perfect, continous function from the training examples, the interpolation filter should have a rectangular frequency response, of BHz bandwidth. This would require an infinite length sinc(x) inpulse response which is impossible. Practical interpolation filters should have a bandwidth of as near as possible B Hz, but it must be recognised that finite impulse response filters will always produce an error in the estimate of the continuous function.

3.3 *Uniformity of Function Sampling Interval*

It is very unlikely that the training examples supplied to a supervised learning machine will be uniformly distributed across the pattern space. This means that the distances between samples of the required mapping function are non-uniform. The Nyquist Sampling Theorem requires at least two samples per cycle of the function if it is to be recovered without loss of information. However, a uniform sample interval is not specified and so the irregularity of the training points does not necessarily mean that the continuous mapping function cannot be recovered. Unfortunately, a simple interpolation filter is unable to recover a continuous function from a set of irregular samples because the function will be non-uniformly scaled in proportion to the density of the samples. This is not a problem if the interpolation is learned by iteratively adjusting the scale factor of the impulse response of the interpolation filter depending on which region of the pattern space it is operating, and an example of this process is shown by the Radial Basis Function system described in the next section.

4.0 Radial Basis Functions As Low Pass Interpolators

The operation of the radial basis function network is illustrated in Fig.5 in which a set of basis functions are added together so that their sum closely fits discrete training examples of a continuous function which is to be learned by the system. Typically the basis functions are multi variate gaussians whose amplitude, mean value, and variance can be scaled to make the network output fit the given examples of the function.

The low pass filter action of the network is easily understood by initially assuming that the training samples of the function are regularly spaced and that one basis function is positioned over every sample in the function domain. In this situation the output of the network is the convolution of the training samples with the radial basis function. Spectrally this is low pass filtering because the Fourier transform of the multi-variate gaussian is a low pass frequency response.

In general the samples are not positioned regularly and the bandwidth and amplitude of each of the basis functions must be individually adjusted to match the sample rate in its locality. Thus, if the samples are very sparse in one region of the pattern space, the bandwidth of the basis function has to be reduced so that proper interpolation can occur. This corresponds to increasing the variance of the basis function.

Very often it is not possible to place a basis function over every single training sample because of computational load, and in this case, the function is sub sampled by using a relatively small number of basis functions for its synthesis. The positions of the basis functions in the pattern space are adapted iteratively to optimise the accuracy of the synthesised function and the bandwidth of each radial basis function is reduced to reflect the lower effective sample rate of the function.

5.0 MLPs As Lowpass Interpolators

Sigmoidal MLPs also perform as low pass interploators and are therefore only capable of generalising correctly on functions whose spectrum is lowpass. This behaviour is easily understood by looking at the three layer scalar to scalar mapping network with one layer of hidden units shown in Fig.6. The output of the network is the weighted summation of the sigmoid functions produced by each of the hidden units and the slope and offset of each of the functions is set by the weights connecting each of the hidden units to the input units. All the weights are adjusted during learning until the desired output function is synthesised as illustrated in Fig.7.

Assuming the weight values feeding into the hidden units are not allowed to become very high, each hidden unit produces a smoothly changing basis function which contributes to the overall function. Any smooth, non-periodic function, such as the sigmoid, will have a low pass spectrum and consequently the MLP will perform generalisation by low pass interpolation and is incapable of correctly generalising functions whose spectrum is bandpass.

6.0 The Single Layer Look Up Perceptron (SLLUP)

The SLLUP is another example of a low pass interpolating system. However, it has clear advantages in terms of learning speed and computation when compared to the MLP and RBF, and it will be described in detail in this paper. The SLLUP is based upon a non-linear adaptive filter proposed by Johnston [5] and is, in part, inspired by the WISARD [4,15,16] system and the technique of n-tuple sampling proposed by Bledsoe and Browning[3]. The operation of the SLLUP is best understood in terms of the WISARD style architecture shown in Fig.8, although it will become apparent that the way in which the SLLUP uses the architecture, and indeed the practical realisation of the SLLUP, are different from WISARD.

The WISARD architecture consists of a retina in which an input pattern, X, is encoded as an image of black and white pixels formed by bits of the code representing the scalar elements of X. Random connections are made onto the pixels in the image and groups of *n* connections are formed into n-tuples which are used to address a large number of RAMs. The RAMs themselves are grouped into 'neuron' blocks called discriminators and the outputs of all the RAMs in the i^{th} block are added to form, y_i, the value of the i^{th} element of the output vector Y.

WISARD is a *pattern classifier* whose function is to produce a high score at the output of a single class discriminator when a pattern belonging to that class is applied to the retina. This is achieved by associating a single discriminator with each of the pattern classes to be recognised and then applying example patterns of each class. A value of '1' is stored in each addressed location of the RAMs in the discriminator and it can be shown [8] that after training with many examples, the output value produced by the discriminator for class C_i is approximately proportional to class conditional probability, $P(C_i|X)$, of the input pattern X. The conditional probabilities are then used to perform Bayesian Classification of input patterns of unknown class. Various modifications can be made to impove the estimate of the class conditional probabilities such as making the value of a RAM location equal to the number of times it has been addressed during training, rather than '1' [9]. However, the fundamental function is to provide a set of probability estimates which can be used for Bayesian Classification.

In contrast to the WISARD *pattern classifier*, most supervised neural nets are designed to learn *arbitrary mapping functions* between input and output vectors. The SLLUP uses n-tuple sampling techniques to implement arbitrary mapping functions in the same way as a conventional neural net but with the computational simplicity and learning reliability of the WISARD system.

The SLLUP can be implemented using the same hardware as the WISARD with the exceptions that each RAM location must be several bits wide so that a wide range of values can be represented, no class is associated with each of the discriminators, and, most importantly, a different training algorithm is used. In reality, it is not practical to implement the SLLUP using a WISARD style architecture because of its inefficient partitioning of memory, and it is usually better to use a single, large multiplexed memory in conjunction with hash addressing rather than small individual RAMs with n-tuple connections.

In the SLLUP mode of operation, the output of each 'discriminator' forms an element of the output vector of the SLLUP and the system is trained by applying a vector X to its input which causes a specific set of n-tuple addresses to be generated that access corresponding contents in each of the RAMs. The summation of the outputs of each group of RAMs produces the elements of the output vector Y. This vector is compared with the desired output T and the resultant error vector, E, is used to modify the values of the currently addressed RAM locations so that next time the same input vector is applied, the output Y is nearer to the desired output T.

Repeated application of different training vectors allows the system to learn the required input - output mapping Y= f(X). It is important to notice that with appropriate choice of n-tuple order and number of RAMs in each neuron block, the system can estimate the best function f(X) to fit a rather sparse training set. That is, it is not necessary to expose the machine to all possible input-output vector pairs because it is able to interpolate the required function between training points. This property will be analysed in more detail later in the paper.

6.1 Analysis of the SLLUP Learning Procedure

The adaptation of the RAM contents to develop the required mapping function is performed using the steepest descent algorithm [10,17] to minimise the mean square error between the actual outputs and target outputs of the system. It is therefore necessary to obtain a value for the derivative of the average error power with respect to the values stored in each location of each RAM so that the required change in each RAM location value can be determined.

Each RAM in the system is addressed by an n-tuple whose value depends on the vector X contained in the input image. Thus, the output of each RAM in the system depends in some complex way on X, such that the output of the j th RAM in the i th 'neuron' block can be expressed as $C_{ij}(X)$. The output of the ith neuron is then given by (1) where Q is the number of RAMs per neuron.

$$y_i = \sum_{j=1}^{Q} C_{ij}(\bar{x}) \qquad (1)$$

If the target output vector when X is input is $T=[t_1...t_N]$, then (1) can be used to express the mean square output error of the system as:

$$\overline{E^2} = \frac{1}{N} \sum_{i=1}^{N} \left\{ \sum_{j=1}^{Q} C_{ij}(\bar{x}) - t_i \right\}^2 \qquad (2)$$

Equation 2 shows that the highest power term involving $C_{ij}(X)$ in the expression for mean square error is two. This indicates a single minimum, quadratic surface, and so convergence to a global optimum is guaranteed using a gradient descent algorithm. The gradient term required is simply calculated from (2) as:

$$\frac{\partial \overline{E^2}}{\partial C_{ij}^{n}(X)} = 2 \, . \, (y_i - t_i) = 2 \, e_i \qquad (3)$$

where e_i is the difference between the output of the ith neuron and its target value. So, the algorithm for modifying the RAM contents becomes:

$$C_{ij}^{n+1}(X) = C_{ij}^{n}(X) + k \, . \, (y_i - t_i) \qquad (4)$$

6.2 The SLLUP Interpolation Kernel:

It can be shown that the SLLUP performs generalisation by low pass filtering the training samples in the same way as a radial basis function network. The basis functions or interpolation kernels are generated implicitly by the architecture of the SLLUP and their form depends strongly on the way in which the input vector X is encoded, but it will be seen that typically they have a double sided decaying exponential shape. The frequency response of a low pass filter having this kind of impulse response is far from an ideal 'brick wall' interpolation filter, but the poor cut-off response is not a problem if the mapping function is sampled at a sufficiently high rate that insignificant amounts of aliasing occur. This means using sufficient numbers of examples in training the network.

The simplest coding, from an analytical point of view, is bar chart coding in which the value of each element of X is represented by the number of 'ON' bits in a bar, and the operation of the SLLUP using this code will be analysed before going on to consider other input vector encoding techniques using binary and Gray codes as well as other codes with controlled redundancy.

6.3 Bar Chart Coding:

A bar chart encoding of a two-dimensional pattern [x_1 x_2] is shown in Fig.9. In this example each vector element has been quantised to one of 8 values and the vector value shown is [3,4].

Consider a single 1-tuple connection made onto the q^{th} pixel of the x_1 bar. If $x_1 > q$ the 1-tuple value will be one, whereas if $x_1 < q$ then its value is zero. In pattern space the value of '1' or '0' on this connection changes as an imaginary threshold line at $x_1 = q$ is crossed. This idea can be extended to the *n* connections forming an n-tuple. The value on each connection tells on which side of the associated threshold line the current input pattern lies. The intersections of the lines delineate particular regions of pattern space which are associated with particular n-tuple values as shown in Fig.9 for the case of a 4-tuple. If many n-tuples are connected onto the input retina, the combination of n-tuple values delineate smaller and smaller regions of pattern space, with each region becoming a regular square as the number of n-tuples becomes very high.

The idea of a pattern space dissected by threshold lines can be used to predict the form of the basis function produced by a SLLUP. Assume that the RAMs have initially zero contents: a single training pattern, X_1, is applied to the system and the contents of the locations in the RAMs addressed by each n-tuple's value are iteratively modified until they produce the specified target output vector, T_1. If another pattern is now applied to the system, starting at

X_1 and gradually moving away from X_1, an increasing number of the threshold lines will be crossed.

Each time a threshold line is crossed, the address formed by an n-tuple of connections on the retina changes value, and one of the RAMS which was contributing to the output value, T_1, switches to another location and produces zero output. Thus the output from the SLLUP slowly drops towards zero as the input pattern is progressively displaced from the training pattern and the shape of the fall in output value defines the form of the implicit basis function.

The shape of the basis function can be derived analytically if it is assumed that large numbers of n-tuples are connected to the input pattern retina. Let there be Q RAMs in each 'neuron' and D dimensions of length W in the pattern space. The output from each 'neuron' which is caused by an input pattern displaced by distance x from the position of the training pattern, is given by s (x).

$$s(x) = t - r(x) \cdot C_{av} \qquad (5)$$

Where t is the output from the neuron when the input is the training pattern, r(x) is the number of RAMs whose addresses change when the input pattern is moved x units away in pattern space, and C_{av} is the average contribution of each RAM to the output value at the training point. But,

$$r(x) = \int_0^x p(x) \, . \, dx \qquad (6)$$

where p(x) is the probability density of crossing a threshold line at position x in the pattern space. Assuming large numbers of randomly connected n-tuples, this density is just the number of active threshold lines, q (x), cutting any axis of the pattern space divided by the pattern space width over which the lines are distributed.

$$\text{ie} \qquad p(x) = \frac{q(x)}{W} \qquad (7)$$

To solve for p (x), r (x) and s (x) we investigate the variation of q (x) as x increases by δx.

$$q(x + \delta x) = q(x) - \frac{n}{D} \cdot \frac{q(x)}{W} \cdot \delta x \qquad (8)$$

The n/D term in the equation arises because crossing a single threshold line renders the other n-l lines within the n-tuple ineffective. i.e subsequent crossing any one of these other lines cannot affect the output from the RAM any more because it is already switched to produce zero output. The n lines which are effectively removed from play are spread over D dimensions and so the average number of lines along a particular axis which are deactivated by crossing just one threshold line is n/D. Rearranging (8) gives:

$$\frac{\delta q(x)}{\delta x} = -\frac{n}{WD} \cdot q(x) \qquad (9)$$

This first order differential equation has a solution of:

$$q(x) = \frac{Q.\,n}{D} \cdot e^{-\frac{n.\,x}{W\,D}} \qquad (10)$$

and can be used in conjunction with equations (5) and (7) to provide the required expression for the interpolation kernel.

$$s(x) = t \,.\, (1 - e^{\frac{-\,n.\,|x|}{W.D}}) \qquad (11)$$

The approximate validity of this expression has been confirmed by a computer simulation in which a SLLUP has been trained on a single input pattern and the output caused by other patterns covering a 2-D input space recorded. The results are shown in Fig.10a for a system using 24 quantization levels for each dimension with 32 RAMS addressed by 4 tuples of connections onto the input retina.

6.4 Simple Binary Coding

A pattern may also be presented to a SLLUP as image pixels whose values correspond to simple binary encoding of the pattern elements. Random n-tuples of connections are made onto the retina in the normal way. However, the effective basis function created when the input vector is binary coded is poorly defined and irregular for the following reasons.

The output from the SLLUP is determined by the number of RAMs in each neuron which are addressed with the n-tuple value on which they were trained. The probability of an n-tuple changing value is proportional to the Hamming distance, $D_{h,}$, between the current input pattern and the training input pattern. Consequently the output of the SLLUP is related to the Hamming distance between the training pattern and current input pattern.The relationship between Hamming distance and signal space distance for Bar Chart code is linear whereas it is non-linear for Binary Code, as shown in Fig.11.

In the latter case, the non-linearity causes the SLLUP output to follow an irregular function as the the input pattern is moved away from the training point in pattern space as shown by the computer simulation example of Fig.10b. In this experiment, simple 5 bit binary coding with

32 RAMs addressed by 4-tuples of retina connections was used. The irregularity of the basis function is evident, but it should also be noted that it is much narrower than in the bar chart system. This is a possible advantage since it means that complex mapping functions can be synthesised if sufficient training data is available.

6.5 Gray Coding

Some improvement in the regularity of the basis function can be obtained by using a Gray code instead of simple binary code. Although the Gray code Hamming distance versus signal space distance function is not monotonic, it is better than the binary code because the Hamming distance only changes by 1 for every 1 unit change in signal. Again, this type of encoding has been tested by simulation and the kernel function obtained using a Gray code is shown in Fig.10c.

6.6 Other Codes

Further improvements in kernel regularity can be obtained by using introducing redundancy into the code. There are a large number of ways of doing this, but a particularly effective technique is *Redundant Gray Coding* [11] in which each signal value is represented by a concatenation of several shifted versions of an R-bit Gray code for the value. Connecting n-tuples onto all the bits in the new code causes the bumps in the relationship between D_h and D_s to be averaged, resulting in a more regular kernel as shown by the simulation results in Fig.10d.

7.0 Speech Recognition and Synthesis Problems Using the SLLUP

Earlier work has shown [12] that the SLLUP is able to perform simple non linear mappings such as the fuzzy EXOR problem. This is achieved using only small amounts of training and with little computation compared to the MLP. In this paper we examine the SLLUP performance on two speech mapping problems of very great complexity on which MLPs and some other neural nets have already been tested.

The first mapping problem is speaker independent recognition of utterances of the letters of the alphabet. A defined cepstral coefficient representation of many utterances of the letters of the alphabet from one large set of talkers must be classified by the SLLUP after it has been trained on examples from a separate set of talkers. The database used in this experiment was compiled by British Telecom Research Labs and is known as the CONNEX S1 data [13]. The SLLUP is trained on approximately 4000 utterances from a balanced mix of 52 talkers and then tested on approximately 4000 utterances from another 52 talkers. The utterance length is normalised by linear time warping and is presented to the SLLUP as a set of 15 frames of 8 Mel Cepstral coefficients.

The second complex problem to which the SLLUP has been applied is text to speech synthesis. In this case orthographic text has to be mapped to a sequence of phoneme codes which are then used to drive a hardware synthesiser. The experiment uses the same database as NETSPEAK [14] and is identical in all respects except that the MLP is replaced by a SLLUP. The SLLUP is presented with a character taken from English orthographic text and has to produce an appropriate phoneme code as output. Clearly the pronunciation of a particular character often depends on the word in which it is embedded and so 3 characters on either side of the target character are simultaneously presented to the SLLUP. Thus, the complete input pattern consists of a context window of 7 characters, each encoded using 11 bits. The output phoneme is represented using a 19 bit code to represent each of 55 phonemes.

It is interesting to consider the types of mapping which the SLLUP has to develop to deal with each of these two problems. In the speech recognition case, input patterns belonging to the same utterance class are likely to cluster together in their N-space and the SLLUP has to map the region of N-space enclosing the cluster to a single specified point in the output space. There may be several clusters belonging to one class but overall the mapping between input and output is smooth, without abrupt transitions. This proposition is supported by the fact that moderately good speech recognition systems can be made using nearest neighbour classification of the input pattern. The task of the SLLUP in this case is to *interpolate* so that previously unseen input patterns which lie between training examples of the same class are mapped to the same output code.

The text to speech mapping is very different. The distances between the codes representing different characters does not have a simple relationship to the distances between the codes for the corresponding output phoneme codes. In other words, the patterns are really symbolic and just happen to be represented in a Euclidean space for processing by the neural net. Thus, the task of the SLLUP is to detect any *logical* structure in the data and failing this, to act as a look up table.

7.1 Experimental Results on The Speech Recognition Problem

Tables 1 to 3 summarise the results obtained using the SLLUP as a speech recogniser. Table 1 shows that a SLLUP using natural binary coding in the retina is able to learn the training set very well, but performs poorly on the test set. Moreover, the performance tends to improve as the order of n-tuple decreases. Taken together, these two factors suggest that the SLLUP is unable to interpolate sufficiently between the training examples because the effective width of the interpolation kernel is too small. Reduction of the n-tuple order causes the kernel to become wider, with a consequent improvement in performance on the test set. Increasing the n-tuple order makes the system behave more like a look up table, giving better recognition of the training set but an inability to generalise.

8 Bit Natural Binary Coding, Retina size 960 bits			
N-tuple Order	Training set	Test Set	Rams per O/P
2	95%	65%	480
3	98%	62%	320
4	95%	52%	240

Table 1. Speech Recognition Results Using SLLUP with Binary Coded Input.

The kernel width produced using natural binary code is very narrow and a possible solution to the poor test set performance is to use a different code in the retina, as demonstrated by the results of Table 2. These results were obtained by quantising each of the cepstral coefficients to 16 levels and representing them by a bar chart code. As expected, the performance improves on the test set and gets worse on the training set. This confirms our hypothesis that the natural binary code leads to an over-specific system. The results in Table 2 show an improvement in performance as the n-tuple order increases, indicating that in this system,the kernel is actually too wide so that with low values of n, over-generalisation is taking place. This is supported by the fact that the system has been unable to accurately recognise the training set.

16 Level Thermometer code, Retina size = 1920 bits			
Tuple Order	Training set	Test Set	Rams per O/P
2	76%	71%	960
3	81%	75%	640
4	83%	77%	480
6	85%	78%	320

Table 2. Speech Recognition Results Using SLLUP With Bar Chart Coded Input.

The recognition accuracy obtained using this system with n=6 is comparable with results obtained using a 2 layer, 25 hidden unit MLP on the same data which produced a test set accuracy of 81%. The best results obtained using the SLLUP on this data are compared to those produced by an MLP in Table 3.

Device	Training Set	Test Set
MLP, 75 Hidden Units (see reference [18])	97.4%	88.3%
SLLUP, Tuple Order 8, 32 level Barchart Code, 480 RAMS per O/P.	97.3%	82.8%

Table 3. Comparison Between Best Results of SLLUP and MLP.

7.2 Experimental Results On The Text to Speech Synthesis Problem.

In these experiments each of the seven characters in the input window are represented by 11 bit codes containing approximately equal numbers of '1's and '0's. This is important when using a SLLUP because an imbalance in the number of'1's and '0's will cause most n-tuple values to always consist of n '1's or n '0's respectively and this renders the n-tuple values insensitive to changes in the input vector X.

The results obtained using a SLLUP in the text to speech application are presented in Tables 4 and 5. Table 4 shows the performance of the SLLUP when the 11 bit codes are placed at approximately equidistant positions in 11-space. This coding is therefore completely unstructured. As expected, the performance is very poor because the input patterns are really symbolic and the interpolation between arbitrary codes effected by the SLLUP is inappropriate. Using a high tuple order of 8 gives improved performance on the training set because the SLLUP starts to operate as a look-up table. However, the test set performance remains poor.

Coding Codes for each character are approximately equidistant, 11 bits long and consisting of 5 '1's and 6 '0's.			Retina Size = 77 bits
Tuple Order	Training Set	Test set	Rams per O/P
4	34.5%	33.4%	20
8	60.8%	55.2%	10
Training: 4 blocks of 10,000 characters			
Testing: 1 block of 10,000 characters			

Table 4. Accuracy of SLLUP as Text to Speech Mapper Using Unstructured Input Coding.

Table 5 shows the performance of the SLLUP working on a modified set of input codes which are chosen so that their mutual distances approximately reflect the perceptual distances between the phonemes which map most frequently to each letter. Using these structured codes, distances in the 11-space have some meaning and so interpolation becomes a more appropriate means of generating an output on unseen input data. Predictably the results in Table 5 are much better, with high accuracies obtained both on training and test data if sufficiently large n-tuples are used. A furthur improvement can be obtained if the frequency of commonly occuring words is relected in the content of the training and test sets. This is because the very common words in English often have irregular pronunciation rules which are hard for the SLLUP to learn unless seen very frequently. McCulloch reports [14] that a 2-layer, 77 hidden unit MLP can give an 86% letter to phoneme mapping accuracy which is slightly better than the SLLUP result. However, the SLLUP converges relatively quickly and shows a trend of improving performance as n-tuple order increases.

Coding Each code is 11 bits consisting of 5 '1's and 6 '0's. Distance between each code reflects the letter group.			Retina Size = 77 bits
Tuple Order	Training Set	Test set	Rams per O/P
4	52.2%	52.2%	20
8	72.7%	71.3%	10
10	78.4%	75.0%	8
10 **	83.9%	79.9%	8
Training: 8 blocks of 10,000 characters			
Testing: 5 blocks of 10,000 characters			
** frequency weighted training and test data			

Table 5. Accuracy of SLLUP as Text to Speech Mapper Using Structured Input Coding.

8.0 Conclusions

It has been argued that the purpose of any supervised learning network is perform generalisation by synthesising a continuous non-linear mapping function from a sparse set of training examples of the function. The continuous function can be generated by *interpolation* between the discrete examples of the function using a low pass filter and radial basis function networks and MLPs are examples of systems which utilise this principle.

Important implications of this argument are that the number of training examples must be sufficient such that the function is sampled at least at the Nyquist rate and that the generalisation which is produced by the filtering is only correct if the spectrum of the function underlying the training data is dominated by low frequencies.

The single layer look up perceptron is another example of a low pass interpolating network which synthesises the required mapping function by effectively convolving the discrete training function samples with a kernel function which is analogous to the impulse response of a low pass interpolation filter.

The SLLUP uses comparable amounts of memory to the MLP for all but the most trivial functions and in general will learn the required mapping function much faster than an MLP because it is a single layer machine in which error gradients used for its adaption can be calculated directly from the output error. Moreover, because it is a single layer machine, the error surface for the SLLUP is quadratic and it therefore always converges to a minimum error.

It has been shown that the SLLUP is able to operate as a speaker independent recogniser with almost as high accuracy as an MLP which suggests both that speech recognition can be effectively performed by interpolation and, perhaps more important, that the MLP also appears to be doing little more than interpolation. This is supported by the use of a SLLUP for text to speech synthesis which again gave a performance only slightly inferior to an MLP.

Although multi dimensional interpolation is a non trivial task, our arguments suggest that many problems to which neural nets such as the MLP are currently applied with enthusiasm might be more efficiently solved by using explicit, classical interpolation techniques. More importantly, there are many logical problems which embody functions which have a bandpass spatial spectrum and which cannot be correctly generalised by low pass interpolation, MLPs, RBFs or SLLUPs. New types of neural network are needed to deal with these classes of problem.

9.0 Acknowledgement.

Acknowledgement is made to the director of British Telecom Research Labs for supporting this work and providing the CONNEX S1 and NETSPEAK databases.

10.0 References

[1] Rumelhart D.E., Hinton G.E. and Williams R.J. *Learning Internal Representations by Error Propogation.* ICS Report 8506, University of California, Sept 1985.

[2] Lowe D. *Adaptive radial basis function non linearities and the problem of generalisation.* Proc. 1st Int. Conf. on ANNs, pp 171-175, London 1989.

[3] W.W.Bledsoe and I. Browning. 1959. *Pattern Recognition and Reading by Machine.* Proc. Eastern Joint Comp.Conf. Boston, Mass.

[4] Aleksander I, Thomas W.V. and Bowden P.A. *WISARD- A Radical Step Forward in Image Recognition.* Sensor Revue July 1984.

[5] Johnston R.D. *Adaptive Recognising Device* . G.B. Patent No 8427165 (1984), U.S. Patent No 4782459 (1988).

[6] J.S Albus, *A new approach to manipulator control:The cerebullar model of articulation controller (CMAC).* Trans. ASME, pp220-227, J.Dynamics Systems Measurement and Control. Sept 1975.

[7] Lapedes A, Farber R, *Nonlinear signal processing using neural networks:prediction and system modelling*. Los Alamos National Lab. Report No. LA-UR-87-2662.Los Alamos FMLP

[8] Tattersall G.D.and Johnston R.D. *Speech Recognisers Based on N-tuple Sampling.* Proc.Institute of Acoustics.pp405-413 Spring Conf.1984.

[9] Sixsmith M.J and Tattersall G.D, *Speech recognition using n-tuple techniques.* British Telecom Technology Journal. Vol.8 No.2, pp50-60, April 1990.

[10] Cauchy A. C.r. Hebd. Seanc. Acad. Sci. Paris, 25,536 (1874).

[11] Tattersall G.D., Johnston R.D. and Foster S.M. *Single layer Look Up Perceptrons* . Patent application No 8906558.5 (March 1989).

[12] Tattersall G.D., Johnston R.D. and Foster S.M.*Single Layer Look Up Perceptrons.* Speech and Language Processing pp307 - 333, edited by C. Wheddon and R. Linggard. Chapman and Hall 1990.

[13] R. Linggard and P. Woodland. *CONNEX S1 Database.* British Telecom Research Labs Ipswich IP5 7RE.

[14] N. McCulloch, W.A. Ainsworth, R.Linggard. *Multi-Layer Perceptrons applied to Speech Technology.* British Telecom Technology Journal. Vol.6 No.2, pp131-139, April 1988.

[15] Aleksander I, Stonham T.J. *Guide to pattern recognition Using Random Access Memories.* Computers and Digital Techniques, Feb 1979 Vol 2 No 1 pp29 - 40.

[16] Aleksander I and Morton H.B. *An Overview of Weightless Neural Nets*. Invited Paper IJCNN, Washington 1990.

[17]*Optimisation* . Edited by R.Fletcher pp307 - 325. Academic Press 1969.

[18]Woodland P.C., *Isolated Word Speech Recognition Based on Connectionist Techniques.* BTTJ Vol 8 No 2, April 1990 pp61 - 66.

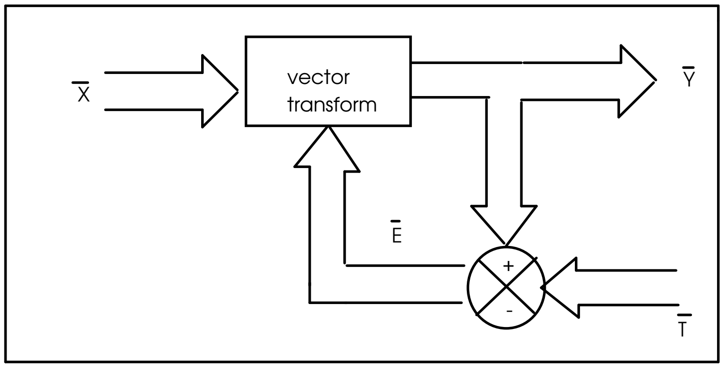


Fig.1. Supervised Learning System

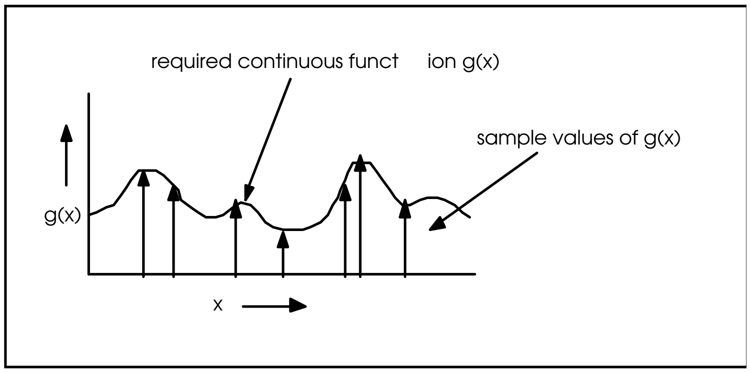


Fig. 2a Training Examples as Samples of The Underlying Function

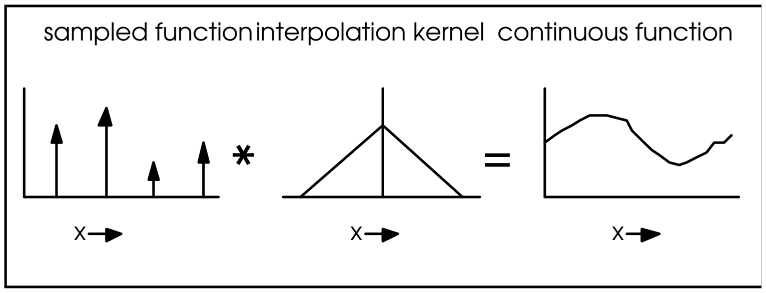


Fig.2b Recovering a Continuous function from its samples by filtering.

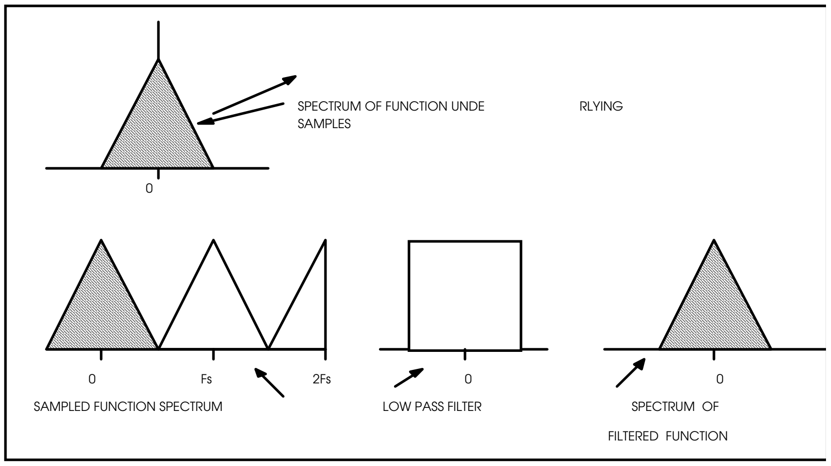

SPECTRUM OF FUNCTION UNDE RLYING SAMPLES
0
0
Fs
2Fs
SAMPLED FUNCTION SPECTRUM
0
LOW PASS FILTER
0
SPECTRUM OF FILTERED FUNCTION

Fig.3 Spectral view of generalisation by low pass interpolation.

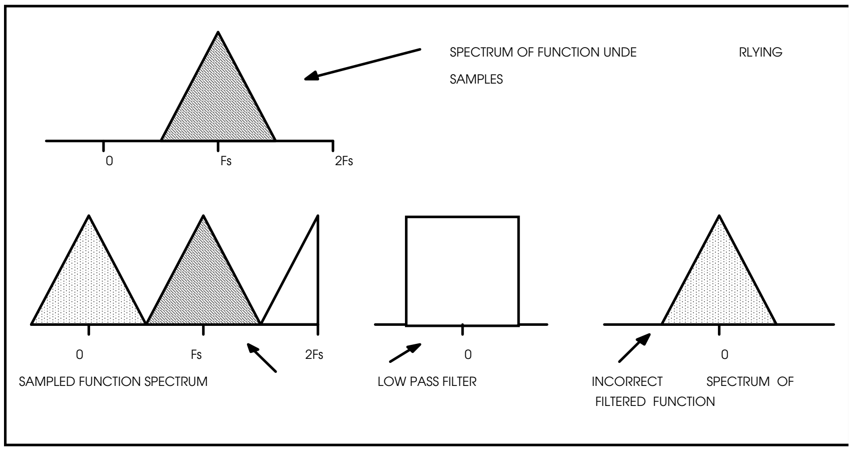

SPECTRUM OF FUNCTION UNDE RLYING SAMPLES
0
Fs
2Fs
0
Fs
2Fs
SAMPLED FUNCTION SPECTRUM
0
LOW PASS FILTER
0
INCORRECT SPECTRUM OF FILTERED FUNCTION

Fig.4 Spectral view of incorrect generalisation of a function with bandpass spectrum.

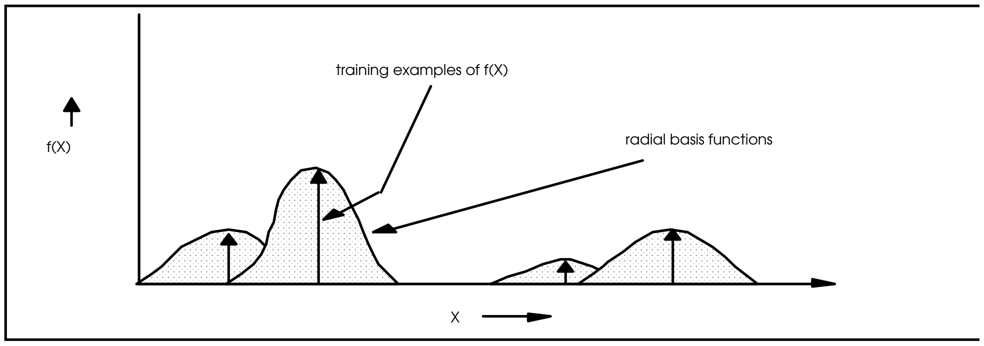
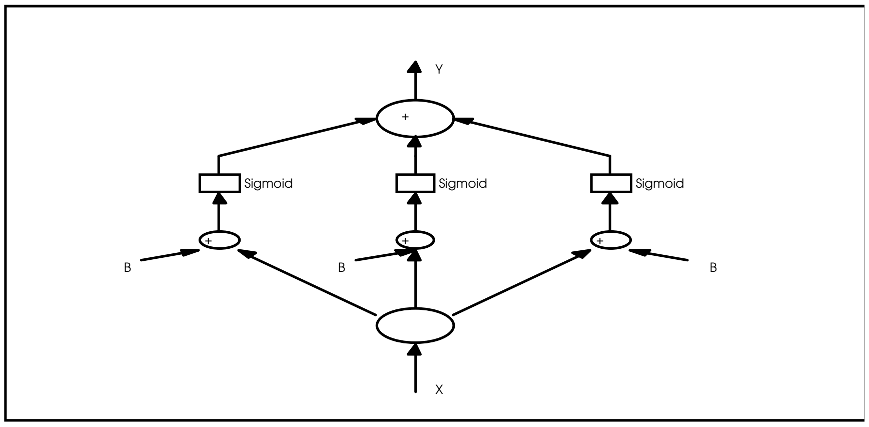


Fig.5 Synthesis of function using radial basis functions.



-

Fig.6 MLP as a synthesiser of a mapping function.

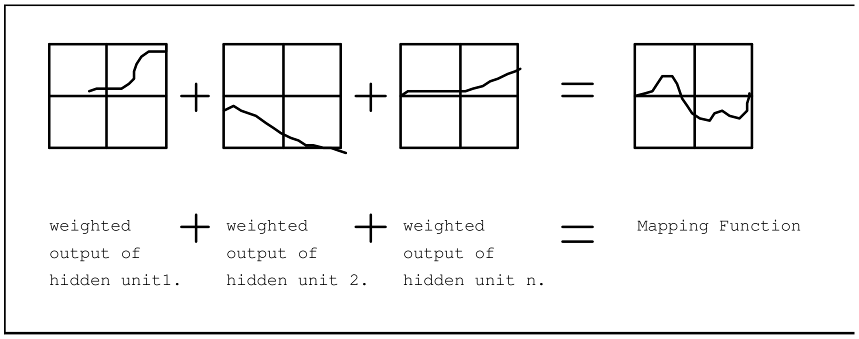

weighted output of hidden unit1.
+
weighted output of hidden unit 2.
+
weighted output of hidden unit n.
=
Mapping Function

Fig.7 Synthesis of mapping function from weighted sum of hidden unit outputs.

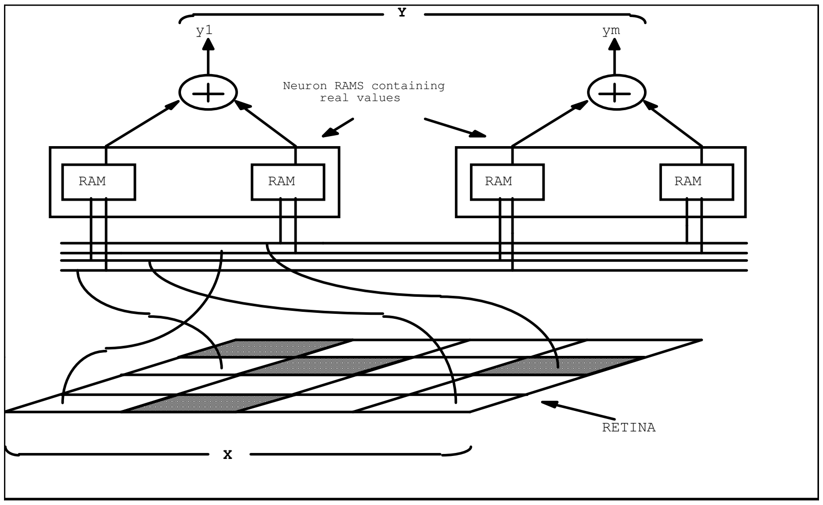


Fig.8 WISDARD Architecture used as a SLLUP.

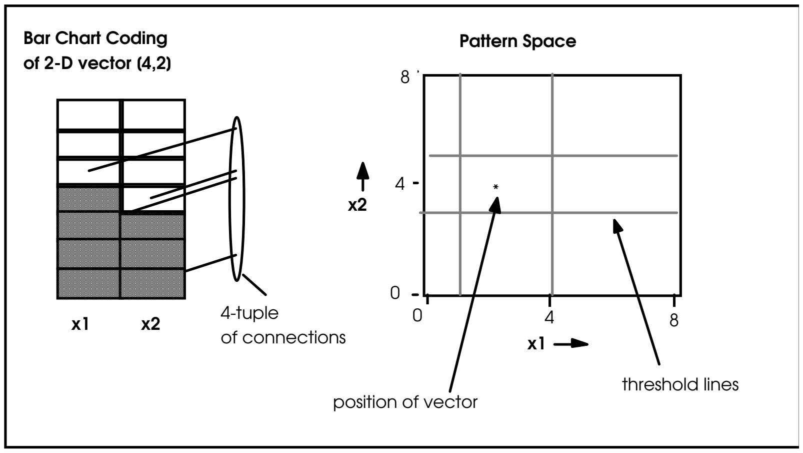


Fig.9 Bar Chart Coding of input vector.

Figs.10a- 10d. Interpolation Kernels of a function Y in a 2- Dimensional pattern space (x_1,x_2), Using a Variety of Input Codings.

KEY: Y = 'high' Y= 'low'

Fig.10a. Bar chart coding.

Fig.10b.Binary coding.

Fig.10c.Gray coding.

Fig.10d.Redundant Gray coding.

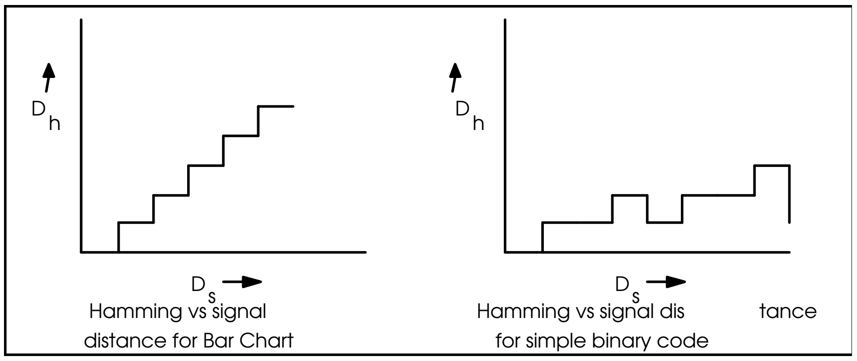


Fig.11 Hamming distance vs signal space distance for bar and binary codings.

List of Captions