IEEE Pacific Rim Conference on Communications, Computers and Signal Processing
June 1st - 2nd, 1989

MULTI-SCALE DYNAMIC NEURAL NET ARCHITECTURES

Les Atlas
Robert Marks II
Mark Donnell
James Taylor

Interactive Systems Design Laboratory
Department of Electrical Engineering, FT-10
University of Washington
Seattle, WA 98195

ABSTRACT

The design of specialized trainable neural network architectures for temporal problems are described. We consider multi-layer extensions of our previous dynamic neural net architectures. Some of the key attributes of these architectures are smoothing and decimation between layers. An analysis of parameters (weights) to estimate suggests a massive reduction in training data needed for a multi-scale topologies for networks with large temporal input windows. The standard back-propagation training rules are modified to allow for smoothing between layers and preliminary simulations results for these new rules are encouraging. For example, a binary problem with an input of size 32 converged in 3 iteration with smoothing and never converged when there was no smoothing. We feel that these new network architectures may be appropriate for speech and sonar pattern classification problems where long temporal context is important.

Introduction

Many artificial neural networks (NN's) are examples of allowing generality while sacrificing potential performance. An important effect of this generality is poor scalability and high implementation complexity. For example, a 2-input exclusive-OR problem is quite easily trainable in a standard back-propagation net [1] while an N input parity problem for this standard net would require $O(N^2)$ interconnections and would, most likely, be impossible to train for large N (e.g. [2,3]). This lack of scalability seems to indicate that for large networks, such as those required for non-trivial practical applications, trainability may not be possible. In this paper, we will use both biological and theoretical arguments to propose new neural network architectures which, at the expense of absolute generality, will scale better and be much easier to implement. We will also argue that these new architectures reduce generality in a way which is natural and appropriate for applications in temporal pattern recognition.

The architectures which we propose are inspired by presumed processing in the mammalian auditory brainstem. Even though the specifics of signal processing in this structure are poorly understood, the neuroanatomy has been fairly well described. These processing pathways, which begin in the cochlea and end in the auditory cortex, involve a relatively small number of neurons. Nevertheless, much processing, especially feature extraction and coding, is presumed to occur in this pathway [4]. It is especially salient that the shortest pathway from the cochlea to the auditory cortex can involve as few a four synapses.

Some of the applications that are considered for NN's have included time series recognition problems such as speech or SONAR and two dimensional image recognition problems. In all of these cases the complexities of implementation and unrealistic training time have thwarted attempts to apply NN's to significant problems. We will be demonstrating, through our examples, how a multi-scale approach substantially ameliorates the scalability and trainability problems of NN's.

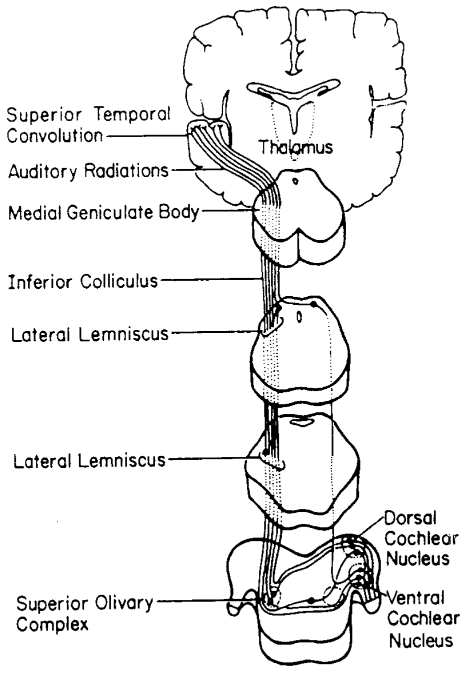


Figure 1. Simplified ascending auditory pathways.

In order to understand how the human system processes sounds it is helpful to look at the structure of some of our auditory pathways. Figure 1 shows a schematized picture of the auditory brainstem. The output of the cochlea, which consists of about 30,000 neurons, enters the ventral coclear nuclei at the right and left bottom corners of the pathway. The pathway is organized into four separate layers or nuclei, where there are bundles of heavily interconnected neurons within each nuclei and relatively few interconnecting neurons between them. We find it very notable that so few layers (with few synapses between each layer) can effectively process the varied sound stimuli we receive. In particular, the relatively sparse interconnections between layers suggests that the fully interconnected models in use for many NN architectures are probably too general. It is also significant that the cochlea's frequency encoding is maintained in a rough fashion throughout the auditory brainstem. For example, degeneration studies have indicated that coclea's frequency map is duplicated in at least two areas [5] of the cochlear nucleus and numerous physiological studies have shown orderly frequency mappings in most areas of the auditory pathways (e.g. see [4]). The sparse interconnections between layers and the maintenance of the frequency map suggest an anisotropic organization of neural connections and a concomitant difference in learning rules. The next section will detail these ideas.

Background—The Dynamic Neuron

We have had on-going research in the theoretical adaptation of the standard sum-of-products neuron to a processing unit which incorporates, in a natural fashion, sampled time. Our first studies in this area have been published previously [6]. The basic idea in this study is shown in figure 2.

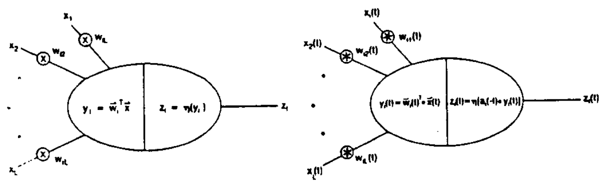


Figure 2. Static and dynamic neuron.

The usual neural element has a function

$$z_i = \eta(\vec{w}_i^{\,T}\, \vec{x})$$

where the basic operations are standard scalar multiplication and addition. Our dynamic modification generalized these two operations to be correlation and scalar sum. Since these operations still constitute a valid mathematical field over time sequences, all linear operations which held for the standard sum-of-products neuron will still hold for the dynamic neuron. The dynamic neuron has a function

$$z_i(t) = \eta[a_i(-t) * \vec{w}_i(t)^{T}\, \vec{x}(t)]$$

where $a_i(-t)$ is an arbitrary linear smoothing function and "$*$" represents correlation. (The reversal of the time index for a_i signifies a standard convolution of the smoothing filter.) The function of η for this dynamic neuron can be a nonlinearity, such as a sigmoid function, and the shape of the $w_{ij}(t)$ ("impulse responses") can also be arbitrary and learned.

Our first application of a group of these neurons was to speech-like transient patterns. We found that we were able to recognize and pin-point (in time) occurences of phoneme-like patterns. However, these experiments were intended to confirm the theory and we would expect that this single layer structure would not be sophisticated enough to form good approximations to difficult class separations. Nevertheless, the demonstrated aspects of this single layer architecture include trainability, time-shift invariance, event-spotting (the output identifies the phoneme and its position in time), and separate training algorithms for the frequency and the time dimension. It is also notable that a multi-layer version of this network, with rectangular impulse responses and back-propagation learning, is identical to a more recently proposed time-delay neural network [7].

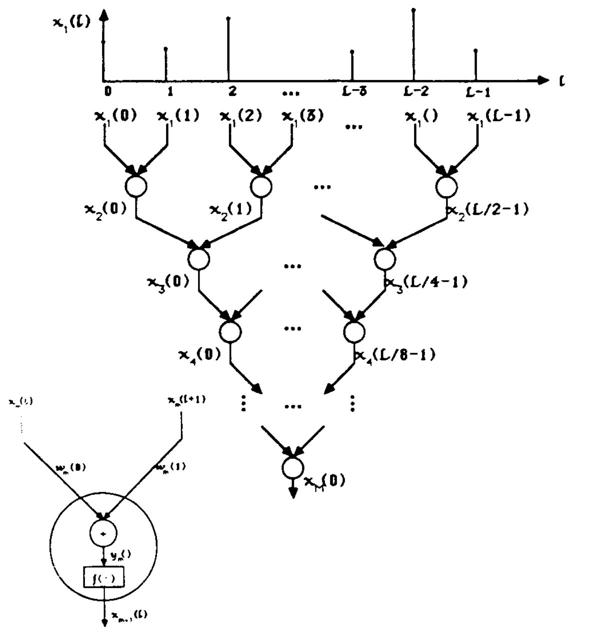
Multi-Scale Dynamic Neural Net

In order to carefully extend the single-layer dynamic neural net to handle difficult problems, it is necessary to allow for flexible classification regions. Another very important extension is to make use of large amounts of temporal context in the recognition of a particular event in time. The key problem here is to make use of a long time window without either sacrificing fine-grain time resolution or allowing the network size to grow unreasonably large. The goals of flexible classification and long time windows can be made consistent. Most importantly, we will propose a new neural net architecture which can attain these goals without requiring an explosion of parameters to estimate.

The multi-scale dynamic neural net (MSDNN) is a multi-layer, dynamic network which decimates (sub-samples) in time from layer to layer. This decimation, as we will show, is the functional foundation of making use of a large amount of context. We propose a technique which uses a very regular structure to self-organize a temporal hierarchy. The hierarchy can make use of successive multi-variable time samples (such as spectra) at the input layer, extracted short-term features at a low hidden layer, extracted long-term features at a high hidden layer, and the longest term units at the highest layer. The architecture allows training annotation to be put at any layer and also allows test outputs to be taken from any layer. Thus, this architecture can allow training with a data structure that includes input, known useful features, annotation or class labels, and temporal context simultaneously. The data structure requires only that different elements be appropriate for the different scales of time.

Figure 3 shows an example of a MSDNN. The MSDNN is a two-dimensional structure which is designed to act like most other fully connected networks in the n dimension and to smooth, downsample, and have shift invariance in the l (sampled time) dimension. The n dimension is shown at the top right of the figure and the inputs can be, for example, DFT coefficients. For the purposes of this explanation, this top right figure represents a "side" view of the MSDNN. Note that the variable l is held constant throughout this side view (to represent a single time slice) and that all layers are fully interconnected.

The top left figure corresponds to a "front" view of the network. The time series shown at the top are L samples of a time waveform. The samples could correspond to the time evolution of a single DFT coefficient. The circles in this structure represent the individual neurons, where the inset in the lower left shows the neuron function in detail. The behavior of each neuron in the l dimension is a sum-of-products followed by a non-linearity, $f(\cdot)$. Note that the there are only two weights and that there labels are dependent on layer number (m), yet independent of l. This independence of l allows shift invariance across the time variable and our learning rule for this network is designed to average errors across a layer to find these two weights. The reason that only two weights are needed for each input in the l dimension is due to: 1) the rich interconnection provided in the n dimension and 2) the intent to reduce unnecessary complexity by allowing more distant comparisons (e.g. between well-separated events in time) to occur in deeper and more smoothed layers. As will be seen in the next sub-sections the effect of this topology is dramatic in the reduction of the number of weights and is a necessary step in the application of a trainable architecture to long (say 50-200 frames) duration inputs.



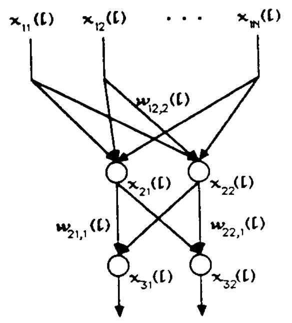


Figure 3. The multi-scale neural network.

Another important facet of this architecture is that outputs can be taken from any "hidden" neuron during use of this net and target codes can be compared to these outputs to find errors to back-propagate. The location of these outputs and target codes is determined by the segmentation locations found in the previous stage of the recognition system.

Reduction of Number of Weights

The reduction in weights to estimate can be shown for a general and regular MSDNN. If there are N dimensions (e.g. spectral samples) for each time frame and if the context window is L frames long, for a M layer MSDNN with a sub-sampling rate of l the number of weights to estimate is

$$W_A = Nl(1+\log_l L)$$

For a static net with M layers and m units per hidden layer and output layer a length L N-dimensional input requires

$$W_B = LNm + (M-1)m^2$$

weights to estimate. Some typical comparison numbers are N=16, L=512 (about 1.5 seconds of 3 mSec frames), l=m=8, and M=3. With these reasonable choices W_A = 512 and W_B = 65,664! Clearly, the complexity of a static multi-layer net for 1.5 seconds of speech (as measured in weights to estimated) is well beyond the training algorithms which are currently available. On the other hand the MSDNN has a complexity which is much more reasonable.

Derivation of Multi-Scale Training Rules

We hypothesize that a requirement for convergent training is that the sub-sampling operation does not allow aliasing. The reason that aliasing can be so damaging to training is due to its non-monotonic property which violates the necessary conditions for convergence in gradient descent. A standard approach would be to have a sub-sampling filter at each neuron's output where this filter would be designed to attenuate the high frequencies generated by the sigmoidal nonlinearity (η). The corner frequency of this filter could be set to be appreciably below half the sampling rate of the next layer thus reducing aliased frequencies to be insignificant.

We have devised an alternative approach to the reduction of aliasing for MSDNN's with a fan-in of 2 (as depicted in Figure 3). This approach relys upon a modification of the cost function in the usual back-propagation training rule [1]. Our reasoning is that if only two weights feed into any neuron, then choosing these weights to be as close to equal as possible would constitute a size 2 running average (in the l dimension) which is an optimal size two smoother. Clearly, a goal of completely equal weights in any layer (in the l dimension) in not very appropriate for effective training. However, a "smoothing" term in the cost function, which can allow an adjustable bias toward weight similarity, may effect much better training for the decimated layers of our MSDNN.

The usual cost function (to be minimized) in back-propagation is

$$J = 1/2(x_M - d_M)^2$$

where x_M is the observed output and d_M is the desired output. We define a new cost function

$$E = \rho E + \gamma D$$

where a small D indicates weights acting as effective smoothers. γ is a smoothing gain and ρ is the usual learning gain. For the output layer of a MSDNN D would be a function of the two weights, $w_{M-1}(0)$ and $w_{M-1}(1)$, which multiply the inputs of the output neuron, $x_M(0)$. A simple function for D could thus be

$$D = 1/2(w_{M-1}(0) - w_{M-1}(1))^2.$$

Note that D is minimum when $w_{M-1}(0) = w_{M-1}(1)$ (perfect smoother) and maximum (for a given magnitude) when $w_{M-1}(0) = -w_{M-1}(1)$ (1st difference).

For a the updates of the two weights to the output layer, the weight changes become

$$\Delta w_{M-1}(0) = \rho\delta x_{M-1}(l) + \gamma w_{M-1}(1)(w_{M-1}(1)^2 - w_{M-1}(0)^2)$$

$$\Delta w_{M-1}(1) = \rho\delta x_{M-1}(l+1) + \gamma w_{M-1}(0)(w_{M-1}(0)^2 - w_{M-1}(1)^2)$$

where $\delta = (d_M - x_M)x_M(1 - x_M)$. The back-propagation of these weight changes to other layers follows the same application of the chain rule as previous derivations [1] and our weights are averaged across the sampled time dimension (l) to allow for shift invariance as in Waibel et al [7].

Experimental Results

In order to make a preliminary assessment of the above training rules, we simulated a 1-dimensional case of the MSDNN. This 1-dimensional topology is as shown in the left half of figure 3. The experiments we tried was the discrimination of single pulse versus no pulse for various sizes of input. In all of the experiments the training set consisted of two binary sequences. The first sequence consisted of all zeros and the second sequences had all but one of the values equal to zero. In order to avoid solutions which simply used a single threshold, we did not use constant offset neurons (as in standard back-propagation). Input sizes of 4, 8, and 32 were tried for various choices of smoothing gain.

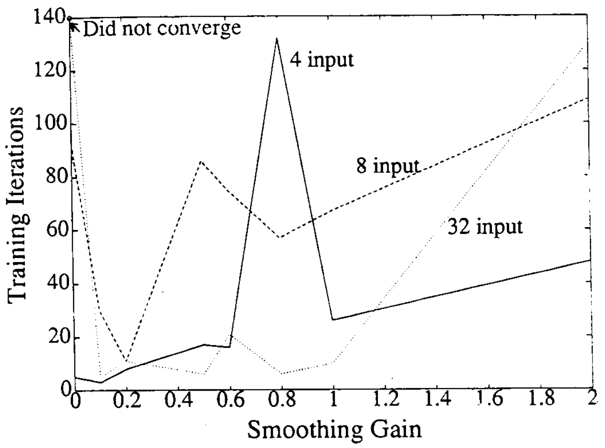


Figure 4. Number of training iterations as a function of smoothing gain (γ).

As can be seen in figure 4, in all cases a small amount of smoothing improved the convergence properties. It is especially notable that for the 32 input case there was no convergence when $\gamma = 0$ and convergence in 3 iterations when $\gamma = 0.1$.

Conclusions

Our preliminary results suggest that an incorporation of smoothing between layers for a tightly constrained multi-scale neural net architecture can be beneficial. There are two reasons that this observation is significant: 1) the hypothesis that the non-monotonic effect of aliasing is deleterious to training is supported, and 2) successively greater amounts of smoothing toward the output layer will be automatically achieved by a γ greater than 0.

We have not yet demonstrated that a MSDNN has practical advantage over a fully interconnected multi-layer neural net. This demonstration, connections to past work in multi-scale signal and image representations (e.g. [8,9]), and the incorporation of higher order optimal smoothers [10] are the subject of our current work.

Acknowledgements

The authors wish to thank Ronald Cole of Oregon Graduate Center and Dick Lyon of Apple Computer for their helpful discussions. Les Atlas was supported by a National Science Foundation Presidential Young Investigator Award and all of the authors were partially supported by the Washington Technology Center.

References

1. D. Rumelhart, G. Hinton, and R. Williams, "Learning Representations by Back-Propagation Errors," *Nature* **323**, pp. 533-536, 1986.
2. M. Minsky and S. Papert, **Perceptrons, Expanded Edition,** MIT Press, 1988.
3. L. Pitt and L. Valiant, "Computational Limitations on Learning from Examples," *J. Am. Soc. for Computing Machinery* **35**, pp. 965-984, October 1988.
4. Atlas, L., "Auditory Coding in Higher Centers of the CNS," *IEEE Engineering in Medicine and Biology Magazine* **6**, pp. 29-32, June 1987.
5. I. Sando, "The Anatomical Interrelationships of the Cochlear Nerve Fibers," *Acta Oto-Laryngology* **59**, pp. 417-436, 1965.
6. T. Homma, L. Atlas, and R. Marks, "An Artificial Neural Network for Spatio-Temporal Binary Patterns: Application to Phoneme Classification," *Proc. IEEE Conf. on Neural Information Processing Systems*, Denver, November 8-12, 1987.

A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme Recognition Using Time-Delay Neural Networks," *IEEE Trans. Acoust. Speech Sig. Proc.* **37**, pp. 328-339, March 1989.

8. A. Witkin, "Scale-Space Filtering: A New Approach to Multi-Scale Description," *Proc. ICASSP '84*, pp. 39A.1.1-39A.1.4, March 19-21, 1984.
9. P. Burt, "Smart Sensing Within a Pyramid Vision Machine," *Proc. IEEE* **76**, pp. 1006-1015, August 1988.
10. R. Schafer and L. Rabiner, "A Digital Signal Processing Approach to Interpolation," *Proc. IEEE* **61**, pp. 692-702, June 1973.