Artificial Neural Networks in Policy Research: A Current Assessment

by Joseph Woelfel, State University of New York at Buffalo

Recent advances in neuroscience, computer science, psychology, and other fields have led to the development of computer programs that are modeled, in principle, on idealizations of organic neural structures. These artificial neural networks (ANNs) exhibit important properties that promise great usefulness for policy researchers. Most important among these are ANNs' ability to learn to identify complex patterns of information and to associate them with other patterns. Furthermore, like their biological predecessors, ANNs can recognize and recall these patterns and associations in spite of noisy, incomplete, or otherwise defective information inputs. ANNs can also generalize information learned about one or more patterns to other related patterns. As a result, ANNs have already found extensive use in areas once reserved for multivariate statistical programs such as regression and multiple classification analysis, and are developing an extensive community of advocates for processing text and other qualitative information.

While modern research has emphasized the extraordinary complexity of organic neural structures and processes, it has, at the same time, given

---

Joseph Woelfel is professor and chair at the Department of Communication, and senior fellow at the Rockfeller Institute of Government at the State University of New York at Buffalo. The author is 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. Terra also provided access to several of its software beta test sites, which provided invaluable information about the applications of artificial neural networks in large-scale marketing and advertising research. David Wassman of J. Walter Thompson and Ford Motor Company, Jim Leiman at MOR-PACE, Gary Dispensa at Elrick & Lavidge Marketing Research, and Steve Parrish of Image Engineering were particularly generous with time and information. Thanks also are due to respondents at the 1992 Advanced Research Techniques Forum of the American Marketing Association at Lake Tahoe for comments on an earlier version of this paper. GALILEO and CATPAC are trademarks of the Galileo company. Rights for the distribution of GALILEO software are assigned to Terra Research and Computing Company, Inc.

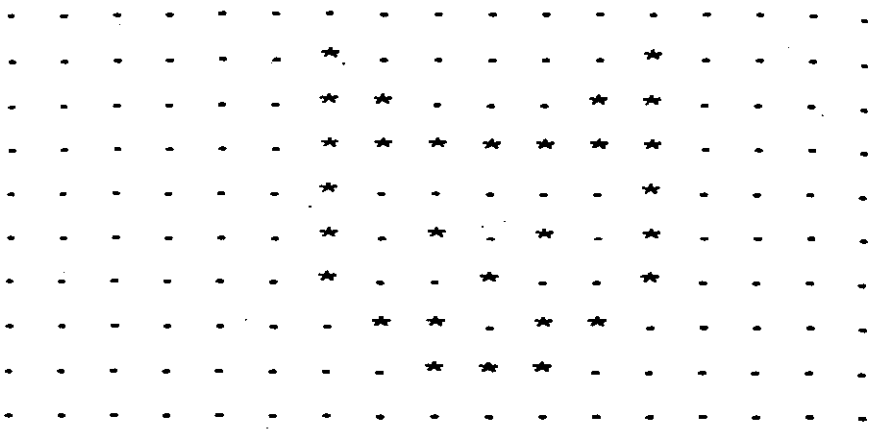

Figure 1. Set of nodes representing a picture of a cat (* represents an active node).

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* (ANNs), even at their present elementary level of development, have already shown great utility in applied areas.

ANNs 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, ANNs are used to model other processes, and it is these other kinds of uses that provide the basis for ANNs 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* by picturing the sequence of the three capital letters *C*, *A*, and *T*. In these figures, an asterisk represents an activated node, while a dot represents an inactive node.

In a video screen, the activation of each pixel is independent of the activation of each of the others. But because the nodes of a neural network are connected to each other by pathways with different conductivities, the activation value of the active node(s) is in turn conducted or commu-

Figure 2. Set o
node).

nicated alon
nections. It i
a network o
capabilities-
associate pa
learned abo

- If the se
linked to ea
the nodes ir
pattern. Th
device.
- If the s
that repres
nodes in th
succeeding
device.

For Figu
weights a
set of noc
the figure
Moreover
sets of nc
nodes in
ture, but a
course, is
Perhaps
all, can be
character
is governe
threshold
the input

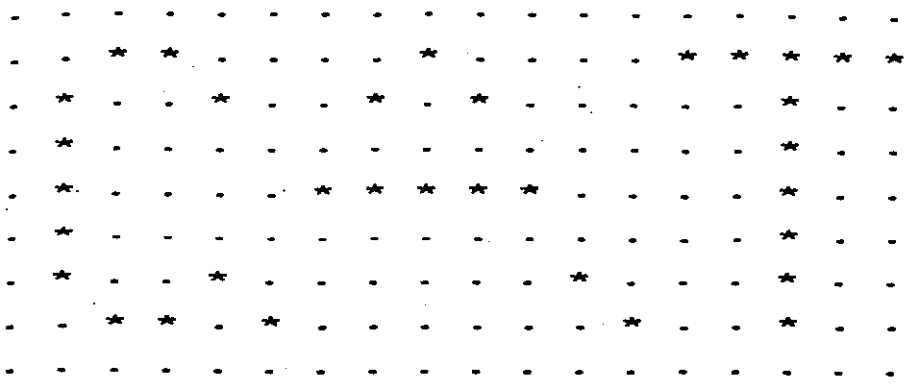

Figure 2. Set of nodes representing the letters *C*, *A*, and *T* (* represents an active node).

nicated 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. For example:

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

For Figures 1 and 2, a neural network can in general find connection weights among the nodes, 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* in Figure 2.) 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 node 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: thus, they are particularly good at problems that serial computer programs do not handle well. They deal effectively with poor or degraded data, and recognize patterns in a holistic way.

Neural networks have an additional striking property: They are able to generalize new information gained about any arbitrary pattern to other related patterns. Since the nodes that represent one pattern are also likely to be a part of the set of nodes that represents a similar pattern (e.g., many of the neurons or nodes that represent the concept "cat" are also likely to be part of the set of nodes that represent the concept "leopard"), information learned about one pattern is automatically generalized to related patterns. Any new connections formed with neurons that form part of the concept "cat" must involve neurons in "leopard," since they are, at least to some extent, the same neurons. If the network depicted in Figures 1 and 2 were to learn to associate "hunts" with "cat," it would also associate "hunts" with "leopard" (see Armstrong & Bauman, 1993).

The significance of these facts for policy research lies in the fact that virtually any pattern may be represented by the network. One pattern, for example, might consist of a set of numerical values that describe the educational structure of a community, while another pattern might represent the educational and occupational attainments of its residents. One input pattern might consist of a set of economic indicators and a set of governmental activities, while the output pattern might include other (future) indicators such as revenues, productivity indices, inflation rates, and the like. Or one pattern might consist of the demographic characteristics of a community combined with the history of governmental actions there, while another might consist of the rates of occurrence of various health problems. Once an artificial neural network has been trained to relate these kinds of patterns, it can make (often excellent) predictions about the rates of health problems for a given community based on known patterns of public health practices.

To be sure, this type of problem can also be approached by traditional methods, like econometric modeling, regression, and the like. But due to its robust character, the network approach can make these kinds of matches even if the input data are incomplete, noisy, or otherwise imperfect (see, e.g., Armstrong & Bauman, 1993). Policy-related problems, perhaps more than others, typically require clear decisions in the face of poorly defined problems, with poorly measured variables, incomplete and inaccurate data, and incomplete theoretical understanding. While the performance of virtually all traditional multivariate statistical procedures deteriorates badly under these conditions, neural networks excel under precisely these circumstances. The general ability of artificial neural networks to learn related patterns of any general type, therefore, forms the basis for a powerful and general applied technology for policy research. Furthermore, the generalizing property of neural networks provides a special ability. For example, if it were to be discovered that a particular

drug or food led to increased risk of cancer or reduced chance of heart disease or the like, a previously trained neural network would immediately suspect that related drugs might produce the same effect without any further retraining.

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 ANNs by the way in which their connection strengths (weights) are modified. This leads to three fundamental types of ANNs: (a) *supervised* models, (b) *unsupervised* or *self-organizing* models, and (c) *hybrid* models. Each of these models has different characteristics and different uses.

Supervised Models

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

Both supervised ANNs 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 (actually, the sum of squared errors) 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 multilayer structure of the network, the network is able to fit virtually any functional form whatever (Funahashi, 1989; Kurkova', 1992). Most important for the policy researcher, however, is the fact that there is no need for the analyst to stipulate the functional form prior to the analysis; the ANN will find it automatically. This can be of vital advantage to the policy researcher, since policy decisions typically must be made even though well-formed theories relevant to the decision do not exist. (Unlike the regression model, however, the neural network is unable to represent the results as an equation with parameters.)

Although variations are common, by far the most typical supervised ANN consists of three layers of nodes: an input layer, one or more middle or hidden layers, 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" that are identical in form to the cases found 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 *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:

$$\text{net}_i = \sum_{j=1}^{n} a_j w_{ij}, \tag{1}$$

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

A commonly used activation function is the logistic function, sometimes referred to as a sigmoid function:

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

where a_{pj} is the activation of the jth node for the pth pattern, and net_{pj} is the net input to the jth node for the pth 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 that 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 in turn can 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, of course, can 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 (Rumelhart, Hinton, & Williams, 1988; Werbos, 1974).

Adjusting the weights to fit the second case generally results in worsening 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 that can provide an optimal mapping of input values onto output values for these cases. 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 predictions about the values of the output characteristics (dependent variables) for those novel cases. Other methods for adjusting the connection strengths (Jacobs, 1988; Rigler, Irvine, & Vogl, 1991; Tolleneare, 1990; van Ooyen & Niehuis, 1992; Vogl, Mangis, Rigler, Zink, & Alkon, 1988) yield equivalent results, often with increased speed, but this *back propagation* algorithm is by far the most commonly used in applied settings.

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.

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 nonlinear and nonmonotonic, the system is highly nonlinear, and several authors (e.g., Cybenko, 1989; Funahashi, 1989; Kurkova', 1992) 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 ANNs 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 ANNs can model any system that can be modeled in the regression framework. Many vendors of supervised ANN software recommend them as a replacement for the general linear regression model and its more sophisticated relatives. A typical claim is "the end result is a marked improvement over conventional methods such as regression analysis, clustering, unequal proportion techniques, or other linear analysis" (Neuralware, 1992, p.2). Advocates of the supervised ANN model claim several significant advantages over the generalized regression model:

1. There is no need to prespecify the correct functional form that relates input to output variables; the network automatically finds the optimal form.
2. All possible interaction terms are automatically considered.
3. ANNs work well with noisy and incomplete data.
4. ANNs are well suited for real-time operation, adjusting their structure as new cases are added.
5. ANNs generally fit the same variables to the same data better than best-fitting regression models.

In spite of the close relationship between self-organizing ANNs and regression, published studies of actual empirical comparisons are scarce. Dispenza and Dasgupta (1992) compared the results of a back-propagation ANN, logistic regression, and linear discriminant analysis in four cases. Dispenza and Dasgupta had available data from a 1,000-case mailed survey, which provided information on each respondent's investment savings, willingness to take financial risks, ownership of investment products, and opinions toward financial product providers, along with unspecified demographic information. These authors were able to match the respondents to the Equifax Marketing Consumer Database[1] to provide third-party demographics and characteristics.

The authors tried to classify (a) respondents from the survey and (b) individuals from the Consumer Database on two dependent dichotomous variables (willingness to take financial risks and willingness to purchase a specific investment product). Independent variables were taken from either the survey or the data base. Within each cell of the resulting 2 × 2 design, all three methods—logistic regression, back propagation, and linear discriminant analysis—were used.

Table 1 shows a somewhat oversimplified summary of the overall re-

[1] This data base is the property of Equifax/Elrick & Lavidge, Inc.

Table 1: Percentage of Cases Classified Correctly by Three Multivariate Models in Four Conditions

Dependent variable	Risk taker/ not risk taker		Bought/ did not buy product	
Independent variables	Survey	Data base	Survey	Data base
Logistic regression	94	100	100	96
Linear discriminant analysis	94	87	96	89
Back propagation	100	82	98	100

Note: Data from Dispenza and Dasgupta (1992).

sults. By and large, the back-propagation model and the logistic regression model perform about equally well, and both outperform the linear discriminant model. Both the logistic regression model and the back-propagation model are influenced by normalization decisions, and these are not described in the Dispenza and Dasgupta paper, so some caution should be used in evaluating these results. Moreover, the simple binary classification problem studied by these authors is very simple and very well suited to logistic regression; it would be reasonable to expect the back-propagation model to outperform the regression model in more difficult cases, particularly those with continuous multiple dependent variables and nonlinear and perhaps nonmonotone functional relations.

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 (Carpenter & Grossberg, 1987; Kohonen, 1982, 1986; Rumelhart & Zipser, 1987). 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 (Hebb, 1949; Kosko, 1989). 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 that 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 activating parts of that pattern.

A self-organizing ANN is typically used for 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 ANNs 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 (Woelfel, Stoyanoff, & Danielson, 1992), an unsupervised neural network that 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 that fall below an arbitrary, user-set frequency of occurrence. For each remaining word, an artificial neuron is constructed that 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 that represents that word is activated (its 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 whether the network has been set to cycle, connection weights among the neurons that represent the words are modified according to the rule

$$w_{ij} = w_{ij} + a_i a_j b, \tag{3}$$

where b represents a "learning constant," and a_i and a_j represent the activations of the ith and jth nodes, respectively. 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 that behave similarly will be increasingly strongly connected.* This results in a square matrix that resembles a correlation matrix, although due to the cycling, which results in spreading activation, 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 be tightly connected to the third word, and so on.

This matrix (called a "weight input network" or WIN matrix) may be considered a kind of similarities matrix, and can 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"

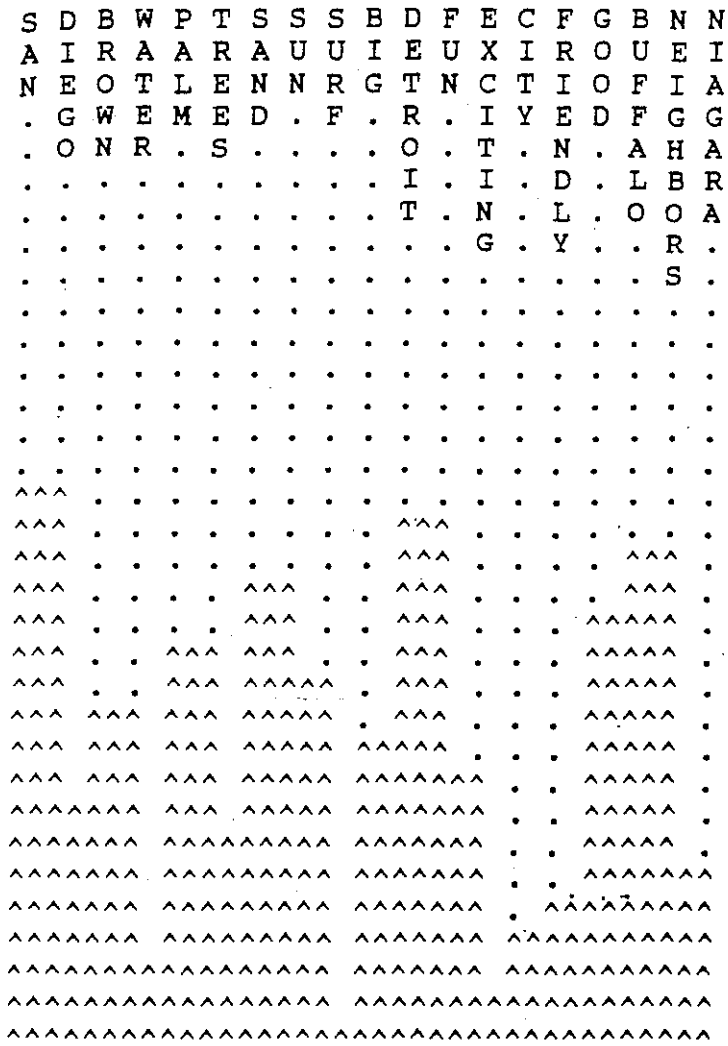

Figure 3. Dendogram produced by CATPAC Shading indicates degree of clustering. Higher shading under neighboring words indicates stronger clustering.

or graphic representation of the clusters in the text. Typically, CATPAC is used to analyze interview data, focus group discussions, texts, and the like. Figure 3 is an example of a CATPAC analysis of three interviews about Buffalo, Detroit, and San Diego. Respondents were asked to describe each of the cities. While the sample is clearly insufficient for generalizations, the extremely small number of cases provides a good understanding of the ability of CATPAC to detect patterns without much data. The text of the three interviews is presented in the Appendix.

CATPAC produces a "dendogram" (see Figure 3), which arrays the words found in the text across the top of the page, and shows how words

cluster together by darkening the area under words that cluster together. (Somewhat oversimplified, words that occurred close to each other in the text are close to each other in the dendogram; how close they are is indicated by the height of the shadowing beneath the words.) Following this rule, at the highest level, CATPAC finds the cluster [*San, Diego*]. At the next level, it finds a new cluster [*Detroit, fun*]. On the next level the cluster [*Buffalo, neighbors*] emerges (Buffalo is called "The City of Good Neighbors"), and, one level down, the cluster [*sand, sun*] begins. Down one more level, the word *good* enters the [*Buffalo, friendly, good*] cluster, then a [*palm, trees*] cluster arises, and next a [*brown, water*] cluster emerges after which *surf* joins the [*sand, sun, surf*] cluster. These three clusters eventually merge with the [*San, Diego*] cluster to result in the cluster [*San, Diego, brown, water, palm, trees, sand, sun, surf*]. The Detroit cluster becomes finally [*big, Detroit, fun, exciting*], while the Buffalo cluster ultimately becomes [*city, friendly, good, Buffalo, neighbors, Niagara*].

Detailed description of the CATPAC dendogram is tedious and can detract from understanding the usefulness of CATPAC. What is significant about this technique is the fact that it read the raw text of three interviews and within seconds recognized that the respondents related Buffalo to good, neighbors, friendly, and Niagara; Detroit to big, city, exciting, and fun; and San Diego to sun, sand, surf, palm trees, water, and brown. Neural networks' capacity to read vast quantities of unprepared text and provide a brief and useful synopsis of their main concepts provides policy researchers with a useful tool not available with conventional technology.

Hybrid Models

ANNs 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 ANNs can be made fundamentally fuzzy in their own right). A model familiar to communication researchers, the Galileo model (Woelfel & Fink, 1980), uses neural networks in which connection weights are measured directly rather than established by the ANN itself. Similarly, combinations of unsupervised ANNs with directly measured communication data are useful in analyzing social networks more commonly studied by conventional software such as NEGOPY (Richards, 1989) and UCINET (Burgatti, Everett, & Freeman, 1992). CATPAC itself is a hybrid program, using conventional parsing and word-counting techniques, along with a conventional cluster-analysis routine, although the similarity matrix among words in the document is computed by a neural network.

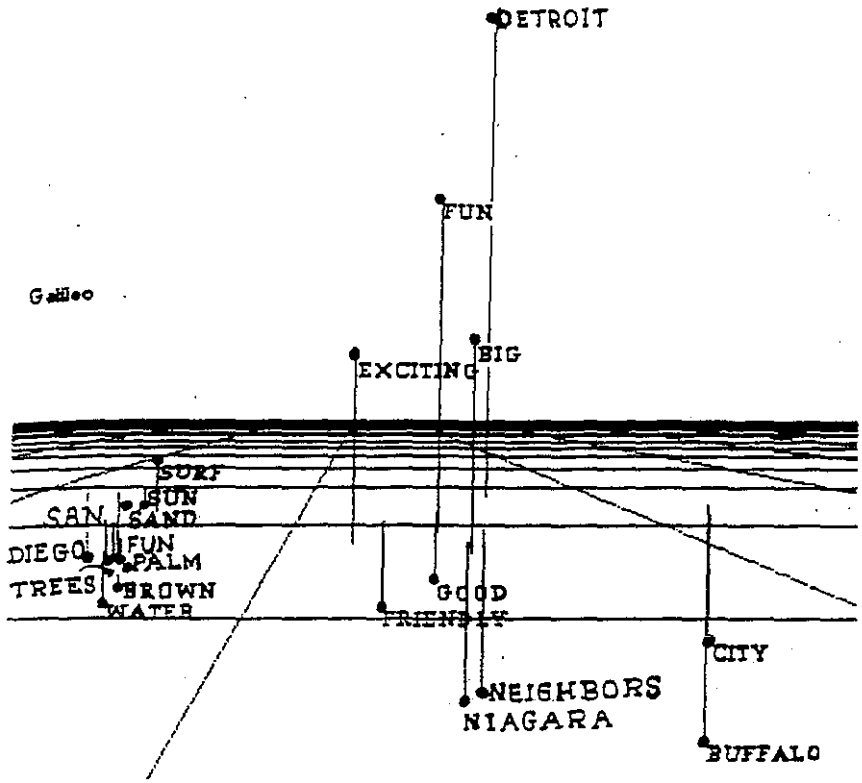

Figure 4. Galileo perceptual map showing similarity relations among words describing Buffalo, Detroit, and San Diego. The similarities data from which the map was produced were generated by CATPAC from the interview texts shown in the Appendix.

Figure 4 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.[2] (For those unfamiliar with perceptual mapping, Figure 4 is arrayed such that concepts that are similar in meaning or that "go together" are located near each other.) Based on this rule, the lower right shows clearly the Buffalo cluster: *Buffalo, friendly, city, good, neighbors,* and *Niagara*. Similarly, at the center of the figure lies the Detroit cluster: *Detroit, big, exciting, fun*. At the left of the figure

[2] While research on what we now call neural networks has a history that goes back well over 100 years, the conventional term "neural networks" emerged only in the late 1980s to describe this area of research. Many technologies and techniques now called neural networks were once called something else. In the most general sense, any abstract entity that could be described by a rectangular array of numbers could be considered a set of interconnected nodes and hence a neural network. A hybrid technology like CATPAC–GALILEO capitalizes on the fact that the square matrix of connection weights (i.e., the weight input matrix) produced by CATPAC's self-organizing neural algorithm is mathematically indistinguishable from the square matrix of real numbers in the centroid scalar products matrix out of which Galileo perceptual maps are made. The *condition* of a neural network at any given moment can thus be displayed as a perceptual map using the GALILEO algorithm.

can be found the San Diego cluster: *San, Diego, sun, sand, surf, palm, trees, brown, water.*

Figure 4 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; even a short while ago few analysts would have considered it likely that a precise perceptual map could be formed from the few lines of text in the Appendix.

Value of the Model

Artificial neural networks show great promise for policy researchers in both the public and private sectors. One way in which neural technology can serve the policy researcher is in providing replacements for technologies already in use, such as regression modeling. While regression modeling is unlikely to be replaced entirely by supervised neural networks, many policy circumstances favor the neural model over the regression model. Policy researchers might consider supervised neural networks instead of regression models when one or more of the following circumstances apply: (a) The problem to be addressed is not well understood theoretically; (b) the variables to be included in the model are poorly or incompletely measured, or are noisy; (c) quick solutions are needed, which preclude careful design and execution of well-planned studies, so policymakers must instead be guided by data at hand; (d) highly skilled multivariate analysts are unavailable for designing complicated causal models; or (e) advice and guidance are needed in real time on a continuous basis.

There are tasks that cannot, in principle, be done by either ANNs or regression-based models. But, circumstances such as those listed above point to tasks that ANNs can do well, when real-world conditions rule out well-constructed regression-based models; these circumstances face the policymaker every day.

Beyond supervised ANNs, self-organizing neural networks provide even more interesting possibilities for the policy researcher. Even at their present early stage of development, they provide capabilities that do not exist in the most highly developed conventional multivariate analytic procedures. The ability of self-organizing ANNs like CATPAC to read and "understand" text is particularly valuable. Policy researchers in particular must make do with data that are primarily and overwhelmingly textual. 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, marketing, and political campaigning.

The ability of ANNs to be used in combination with conventional methods in hybrid models allows users a familiar format for data entry and presentation of results. And the ability of ANNs to grasp patterns from incomplete, noisy data opens up possibilities for rigorous analysis previously unavailable with conventional techniques. Equally important, the ability of self-organizing models to deal easily with textual data provides precise, quantitative analysis for the most widely available source of policy-related data. Most policy researchers and clients are faced with vaguely defined questions that can best be approached by open-ended questions, and even when structured scales are used, these are often supplemented by open-ended questions. Lieman (1992) has investigated the effects of alternative parameter choices for CATPAC on data of this type, and shows useful results.

A typical and important policy-related use of CATPAC is investigation of the public acceptability of electric-powered vehicles (Wassman, 1992). The decision to produce and distribute electric-powered vehicles on a large scale is a policy decision of major consequence to the nation and to automobile manufacturers. An important aspect of the decision must be based on the receptiveness of potential consumers. Wassman analyzed verbatim responses of visitors to auto shows in which electric-vehicle exhibits were displayed to determine the clusters of attributes potential buyers used to define electric vehicles and determine their acceptability. Automobile executives use these data to determine whether the attributes required by consumers to make vehicles saleable are within the range of technical feasibility.

An early example of neural technology, mainframe versions of CATPAC, are already large enough and fast enough to scan large electronic data bases automatically and continually. ANNs promise widespread deployment of automated text-analysis programs to scan the huge, worldwide electronic data base and provide information related to important policy decisions.

By far the most distinctive feature of all ANNs, whether supervised or self-organizing, is their learning capability. All ANNs continuously reconfigure themselves as they run, either to conform to a preset pattern, as in supervised ANNs, or to become increasingly "similar" to the input information passing through the network, as in self-organizing models. Due to ANNs' generalizing ability, information learned about one or more patterns of information applies immediately to other related patterns. As a result, ANNs are particularly well suited to on-line operation, taking in additional cases or information, even in real time, adjusting to the new information, and outputting information continually. Insofar as policy research entails defining and adjusting plans to guide governments, organizations, or other entities through a continuously changing environment, neural technology is particularly well suited to the needs of policy researchers.

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 ANNs' ability to successfully use noisy and incomplete data sets, analysts can be misled into overanalyzing poor-quality or useless data. A back-propagation network, for example, cannot develop a useful model to predict the values of a given set of variables if the variables that 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, every text may not 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 constitute a useful and powerful technology for the policy researcher.

Appendix

Interview Transcripts (each paragraph indicates one response):

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 it's 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.

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.

San Diego is warm, even too hot sometimes. It's brown, and water is scarce. There
a water shortage. Usually a 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.

Detroit is scary. It's big and exciting, but it's scary. There's the Tigers, and a great city.

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.

San Diego! Palm trees! Beaches! Sand! Sun! Surf! Yippee!

Detroit is excellent. Very exciting. A lot of fun. Somewhat dangerous, but worth it. Fun. A big city, very big. Action.

Buffalo is the friendly city. Buffalo is the city of good neighbors. Buffa-

lo and Niagara Falls are neighbors. A party community. Laid back. A good time.

San Diego is sun, sand, surf, and sex. Palm trees. Brown, drought, expensive.

References

Armstrong, G. B., & Bauman, I. (1993). A mathematical model for intercultural interactions: Making connections and increasing harmony. *Journal of Communication, 43*(1), 81–100.

Burgatti, S., Everett, M., & Freeman, L. (1992). *UCINET IV network analysis software user's guide*. Columbia, SC: Analytic Technologies.

Carpenter, G. A., & Grossberg, S. (1987). ART2: Self-organization of stable category recognition codes for analog input patterns. *Applied Optics, 26*, 4919–4930.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems, 2*, 303–314.

Dispenza, G., & Dasgupta, C. (1992, June). *Comparisons between neural network models and multivariate statistical techniques for marketing research analysis*. Paper presented at the Advanced Research Techniques Forum of the American Marketing Association, Lake Tahoe, NV.

Funahashi, K. (1989). On the approximate realization of continuous mappings by neural networks. *Neural Networks, 2*, 183–192.

Hebb, D. O. (1949). *The organization of behavior*. New York: Wiley.

Jacobs, R. A. (1988). Increased rates of convergence through learning rate adaption. *Neural Networks, 1*, 295–307.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics, 43*, 59–69.

Kohonen, T. (1986). *Self-organization and associative memory*. Berlin: Springer.

Kosko, B. (1989, June). *Unsupervised learning in noise*. Paper presented at the International Joint Conference on Neural Networks, Washington, DC.

Kurkova', V. (1992). Kolmogorov's theorem and multilevel neural networks. *Neural Networks, 5*, 501–506.

Leiman, J. (1992, June). *Using a neural network (CATPAC) to map the conceptual structure of verbatim responses*. Paper presented at the Advanced Research Techniques Forum of the American Marketing Association, Lake Tahoe, NV.

Neuralware. (1992). *Applying neural networks to target marketing*. Pittsburgh, PA: Neuralware, Inc.

Richards, W. D. (1989). *The NEGOPY network analysis program*. Barnaby, BC: Department of Communication, Simon Fraser University.

Rigler, A. K., Irvine, J. M., & Vogl, T. P. (1991). Rescaling of variables in back propagation learning. *Neural Networks, 4*, 225–229.

Rumelhart, D., Hinton, G., & Williams, R. (1988). Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland, & the PDP Research Group (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition: Vol. 1. Foundations*, (pp. 318–362). Cambridge, MA: MIT Press.

Rumelhart, D., & Zipser, D. (1987). Feature discovery by competitive learning. In D. E. Rumelhart, J. L. McClelland, & the PDP Research Group (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition: Vol. 1. Foundations* (pp. 151–393). Cambridge, MA: MIT Press.

Tolleneare, T. (1990). SuperSAB: Fast adaptive back propagation with good scaling properties. *Neural Networks, 3,* 561–573.

van Ooyen, A., & Niehuis, B. (1992). Improving the convergence of the back-propagation algorithm. *Neural Networks, 5,* 465–471.

Vogl, T. P., Mangis, J. K., Rigler, A. K., Zink, W. T., & Alkon, D. L. (1988). Accelerating the convergence of back-propagation with good scaling properties. *Neural Networks, 3,* 561–573.

Wassman, D. A. (1992, June). *Using CATPAC to read qualitative data.* Paper presented at the Advanced Research Techniques Forum of the American Marketing Association, Lake Tahoe, NV.

Werbos, P. (1974). *Beyond regression: New tools for prediction and analysis in the behavioral sciences.* Unpublished doctoral dissertation, Harvard University.

Woelfel, J., & Fink., E. L. (1980). *The measurement of communication processes: Galileo theory and method.* New York: Academic Press.

Woelfel, J., Stoyanoff, N., & Danielsen, S. (1992). *CATPAC user manual.* Troy, NY: Terra Research and Computing Co.