Functional link nets: removing hidden layers

Author: Yoh-Han Pao
Date: Apr. 1989
From: AI Expert(Vol. 4, Issue 4)
Publisher: UBM LLC
Document Type: Article
Length: 1,904 words

Abstract:
Neural networks tend to be computationally expensive. Good algorithms exist for specific programming tasks, but the algorithms are based on differing network architectures, processes, I-O formats and requirements. Passing results efficiently from one functional module to another, as is often required in real-world problems, can be difficult. Supervised learning of I-O pairs as implemented using a semilinear feedforward network architecture with error backpropagation via the generalized delta rule (GDR) converges too slowly. Modification of the supervised learning paradigm simplifies network architecture and reduces convergence time. This sort of an architecture can also implement unsupervised learning procedures and associative retrieval algorithms.

Full Text:
FUNCTIONAL LINK NETS: Removing Hidden Layers

Good neural net algorithms exist for specific programming tasks such as clustering, I/O pair learning and associative retrieval, but these algorithms are based on diverse network architectures, processes, I/O formats, and requirements. Consequently, it is difficult to pass results efficiently from one functional module to another, as is often required, to solve a multifaceted, real-world problem. In addition, supervised learning of I/O pairs as implemented using a semilinear feedforward network architecture with error backpropagation via the generalized delta rule (GDR) converges too slowly for practical use.

A modification of the supervised learning paradigm can simplify network architecture and significantly reduce convergence time. Architecture designed this way can also implement unsupervised learning procedures such as ART and associative retrieval algorithms. Additionally, hidden layers in the GDR network impede the unification of supervised and unsupervised learning. Because they are unnecessary in our network, both supervised and unsupervised learning are possible using the same architecture.

FUNCTIONAL LINK NET APPROACH

Our functional link network performs a nonlinear transformation of the input pattern before it is fed to the input layer of the network. The essential action is therefore the generation of an enhanced pattern to be used in place of the original. Both mathematical and pragmatic evidence show that supervised learning can be achieved very well using a "flat" net (one that has no hidden layers) and the delta rule if this enhancement is done correctly. The concept of a functional link is illustrated in Figure 1. The idea is that once node k is activated, different additional processes f.sub.1(0.sub.k)..., f.sub.n(0.sub.k) may also be activated through a functional link. D. J. Sobajic's theoretical analysis sheds light on the validity of the method.

Two models of pattern enhancement for the functional link will be explored in this article: the functional expansion model and the tensor, or outer-product, model. (Introduction of higher-order effects through the outer-product operation has also been proposed and investigated by Lee Giles and coauthors.) Different effects are obtained depending on the details of the model used, and the models may be used in tandem to great effect.

FUNCTIONAL EXPANSION MODEL

In the functional expansion model, the functional link acts on each node singly (Figure 2). In this model, the functional link acts on each component 0.sub.k of the input vector to yield the quantities f.sub.1(0.sub.k), f.sub.2(0.sub.k)..., f.sub.n(0.sub.k).

In some circumstances, f.sub.1(x) might simply be x itself and f.sub.2(x) might be x.sub.2, or the functions might be a subset of a complete set of orthonormal basis functions spanning an n-dimensional representation space, such as sin [pi]x, cos [pi]x, sin2 [pi]x, cos2, or [pi]x. The net effect is to map the input pattern onto a larger pattern space. Each component x is associated and represented with the quantities (x, x.sub.2,x.sub.3, ...) or with (x, sin [pi]x, cos [pi]x, sin2 [pi]x, cos2 [pi]x,...), depending on the set of functions we deem appropriate. No intrinsically new information is introduced, but the representation is enhanced.

In the tensor model, each component of the input pattern multiplies the entire input pattern vector. The functional link generates an entire vector from each of the components instead of generating a finite (or infinite) array of functional values as the functional expansion model does. Thus, the outer-product model is actually a special case of the functional expansion model--and not a very expansive one at that. This process may be described as the formation of an outer product between two vectors, one being the original pattern vector and the other being the same vector augmented by an additional component of value unity. Augmenting the vector regenerates the original pattern and the higher-order terms.

In both models, the process of representation enhancement may be used recursively, as appropriate. In the tensor model, the pattern might go through a sequence of transformations such as: [xi] * [xi,xixj] * [xi,xixj,xiXjXk] * .... j[is greater than or =]i j[is greater than or =]i k[greater than or =]j[greater than or =]i and the numbers of terms retained at any stage might be pruned by omitting the terms for which:

In equation 2, the values of x.sub.i and x.sub.i range from -1 to +1.

The effects of such transformations are generally dramatic, as demonstrated in the parity 3 problem shown in Figure 3. For supervised learning, not only is the learning rate usually greatly increased, but a flat net with no hidden layer suffices. This means that supervised learning can be carried out with the same net architecture as that used for unsupervised learning (as in nets for the ART algorithm).

The outer-product enhancement procedure introduces additional facets of representation that support learning. The degree of enhancement is often low. It is effective whenever explicit information on joint activations is helpful. But experience indicates that a low degree of functional expansion followed by outer-product enhancement is often efficient and effective.

In the case of the parity 3 problem (Figures 3 and 4), it was essential that two successive outer-product operations be carried out. If we had stopped at the first step, the flat net and delta rule could not have solved the problem, as shown in Figure 5, where curve (A) indicates failure to achieve learning. Since such empirical results can be misleading, a clearer understanding of what happens can be attained by examining Sobajic's analysis, which indicates this result was to be expected (see sidebar).

NET REPRESENTATIONS

Learning to represent an arbitrary mathematical function is a stringent test of a net's abilities. In this task, we present the net with 20 sampled points of a curve, ask the net to learn the I/O pairs, and then generate an estimate of the original function for any value. In essence, we learn a network representation of the function and provide estimates of the value of the function at all other points.

In this case, we use the functional expansion model and map the one component of the output pattern onto the space spanned by the set of orthogonal functions [sin (n[pi]x), cos (n[pi]x); n = 1,2,...]. One example of such mapping is presented in Figure 6A.

The architectures of the flat functional link net and the multilayered, semilinear GDR net are compared in Figures 6A and 6B. The respective learning rates attained with these nets are shown in Figure 7 and the regenerated figures are shown in Figure 8. The learning rate is about 200 times faster in the functional link net, and the estimated values are slightly better. These figures demonstrate the representation and generalization power of the functional link net. It far surpasses other methods, including the use of spline curves. Networks can also learn to represent surfaces: functions of two variables. Here we combined functional expansion and outer-product pattern procedures to use a flat net architecture for learning to represent a surface synthesized from two-dimensional Gaussians.

The original two-component input pattern (x.sub1., x.sub.2) was enhanced to a 10-component pattern with components: x.sub.1, sin ([pi] x.sub.1), cos ([pi] x.sub.1), sin (2 [pi] x.sub.1), cos (2 [pi] x.sub.1], sin (3 [pi] x.sub.1), cos (3 [pi] x.sub.1), x.sub.2, sin ([pi] x.sub.2), cos ([pi] x.sub.2), sin (2 [pi] x.sub.2), cos (2 [pi] x.sub.2), sin (3 [pi] x.sub.2), cos (3 [pi] x.sub.2), x.sub.1x.sub.2, x.sub.1 sin ([pi] x.sub.2), x.sub.1 cos ([pi] x.sub.2), x.sub.2 sin ([pi] x.sub.1) and x.sub.2 cos ([pi] x.sub.1). A comparison of the actual and estimated values of the surface along the section x.sub.2=0.75x.sub.1 demonstrates that reasonably close agreement was attained.

A representation was also learned using the generalized delta rule for the nonenhanced patterns. Two hidden layers were used, with eight nodes in the first hidden layer and 10 notes in the second hidden layer. A comparison of the learning rates for the two types of nets again indicates a significant increase in the rate of learning of the functional link net. Twenty-five values estimated by the two types of nets were compared with actual values. Reasonable agreement was obtained in both cases, but the functional net results were slightly better.

INTEGRATED ENVIRONMENT

The functional link approach tries to produce a sufficiently enhanced representation for the input patterns so that the associated (enhanced I/O) pairs might be learned with a flat net. One advantage of using a flat net is that both supervised and unsupervised learning can be carried out using the same net architecture with no need to reconfigure or shuffle the data in moving it from one paradigm to the next. This is very important in any pattern recognition task that might require the capability to infer classification procedures and estimate attributes.

To illustrate, a neural net-based system for automatic soybean grading and processing is described. Soybean samples are described by patterns of attributes, and unsupervised learning discovers four clusters in attribute space corresponding to the superior, regular, industrial, and substandard grades. Actions are associated with each of these clusters (Figure 9). Thus, if a soybean lot is recognized as superior, it is processed and bagged in a manner to maintain it suitable for human consumption. The regular grade lots are

processed for feed. Prices and differentiation within each cluster are estimated based on training examples provided by experts.

To use this system, a soybean sampling attribute pattern is given to the automatic processing system. If the soybeans are "superior," associated processing procedures are activated. In addition, the same net is now made ready for supervised learning or estimating output values--in this case, the selling price. This output is obtained, and associated patterns or procedures can be attached to such outputs (Figure 10).

Taking each cluster, one at a time, we can determine if the class labels of the patterns are indeed "similar" to the extent that class membership is also the same. If it is, we might conclude the patterns are described realistically. We can then also synthesize a set of weights that will yield the desired output whenever one of those patterns is presented as input. The output model(s) is then associated with the node used for unsupervised learning. During the supervised learning, only a few of the (possibly many) patterns need to be used in the supervised training because they are all quite alike. All others can be discarded. When dissimilar patterns have the same class label, the disjoint volumes in pattern space are labeled with the same class label.

This easy coexistence between supervised and unsupervised learning relieves a net of the burden of trying to associate many dissimilar patterns with the same output. In addition, often if two unlike groups end up in the same cluster, either the pattern description is inadequate or more vigilance needs to be exercised when considering what patterns are similar. This scenario is one of many in which supervised learning, unsupervised learning, and associative recall can be used in intermixed manner, made possible primarily by using the functional link net with the flat net architecture.