# ARTIFICIAL NEURAL NETWORKS AND CRIME MAPPING

by

**Andreas M. Olligschlaeger**
**Carnegie Mellon University**

***Abstract:*** *The recent change in emphasis from reactive to proactive law enforcement, as evidenced by such concepts as community-oriented policing, has resulted in the need for tools to support these efforts. While geographic information systems (GISs) have been very successful at tracking criminal activity, proactive law enforcement requires systems that anticipate the emergence of criminal activity. One such system under development at Carnegie Mellon University and the Pittsburgh (PA) Bureau of Police is an early warning system that incorporates a GIS, previously developed to track criminal activity, and a relatively new technology — artificial neural networks — to predict the emergence or "flare ups" of drug hot-spot areas. The system obtains its input from cell-aggregated GIS-based data, processes the data with a previously trained artificial neural network and outputs the results to a choropleth map indicating those areas for which the network has predicted a relatively high number of 911 calls for service for drugs. The focus of this paper is to describe how the early warning system was developed, and to explain some of the underlying theory behind neural networks. In addition, the performance of the network is compared to some of the more traditional geographic forecasting methods.*

## INTRODUCTION

Computerized mapping has come a long way since the first mainframe applications (such as SYMAP) produced "maps" on high-speed line printers that shaded choropleth (or thematic) maps using differ -

Address correspondence to: Andreas M. Olligschlaeger, H. John Heinz III School of Public Policy and Management, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213.

ent combinations of ASCII characters printed one on top of the other. However, the concept of geographic information systems (GIS) and all that it implies is still relatively new to researchers and practitioners alike. In part this is because people are only just beginning to realize that GIS can be much more than merely an automated mapping system, and in part because the quality of spatial data contained in a GIS can be so high that researchers are continuously finding new uses for it. In fact, it can be a one-stop shopping center for information. GIS is indeed an exciting field, especially within the context of law enforcement.

There is no doubt that GISs have proven themselves to be an invaluable tool for law enforcement. Examples of successful uses of GIS from a research perspective abound for a variety of law enforcement support functions, such as measuring the geographic displacement of drug offenders (see, for example, Green, 1993), monitoring the effects of law enforcement strategies on nuisance bar activity (Cohen et al., 1993), and point pattern analysis of crime locations (Canter, 1993). Other examples of more general purpose crime mapping systems for law enforcement include the Drug Market Analysis Program (DMAP) effort undertaken in Jersey City, Hartford, San Diego, Pittsburgh and Kansas City (McEwen and Taxman, 1994; Maltz, 1993), and PA-LEGIS (Pennsylvania Law Enforcement Geographic Information System), an integrated GIS and police records management system developed for smaller police departments (Bookser, 1991).

The impact of GIS in law enforcement is further illustrated by the fact that today virtually all commercially available police record management and emergency operations systems include a GIS component. As police organizations automate their operations and implement more modern computer systems, taking advantage of advances in information science such as open-architecture database systems, enterprise-wide computer applications and ever-increasing microprocessor and network speeds, more and more information will become available to police officers at the click of a mouse. Moreover, all of this information will be linked together from various sources and organized in ways that were previously unheard of. Police investigators will likely find this wealth of information a boon to their work, but crime analysts may well find themselves faced with information overload.

At the same time that police departments are making greater use of computer technology, they are also undergoing a change in law enforcement philosophy. Evidence of this change can be seen in the fact that many departments are implementing community-oriented policing (COP) in an effort to emphasize proactive rather than reactive law enforcement. While the concept of COP is certainly not new (for a review of early initiatives, see Trojanowicz, 1986), the way in which information is utilized in COP has changed over the years. In many cities, desktop personal computers have replaced the daily log for foot patrol officers, and in some cities the time-honored tradition of a notebook and pencil has given way to hand-held, pen-based mobile computers.

The current decade has shown that providing police officers with automated tools to collect and access information is not a problem. While they will undoubtedly become more sophisticated, such tools already exist in many police departments. Given the right amount of funding, careful planning and proper implementation, there are abundant technical solutions available to those departments wishing to automate information gathering and dissemination. Rather, the challenge in the next decade will be to provide crime analysts and police administrators with the tools to support proactive law enforcement efforts. This, in turn, will require the use of increasingly sophisticated data analysis methods and statistical techniques.

This chapter addresses one way in which GISs can provide such tools in the next decade. Specifically, it describes how another relatively new technology — artificial neural networks — was incorporated into a GIS-based early warning system for street-level drug markets implemented by the Pittsburgh Bureau of Police as part of the MAP. The resulting computer system predicts the emergence, or "flare-ups" of drug "hot-spot" areas. Input for the system is obtained from cell-aggregated GIS-based data that is processed by a previously trained artificial neural network. The output is then displayed on a choropleth map indicating those areas for which a relatively high number of 911 calls for service for drugs are anticipated.

The chapter proceeds in the following manner. First, the existing early warning system used by the Pittsburgh Bureau of Police, as well as some of the pitfalls encountered during its development, is described. Second, a brief overview is given of some of the more prevalent models used in space-time forecasting. The third section provides a brief overview of artificial neural networks. Fourth, an artificial neural network used for space-time forecasting is introduced. The fifth section presents a case study of predicting flare-ups of 911 calls for service for drugs, and describes how the artificial neural network was incorporated into the early warning system. In the conclusion the chapter is summarized and future work is outlined.

## THE PITTSBURGH DMAP SYSTEM

The Pittsburgh DMAP computer system is a fully integrated GIS linking a variety of sources of information into a single user interface. Specifically, these sources consist of 911 calls for service, police incident and arrest data, property tax and ownership information, liquor license data, and map sheet coverages from the Pittsburgh-Allegheny GIS (PAGIS). The map sheets include streets, property parcels, building footprints, parks, cemeteries, schools, public housing projects, Census tracts, neighborhoods and police zones.

The DMAP system is designed to provide support for both investigative and administrative police personnel. Users can query the system by any geographic area (such as neighborhood or census tract) or by a single address. Depending on the type of query users can, for instance, map out police or 911 incidents, determine property ownership, produce reports of incidents involving persons residing at a particular address, or map geographic displacement of criminal activities. One component of DMAP that is of particular interest to this chapter is an early warning system that allows administrative personnel to analyze crime pattern trends by geographic area.

Development of the early warning system was possible in part because of a key feature of GISs: the ability to associate xy coordinates with the address of an incident. This concept is known as "geocoding." During geocoding, an address is matched against a known data set (also referred to as an "address coverage") containing the xy coordinates of all addresses located within a particular area. There are two types of address coverages: point-based and line-based. Line-based (or street) address coverages are by far the most commonly used, in part because they are much easier to construct and maintain and in part because they are available at low cost from a variety of sources. Street-address coverages consist of nodes (points in space that can represent either curves or intersections) and arcs. For example, an arc connecting two intersections can represent a street block. Each arc has a number of attributes associated with it, including the street name, street type, street direction, and the left and right beginning and ending address. Geocoding using line-based address coverages proceeds in the following manner: the system locates the arc that not only shares the same street name, type and direction of the address to be matched, but where the street number also falls into the left or right beginning and ending address range. Once the arc has been located, the xy coordinate of the address is determined by interpolation between the beginning and ending node of the arc. For example, if the left address range of an arc is between 100 and

198, and the street number of the address to be matched is 150, then the xy coordinate of the matched address will be halfway between the two nodes.

Point-based address coverages are different in that the xy coordinates of an address are determined not by the street the address is located on, but some other key identifier such as a building footprint or the center of a property parcel. Because one building or property parcel may contain several different addresses, one or more addresses can share the same xy coordinate. In order to match an address, all the system has to do is find an exact match in the address coverage and retrieve the corresponding xy coordinate. Geocoding using point-based address coverages is therefore much faster because the algorithm requires fewer steps. The implementation of a point-based address coverage is more costly and time-consuming than line-based address coverages, but yields a far more accurate data set for geocoding. The Pittsburgh DMAP system uses a point-based address coverage where the xy coordinates of addresses were determined using the geographic centers of property parcels. For each parcel the lot and block number was related to the property tax file, resulting in one or more valid addresses for each parcel. These addresses, along with the xy coordinate, were then added to the address coverage. The final result contains one record for each valid address in the City of Pittsburgh.

Problems arise during geocoding when an address cannot be located in the address coverage. If this is the case, the data are lost and cannot be displayed or located on a map. The two most common causes of unmatched addresses are inaccurate address coverages and inconsistently spelled street names. In Pittsburgh both of these causes were encountered. Most GIS software packages have a component that allows users to manually select an address from a list of candidates in the event that an exact match cannot be found. This is fine when only a few addresses are to be matched, but too time-consuming when hundreds of thousands of addresses are to be matched, as is the case in DMAP. As a result, the Pittsburgh DMAP system utilizes its own geocoding algorithm where addresses are preprocessed to eliminate spelling inconsistencies and candidates are automatically selected according to a specific set of rules. Because of this algorithm DMAP successfully matches about 97% of all addresses.

The accuracy of geocoding in DMAP means that little information is lost in the geocoding process. In addition, the geocoding algorithm ensures that data from various otherwise incompatible sources can be consistently collated via the address, resulting in a large data set that can be aggregated according to any spatial unit.

The early warning system utilizes this data set to provide choropleth maps of changes in criminal activity based either on 911 calls for service or police incident data for user-selected crime types. Users choose the time period (for example, changes over the past month) as well as the areal unit by which they would like to aggregate the data. The areal units can either be pre-defined, such as by Census tracts or patrol sectors, or user-defined cells derived by overlaying a grid on the City of Pittsburgh. Finally, the user chooses the class intervals for shading the map. These are either calculated automatically according to standard deviations or frequencies, or defined by the user. The output of the early warning system is a citywide map that shades areas according to changes in criminal activity relative to other areas in the city. Those areas that experienced negative changes are shaded from light blue to deep blue depending on the relative magnitude of change ("cold areas"), whereas those with positive changes, or increases, are shaded from light red to deep red ("hot areas"). Users can also zoom in to hot areas to look at specific addresses or intersections responsible for the increase in criminal activity.

The early warning system described above was a significant step ahead in terms of providing police administrators with an automated tool to analyze changes in spatial crime patterns on a citywide basis. However, the system does not provide for any predictive capabilities. As mentioned earlier, police administrators also have a need for tools allowing them to anticipate changes in criminal activity. Thus, the following pages outline how the early warning system was modified to provide space-time forecasts of changes in 911 calls for service for drugs.

## SPATIAL FORECASTING MODELS AND METHODS

Somewhat surprisingly, it was not until the early 1950s that the first models were developed that took spatial context into consideration, even though temporal context, or time series modeling, was introduced as early as the 1920s (Cliff et al., 1975). Since then geographers and regional scientists have devised a variety of techniques for space-time forecasting, many of which are spatial extensions of time-series models. Anselin (1988) notes that the regional science and geography literature provides much evidence that the effects of space are heterogeneous rather than homogeneous. As a result, he argues, modeling strategies have to account for regional, or local, features.

A univariate example of space-time models is the Space-Time Autoregressive model, or STAR (Tobler, 1969). STAR, an extension of the purely temporal autoregressive model originating with Box and Jenkins (1970), assumes that the influence of neighboring observations declines with distance from the current observation according to a set of pre-defined spatial weights. A related model is the Space-Time Autoregressive Integrated Moving Average model, which incorporates repeated differencing for trend elimination and the exponential smoothing model (Cliff et al., 1975).

One of the most basic multivariate models that takes into account local context is the spatially varying parameter model, defined as follows:

$$Y_i = \sum_{k=0}^{p} \beta_{ki} x_{ki} + \varepsilon_i, i \in C \qquad (1)$$

where C is an index for the space-time context of parameter variation, i is an index in C, $Y_i$ is the independent variable at observation i, $\beta_{ik}$ is the parameter for the $k_{th}$ independent variable for observation I (k = 0,1,2...p), and $\varepsilon_i$ is the error term for observation i. Implementing this model in unconstrained form is impossible because the number of parameters increases with the number of observations (Anselin, 1988). Consequently, researchers have devised a number of strategies to counter this problem.

Two such methods are locally weighted regression (see, for example, Cassetti, 1982, and Cleveland and Devlin, 1988) and Kriging (David, 1977; Haining, 1990). Locally weighted regression techniques require that the weights are specified a priori. This often occurs via trial and error as an attempt is made to see which set of weights produces the best fit of the dependent variable. Kriging — or, in the multivariate case, cokriging — uses an empirically estimated function, called a variogram, to determine the spatial weighting of data observations. Both methods assume that the influence of other observations declines with distance from the current observation.

Model (1) is also often used for exploratory data analysis (Gorr and Olligschlaeger, 1994): maps of residuals can show undetected spatial heterogeneity suggesting additional theory or model structure, and maps of estimated spatially varying parameters are useful in determining the functional form of parameter variation. This type of exploratory data analysis is also used in expansion modeling. One example is a stepwise regression model using polynomial or other functions of time and space coordinates that interact with an initial model's variables (see, for example, Cassetti, 1982 and Cassetti and Jones, 1992).

Two final examples of estimating spatially varying parameter models are spatial adaptive filtering (SAF) and weighted spatial adaptive filtering (WSAF), both of which are based on adaptive filtering originating with Widrow and Hoff (1960). Foster and Gorr (1986) introduced SAF as an extension of multivariate damped negative feedback estimation by using a heuristic approach to optimizing individual damping factors for each $\beta_k$ in model (1). WSAF was introduced by Gorr and Olligschlaeger (1994) as an extension of SAF. It incorporates a pattern recognizer into SAF that reduces an inherent bias created by applying equal weights to feedback signals from neighboring observations. Based on the magnitude of forecast errors using the pk of neighboring observations, WSAF automatically assigns appropriate weights to feedback signals: those observations with small forecast errors receive relatively large weights, whereas those with larger errors receive smaller weights. The resulting weighting scheme is similar to those used in time-series combination forecasting (see, for example, Bates and Granger, 1969 and Clemen, 1989).

At this point it is important to note that both SAF and WSAF use a feedback scheme very similar to the Widrow-Hoff (1960) rule and the perceptron convergence procedure originating with Minsky and Papert (1969). In addition, the scheme resembles the generalized Delta Rule employed in estimating feed-forward artificial neural networks with backpropagation (Rumelhart and McClelland, 1988). The significance of this will become apparent in a later section.

All of the examples discussed above assume that the distribution of data fits a certain functional form. As practitioners know, this is only very rarely if ever the case, especially when it comes to crime data. Second, with the exception of WSAF and Kriging, the functional form of the spatial variation of parameters as well as the weights defining the influence of neighboring observations must be specified a priori. From a practical viewpoint, this requires much exploratory analysis and experimentation with different model specifications in order to arrive at a model with good predictive capabilities. WSAF, while generally more efficient at automatically detecting spatial patterns, is very sensitive to the heuristics used in determining optimal damping factors and sometimes tends to overfit the model. In addition, it is more of an exploratory technique rather than a predictive one.

Law enforcement practitioners rarely have the time or the expertise to arrive at predictions of criminal activity using the above models. Rather, they require techniques that provide reasonably accurate forecasts of crime patterns with a minimum of user intervention, with results being displayed in such a way that they are easy to understand and intuitive. In addition, the forecasting models used to make predictions must be adaptive, i.e., they must be able to automatically recognize and adapt to changes in the functional form underlying spatio-temporal crime patterns. For example, drug dealers are constantly changing their method of operation in response to various law enforcement efforts as well as changes in the nature of their business. This, in turn, leads to changes in behavior as evidenced, for example, by the geographic displacement of drug markets. One technique that could accomplish the above-mentioned goals are artificial neural networks.

## A BRIEF OVERVIEW OF ARTIFICIAL NEURAL NETWORKS

As with many newer technologies, there is still much confusion and skepticism among applied researchers over what artificial neural networks are and how useful they can be. There is confusion because there are so many different types of artificial neural networks (also known as "connectionist" or "parallel distributed processing" models) and because little, if anything is known about their statistical properties. On the one hand, artificial neural networks seem to be able to do things that no other statistical method can do, but on the other no one quite understands how and why they can do it. Skeptics argue that, like expert systems, neural networks are just another much-ballyhooed technology that may have some usefulness for a small set of well-defined applications but that they are not quite the greatest thing since sliced bread that their proponents make them out to be.

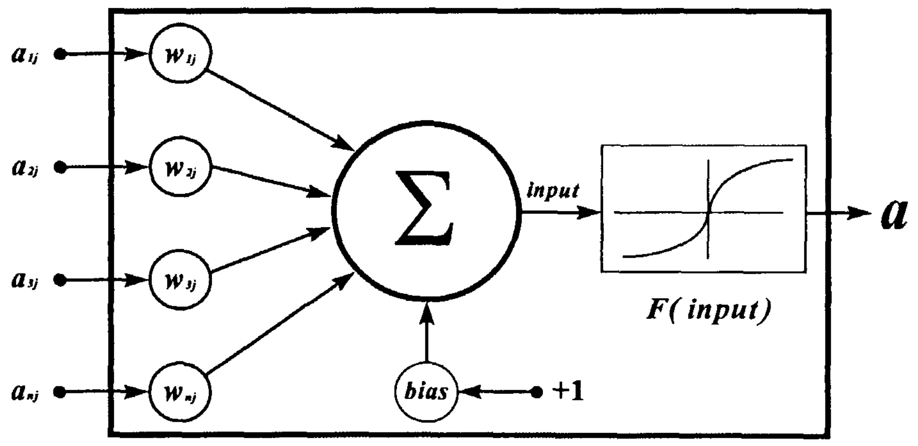
Exactly how widespread the use of neural networks will be remains to be seen. However, some very important developments have recently occurred in the field that show great promise for a wide range of potential applications. All indications are that research into neural networks is not going to be as short-lived as some people believe. The current interest in neural networks actually represents the second wave of research into the area. The first wave occurred after McCulloch and Pitts (1943) introduced a simple form of neurons as an attempt to emulate biological neurons. The authors envisioned using these mathematical neurons as components that could perform computational tasks in electronic circuits (Kroese and Van der Smagt, 1993). After two decades or so of research by a variety of authors, Minsky and Papert (1969) showed that perceptrons (their term for neural networks) had a number of deficiencies that prevented their use for general purposes. One of the most important deficiencies was the fact that they were unable to perform non-linear calculations. Consequently, many researchers left the field, although some key authors continued their work (Rumelhart and McClelland, 1988).

A resurgence of interest in the field occurred when Rumelhart and McClelland (1988) published their two volume set *Parallel Distributed Processing.* It consisted of a series of articles written by the PDP Research Group, a group of researchers who had continued to work on neural networks during the 1970s and 1980s. One of the chapters that is of particular interest to this paper outlined an approach incorporating multiple layers of neurons and nonlinear signal processing that allowed perceptron-like neural networks to estimate nonlinear functions. Since then there has been a remarkable amount of research in the field, including efforts at using neural networks for prediction (see, for example, Poli and Jones, 1994, and White, 1988).

Due to the large variety of artificial neural networks, it is beyond the scope of this paper to discuss even a representative sample (for an excellent introduction to the topic, see Rumelhart and McClelland, 1988 or Carpenter and Grossberg, 1991). However, one neural network architecture — feed-forward networks with backpropagation — will be outlined because it forms the basis of the neural net architecture of the early warning system described in the next section. Multi-layer feed-forward networks with backpropagation are probably the most studied type of artificial neural network, and are essentially a non-linear extension of Minsky and Papert's (1969) perceptrons.

Regardless of type, all artificial neural networks consist of a number of processing units that send signals to one another via a large number of weighted connections (Kroese and Van der Smagt, 1993). The main difference between network architectures is how signals are processed by each unit and the way in which the weights on each connection are updated. The internal representation of a processing unit in a backpropagation network is shown in Figure 1.

**Figure 1: A Neural Processing Unit**



Each processing unit, via weighted connections, receives input in the form of the outputs (activations) from the processing units in the previous layer. These outputs are multiplied by the weight on each connection, and, together with a bias, are summed to form the net input for the processing unit. More formally:

$$net_{pi} = \sum_j w_{ij} a_{pj} + \theta_{pi} \qquad (2)$$

where $net_i$ is the net input for processing unit i for input pattern p, $w_{ij}$ is the weight of the connection between processing unit i and processing unit j in the previous layer, $a_{pj}$ is the activation of unit j in the previous layer, and $\theta_{pi}$ is the bias associated with unit i. The function used to process the net input varies, but generally takes on a sigmoid functional form. The most commonly used function is the logistic function. It yields values in the range [0,1] and determines the activation of the processing unit at the next step as follows:

$$a_{ip} = \frac{1}{1+e^{-net_{ip}}} \qquad (3)$$

where $a_{ip}$ is the activation of unit i for input pattern p.

**Figure 2: A Multilayer Feed-forward Network**

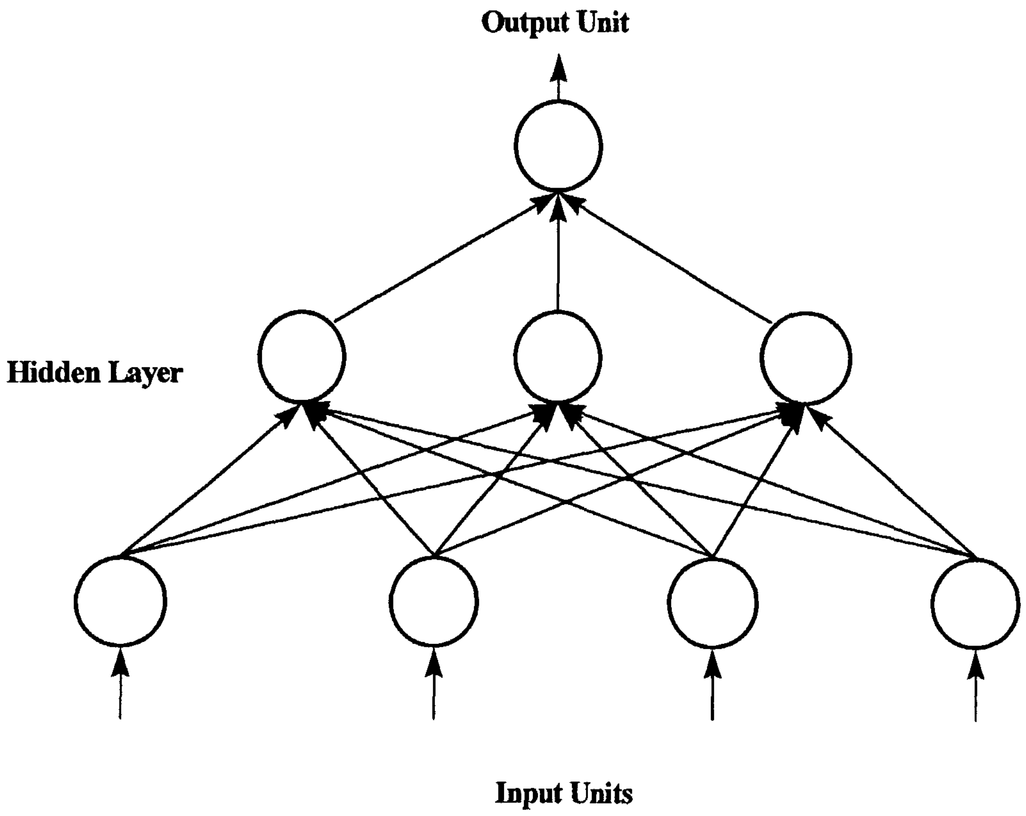


Figure 2 shows an example of a simple multi-layer neural network architecture. The network consists of three layers: input, output and hidden. Each layer consists of a number of units (neurons) that process information passed to them by units in the layer below, as described in (2) and (3). The input layer has four nodes and the hidden layer has three. In this example there is only one output node, although it possible to have more than one. In feed-forward networks information is input into each node, processed, and then passed on to each node in the layer above. In the case of an output node, information is simply passed out of the network.

The goal is to map the input units to a desired output similar to the way in which the dependent variable is a function of the independent variables in regression analysis. The difference is that regression analysis uses linear direct mapping whereas multi-layer feed-forward networks use non-linear indirect mapping. The hidden layer creates an internal representation of the patterns to be mapped. The internal representation is then mapped to the output unit. It is the hidden layer, along with the use of a non-linear activation function, that allows multi-layer networks to map far more complex functions than simple direct input-to-output unit mappings.

Feed-forward networks with backpropagation "learn" to map the input units to the output units by adjusting the weights on the connections in response to error signals transmitted back through the network. During training, the network is presented with each input pattern and computes the activation of the output unit(s) using the current network weight structure (the weights are initialized randomly prior to training). The difference between the output of the network and the target mapping constitutes the error signal. This signal is then propagated back through the network via the processing units, and their connections and the weights are updated. The goal is to continually update the weights until the sum of all error signals is minimized.

In backpropagation networks, weight updates are performed using the generalized delta rule derived by Rumelhart et al. (1988) from the perceptron convergence procedure originating with Minsky and Papert (1969). The latter, in turn, is a variation of the delta rule proposed by Widrow and Hoff (1960). The generalized delta rule can be summarized in three equations (for a formal derivation of the generalized delta rule, see Kroese and Van der Smagt, 1993 or Rumelhart et al., 1988). The first specifies that the weight change should be proportional to the product of the error signal sent to a receiving unit along a connection, and the activation of the sending unit. More formally,

$$\Delta_p w_{ij} = \eta \delta_{pj} a_{pi} \qquad (4)$$

where $\Delta_p w_{ij}$ is weight change for training pattern p and the connection between processing units i and j, $\delta_{pj}$ is the error signal sent to unit j, $a_{pi}$ is the activation of unit i for input pattern p, and $\eta$ is the "learning rate." The learning rate is usually some small number less than 1. The definition of the error signal differs between output units and hidden units. For output units using a logistic activation function, it is defined as:

$$\delta_{pj} = (t_{pj} - a_{pj})a_{pj}(1 - a_{pj}) \qquad (5)$$

where $t_{pj}$ is the target activation for the output unit. For hidden units the error signal is given by

$$\delta_{pj} = a_{pj}(1 - a_{pj})\sum_i \delta_{pi} w_{ij} \qquad (6)$$

The generalized delta rule implements a gradient descent in the error term. Training of the network proceeds by repeatedly presenting all input patterns and adjusting the weights until the sum of all errors is minimized, i.e., the network converges to a solution. In this respect, it is crucial to select a learning rate that during training will allow the network to iterate toward a true global minimum rather than getting stuck in local minima. Too-large learning rates can lead to oscillations between local minima, whereas small learning rates can require hundreds of thousands of iterations to converge. While it is theoretically possible that even with small learning rates the network will converge to a local minimum, empirical evidence suggests that this is rarely the case (Weiss and Kulikowski, 1991). One way to avoid this is to train the network several times with different random initializations of the weights, and to compare the results.

When implementing a backpropagation network, there are a number of factors to take into account. For example, the derivation of the generalized delta rule assumes that the network weights are updated each epoch, i.e., the error signal used in equation (4) is taken to be the sum of all error signals computed for each input pattern in the training data set. It has been shown that in some cases, updating the weights after each input pattern is presented can yield better results, i.e., the total mean squared error is smaller (Weiss and Kulikowski, 1991).

A further consideration is to determine how many processing units to include in the hidden layer. It appears that the number is directly related to the complexity of the function to be estimated (Hecht-Nielsen, 1990). Since in empirical applications the functional form of the input-to-output unit mapping is rarely known, the number of processing units has to be determined by trial and error. Too few hidden units may produce sub-optimal results, whereas too many may result in overfitting, i.e., the network simply "memorizes" all of the input and output patterns and yields zero error. While in some cases this may be desirable it is inappropriate for forecasting. One commonly used approach to determine the correct number of hidden units is to use about two thirds or three-quarters of the sample data for network training, and to keep adding hidden units until the network no longer generalizes well. Network generalization refers to how well the network performs with "unseen" data, i.e., data that was not used in training. If the performance is significantly worse, then the network has overfitted the data.

Depending on the number of weights present in the network and the type of architecture, feed-forward networks with backpropagation can require a large number of data points to train. If the sample size is not large enough, the network once again will simply "memorize" the input-to-output mappings and overfit the data. This can pose a problem for spatial data. Often only annual data are available at the Census tract level, so that even with a relatively large study area (say, on the order of 150 Census tracts) and 10 years worth of data the total number of observations would only be 1,500. For social science data this is typically not enough to train a neural network without overfitting the data. Depending on the complexity of the function to be estimated, neural networks can require tens of thousands of observations. As will be demonstrated in a later section, GISs allow for the production of very large data sets due to the nature in which data is stored and the manner in which it can be queried.

Depending on the sample size and the size of the network, the task of determining the ideal network structure for a specific application can prove to be very time-consuming. However, artificial intelligence algorithms do exist that can automatically design and optimize application-specific network structures. The genetic algorithm due to Holland (1975), which "evolves" the network architecture based on a "survival of the fittest" scheme, has been used quite successfully (see, for example, Harp et al., 1990 and Rogers, 1990). Regardless of how the network architecture is optimized, once this has been accomplished the network can readily adapt to changes in the functional form of the input-to-output unit mapping. In a time series, for exampie, the network is simply retrained as new data are collected using the previous network weight structure as initial weights.

Neural networks have one drawback in their potential as a spatial modeling tool, however. The algorithm described above requires that the number of input units is the same for all input-to-output mappings, i.e., there can be no missing variables in the sample data. Many spatial models (such as the Spatial Adaptive Filter discussed in the previous section) use a contiguity matrix to determine the neighborhood of the current observation. A contiguity matrix is an n by n dimensional matrix of ones and zeroes, where n is the total number of geographic regions in the study area. If a region is a neighbor of the current observation, then the matrix entry for those two regions has a value of one. If the two regions are not neighbors, the matrix entry has a value of zero. One of the most common geographic units used to estimate spatial models are Census tracts. The problem with using Census tracts as a geographic basis for spatial modeling using neural networks is that each Census tract has an inconsistent number of neighbors. Including neighboring observations as input units, as is most often the case in spatial modeling methods violates the rule that the number of input units must be the same for all observations. This problem will be addressed and a solution to it provided in the next section.

Feed-forward networks with backpropagation have some very important properties that make them suitable for the kind of modeling discussed in the previous section. First, they do not require that the functional form of the input-to-output unit mapping is specified a priori. The theoretical underpinnings of many statistical models, on the other hand, require that the data sample and the input-to-output mapping have a certain functional form. For example, in Kriging or Cokriging the functional form of parameter variation must be specified a priori. While there are certainly many types of statistical models with different assumptions as to the distribution of data or parameters, they often require much exploratory analysis and experimentation with different functional dependencies before satisfactory results are achieved.

A second important discovery with regard to the nature of backpropagation networks was made by Hornik et al. (1989), who proved mathematically that, provided sufficiently many hidden units are available, multi-layer feed-forward networks with backpropagation form a class of universal approximators capable of estimating any functional form to any desired degree of accuracy.

## A NEURAL NETWORK ARCHITECTURE FOR SPACE-TIME FORECASTS

In summarizing the previous sections it is possible to define the following requirements for a neural network architecture for space-time forecasting:

(1) The number of input units must be constant across all input-output mappings, i.e., each observation must have an equal number of observations in its neighborhood.

(2) The size of the data set must be sufficiently large to train the network without overfitting, or "memorizing," the input-to-output mappings.

(3) The number of hidden processing units must be large enough to facilitate an internal representation of the input-to-output mappings, but not too large to cause overfitting of the data.

(4) The network must be able to generalize, i.e., the weight structure arrived at during training should perform reasonably well on data that were not used during training.

(5) The network should not get stuck in a local minimum, i.e., should perform equally well with different random initializations of the connection weights.

The idea for a neural network architecture for space-time forecasting that satisfies the above criteria was originally conceived when the author was conducting research on cellular automata and chaos theory. Chaos theory, like artificial neural networks, has been the focus of much attention in recent years. It basically involves the study of phenomena or systems that are very sensitive to initial conditions. In chaotic systems or equations minute changes in parameters can result in very different outcomes — ranging from long-term stability to apparently random and unpredictable chaotic behavior (for an excellent introduction to chaos theory, see Schroeder, 1991). Some real-world examples of chaotic systems include weather patterns, neurological and cardiac activity, and the stock market. Chaos theory postulates that although chaotic systems seem to display totally random and unpredictable behavior, they are actually following strict mathematical rules that can be derived and studied (Pickover, 1990). These rules can range in sophistication from simple decision trees to complex non-linear functions.

Cellular automata are a specific type of chaotic system. They differ from other chaotic systems in that they act on discrete space or grids rather than a continuous medium such as a surface. In a cellular automata machine, each frame (representing all cells in a population) is replaced by a new one according to a specific "recipe," or rule, in the next epoch (Toffoli and Margolus, 1987). A key determinant of cellular automata rules is how each cell is influenced by neighboring cells. Consider the example given below.

**Figure 3: A Game of Life**

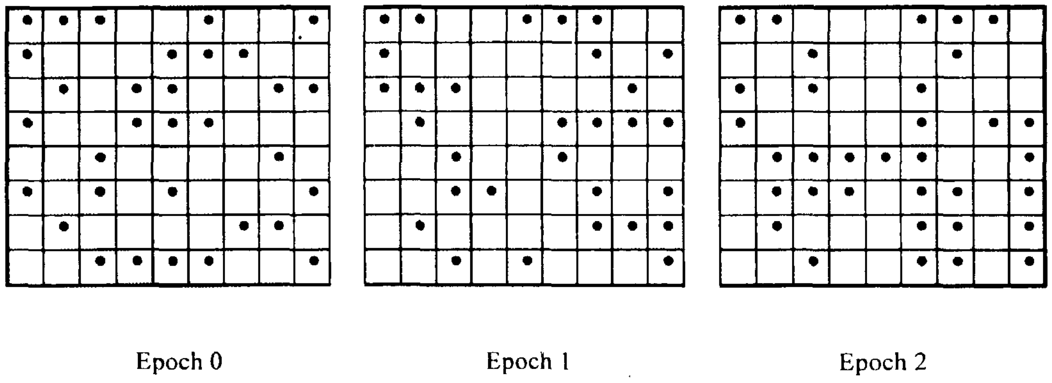


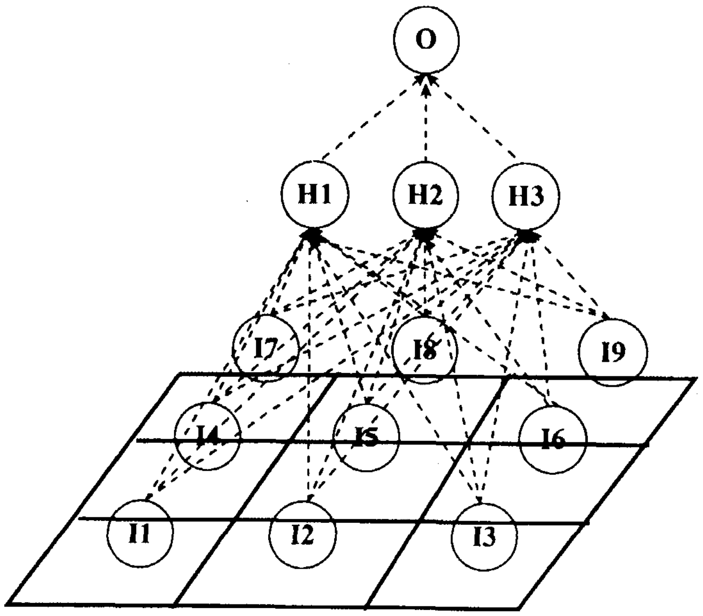
Figure 3 shows three epochs in the life of a population of cells occupying a 9 x 9 grid. An empty square represents a dead cell, whereas a dotted square represents a live cell. Upon initial examination of the three epochs it appears that cells are born and die randomly. As it turns out, however, this is not the case. The behavior of the cell population from one epoch to the other actually follows a very simple set of rules introduced by mathematician John Conway in his game of "life" (Gardner, 1970). Assuming that the "neighborhood" of a cell consists of all immediately adjacent cells, the rules are as follows:

(1) A live cell will only stay alive if it has two or three living neighbors. Otherwise it will die of either "overcrowding" or "loneliness."

(2) A dead cell will come to life if it has exactly three living neighbors. Otherwise it remains dead.

The above example can easily be expressed in terms of a neural network. Designate the target output of the network as the state of a cell in the next epoch (where the target output is one if the cell is alive and zero otherwise). The next step is to determine the input units. Since the rules of the game consider the state of all adjacent cells, as well of the current observation, we will need a total of nine input units representing a nine-cell square with the current observation being in the middle. If the current observation is located at the edge of the grid then all neighbors not on the grid — i.e., "missing" neighbors — are assumed to have a value of zero. Incidentally, this definition of a neighborhood is identical to that used in spatial adaptive filtering and weighted spatial adaptive filtering outlined earlier (see Gorr and Olligschlaeger, 1994). However, in the original Game of Life, cells are assumed to "wrap around" to the other edge of the grid (see Toffoli and Margolus, 1987). The mapping of the input-to-output units is a simple binary-to-binary mapping, i.e., a combination of nine zeros and ones map to either a single zero or a single one. If we assign three hidden units to the network the architecture would look like that in Figure 4.

**Figure 4: Game of Life Neural Network**



The neural net architecture in Figure 4 was in fact trained to "learn" the rules of the Game of Life when the author was first exploring the idea of using neural networks for space-time modeling. The algorithm used in training centered the network on each observation, and used the input mappings of the nine cells in the neighborhood to arrive at the output of the network for the current observation in the next epoch. The difference between the target output and the actual output of the network constituted the error signal. Thus, for each epoch (or time period) the number of patterns presented to the network is equal to the number of cells in the grid. The total number of training patterns is equal to the number of cells in the grid, multiplied by the number of epochs in the game presented to the network.

Presenting the results of the network outlined above would be beyond the scope of this paper. Suffice it to say, however, that the network was able to learn the rules of the Game of Life perfectly, i.e., without error. Moreover, the network performed flawlessly even when shown patterns that were not used during training: different random initializations of the grid cells had no effect on the performance of the network. In addition, it was able to predict successive generations of cells ad infinitum, requiring only the first epoch of randomly initialized cells to do so. This indicates that the network is very robust and able to generalize well, at least for this particular problem.

The rules of the Game of Life are certainly very simple. In the real world, rules governing space-time phenomena are far more complex. However, as demonstrated earlier, feed-forward networks with backpropagation are capable of "learning" extremely complex input-to-output mappings. In addition, the inputs or outputs need not be binary; they can take on any functional form (continuous or discontinuous) and do not have to fall into the [0,1] range. Finally, there is no reason why each cell in the neighborhood need only have one input unit. The rules can depend on more than one input that is unique to each cell.

It should therefore be possible to extend the neural network architecture outlined in Figure 4 in order to accommodate a variation of model (1). If for the moment we assume that we can obtain grid-based data for a geographical area, then using the same definition of a neighborhood as that for the Game of Life we can rewrite (1) as follows:

$$Y_i(t+1) = \sum_{k=0}^{p} \beta_k x_{ik}(t) + \sum_{j=0}^{8} \sum_{l=0}^{p} \beta_{jl} x_{jl}(t) + \varepsilon \qquad (7)$$

where p is the number of independent variables and j represents each observation (1,2...8) in the neighborhood. Model (7) assumes that the parameters are spatially constant. For the spatially varying parameter case we can write:

$$Y_i(t+1) = \sum_{k=0}^{p} \beta_{ik} x_{ik}(t) + \sum_{j=0}^{8} \sum_{l=0}^{p} \beta_{jlm} x_{jl}(t) + \varepsilon; i, m \in C$$

(8)

where i and m are indexes in C, the context of spatial parameter variation. Therefore, the only difference between (7) and (8) is that in model (8) each cell has its own set of parameters. Note that models (7) and (8) also assume that the dependent variable is a linear function of the independent variables. Feed-forward networks with back-propagation do not require this assumption since functional dependencies do not need to be specified a priori.

A neural network architecture to estimate models (7) and (8) would look very similar to that presented in Figure 4. The only difference is that the number of input units for each cell is equal to p, and the number of hidden processing units will presumably be greater. For the spatially varying case, each cell would have its own unique set of input-to-hidden unit connections along with a unique set of weights. The hidden-to-output unit weight structure would be the same for all observations.

The neighborhood defined above satisfies requirement 1 outlined earlier, that the number of input units must be constant for all input-to-output mappings. Requirement 3 is usually determined via trial and error, i.e., the number of hidden units is increased over repeated training sessions until the network performance does not significantly improve. The fourth requirement can be determined by comparing the performance of the network to patterns previously unseen, i.e., not used during training. Finally, requirement 5 can be satisfied by repeatedly training the network using the same number of hidden units but different random initializations of the weight structure, and comparing the results. Thus requirements 3, 4 and 5 are satisfied during the optimization of the network, i.e., by repeated training and comparing of results.

What remains to be determined, however, is how to obtain not only grid-based data but also a large enough number of observations to train the network (requirements 1 and 2). This is where GIS and geocoding, discussed in section 2, come into the picture. Recall that one of the main advantages of GIS-based data is that it can be queried according to any geographic area, including user-defined ones. If we create a grid-based polygon coverage and overlay it on a study area such as a city, then we can easily determine the number of data points that fall within each grid cell. In addition, we can break down the data further according to other characteristics. For example, with 911 or police data we can count the frequencies of drug calls for service, robbery arrests or burglaries. In addition, if property ownership data is geocoded by address, then for each grid cell we can also determine variables such as the average assessed value of homes, the proportion of commercial properties, etc. Thus requirement 1 is also satisfied.

The final requirement that still needs to be satisfied is the number of data points. If we have four years' worth of annual data and 400 grid cells, we would have 1,600 observations distributed over four epochs. This is not enough to train a neural network. However, if the geocoded data points also have a date associated with them, then we can break down the data even further into time slices such as months or weeks and thus increase the total number of data points considerably. For example, four years' worth of monthly data and 400 grid cells would yield 19,200 observations distributed over 48 epochs. This would conceivably be enough. Thus requirement 2 would also be satisfied.

The next section describes how the neural network architecture discussed in this section was used and trained to create an early warning system for 911 drug calls for service. The data for the early warning system was created using the DMAP GIS outlined in the second section.

## A NEURAL NETWORK-BASED EARLY WARNING SYSTEM FOR 911 DRUG CALLS FOR SERVICE

Like many other cities of its size, Pittsburgh experienced a marked increase in street-level drug trafficking during the late 1980s and early 1990s as a result of the crack epidemic. Although crack cocaine use was already prevalent in larger cities such as Los Angeles, Detroit and New York before that time, historical evidence shows that it generally takes a few years for new illicit drugs to disperse to smaller cities that are not ports of entry for drug smugglers. Prior to the appearance of crack cocaine, street-level drug dealing in Pittsburgh was largely confined to two areas that specialized primarily in heroin and marijuana. Other sporadic areas of open-air drug dealing did exist, but were mainly limited to the sale of prescription drugs such as painkillers and the "Yuppie" drug powder cocaine.

In the summer of 1991, Pittsburgh also experienced a surge in gang-related violence. While initially most gangs were merely loosely organized groups of adolescents, experienced gang members from larger cities quickly attempted to gain a foothold in what they perceived as "virgin territory" for crack cocaine sales. Street-level drug markets in other major cities were already saturated by dealers, and there was little opportunity for entry into a market tightly controlled by gangs. Pittsburgh, on the other hand, was still a "free-for-all": demand was greater than supply. Thus, at least a part of the increase in violence can be attributed to street gangs setting up and defending "turfs" within which they conducted their illicit drug trade.

In reacting to the increase in street-level drug dealing, the Pittsburgh Bureau of Police used disruptive enforcement strategies that were proven highly successful in other cities: reverse stings, undercover buys, on-sight arrests of drug dealers after having observed illicit transactions and placing community-oriented police officers in plain view of established drug hot spots. These strategies were used because street-level drug dealing is widely regarded as a weak link in the chain: once a street market has been disrupted it is very difficult for dealers to relocate (Cohen et al., 1993). They are unable to advertise their new location, and are severely restricted in establishing new ones because they might infringe upon turfs already claimed by other drug dealers. However, there were a few instances where new hot spots did eventually surface.

While DMAP performed quite well at tracking the geographic displacement of drug dealers via its ability to plot the locations and frequencies of the number of drug calls for service and drug arrests, it did not perform as well at identifying emerging drug markets. The reason for this is twofold: first, police officers rarely make arrests in areas in which they are unaware that street-level drug dealing is going on unless they happen to stumble upon a transaction. Street sweeps tend to concentrate on known drug markets. Second, residents of areas in which street-level drug dealing is a new phenomenon frequently are unaware of what is going on. They initially do not perceive the activity as drug dealing. Rather, they tend to notice an increase in the level of crimes associated with street-level drug dealing such as robberies, burglaries and assaults. Thus there is a lag between the time a drug market has established itself and when residents begin to make drug-related calls for service.

An early warning system for emerging street-level drug markets must therefore be able to predict drug calls for service based on factors other than the level of drug calls for service in previous time periods. Based on the results of previous work and the availability of data from DMAP, it was decided to use three types of calls for service as indicators of emerging drug activity: weapon-related calls (shots fired, person shot, person with a gun, etc.), robbery calls and assaults-related calls. Cohen et al. (1993) showed that ecological factors such as the proportion of commercial properties in an area are important contributors to the level of drug calls for service. Commercial areas, especially older ones, lend themselves more to open-air drug dealing because of factors such as the relative lack of population outside of business hours (there are fewer residents to observe drug dealing). Thus the proportion of residential and commercial properties were included as indicator variables. Open-air drug dealing is a seasonal phenomenon: in the winter months drug dealers tend to stay inside not only because it is cold, but also because fewer people are on the streets and they become more visible. A seasonality index was therefore also included.

The data for the early warning system were obtained by superimposing a grid on the area of the city of Pittsburgh (see Figure 5), and aggregating data for each grid cell. The cells are 2,150 feet square, resulting in a total number of 445 cells. It was important not to make the cells too small. Otherwise, only few cells would have more than one or two calls for service, while cells that were too large would have resulted in too few data points for neural net modeling.

Call-for-service data were obtained by counting the number of calls per month within each cell using the xy coordinates of the geocoded locations. The data spanned the years 1990 to 1992, resulting in 35 months' worth of data (December 1992 could not be used since there was no value for the number of drug-related calls for service in January 1993). With 445 cells, the total number of data points was therefore 15,575. The relative proportions of commercial and residential properties were arrived at by relating property ownership information to parcel polygons via the lot and block number. The xy coordinates of the geographic center of a parcel were then used to determine which grid cell a particular property falls into. The zoning classification for each property provided the basis for the relative frequencies. Finally, the seasonal index was arrived at by assigning values between 0.1 and 0.9 in equal increments to each month, where a value of 0.9 was assigned to June and July and 0.1 to December and January.

**Figure 5: EWS Tile Structure vs. 1990 Census Tracts**

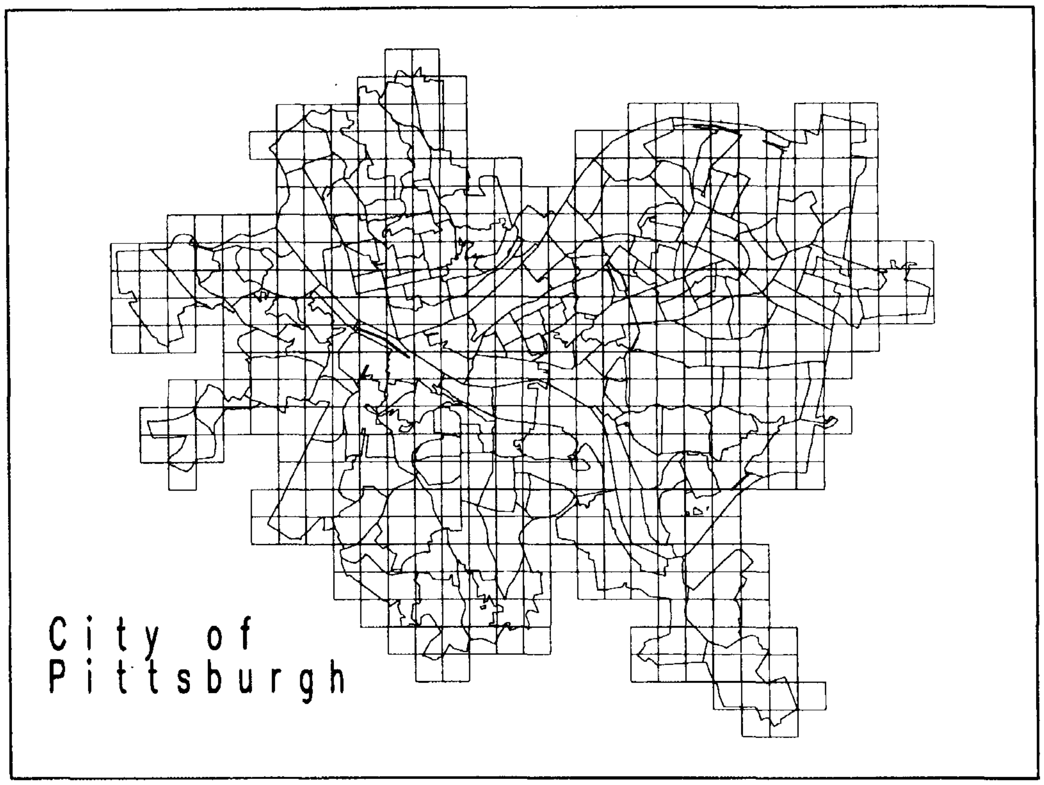


**Table 1: Annual Total Number of Calls for Service**

| | Drugs | Weapons | Robberies | Assaults |
|---|---|---|---|---|
| 1990 | 5053 | 3580 | 1922 | 8618 |
| 1991 | 6397 | 4523 | 2309 | 7154 |
| 1992 | 6223 | 6622 | 2699 | 6147 |

Table 1 shows the total number of calls for service, by year, for each of the four nature codes used in this study. Notice the remarkable increase in the number of weapon-related calls for service of almost 100% over three years. The number of drug- and robbery-related calls for service also increased; only assaults showed a decline. Figure 6 shows the number of calls for each nature code broken down by month. The seasonal variation is quite noticeable, especially for drugs, weapons and assaults. The figure also shows how, with the exception of assaults, most of the increase in the number of calls for service is accounted for in the summer months.

Using the data described above, three methods were employed to estimate the one-step-ahead forecasting model specified in equations (7) and (8). Ordinary least squares regression with six independent variables for each cell in the neighborhood and a neural network with an architecture similar to that shown in Figure 4 were used to estimate model (7). The only difference between the network shown in Figure 4 and that used to estimate (7) was that each cell in the neighborhood had six input units instead of just one, and the number of hidden units was nine instead of three. Model (8) was estimated using a neural network with spatially varying input-to-hidden unit weights. In other words, each cell had its own unique set of weights between the input and hidden layers. The neural network with spatially varying weights also had six input units per cell and nine hidden units.

Both neural networks were estimated using a value of 0.001 for the learning rate. In order to determine whether the neural networks were overfitting the data and to compare the robustness of each methodology, only two years' worth of data (1990 and 1991) were used to estimate the regression parameters and to train the networks (the training data set). The 1992 data were used to test how well each method performed on data not used during training or for estimation (the unseen data set). The program utilized to estimate the neural networks was custom written using the C programming language, and was run on a Sun Microsystems Sparc 20 workstation with 128 Megabytes of RAM. In order to limit the amount of computer time used, the number of iterations was limited to 15,000. In spite of the relatively powerful hardware and code optimization techniques, it took between three and six days for each network to either converge or reach the limit of 15,000 iterations.

Each network architecture was estimated four times with different random initializations of the weights in order to determine whether the results were consistent and the networks were not stuck in a local minimum. This was indeed the case. For each network architecture and in all four replications the residual sum of squared errors differed by no more than 0.5%.

Table 2 shows a comparison of the results of the three methodologies. For reasons of brevity, the estimated regression parameters (of

**Figure 6: Monthly Calls for Service**

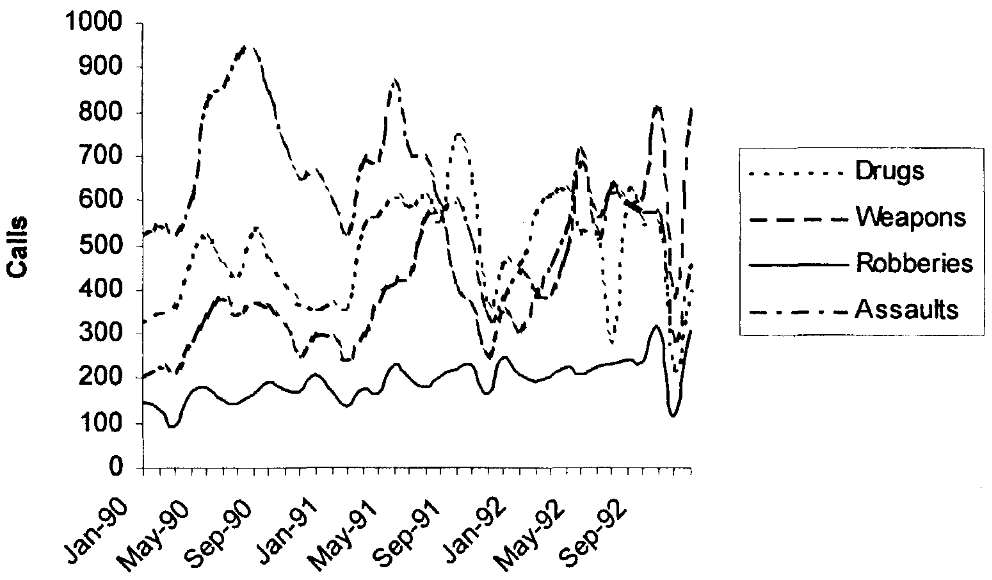


**Table 2: Performance Comparison of Forecasting Methods**

| | Regression | | Constant Weights | | Varying Weights | |
|---|---|---|---|---|---|---|
| | Residual Sum of Squares | $R^2$ | Residual Sum of Squares | $R^2$ | Residual Sum of Squares | $R^2$ |
| Training | 120,481 | .4185 | 120,589 | .4117 | 95,032 | .5343 |
| Unseen | 61,345 | .3457 | 51,719 | .3983 | 41,257 | .5326 |

which there were 47) are not presented and only one replication of each of the network architectures is shown. The constant weight version of the neural network took 9,447 iterations to converge to a minimum, whereas the varying weight version was stopped at 15,000 iterations. On the training data set, the regression model and the constant weight network architecture performed about the same while the varying weight architecture did significantly better.

The differences in performance were more pronounced when the estimated parameters and weights were used on the unseen data set.

The fit of the regression model dropped sharply, from an r square of .4185 to .3457, while that of both network architectures only did so very slightly. This is a strong indicator, although by no means statistical proof, that neural networks are a robust and consistent spatial forecasting methodology. The constant weight architecture performed 15.2% better than regression on the unseen data set, whereas the varying weight network outperformed regression by 54.1%. This is a highly significant difference.

Figures 7 through 10 are choropleth maps of actual and predicted drug calls for service, by grid cell, for the month of August 1992. This month is part of the unseen data set. Unfortunately, some of the detail on the maps is lost due to the lack of color (black-and-white choropleth maps allow for fewer class intervals; color maps show the differences to be more pronounced). It is nevertheless clear that the neural network architecture with spatially varying weights more accurately predicts hot spots of drug activity than the other two methods.

In looking at the map of actual calls for service it is apparent that most cells are zero, i.e., had no drug calls for service during the month of August 1992. Notice how both the regression and the constant weight models tend to perform relatively poorly on those cells with no calls. Table 3 shows the mean absolute percent forecast error for all cells and for those with at least one drug call for service. All methods tend to perform better on non-zero cells. However, the difference in performance is much less pronounced for the neural network with spatially varying weights. Finally, both neural network models perform better than regression for both zero and non-zero cells. Again, this is highly significant: the mean absolute percent error is often viewed by forecasters to be more indicative of a model's forecasting capability than the r squared.

**Table 3: Mean Absolute Percent Forecast Error on Unseen Data Set**

| | **Regression** | **Constant Weights** | **Varying Weights** |
|---|---|---|---|
| All Data Points | 678.68 | 549.20 | 253.86 |
| Non-Zero Cells | 137.49 | 127.34 | 110.98 |

**Figure 7: Drug Calls for Service, by Tile, August 1992**

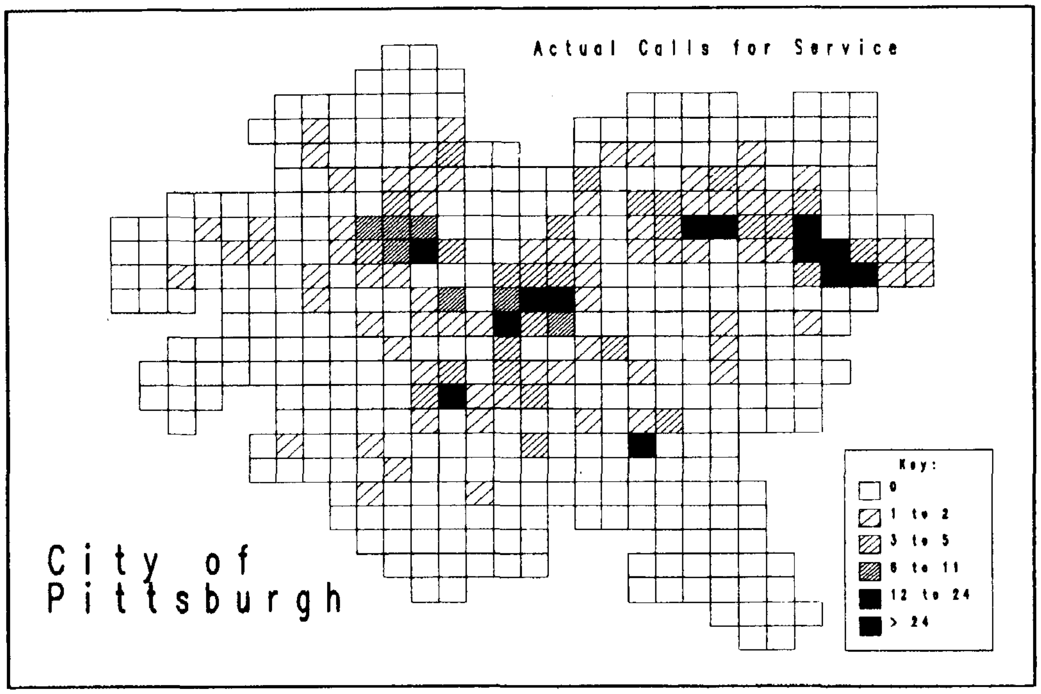


**Figure 8: Drug Calls for Service, by Tile, August 1992**

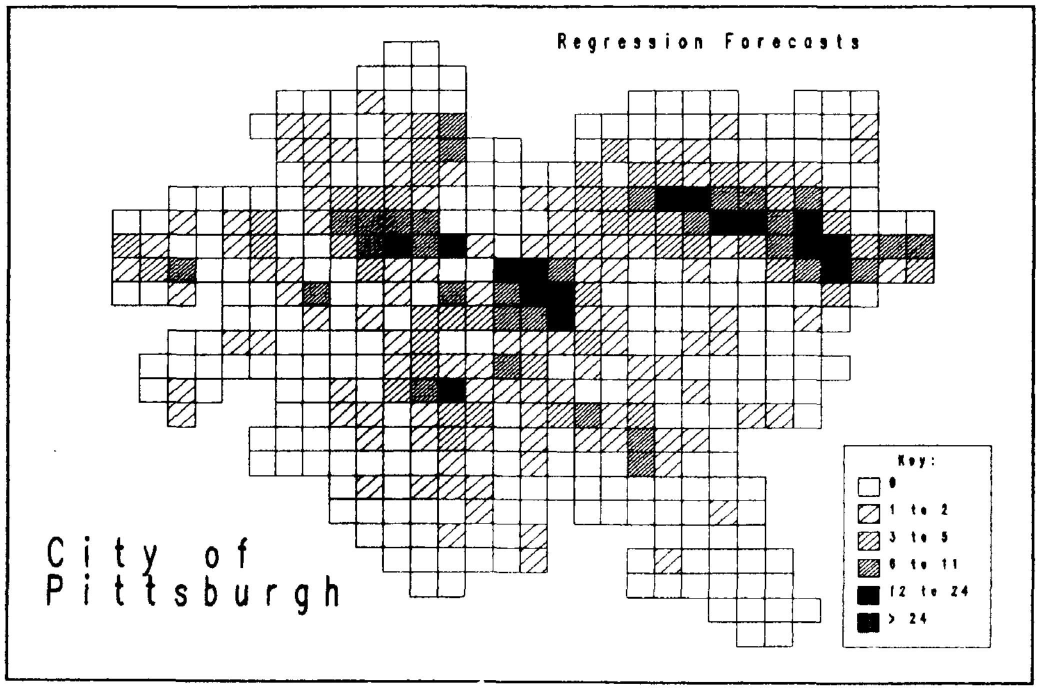


**Figure 9: Drug Calls for Service, by Tile, August, 1992**

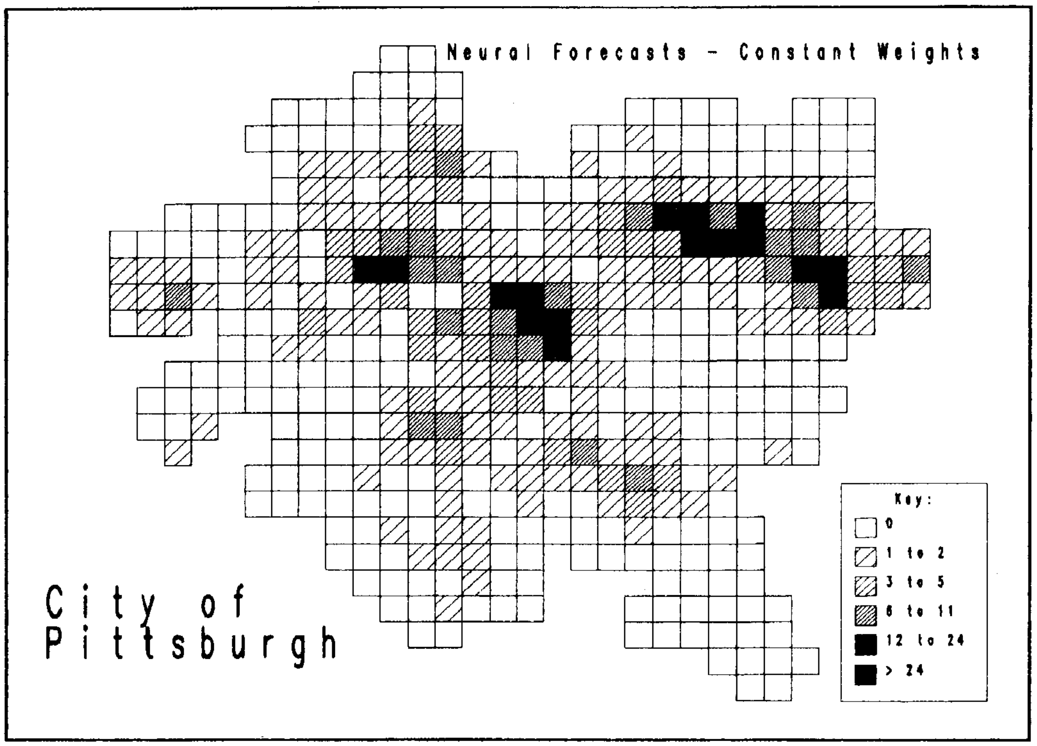
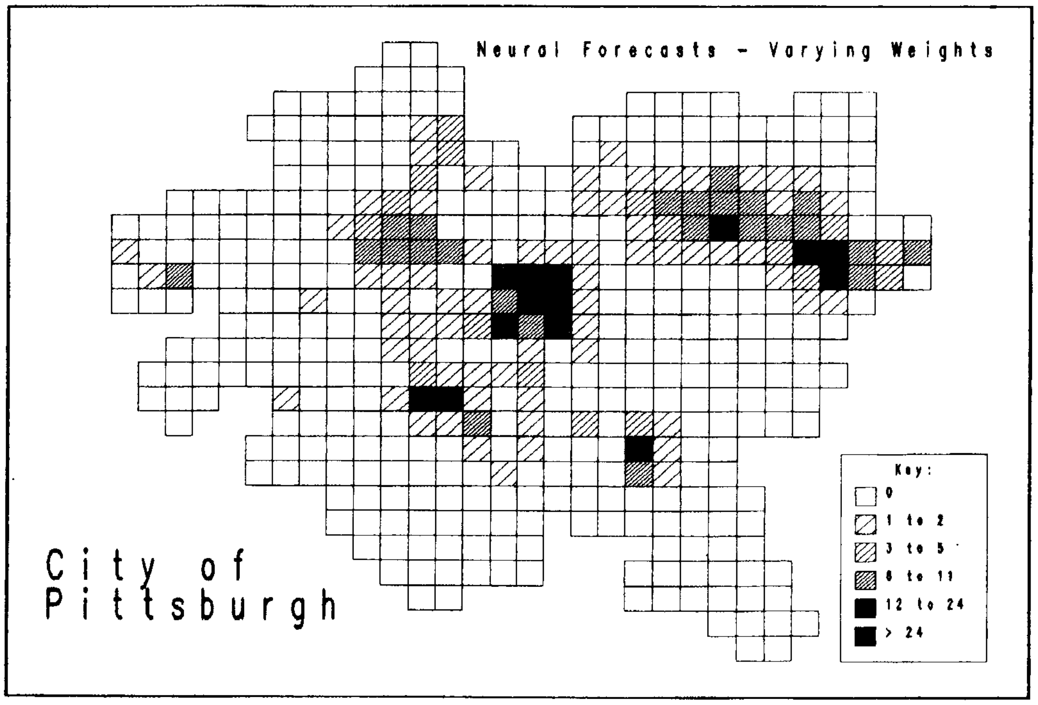


**Figure 10: Drug Calls for Service, by Tile, August 1992**



## CONCLUSION

The results described in the previous section are very encouraging. Both neural network architectures performed at least as well in terms of model fit as a fairly complex ordinary least squares regression model, and, in one case, significantly better. In both cases artificial neural networks appear to be more robust at estimating forecasts than regression. One disadvantage of neural networks is that there currently are no tests of statistical significance for the estimated weight structures. However, if the main goal of a model is to provide good forecasts rather than to analyze relationships between dependent and independent variables, then this should not be an issue.

Producing the data set, estimating the neural networks and mapping the results took a lot of computing power. The reader should not be discouraged by this fact. While the average desktop computer in use today does not have the computing capabilities of the hardware employed in this study, recent advances in computer technology indicate that this will change in the very near future. For example, the code used for the neural networks was developed on a now five-year-old Sun Microsystems Sparc 2 workstation with only 32 megabytes of RAM. At the time it was purchased the Sparc 2 was one of the fastest workstations commercially available, at a cost of around $25,000. In contrast, a high-end Pentium PC available for about $4,000 today is two or three times faster than the Sparc 2.

The early warning system described in this study represents only a first attempt at using artificial neural networks for GIS-based space-time forecasting. As far as neural networks are concerned the algorithm used is a fairly simple one. In developing the algorithm, the primary focus was on adapting existing neural network technology to work with spatial data and to simply see if it would work. However, the fact that even two fairly simple artificial neural network algorithms are able to outperform a fairly complex statistical model is highly significant. Certainly more work remains to be done in order to determine whether this is also true for other data sets. Rigorous testing with Monte Carlo data would also provide more insight.

A further next step is to employ some of the recent advances made by researchers in other areas of neural network applications. For example, there are many ways in which feed-forward networks with backpropagation can be modified to converge more quickly to a solution. An additional improvement would be to employ genetic algorithms to develop self-optimizing network architectures.

Finally, there are a variety of types of neural networks, many of which can potentially be adapted for spatial modeling and integration into GIS. In addition, forecasting is only one of countless ways in which GIS can be used for modeling. Exploring and improving the ways in which neural networks can be applied to GIS promises to be an exciting field in the years to come.

◆

**Acknowledgments:** Funded in part by the U.S. National Institute of Justice, Drug Market Analysis Program, Grant 90-IJ-CX-007. Special thanks to Darrell Packer and Glenn Bigler for the extra CPU time.

## REFERENCES

Anselin, L. (1988). *Spatial Econometrics: Methods and Models.* Dordrecht, NETH: Kluwer.

Bates, J.M. and C.W.J. Granger (1969). "The Combination of Forecasts." *Operations Research Quarterly* 20:451-68.

Bookser, M.C. (1991). "The Future of Information Management in Pennsylvania Law Enforcement." Paper presented at the International Symposium on the Future of Law Enforcement, Quantico, VA, April.

Box, G.E.P. and G.M. Jenkins (1970). *Time Series Analysis, Forecasting and Control.* San Francisco, CA: Holden-Day.

Canter, P. (1993). "State of the Statistical Art: Point Pattern Analysis." In: C.R. Block and M. Dabdoub (eds.), *Workshop on Crime Analysis Through Computer Mapping Proceedings.* Chicago, IL: Illinois Criminal Justice Information Authority.

Carpenter, G.A. and S. Grossberg (1991). *Pattern Recognition by Self-Organizing Neural Networks.* Cambridge, MA: MIT Press.

Cassetti, E. (1982). "Drift Analysis of Regression Parameters: An Application to the Investigation of Fertility Development Relations." *Modeling and Simulation* 13:961-66.

——and J.P. Jones (1992). *Applications of the Expansion Method.* London, UK: Routledge.

Clemen, R. T. (1989). "Combining Forecasts: An Annotated Bibliography." *International Journal of Forecasting* 5:559-83.

Cleveland, W.S. and S.J. Devlin. (1988). "Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting." *Journal of the American Statistical Association* 83:596-610.

Cliff, A.D., P. Haggett, J.K. Ord, K. Bassett, and R. Davies (1975). *Elements of Spatial Structure.* Cambridge, UK: Cambridge University Press.

Cohen, J., W.L. Gorr, and A.M. Olligschlaeger (1993). *Modeling Street-Level Illicit Drug Markets,* Working Paper #93-64. Pittsburgh, PA: H. John Heinz III School of Public Policy and Management, Carnegie Mellon University.

David, M. (1977). *Geostatistical Ore Reserve Estimation.* Amsterdam, NETH: Elsevier.

Foster, S.A. and W.L. Gorr (1986). "An Adaptive Filter for Estimating Spatially Varying Parameters: Application to Modeling Police Hours Spent in Response to Calls for Service." *Management Science* 32: 878-89.

Gardner, M. (1970). "The Fantastic Combinations of John Conway's New Solitaire Game 'Life.'" *Scientific American* 223(4):120-123.

Gorr, W.L. and A.M. Olligschlaeger (1994). "Weighted Spatial Adaptive Filtering: Monte Carlo Studies and Application to Illicit Drug Market Modeling." *Geographical Analysis* 26:67-87.

Green, L.A. (1993). "Drug Nuisance Abatement, Offender Movement Patterns, and Implications for Spatial Displacement Analysis." In: C.R. Block and M. Dabdoub (eds.), *Workshop on Crime Analysis Through Computer Mapping Proceedings.* Chicago, IL: Illinois Criminal Justice Information Authority.

Haining, T. (1990). *Spatial Data Analysis in the Social and Environmental Sciences.* Cambridge, UK: Cambridge University Press.

Harp, S.A., T. Samad, and A. Guha (1990). "Designing Application-Specific Neural Networks Using the Genetic Algorithm." In: D.S. Touretzky (ed.), *Advances in Neural Information Processing Systems* vol. 2. San Mateo, CA: Morgan Kaufmann.

Hecht-Nielsen, R. (1990). *Neurocomputing,* Reading, MA: Addison-Wesley.

Holland, J. (1975). *Adaptation in Natural and Artificial Systems.* Ann Arbor, MI: University of Michigan Press.

Hornik, K., M. Stinchcombe, and H. White (1989). "Multilayer Feed-forward Networks are Universal Approximators." In: *Neural Networks,* vol. 2. Elmsford, NY: Pergamon Press.

Kroese, B. J. and P.P. Van der Smagt, (1993). *An Introduction to Neural Networks.* Lecture Notes. Amsterdam, NETH: University of Amsterdam.

Maltz, M.D. (1993). "Crime Mapping and the Drug Market Analysis Program (DMAP)." In: C.R. Block and M. Dabdoub (eds.), *Workshop on Crime Analysis Through Computer Mapping Proceedings.* Chicago, IL: Illinois Criminal Justice Information Authority.

McCulloch, W.S. and W. Pitts (1943). "A Logical Calculus of the Ideas Imminent in Nervous Activity." *Bulletin of Mathematical Biophysics* 5:115-33.

McEwen, J.T., and F.S. Taxman, (1994). "Applications of Computerized Mapping to Police Operations." In: J. Eck and D. Weisburd (eds.), *Crime and Place.* Crime Prevention Studies, vol. 4. Monsey, NY: Criminal Justice Press.

Minsky, M. and S. Papert (1969). *Perceptrons.* Cambridge, MA: MIT Press.

Pickover, C.A. (1990). *Computers, Pattern, Chaos and Beauty.* New York, NY: St. Martin's Press.

Poli, I. and R.D. Jones (1994). "A Neural Net Model for Prediction." *Journal of the American Statistical Association* 89:117-121.

Rogers, D. (1990). "Predicting Weather Using a Genetic Memory: A Combination of Kanerva's Sparse Distributed Memory with Holland's Genetic Algorithms." In: D.S. Touretzky (ed.), *Advances in Neural Information Processing Systems,* vol. 2. San Mateo, CA: Morgan Kaufmann.

Rumelhart, D.E. and J.L. McClelland (1988). *Parallel Distributed Processing,* vols. 1 and 2. Cambridge, MA: MIT Press.

——G.E. Hinton, and R.J. Williams (1988). "Learning Internal Representations by Error Propagation." In: D.E. Rumelhart and J.L. McClelland (eds.), *Parallel Distributed Processing,* vol. 1. Cambridge, MA: MIT Press.

Schroeder, M. (1991). *Fractals, Chaos and Power Laws.* New York, NY: W.H. Freeman.

Tobler, W.R. (1969). "Geographical Filters and their Inverses." *Geographical Analysis* 1:234-253.

Toffoli, T. and N. Margolus (1987). *Cellular Automata Machines: A New Environment for Modeling,* MIT Press Series in Scientific Computation. Cambridge, MA: MIT Press.

Trojanowicz, Robert (1986). *Community Policing Programs: A Twenty-Year View,* Community Policing Series No. 10. East Lansing, MI: National Neighborhood Foot Patrol Center.

Weiss, S.M. and C.A. Kulikowski (1991). *Computer Systems That Learn.* San Mateo, CA: Morgan Kaufmann.

White, H. (1988). "Economic Prediction Using Neural Networks: The Case of IBM Daily Stock Returns." *Proceedings of the IEEE International Conference on Neural Networks.* San Diego, CA.

Widrow, G. and M.E. Hoff (1960). *Adaptive Switching Circuits,* Western Electronic Show and Convention Record, Part 4, pp.96-104. New York, NY: Institute of Radio Engineers.