Artificial Neural Networks for Advertising and Marketing Research: A Current Assessment[1]

Joseph Woelfel

University at Buffalo

[1] I am grateful for the kind assistance of N. J. Stoyanoff and Scott Danielsen of Terra Research and Computing, Inc., for use of Terra Software, proprietary data and technical assistance. GALILEO and CATPAC are trademarks of The Galileo Company. Exclusive rights for the distribution of GALILEO software are assigned to Terra Research and Computing Company, Inc.

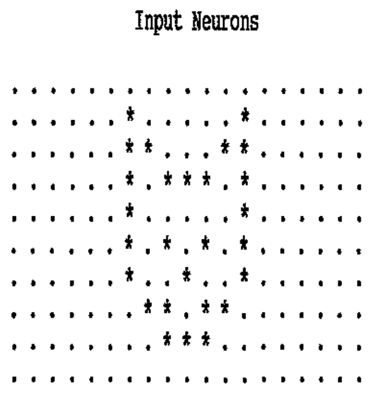
ABSTRACT

While modern research has emphasized the extraordinary complexity of organic neural structures and processes, at the same time it has given rise to the belief that certain common and reasonably simple mathematical forms may underlie this complexity. This in turn has led to the very rapid development of mathematical models of elementary neural processes, as well as to computerized implementations of these mathematical models. These *artificial neural networks* or ANN's, even at their present elementary level of development, have already shown great utility in applied areas, particularly in areas of policy.

ANN's are mathematical models in two senses. First, they represent idealized models of biological neural systems, and are often used as tools to help better understand the functioning of biological systems. But more frequently, ANN's are used to model other processes, and it is these other kinds of uses that provide the basis for ANN's as an applied technology.

In their simplest form, these mathematical models consist of artificial neurons or *nodes*, which are connected to each other by communication channels of varying strength. Multiple inputs to any node are summed in a *transfer function*; this sum is then entered into the node's *activation function*, which determines whether the node will become active, and, in the case of continuously variable nodes, how active it will become.

Sets of such nodes represent patterns in the same way as the activation of subsets of pixels on a video screen represent patterns on the screen. Changing patterns of activation represent changing images represented by such a system. Any arbitrary pattern may be represented. Figure 1 shows a set of nodes representing a crude pictorial image of a cat, while Figure 2 shows a similar set of nodes representing the word "CAT."



1 Figure 1: Input nodes Depicting a Cat

Because the nodes are connected to each other by pathways of various different conductivity, the activation value of the active node(s) is in turn conducted or communicated along the connections as a function of the strength of these connections. It is this process of communicating activation levels throughout a network of nodes that gives a neural network its information processing capabilities, which include the ability to represent, store, retrieve and associate patterns of arbitrary complexity, and to generalize information learned about a given pattern to other related patterns:

```
              Output Neurons

. . . . . . . . . . . . . . . . . . .
. . * * . . . . * . . . . * * * * * .
. * . . * . . * . * . . . . . * . . .
. * . . . . . . . . . . . . . * . . .
. * . . . . * * * * * . . . . * . . .
. * . . . . . . . . . . . . . * . . .
. * . . * . . . . . . . * . . * . . .
. . * * . * . . . . . . . * . . * . . .
. . . . . . . . . . . . . . . . . . .
```

2 Figure 2: Nodes representing "CAT"

o If the set of nodes which represent a given pattern are more strongly linked to each other than to other nodes, then the activation of some of the nodes in the pattern is likely to activate the remaining nodes in the pattern. Thus the network can serve as a pattern storage and recognition device.

o If the set of nodes which represents one pattern is linked to the nodes which represent another pattern, activation of a sufficient subset of the nodes in the first pattern can serve to activate the nodes which represent the succeeding pattern. Thus the network can serve as a pattern association device.

o Since the nodes which represent one pattern are also likely to be a part of the set of nodes which represents a similar pattern (e.,g., many of the neurons or nodes whose which represent the concept "cat" are also likely to be part of the set of nodes which represent the concept "leopard"), information learned about one pattern is automatically generalized to related patterns. (I.e., any new connections formed with neurons which form part of the concept "cat" must involve neurons in "leopard", since they are, at least to some extent, the same neurons.)

In terms specifically of the example provided in Figures 1 and 2, a neural network can in general find connection weights among the nodes in Figure 1, for example, such that the activation of a large enough subset of nodes in the picture of the cat will activate the rest of the nodes in the figure. (The same is true, of course, for the word "CAT.") Moreover, it is also possible in general to find weights between the two sets of nodes such that the activation of a large enough subset of the nodes in the picture will activate not only the remaining nodes in the picture, but also the nodes representing the word "CAT." (The reverse, of course, is also possible.)

Perhaps more important than the ability to link patterns (which, after all, can be linked in an ordinary serial computer program), is the robust character of the link. Since the decision of each nodes as to whether to fire is governed by a weighted sum of many inputs and, typically, by some thresholding function, the proper association can be made even if part of the input is left out, or if part of the input is erroneous. Neural networks, both real and artificial, are highly tolerant of input noise. Neural networks, then, are particularly good at problems that serial computer programs are bad at. They deal well with poor or degraded data, and recognize patterns in a holistic way.

The significance of these facts for marketing and advertising research, of course, lies in the fact that virtually any pattern whatever may be represented by the network. One pattern, for example, might consist of a set of numerical values which describe the credit history of an applicant for a franchise, while another pattern might represent the numerical likelihood that the applicant will succeed in the franchise operation. Or one pattern might consist of the demographic and psychographic characteristics of a potential new car buyer, while another might consist of the consideration set of cars he or she might buy. Moreover, the network can make these kinds of matches even if the input data are incomplete, noisy or otherwise imperfect. The general ability of artificial neural networks to learn to related patterns of any general type, therefore, forms the basis for a powerful and general applied technology.

A network learns to recognize patterns and associate them with other patterns by forming and changing the connections among nodes. Although alternative taxonomic models exist, it is most common to classify ANN's by the way in which their connection strengths (weights) are modified. This leads to three fundamental types of ANN: *supervised* models, *unsupervised* or *self-organizing* models, and *hybrid* models. Each of these models has different characteristics and different uses.

Supervised models:

Supervised ANN's are by far the most common models available, and form the lion's share of the applied market. Supervised ANN's bear a functional resemblance to the generalized regression model. Like the regression model, supervised ANN's consist of input (independent) variables and output (dependent) variables. Also like the regression model, data for supervised ANN's consist of *cases*, which are sets of values of inputs and outputs, with each set of values making up a case.

Both supervised ANN's and the regression model then attempt to fit inputs to outputs, but in different ways. The regression model attempts to find coefficients or slopes for each of the terms in a function provided by the analyst which minimize the degree of error in predicting the output values of the cases given their input values. The supervised ANN also attempts to predict the values of the outputs from the (given) values of inputs for each case, but it does so by reconfiguring the pattern of connections between input and output nodes in the network itself. Because the transfer functions of the nodes in the network are typically nonlinear (generally logistic), and because of the multi-layer structure of the network, the network is able to fit virtually any functional form whatever. What's more, there is no need for the analyst to stipulate the functional form prior to the analysis; the ANN will find it automatically.

Although variations are common, by far the most typical supervised ANN consists of three layers of nodes, an input layer, a middle or hidden layer, and an output layer. Nodes within each layer are generally not connected to each other, but the nodes in the input layer have (initially random) connections with the hidden layer, and the nodes in the hidden layer are (randomly) connected to those in the output layer. (Typically, each node in a layer will be connected to all the nodes in the following layer, although the values of the connections will vary randomly.)

Data are entered into the network in the form of "cases" or "patterns" which are identical in form to the cases familiar in regression models. Each case or "input pattern" consists of a set of values for a set of *input characteristics* (independent variables) and a set of values for a set of values for a set of *output characteristics* (dependent variables). Again typically, each input characteristic corresponds to a node in the input layer, and the *activation value* of the input node corresponding to a given input characteristic is set to the value of that input characteristic for that case.

These input activations (which correspond to the values of the input characteristics for a single case) are then communicated through the (random) connections to the hidden layer. Each node in the hidden layer then sums the inputs from all input nodes as in equation 1:

$$net_i = \sum_{j-1}^{n} a_j w_{i,j}$$

where a_j represents the activation value of the j_{the} input node, and w_{ij} represents the weights or connection strengths between the j_{the} input node and the i_{the} hidden node.

A commonly used activation function is the logistic function, sometimes referred to as a "sigmoid" function, because its shape when plotted resembles a capital Greek sigma:

$$a_{pj} = 1/(1+e^{-net_{pj}}) \tag{2}$$

where:

a_{pj} = the activation of the j_{the} node for the p_{th} pattern, and

net_{pj} = the net input to the j_{the} node for the p_{th} pattern from all input nodes.

This results in the nodes in the hidden layer taking on various (random) activation values, which are in turn communicated to the output nodes in the same fashion. The output nodes then take on a pattern of activations which represent the network's "best guess" as to what the actual values of the dependent variables might be for that case. Since the initial weights are simply random numbers, however, these output activations represent a random guess. These random activations are then subtracted from the "correct" activations as represented by the values of the output characteristics (or dependent variables) in the input pattern or case. This subtraction yields an *error term* for each output node. The essence of a supervised neural network is that it uses this error term to reconfigure the network -- in essence the actual output values of the cases in the training set serve as a "supervisor" or teacher, and the network is "trained" to minimize these errors.

The most common procedure for modifying the weights is called "back propagation." The errors may be expressed as a function of the output activations, which can in turn be expressed as a function of the connection strengths or weights from the hidden nodes to the output nodes and the activation values of the hidden nodes. These activations of the hidden nodes, can, of course, be expressed as a function of the connection weights from the input to hidden nodes. By the chain rule, it is possible to find a *weight error derivative* for each weight, which represents the slope of the weight relative to the error. Each weight is then multiplied by its appropriate weight error derivative, which modifies the weights so that the network will produce the correct output activations for the input values in that case. The next case is then entered and the process is repeated.

Adjusting the weights to fit the second case generally results in a worsening of the fit for the first case, so in practice the process is repeated until all cases have been entered, then the network cycles through all the cases repeatedly until the total error summed across all cases either falls below a preselected threshold or until no further improvements occur. The result is a network which can provide an optimal mapping of input values onto output values for the cases studied. Because of the generalizing ability of networks described above, the network is then able to read input values for cases not in the training set and make (usually excellent) predictions about the values of the output characteristics (dependent variables) for those novel cases.

Researchers increasingly understand that the ability to learn to predict the output values of cases in the training set as accurately as possible (the rough equivalent of minimizing a least squares error term in a regression model) is not always an optimal criterion for determining when a network has been well trained. Much more important is the ability to predict the output values of cases not contained in the original training set. Many researchers have noted, in fact, that "overtraining" the network can *reduce* its ability to generalize to new cases; such networks are said to have simply memorized the training set. As a result, in practical use, the researcher sometimes holds back a subset of the original dataset to use in testing the generalizability of the network to cases other than those in the training set. Many recent releases of packaged neural network software contain provisions for using the generalizability of the network as a criterion for ending the learning or training phase.

Other methods for adjusting the connection strengths exist, but this *back propagation* algorithm is by far the most commonly used in applied settings; other methods yield equivalent results and differ only in their computational characteristics.

Whatever procedure is used to adjust the weights, the result is a network whose connection weights are so calculated that the network can match a set of input patterns to a set of output patterns with minimum error. Since the logistic function which forms the basic operating characteristics of the network is non linear and non monotonic, the system is highly non linear, and several authors have shown that any function can be approximated as a sum of logistics. This provides one of the most significant advantages of the supervised neural network over the regression model: the neural network can represent relationships of any degree of curvilinearity automatically; the researcher does not need to specify the functional form of the relationships among the input and output variables in advance.

Supervised ANN's may have any number of input and output nodes (variables), may contain feedback loops, and may be cascaded, with outputs from one network serving as inputs to another. In general, a set of cascaded supervised ANN's can model any system that can be modelled in the regression framework. As a rule, many applied users of supervised ANN's consider them a replacement for the general linear regression model and its more sophisticated relatives, such as LISREL. Advocates of the supervised ANN model claim several significant advantages over the generalized regression model:

o There is no need to prespecify the correct functional form which relates input to output variables. The network automatically finds the optimal form,

o All possible interaction terms are automatically considered, o ANN's work well with noisy and incomplete data,

o ANN's are well suited for real time operation, adjusting their structure as new cases are added,

o ANN's generally fit the same variables to the same data better than best-fitting regression models.

Self-Organizing models:

In supervised networks, the output values for any case serve as the "correct" outputs the network should provide when supplied with the given input values for that case. In self-organizing networks, there is no "correct" pattern of outputs. Instead, the network changes its internal connection strengths to recognize recurrent patterns of inputs. Although many types of self-organizing or unsupervised networks have been developed, in general most tend to strengthen connections among nodes that are frequently coactive. Over time this leads the network to modify its own internal structure to become increasingly similar to its environment.

The set of activation values of the nodes of a network at a given time may be considered a "pattern." If a given pattern of activations is presented to the network as input relatively frequently, the nodes which make up the pattern will become fairly tightly connected. The result will be that, due to their interconnections, the activation of a large enough subset of the nodes in the pattern will usually cause the remaining nodes to become active as well. This means that the total pattern may be stored in the network and retrieved later by activati~~on~~ng parts of that pattern.

A typical use for a self-organizing ANN is in information storage and retrieval systems. Users may request several documents during a search; if this set of documents is a "typical" search pattern, then the network will learn this pattern and tend to retrieve the entire set of documents when some subset is requested.

Self-organizing or unsupervised ANN's have not as yet developed as large a base of commercially available technologies as have their supervised counterparts, although several are commercially available. One such program is CATPAC, an unsupervised neural network which is designed to read and "understand" text. CATPAC reads any ASCII text, discards minor words such as articles, prepositions, and the like from a prewritten exclude file,and discards additional words which fall below an arbitrary, user-set frequency of occurrence. For each remaining word, an artificial neuron is constructed which represents that word. A scanning window of user-set size is then passed through the text. Whenever a given word is in the scanning window, the neuron which represents that word is activated (it's activation value is set to 1.0). At the user's option, CATPAC may be set to "cycle" up to four times. During each cycle, activations of the nodes are propagated along the connections and other nodes connected to the active nodes may also be activated (or deactivated in the case of negative connection weights). Regardless of how whether the network has been set to cycle, connection weights among the neurons which represent the words are modified according to the rule:

$$w_{ij}=w_{ij}+a_i a_j h$$

where h represents a "learning constant". (In CATPAC, where connection strengths are normalized such that the largest weight equals 1 or -1, the value of the learning constant is .05). This means that the connection strengths among nodes will be strengthened when they are simultaneously active to the extent that they are active. In essence, this means that nodes which behave similarly will be increasingly strongly connected. This results in a square matrix which resembles a correlation matrix, although, due to the spreading activation due to cycling, the matrix need not in general be symmetric. Each value in the matrix represents the degree to which the nodes representing their corresponding words are "similar". In practice, this means that words that are close to each other in the text will be tightly connected; it also means that a word similar to another word which is similar to a third word will also be tightly connected to the third word, and so on.

This matrix (called a "weight input network" or WIN matrix) may be entered into a variety of multivariate analytic procedures to reveal the underlying structure of the text. CATPAC routinely provides two familiar techniques, cluster analysis and multidimensional scaling ("perceptual mapping"). The hierarchical clustering algorithm built into CATPAC produces a dendogram of the clusters in the text. Typically, CATPAC is used to analyze interview data, focus groups, texts, and the like. Below is an example of a CATPAC analysis of a series of interviews about Buffalo, Detroit and San Diego (-1's delimit cases):

```
Detroit is exciting! You can feel the excitement of it. It's big, and brawling a
        You have to watch your hat and coat, but its worth it. You can have a good time in
detroit.
        There's lots of things to do in Detroit. Detroit is a fun city. Lots of driving: a ca ci
-1
Buffalo is a friendly city. They call it the city of good neighbors, and that's
true. Buffalo is the friendliest city. Buffalo is a working town, but the people
a party there.
-1
San Diego is warm, even too hot sometimes. It's brown, and water is scarce. There
a water shortage. Usuallya drought. San Diego is brown. Shortage of water is a
problem.
        There's palm trees, and ocean and beach. If you like beaches, San Diego
is the place for you. Warm, sand, sun, surf. Beautiful city.
-1
Detroit is scary. It's big and exciting, but it's scary. There's the tigers, and
a great city.
-1
Buffalo is on the niagara river; it's near niagara falls. The university of
Buffalo is there, in Amherst. It's windy. Very windy. The people are friendly.
-1
```

```
San Diego! Palm trees! Beaches! Sand! Sun! Surf!. Yippee!
-1
male
-1
Detroit is excellent. Very exciting. A lot of fun. Somewhat dangerous, but
worth it. Fun. A Big city, very big. Action.
-1
Buffalo is the friendly city. Buffalo is the city of good neighbors. Buffalo
and Niagara Falls are neighbors. A party community. Laid back. A good time.
-1
San Diego is sun, sand surf and sex. Palm trees. Brown, drought, expensive.
-1
```

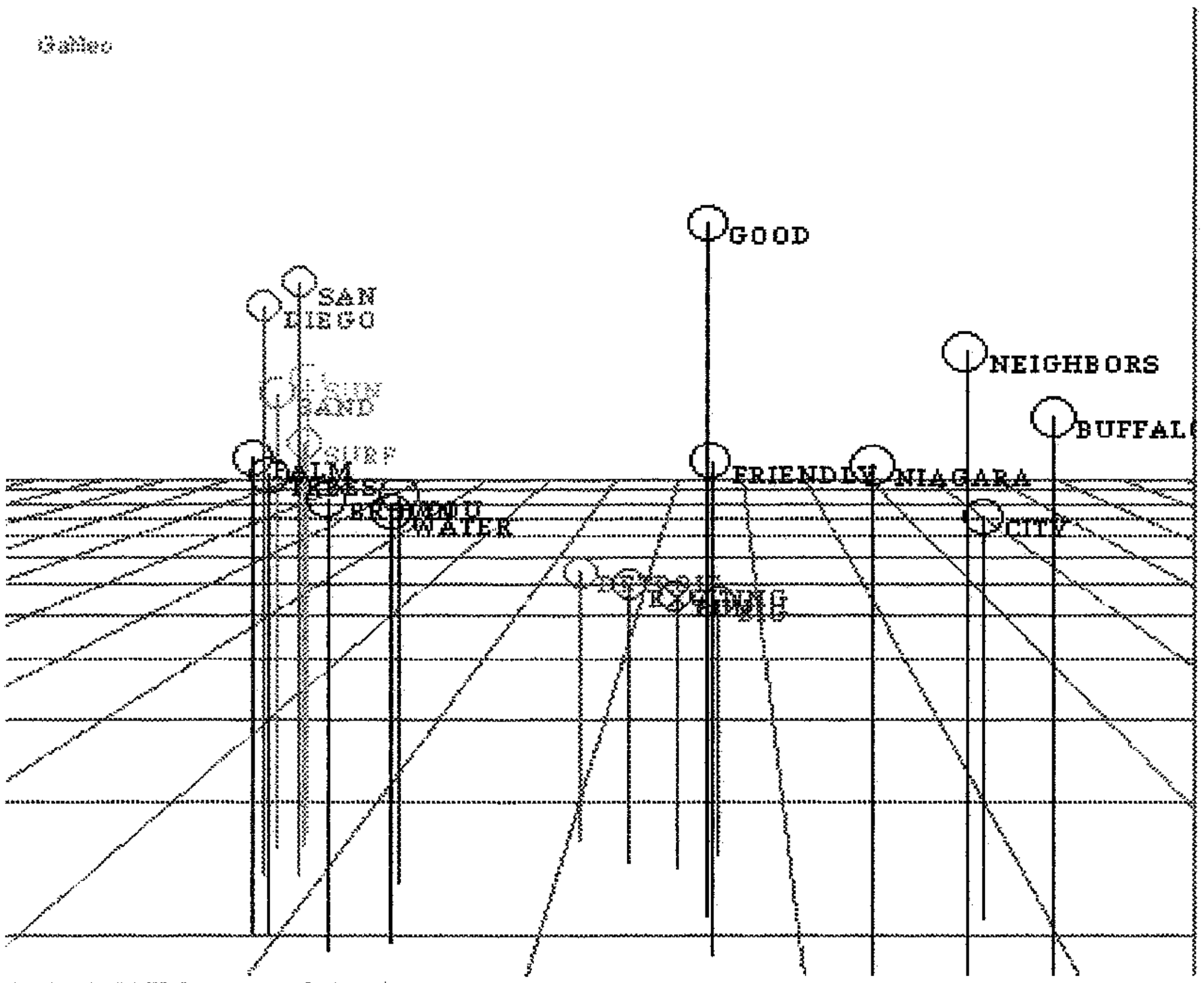
CATPAC produces the following output for these data:

```
        CATNET_PC   1.01      09/16/91      14:47:34

 TITLE: test

 TOTAL WORDS            81     THRESHOLD             .000
 TOTAL UNIQUE WORDS     20     RESTORING FORCE       .100
 TOTAL  WINDOWS        144     CYCLES                   1
 TOTAL LINES            23
 WINDOW SIZE             7     CLAMPING               YES
 SLIDE SIZE              1

                  W Y B S D P T S S S D E B F C F G B N N
                  A O R A I A R A U U E X I U I R O U E I
                  T U O N E L E N N R T C G N T I O F I A
                  E . W . G M E D . F R I . . Y E D F G G
                  R . N . O . S . . . O T . . . N . A H A
                  . . . . . . . . . . I I . . . D . L B R
                  . . . . . . . . . . T N . . . L . O O A
                  . . . . . . . . . . . G . . . Y . . R .
                  . . . . . . . . . . . . . . . . . . S .
                  . . . . . . . . . . . . . . . . . . . .
                  . . . . . . . . . . . . . . . . . . . .
                  . . . . . . . . . . . . . . . . . . . .
                  . . . . . . . . . . . . . . . . . . . .
                  . . . . . . . . . . . . . . . . . . . .
                  . . . . . . . . . . . . . . . . . . . .
 .64522210E+00    . . . . . . . . . . . . . . . . . ^^^ .
 .48450430E+00    . . . ^^^ . . . . . . . . . . . . ^^^ .
 .48298940E+00    . . . ^^^ . . . . . . . . . . . . ^^^^^
 .41938970E+00    . . . ^^^ . . ^^^ . . . . . . . . ^^^^^
 .37922520E+00    . . . ^^^ . . ^^^^^ . . . . . . . ^^^^^
 .31803620E+00    . . . ^^^ ^^^ ^^^^^ . . . . . . . ^^^^^
 .23784230E+00    . . . ^^^ ^^^^^^^^^ . . . . . . . ^^^^^
 .21497290E+00    . . . ^^^ ^^^^^^^^^ . . . . . . ^^^^^^^
 .21425270E+00    . . . ^^^^^^^^^^^^^ . . . . . . ^^^^^^^
 .15372330E+00    . . . ^^^^^^^^^^^^^ . . ^^^ . . ^^^^^^^
 .14148200E+00    . . ^^^^^^^^^^^^^^^ . . ^^^ . . ^^^^^^^
 .12773930E+00    . . ^^^^^^^^^^^^^^^ . . ^^^ . ^^^^^^^^^
 .88879470E-01    . . ^^^^^^^^^^^^^^^ . ^^^^^ . ^^^^^^^^^
 .71331880E-01    . ^^^^^^^^^^^^^^^^^ . ^^^^^ . ^^^^^^^^^
 .65057890E-01    . ^^^^^^^^^^^^^^^^^ ^^^^^^^ . ^^^^^^^^^
 .52048050E-01    ^^^^^^^^^^^^^^^^^^^ ^^^^^^^ . ^^^^^^^^^
-.24098470E-01    ^^^^^^^^^^^^^^^^^^^ ^^^^^^^ ^^^^^^^^^^^
-.84040960E-01    ^^^^^^^^^^^^^^^^^^^ ^^^^^^^^^^^^^^^^^^^
-.48337410E+00    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

The CATPAC dendogram shows clusters by darkening the area under words which cluster together. At the highest level, CATPAC finds the cluster [Buffalo, neighbors] (Buffalo is called the City of Good Neighbors). Next it finds [San, Diego]. On the next level, Niagara enters the Buffalo cluster (Buffalo and Niagara Falls are about 30 miles apart.) Next the cluster [sand,sun] appears; in the next level "surf" joins this cluster. Next CATPAC finds [palm,trees], and this new cluster merges with the [sun,sand,surf] cluster at the next level. This new cluster in turn merges with the [San, Diego] cluster, which ultimately grows to [San, Diego, sand, sun, surf, palm, trees, brown, water, you] (several respondents mentioned that San Diego was brown due to draught.) The final Buffalo cluster becomes [Buffalo, neighbors, Niagara, good, friendly, city], and the Detroit cluster becomes [big,fun,exciting,Detroit].

Hybrid models

ANN's are often found in mixed environments which employ other technologies in addition to neural models. Most commonly, neural networks are combined with serial software for data access and handling. A typical combination is neural networks and rule-based expert systems, and neural networks and fuzzy-logic modules (although ANN's can be made fundamentally fuzzy in their own right). A model familiar to Communication researchers are neural networks in which connection weights are directly measured rather than established by the ANN itself (Galileo is such a model). Similarly, combinations of unsupervised ANN's with directly measured communication data provide useful means for the analysis of social networks more commonly studied by conventional software such as NEGOPY and UCINET. CATPAC itself is a hybrid program, using conventional parsing and word counting



3 GALILEO map of CATPAC output.

techniques, along with a conventional cluster analysis routine, although the similarity matrix among words in the document is computed by a neural network. Figure 3 shows one of the most powerful hybrid uses, using a similarities matrix generated by the neural network in CATPAC and the perceptual mapping algorithm from GALILEO: Figure 3 is instructive in two ways: First, it gives a clear indication of how neural programs may be combined with conventional algorithms to produce useful results in a form already familiar to analysts and clients. Second, it shows clearly the kind of power available from the new neural algorithms; few analysts even a short while ago would have considered it likely to form a precise perceptual map from the few lines of text on pages 9 and 10 above.

Summary

Artificial Neural Networks show great promise for advertising and market research. Even at there present early stage of development they provide capabilities not present in the most highly developed conventional multivariate analytic procedures. Perhaps most interesting of all, their similarity to human reasoning provides analyses that are more like the intuitive "gut" pattern recognition long favored by many in advertising and marketing. Their ability to be used in combination with conventional methods in hybrid models allows users a familiar format for presentation of results. And the ability of ANN's to grasp patterns from incomplete, noisy data opens up possibilities for rigorous analysis previously unavailable with conventional techniques.

At the same time, no computational procedure is immune to human blunders, and it is quite possible to produce misleading or erroneous results with the new technology. Particularly because of their ability to make use of noisy and incomplete datasets, there is a fair danger analysts can be misled into overanalyzing poor quality or useless data. A backpropagation network, for example, cannot develop a useful model to predict the values of a given set of variables if the variables which control those values have not been included in the model, just as a regression model cannot accurately explain the values of a set of dependent variables if the right independent variables have not been measured. Similarly, not all texts contain sufficient information to produce a useful cluster analysis or perceptual map, regardless of the sophistication of the analytic tools.

These cautions notwithstanding, currently available artificial neural networks make up a useful and powerful technology for the advertising and market researcher, subject to the usual cautions any prudent researcher would take.